16. Automatic Score Extraction with Optical Music Recognition (OMR)

Ichiro Fujinaga, Andrew Hankinson, Laurent Pugin

Optical music recognition (OMR) describes the process of automatically transcribing music notation from a digital image. Although similar to optical character recognition (OCR), the process and procedures of OMR diverge due to the fundamental differences between text and music notation, such as the two-dimensional nature of the notation system and the overlay of music symbols on top of staff lines. The OMR process can be described as a sequence of steps, with techniques adapted from disciplines including image processing, machine learning, grammars, and notation encoding. The sequence and specific techniques used can differ depending on the condition of the image, the type of notation, and the desired output.

Several commercial and open-source OMR software systems have been available since the mid-1990s. Most of them are designed to be used by individuals and recognize common (post-18th-century) Western music notation, though there have been some efforts to recognize other types of music notation such as for the lute and for earlier Western music.

Even though traditional applications of OMR have focused on small-scale recognition tasks, typically as an automated method of musical entry for score editing, new applications of large-scale OMR are under development, where automated recognition is the central technology for building *full-music* search systems, similar to the large-scale full-text recognition efforts.

16.1 History

Computer-based optical recognition technologies, including both OMR and OCR, have been under development since the early days of computing technology. Computerized OCR was first developed in 1951 [16.1] and sold to large corporations, such as Reader's Digest and AT&T, to help process subscription and billing information [16.2].

Pruslin [16.3] demonstrated the first optical music recognition system. This system operated on a single measure of common Western music notation (CWMN), and was capable of recognizing a limited set of musical symbols. Several years later, *Prerau* [16.4] introduced the *DO-RE-MI* OMR system, capable of recognizing three measures of printed CWMN consisting of a single voice on two staves in a single font.

The goal of most early OMR development was a universal recognition system, capable of recognizing the entirety of music notation output, in much the same way that early OCR systems were envisioned as a tool capable of complete and accurate transcription of all textual documents [16.5, 6]. Despite understanding the limitations of the early OMR systems, Prerau concludes that the technique *should be able to be expanded to the recognition of all printed music* [16.7]. This early optimism was driven by a näive (in hindsight) understanding of the diversity and complexity of document contents. For text, page features such as

R. Bader (Ed.), *Springer Handbook of Systematic Musicology*, https://doi.org/10.1007/978-3-662-55004-5_16

columns, figures, tables, footnotes, and even headings posed a challenge for accurate textual recognition. For music recognition, the troubles were even more acute since music notation styles and practices would vary by composer, publisher, repertoire and historical practices. By the 1990s, the goal of creating a universal recognition system had been largely abandoned in favor of repertoire and application-specific OMR systems capable of transcribing a well-defined subset of music documents [16.8]:

> [...] *in practice, composers and publishers often feel free to adapt old notation to new uses, and invent new notation, as they see fit. There are in fact national dialects of music notation, and musical works use many different levels of notational complexity. Thus it may not be possible to devise a single recognition system capable of recognizing all music notation.* [Pruslin 1966] *states that a complete solution to the music recognition problem is the specification of: which notes are present, what order they are played in, their time values or durations, and volume, tempo, and interpretation. This level of recognition suffices for only some of the applications listed* [later in this paper].

As research and development continued through the 1990s and 2000s, OMR systems were developed to specialize in transcribing specific repertoires or styles of notation. In addition to CWMN recognition systems, repertoire-specific recognition systems exist for many different music notation styles, including lute tablature [16.9–11], Byzantine chant [16.12, 13], mensural notation, both in print [16.14] and manuscript sources [16.15], and others.

Several OMR toolkits have also been developed to help assemble bespoke OMR systems. These toolkits have been used to build several of the aforementioned systems, and provide a generalized structure and toolset from which these customized OMR systems may be built. Examples of these frameworks include the CANTOR system [16.16] and the Gamera system [16.17]. While these systems present a more flexible approach to OMR, they require significantly more development expertise to create and run OMR than *turnkey* systems and, as such, are generally only used in research contexts.

16.2 Overview

The recognition process can generally be defined as identifying and contextualizing from a signal the information contained in it. It is a discriminative response to a specific stimulus that makes it possible to assign each object to a particular class. Recognizing shapes and reading are typical recognition processes performed by humans, and cognitive research has studied the complexity of this task. *Marr* [16.18], a pioneer in cognitive science, has proposed a theory of vision describing it as a bottom-up process, in which an image is first deconstructed into bidimensional primitives and then reconstructed as a spatial object. Since then, other studies [16.19] have shown that such a theory, however rational, is too restrictive and that a top-down activity is part of the human recognition processes, in particular when the subject does not easily understand the scene or the context. For example, when a human is reading a highly degraded document, the type of document and the context will be instrumental clues to deciphering its contents. For example, knowing that the document is a string quartet and reading the beginning of the bottom staff provides the reader with the information that the content to be read at that place is quite likely to be an F clef. Studies on human text reading have highlighted different reading techniques [16.20]. In most cases, the brain does not operate by letter decoding but rather by adopting a more global approach based on the linguistic context. This explains why reading known words within which letters are missing or are inverted can usually be achieved without problems. Conversely, when one has to read an unknown word, reading is performed via letter decoding and assembly.

The goal pursued in optical recognition of music scores and of documents is similar to the recognition process as described above. A major problem, however, is that it is extremely difficult to formalize this process algorithmically, especially if the aim is to design a system that can operate simultaneously with the bottom-up and top-down analysis mechanisms.

Among the various applications developed in the field of document recognition, the most advanced ones are certainly the optical character recognition (OCR) applications for which many functional and business solutions already exist. In fact, very early on, the results obtained in this field allowed complete solutions to be implemented for the recognition of typewritten texts or fonts, at least for fairly good documents having a relatively simple structure. In most systems, the recognition is performed by a sequential process of preprocessing, segmentation, classification and validation. This approach provides excellent results for two main reasons: firstly because character segmentation is

relatively easy and can be done simply by analyzing verticals projections, and secondly because classifiers such as neural networks (NN) or *k*-nearest-neighbor (*k*-NN) that are often used in these systems perform particularly well for this type of operation and on this type of data.

The results obtained in the recognition of cursive handwriting, or related tasks, are significantly less successful and it is unanimously agreed that this task is much more difficult [16.21]. In addition, it is striking to see how difficult it is, if not impossible, to adapt techniques traditionally used for recognition of printed texts to the recognition of handwritten text. In fact, the problem faced with handwriting recognition is a problem that can be summed up in a well-known paradox stated by *Sayre* [16.22]: *to recognize, we need to segment the input signal, but in order to segment appropriately we need to recognize*. The best results in handwriting recognition were obtained using other techniques than NN or *k*-NN that enable validating the recognition at different linguistic levels. The approach that has been the most widely used in recent years is hidden Markov models (HMM), especially because of their excellent noise absorption capacity and, above all, because of their linguistic context integration capacity [16.23].

16.3 OMR Challenges

OMR is a special case of document recognition. From a technical point of view, it has many similarities with OCR. Many of the challenges encountered in character recognition also arise with music. A recurring problem is the quality of documents considered. Some techniques can show promising results and yield high recognition rates when the documents considered are very clean, are straight, and show perfectly printed symbols. The difficulty of the task, however, increases dramatically with document degradation, if the document is skewed or curved, or if some symbols are poorly printed or partially deleted. In real-life cases, however, it is very rare that the documents to be processed are clean, straight, and perfectly printed. In most cases, we can expect imperfections of various types. They can be grouped into two distinct categories: the imperfections in the document itself and degradation introduced by the document acquisition phase.

Imperfections in the document can vary from one type of document to another, from one document to another, and sometimes even from one page to another. The following problems are commonly encountered:

- Printing imperfections (such as uneven absorption of ink by the paper)
- Partial erasure of symbols
- Ink bleed-through from the opposite side of the paper
- Stains
- Paper degradation (yellowing, foxing, mold and mildew etc.)
- Holes or tears.

An essential step in the recognition process is digitization, where the analog signal (an original document, a photo etc.) is converted into a digital image. This document acquisition phase is not always limited to a single capture event. In practice it is composed of several successive steps, and in the recognition stage the user may not have a clean and clear image of the original physical document but only a version that previously went through different imaging processes. In some cases for instance, the document is photographed before being digitized. A source may have been photographed and then transferred to a microfilm or microfiche, and ultimately only the microfilm is available to be digitized. Each step of the of document acquisition can introduce various types of unwanted artifacts:

- Document image skewing
- Noise
- Bulged or curved appearance of the document image
- Nonuniform lighting
- Borders around the document image
- Partial content of other pages around the document image.

An important consideration during this phase is the digitization resolution. The resolution has a direct effect on the size and definition of details that can be captured from the physical object. Musical notation contains specific features that depend on relatively small symbols, for example staccato, dotted notes and accidentals. *Ng* and *Boyle* [16.24] showed that with a document scanned at a resolution of 300 dpi, the distinction between a sharp and a natural could be ambiguous. Several studies [16.25, 26] have shown that as a result the appropriate resolution for capturing details from a music document is higher than that for text.

While OCR and OMR are similar tasks in many aspects, they also have fundamental differences that make OMR a more difficult task than OCR. The first difference concerns the placement of symbols on a page

and how they are arranged. Text is, for the most part, unidimensional. Characters and words are typically composed in a horizontal line, and each line has no vertical relationship with the parallel lines above or below it. This approach is not appropriate for music, because it is necessary in music to consider both the vertical and the horizontal dimensions simultaneously. The addition of the vertical dimension poses challenges in accurately determining the pitch of a note, for example, or the component pitches of a chord, while simultaneously information embedded in the horizontal dimension determines co-occurrences of both sounding and nonsounding events (i. e., notes that sound at the same time, or ties that control the sounding duration of a note).

Another difference is the musical characteristic that symbols are superimposed on staff lines. Superimposition of the elements is known to be a difficult problem of recognition of forms [16.27]. Many segmentation algorithms widely used in textual recognition are ineffective with music scores because they operate by edge detection. Thus, in OMR, the superposition of the symbols makes segmentation a particularly critical phase. Most musical recognition systems include the removal of the staff lines, which is particularly difficult when a musical symbol merges with a staff line, for example on top of an F clef. In practice, several factors can make staff line detection and removal difficult:

- The lines are usually not perfectly straight
- The line thickness is often variable
- The lines may be interrupted at certain points.

Many musical symbols have similar characteristics to each other (e.g., a half note and a quarter note are both made of a stem and a note head of the same size), but must be interpreted in vastly different ways. The visual difference between a note with a dotted duration and a note with a staccato is simply a slight shift in the position of the dot, but one means to lengthen the sounded note, and the other means to shorten it! This often leads to classification problems. Many musical symbols are not made of a single graphical element, but are *compound* symbols composed of two or more distinct elements. This is the case, for example, of an F clef, where the entire symbol is composed of three components: a curved line and two dots. Elements of compound symbols can belong to different types of symbols. A small dot can be the point of an F clef, a point of a dotted note, or a staccato point. Similarly, a sharp can be part of a key signature, the accidental of a note, or belong to a figured bass.

16.4 Technical Background

Several studies cover the early OMR techniques exhaustively, including *Blostein* and *Baird* [16.8], *Selfridge-Field* [16.28], *Bainbridge* and *Bell* [16.27], and more recently *Rebelo* et al. [16.21].

The nature of musical symbols has meant that from very early on OMR research had to look at structural recognition methods. The earliest research quickly showed that a functional recognition approach would work only for excessively simple cases. In general, symbols are grouped into two distinct categories: on one side, the symbolic, which can be treated as characters (keys, alterations, rests etc.), and on the other side, those that *Martin* and *Bellissant* [16.29] name the *iconic*, or assembled (e.g., note heads, stems, flags and beams), which are made of different primitives that may undergo various transformations and are assembled following certain rules.

Most OMR systems developed to date have more or less the same pipeline architecture that can be broken down into distinct phases:

1. Preprocessing of the image
2. Detection and removal of staff lines
3. Segmentation of the objects
4. Reconstruction of musical symbols
5. Classification and interpretation.

In many systems, musical rules are applied to increase accuracy. This can happen during several phases of the process, but most of the time the rules are applied during the reconstruction of the musical symbols from the primitives and during the classification of the symbols and the interpretation of the music content.

Using grammar to describe the musical notation and musical knowledge is a common technique [16.30]. Various methods have been proposed to integrate grammatical methods developed originally for languages into the bidimensional space of OMR. The transposition from one to two dimensions increases the complexity of the grammar, in both the design of the grammars and their use. This results in grammars that are either fairly simple but incomplete, or very complex but challenging to manage. One approach to simplify the problem is to have two distinct grammatical levels: low-level grammars, for describing the structure of a musical symbol constructed from primitives, and high-level grammars

for the description of the organization of musical notation itself [16.31].

16.4.1 Preprocessing

In many OMR research projects no particular attention is given to the preprocessing. The reason is that many of the problems to be solved are common to that faced by OCR, or even more generally in document recognition. In most researches a reference is made to the solutions proposed in these two research areas.

However, some more specific studies focusing on old documents looked more precisely at the thresholding problem during the binarization phase. This is the case with *MacMillan* et al. for the Levy project [16.32]. It shows that in this practical case, it is necessary to use more advanced methods of binarization. As does Ng for manuscript sources, MacMillan et al. use some locally adaptive thresholding algorithms. This technique allows the binarization to be optimized for document images with nonuniform lighting [16.33]. Furthermore, *Burgoyne* et al. [16.34] found that binarization methods that worked well for text documents did not work well for music documents and vice versa.

16.4.2 Staff-Line Detection and Removal

The detection and the removal of the staff lines is a key phase in OMR systems. This phase is often part of the preprocessing phase since usually the detection of the staff lines is used also to correct the skew of the document. Ideally, the staff lines are straight, parallel to each other, have constant thickness, and are horizontal. In reality, for various reasons, the lines may be curved, not parallel to each other, have varying thickness, and be skewed. A wide range of solutions has been proposed of tackle these challenges [16.21, 35]. These include projections, line tracing, two-dimensional vector fields, skeletonization, and graph-based approaches.

Once detected, staff lines are usually removed without removing the musical symbols. This step is critical, in particular with symbols touching the staff lines. In such cases, the removal of the staff lines may breakup the musical symbols into smaller pieces, which will then make their classification difficult without reconstruction. The difficulty of the task is therefore to find a solution that does not erase the staff line when it intersects with a symbol.

16.4.3 Recognition Architectures

The techniques used in OMR for locating and classifying symbols varies significantly from system to system and have evolved considerably since the first systems. This has been made possible by the considerable and steady increase of computational resources offered by new machines. Despite this diversity, some techniques are recurrent and are to be found similar in many approaches.

The simplest approach to locating and classifying the symbols is to look for the connected components and to classify them. This technology, already used by Prerau, generally uses simple measures such as the size of the bounding box of the symbol or the surface of the symbol. As simple as it is, this approach is still relatively effective because of the morphological diversity of some musical symbols (including in size) and can still be used today for a first classification phase.

Another approach widely used for the localization of the umsic symbols is to look for some easily detectable primitive and then to look around it for other primitives belonging to the expected symbol. For example, *Martin* and *Bellissant* [16.29] locate the note stems and then search for ellipses of note heads in the adjacent area. *Miyao* [16.36] uses a similar technique tailored for recognizing polyphonic passages. First, the note stems are located together with all the note heads and with stem flag candidates around each stem. Then, two neural networks are used to identify from the candidates those that are attached to a specific stem. With *Rossant* [16.37], vertical segments are not only used to locate the stems but also a whole series of symbols in which a vertical segment of a certain size is included: these symbols belong to various categories, such as notes, accidentals, bar lines, or certain types of rests. For the remaining symbols, Rossant performs a template correlation on limited zones: for locating whole note or half note rests, this area can be limited to a staff range, whereas the location of the whole notes, the correlation must be performed on a wider area.

Fujinaga [16.38] uses mainly vertical and horizontal projections. The symbols are located by vertical projection of the area between the upper line and the lower line. Different local projections are used to calculate the height, width, area, and the number of peaks to the vertical projection of the symbol. The data is used for classification in conjunction with some syntactic rules. For example, the first symbol on a staff is expected to be a clef; or a beam of eighth notes can contain only notes, rests or accidental; or a duration dot can be placed only after a note or rest. Classification is performed using a k-NN classifier together with a genetic algorithm.

Kato and *Inokuchi* [16.39] use a blackboard expert system where knowledge of musical notation is used in the process to remove ambiguities. The recognition is performed measure by measure once the clef, the

key signatures and the time signatures have been recognized. To allow optimal recognition of the different musical symbols that differ significantly in their size, their position, and their possible appearance frequency or importance, a wide variety of recognition methods are used. The architecture consists of four separate modules, which communicate with each other through a shared common data structure representing a measure. The role of the various modules is as follows:

1. Extraction of the primitives
2. Reconstruction of the symbols
3. Recognition of the symbols
4. Semantic analysis.

The memory shared by the four modules consists of five distinct layers representing five different levels of abstraction for a measure:

1. Pixels of the image
2. Primitives (note heads, stems, accidentals etc.)
3. Musical symbols (reconstituted notes, rests etc.)
4. Musical meaning of symbols (pitch, duration etc.)
5. Possible interpretation of the content of the measurement.

The recognition process is guided by a variable threshold that controls the recognition. Each layer makes assumptions that are verified by the upper layer from a tight line. If an upper layer considers a hypothesis unacceptable, the threshold is released and the treatment is reperformed at the lower level with the new threshold. For example, an unrecognized primitive will be put back in the pixels of the image. A document of good quality will be recognized with tighter thresholds and faster than poor-quality paper.

Ng and *Boyle* [16.24] use an iterative process of segmentation and recognition. Recognition is performed at different phases of the segmentation process, rather than requiring all the symbols to be segmented into primitive or broken signs prior to recognition. The purpose of this method is to avoid over-segmentation of the symbols. A first pass of recognition is performed directly after the removal of staff lines based on simple rules and using a k-NN classifier. For nonrecognized symbols, different types of subsegmentation and heuristics are performed.

Coüasnon [16.40] proposes a solution that uses the syntactic structure of musical notation. The idea is to use the a priori knowledge given by the music writing rules in order to guide the process of segmentation and labeling of objects. Coüasnon distinguishes between two types of information: the *physical* information corresponding to the arrangement of notes and attributes on the partition, and the *logical* information corresponding to the transformation of the notes in music writing. He defines a grammar that models this separation into two levels. The terminals of this grammar are the basic entities of the document description and are recognizable without contextual information in order to have information on which the recognition process can be based. They consist of terminal segments and symbols. The recognition process is performed in two phases corresponding to the two levels of information: graphic recognition and syntactic recognition. The first phase deals with the notes and recognizes the relative positioning of the attributes, while the second phase recognizes the symbols connected with a voice (slurs, dynamics etc.) and assigns the notes to the different voices depending on the vertical alignment and the number of beats per measure. The method specifically targets orchestral scores, where a distinction must be made between voices and staves: there may be up to three voices per staff and, conversely, a voice may be written on more than one staff.

Bainbridge and *Bell* [16.31] provide an extensible solution also based on the use of a grammar. The solution is meant to allow the recognition of different types of music notations and not be limited to CWMN. The system, named CANTOR, consists of distinct modules to be applied after the removal of stave lines:

1. Primitive recognition
2. Primitive assembly
3. Musical semantics.

Recognition of primitives is defined through a language specifically designed for the task. The idea is to accommodate a wide range of music writing by easily assigning a primitive recognition for each shape (e.g., projections, Hough transform etc.). Assembly of the primitive, the second module, is based on a slightly modified definite clause grammar, which has the limited scope of describing the taxonomy of musical notation. The purpose of keeping the scope of the grammar limited is to prevent it from becoming too complex and difficult to manage. The purpose of the last phase is to combine the symbols recognized by looking at their position in order to produce a structure representing the musical content of the image.

16.4.4 OMR Aggregation

Several researches have tried to improve OMR results by aggregating multiple commercial OMR tools. This approach relies on the fact that the commercial OMR tools all have their strengths and weaknesses and thus it should be possible to improve the overall recognition results by comparing the output of several tools. The first attempts were made by *Knopke* and *Byrd* [16.41]. Their approach involves two steps. First, the strengths

and the weaknesses of each tool that will be used are evaluated. Then the results of this evaluation are used for weighting the output of each tool when comparing their output and when the tools do not agree. *Bugge* et al. [16.42] propose a similar approach together with the definition of a dedicated format (MusicXiMpLe), a subset of the MusicXML file format used for the alignment of the tool's output and that is meant to facilitate the alignment of the data. More recently, *Church* and *Cuthbert* [16.43] have proposed a different approach that integrates rhythmic analysis in the alignment process.

16.5 Adaptive OMR

A wide range of music printing techniques have been used throughout history [16.44, 45], each having their own graphic particularities. This has led to the creation of highly varied music document symbols and notation practices. For each printing technique, publisher, editor, composer, or musical repertoire, the appearance of the notation can vary significantly. Printers often had their own distinctive font. The font shapes also vary depending on the size of the book, ranging from small in-octavo formats to larger in-folio or even in-plano formats. Font designs also have trends and changes occur over time. For example, the shape of the note heads in music fonts of the 16th and 17th centuries was generally diamond shaped. It then gradually changed and by the 18th century round note heads had become the trend.

In many OMR systems, the recognition of symbols is performed using supervised machine-learning algorithms. Supervised learning algorithms require ground-truth datasets of symbols in order to be trained. For each ground-truth symbol (e.g., a note head), features are extracted and fed to the algorithm for training. Once trained, the system will be able to identify similar symbols of the same category. When building an OMR system, gathering the ground-truth data for training is a highly time-consuming task since the amount of data required can be quite high, the data should ideally come from a wide range of sources, and each symbol needs to be correctly labeled. The high variability in the data makes it unrealistic to build an OMR system optimized for recognizing any document, even if targeting documents from a reduced historical period or one of restricted type.

A solution for tackling this issue is adaptive OMR [16.38, 46]. The assumption with adaptive OMR is that the system is not likely to consistently reach 100% accuracy and that most of the time, OMR workflows require human verification and correction to achieve usable results. In this context, adaptive OMR uses these correction results as further training data, feeding them back to the system and retraining the recognition system. As a result, systems will *learn* from their previous mistakes, correctly identifying previously misrecognized symbols through the expansion of its training data.

Pugin et al. [16.47] describe how this can be achieved using maximum a posteriori (MAP) adaptation, a technique widely used in handwriting and speech recognition. In a preliminary phase, a book-independent (BI) model is trained using ground-truth data taken from a selection of different books drawn from different printers and featuring variations in font shape and size, and is used as a seed for the recognition system. Thus the BI model gives acceptable results in general but is not specifically optimized for a particular source. In real-world usage, as each page of a book is recognized and corrected, the BI model is amended and optimized with MAP adaptation for the symbols in that book. As soon as the user has corrected the recognition errors on a newly processed page, that page is used as ground-truth to adapt the BI model. Eventually, after a user has corrected a number of pages, a book-dependent (BD) model emerges that is optimized for a particular set of sources (e.g., from a specific printer). The BD model can be saved and used for recognizing similar documents, creating a more optimal *bootstrap* than the general BI model.

16.6 Symbolic Music Encoding

The output of an OMR process is a machine-readable encoded music score. Whereas for text recognition, unformatted text files can serve as a basic encoded output of OCR processes, there is no equivalent for music encoding. There is no uniformly recognized basic music encoding scheme equivalent to unicode or to ASCII (American Standard Code for Information Interchange). For OMR applications, this means that

the encoded output has to be structured according to a defined music code, be it designed specifically for the OMR application, such as the Liszt format for the SharpEye commercial application, for example, or a more generic musical code.

Over the years, hundreds of musical codes have been proposed, illustrating both the complexity of music notation and the wide range of applications codes can serve [16.48]. Selfridge-Field groups them into three categories:

1. The codes targeting sound applications
2. The codes targeting notational applications
3. Those targeting analytical or more abstract applications.

Looking at how note pitches are handled is a simple way to help understand the difference between these code categories. In the first category, the most widely used code is undoubtedly MIDI (Musical Instrument Digital Interface), whose primary goal was to allow sound information to be exchanged between instruments. One particularity of MIDI regarding pitches is that it does not make the difference between, say, an F-sharp and a G-flat even if they were noted differently in the score. The second category of codes is meant to capture visual characteristics of music notation. Musical codes that belong to this category include DARMS (Digital Alternative Representation of Music Scores), one of the oldest computer codes; SCORE, the code used by the Score music notation software application developed by Leland Smith; or the notation interchange file format (NIFF). One particularity of the codes for notational applications is that they often do not have a concept of pitch. They do not store the notes by referring to the pitch name and the octave but instead by referring to the staff line on which they appear. The C_4 (C of the fourth octave) with the treble G-clef will be coded with the same staff line parameters as an E_2 with the bass F clef. Codes from the later category that target analytical applications, such as the plain and easy code, the kern representation, or the Essen associative code (EsAC), to mention only a few, store the note information through its pitch name and its octave.

Codes for notational applications are well suited to OMR applications. In fact, NIFF was developed with OMR in mind and is still used by some commercial software applications. NIFF was designed in the mid-1990s and subsequently came to be supported by a fair range of commercial OMR systems, including SharpEye, SmartScore, and PhotoScore. The use of NIFF remained limited, however, and it never really took off beyond its use by commercial OMR applications, which eventually abandoned it. Nor did another attempt in music code definition that was to extend MIDI for OMR with the expressive MIDI [16.48].

One reason NIFF failed to establish itself as a standard file format may be that it is a binary format. (Binary codes had the advantage of being more compact than ASCII codes, but disk space is no longer an issue for this type of data). But another reason might be that it is a notational code, with the limitation that only the graphical component of music notation is taken into account. In OMR, the recognition task acts as an encoding process that moves from a graphical domain to a logical domain. With a notational code such as NIFF, the processing of the data remains limited. For example, the pitch name and the octave of a note can remain unknown since only the position on the staff line is stored. In many uses of OMR data, however, further processing will be desirable in order to move to a code of the third category, the codes for an analytical or more advanced application. Typically, this will mean further processing the data in order to determine the pitch names and the octaves by taking into consideration the clef of the staff (or possible intermediate clef changes) and the key signature (or possible accidentals appearing previously in the measure). When the goal of the OMR process is the reediting of the original image, then a notational code can suffice, assuming that the editing tool can read the notational code produced by the OMR process. (It should to be noted that such an approach can have advantages regarding the impact of some recognition errors. For example, if a clef of a staff was wrongly recognized, it will have no impact on the accuracy of the content of the staff itself). This is, however, only one possible use case of OMR. Typical use cases of OMR require analytical data, for example to make an arrangement or transposition of the musical content. The analytical data can be derived from notational codes, but this is not necessarily trivial to do. It is one reason why encoding output formats other than NIFF were desirable and why it eventually became obsolete.

One analytical format that is widely used as an output of OMR processes is MusicXML, which is a code that started as an XML (extensible markup language) representation of MuseData [16.49]. It was developed by Michael Good and is now owned by MakeMusic. MusicXML is primarily an interchange format for exchanging music data between computer applications. This makes it well suited for exporting data from an OMR system to other types of applications, but it is not designed to represent and store the information of an OMR process.

Some XML alternatives to MusicXML were proposed for encoding music notation. One targets a wide range of applications, including OMR: the IEEE 1599–2008 standard designed by the IEEE 1599 Working

Group for XML Musical Application [16.50]. To our knowledge this standard is not currently used by any OMR systems.

16.6.1 The Music Encoding Initiative (MEI)

Over the last few years, a community-based project began to occupy an increasingly important role: the Music Encoding Initiative (MEI) [16.51]. The project was started around 2000 by Perry Roland from the University of Virginia. It was directly inspired by the Text Encoding Initiative (TEI), a leading project in text studies that became over the years the commonly accepted standard for representing and encoding texts. MEI pursues similar goals for musical documents. It is expressed in the form of an XML schema that defines the structure of the corresponding XML musical data. The first version of the MEI schema was released in 2010 in ODD (one document does it all), which is a schema definition solution developed by the TEI community that regroups in one document the schema definition and all the documentation related to it. The MEI schema is regularly updated to incorporate the latest improvements.

One feature of MEI in contrast to many attempts to define a musical code is that it is driven by an open community of contributors representing a wide range of backgrounds and interests. They include technologists, musicologists with various repertoires of expertise, and librarians. Beside the fact that MEI is community driven, it also has the particularity of accommodating a wide range of music notation and not being limited to CWMN – even though this is its first target. This flexibility is greatly facilitated by its modular approach and its nonmonolithic design. Each music notation type can be defined as a separate module, for example, MEI already includes in its core set of modules specialized modules for mensural and neume notations. The modularity of MEI does not serve only the separation of music notation types. MEI includes distinct modules for the metadata, for pointers and references, for linking with facsimile or for the definition of graphics, shapes, and symbols. Furthermore, each module can be modified or new modules added, should the default settings of MEI not be appropriate. This can happen when encoding a different notation type or when another type of application is targeted by the encoding. MEI includes a so-called customization service that allows part of the schema to be redefined or new encoding concepts to be introduced [16.52].

Even though MEI was not designed specifically for OMR applications, its richness and flexibility makes it perfectly appropriate for them [16.53]. The aforementioned module for linking with facsimile images is of particular interest. It makes it possible to easily and precisely refer from the encoding to zones in an image. This is similar to what is achieved with the hOCR format developed by Google for their open-source OCR project OCRopus. Since the output is text with OCR, they can use HTML (hypertext markup language) to mark up the text output [16.54]. However, the hOCR format enriches the HTML with additional information on layout, referring to an image, and data such as character recognition confidences. This is done in a way that does not affect the structure of the HTML content, which remains standard HTML. In a similar (though not identical) way, the MEI can include additional information, including references to image zones without the logical structure of the MEI to be modified. With MEI, references to images work by defining a facsimile subtree in the MEI file that regroups a set of surfaces (typically each one representing a page) that will contain a reference to an image and a list of zones in it. Each zone is identified with a unique identity to which any element in the encoding (a clef, for example) can refer. This not only enables robust linking between the encoding and the image to be generated, but it also has the advantage of keeping the information concerning the references to the images separate from the logical musical data.

16.7 Tools

16.7.1 Commercial OMR Software

Although several commercial OMR software packages have been marketed, currently only a handful products have stood the test of time. One of the survivors is also the first commercial OMR software ever published. Initially under the name MIDISCAN, Musitek released its product in August 1993 [16.55]. It was renamed later as SmartScore and its Lite version was bundled with a music editing software Finale 2003 in the spring of 2002. Another popular music editing software Sibelius (now owned by Avid Technology) uses Photoscore [16.56], which was originally released by Neutron for the Acorn system in 1997 (for Windows in 1999 and for the Mac in 2000). The ScoreMaker by Kawai has been available on the Windows system (in Japanese only) since 1995 and the capella-scan [16.57] from Germany has been around since about 2000 [16.58]. According to the developer, Graham Jones, the development of SharpEye started in 1996 [16.59], but it has not been updated

since 2006 (version 2.86) when he *transferred rights in SharpEye to another company* [16.60]. All of the commercial OMR software mentioned above are designed to work with CWMN.

16.7.2 Open-Source Tools and Toolkits

A few OMR tools are available as open-source. We can make the distinction between end-user ready-to-use desktop applications and toolkits available for creating custom OMR applications.

Desktop Applications

For CWMN, the only open-source tool available is Audiveris [16.61]. It is written in Java and is available under the General Public License (GPL) v2 open-source license. For text recognition, Audiveris uses the Google OCR Tesseract engine. The recognition engine is based on neural networks that can be retrained for specific sources. Audiveris includes a user interface for data correction and exports to MusicXML. Version 5 is currently in preparation.

One open-source OMR project specifically targets Renaissance music prints: the Aruspix project [16.62]. The project focuses on the development of techniques and tools for processing early typographic music prints from the 16th and 17th centuries. The Aruspix software application is a desktop application written in C++ (cross-platform) and is available under the GPL v3 license. Aruspix uses HMM (hidden Markov models) for the recognition with an original approach without staff removal [16.14]. It includes a user editor for correcting the results and an adaptive feature based on MAP adaptation [16.47]. Aruspix uses MEI as internal and output format.

Toolkits

The most widely used toolkit for building OMR systems is Gamera [16.17, 63]. Gamera is written in C++ and Python and is available under the GPL v2 license. It is designed as a toolkit for building pattern recognition systems with a strong focus on document recognition and OMR in particular. It includes a whole range of image processing algorithms together with a k-NN classifier. With this tool, the users can build their own recognition system by putting together scripts that will perform selected operations. These can include various preprocessing operations, such as noise removal, blurring, deskewing, contrast adjustment, sharpening, binarization, and morphology, for example. At the core of the Gamera system is the segmentation and the classification. Several algorithms are provided for these tasks together with a user interface for labeling the data. Both the image processing and the classifier parts are easily extendable, if necessary, with either C++ or Python extensions.

Gamera is distributed with add-on toolkits specifically designed for building OMR applications. The MusicStaves toolkit implements various algorithms for removing the staff lines [16.35]. It can be used in an interactive mode through the Gamera graphical user interface or in a noninteractive mode from the command line or scripts. The OTR (optical tablature recognition) toolkit is a package for the recognition of lute tablature [16.9, 11]. It includes scripts for the recognition of French, German, and Italian tablatures, all being slightly different tablature notations. Gamera also includes a Psaltiki recognition toolkit for the recognition of Byzantine chant notation used in chant notation of the Eastern churches [16.13]. It also distributes an OCR toolkit with custom page segmentation algorithms and heuristics rules for dealing with diacritics.

Gamera has been used for projects on other music notation, including neumes [16.64]. Gamera is also often used for evaluating research techniques, such as with Aruspix [16.65] where two OMR approaches are compared, or on more specific OMR steps such as staff line removal [16.66–68].

16.8 Future

Recent directions taken in OMR developments together with changes in technology make it possible to envisage completely new OMR paradigms. For decades, the goal of OMR has focused almost exclusively on providing image transcriptions for further processing with external software applications. Commercial OMR applications are usually desktop software applications that take an input image and output an encoded score without making further use of the link established between the image and its content.

Recent advances that can change this paradigm are manifold. First of all, open-source developments, such as Gamera, open new perspectives. The OMR technology is no longer embedded in a black box but instead remains open in modules that can be modified and assembled differently according to the needs and type

of documents to be processed. Many music documents raise similar though not identical challenges, and being able to adjust the tools is essential. The aforementioned Gamera MusicStaves toolkit is a perfect example in that regard.

Adaptive-OMR design is another direction that opens new perspectives and that also differs from the design of most commercial OMR desktop applications. Having systems that improve themselves over time is essential for bringing OMR to a next level. Correcting OMR errors is a highly time-consuming task, and the advantages gained by being able to feed this knowledge back into the system appear self-evident. Nonetheless, for years users have been using OMR systems and correcting their output without exploiting this data goldmine. This wasteful practice can be changed, but only if the output of the OMR process preserves the link between the output data and the image. It appears that the developments of MEI will play a key role in this endeavor. Having a standard format for preserving OMR data will allow the creation of large datasets. They will be usable for training or improving any OMR systems, which in turn will greatly facilitate the development and improvement of OMR technology.

One significant technological change that has occurred over the last few years is the emergence of online applications that can run in web browsers. It is now possible to develop software applications that run online without any application or plugin to be downloaded and installed locally. This radically changes the way software applications can be made available to the users. Over the next few years we can expect to see online OMR tools appearing that will be a significant breakthrough from the desktop applications currently available. In this context, the development of MEI engraving tools, such Verovio [16.69, 70] will be essential for making the OMR output editable online. For OMR, having online tools will also make it possible to develop adaptive systems where the corrections of one user can be immediately incorporated into the system and benefit all users, not only the one who made the corrections.

Online technology also transforms the way data can be made available. We now have access to thousands of images of music sources that are being digitized and made available by music libraries and archives all around the world. These images are an unparalleled resource for musicians, musicologists, and other scholars alike. However, only OMR technology can fully revolutionize the way they are made available to the user. The recent developments of Diva.js [16.71, 72] offer a glimpse of the future, where the output of the OMR process in MEI is displayed on top of high-resolution images directly in the web browser. This is a setup that is widely known for books but is still lacking for music, and large-scale online OMR technology is the key to fill this gap.

As large amounts of score data become available in symbolic formats, the next challenge is how to effectively use these large corpora of musical data. There are two basic issues: searching and analysis. Searching music is complex: there are pitches, rhythms, text, multiple voices sounding simultaneously, chords, different instruments etc. Linking metadata for sources and works (e.g., date of composition, location, or genre) and musical content is also essential. Analyzing music is also challenging given the large amounts of symbolic data that were not available previously. Thus, there will be questions such as what are the best ways to search through these corpora? What should the queries look like? What kinds of user interfaces are needed for queries and displays? What types of analysis of music are possible given these large datasets? The answers to these and other questions will create new research avenues paved with the aid of OMR technology.

References

16.1 D.H. Shepard: Apparatus for reading, Patent Application 2664758 (1951)

16.2 D. Martin: David H. Shepard, 84, Dies; Optical Reader Inventor, New York Times, 11 December 2007

16.3 D. Pruslin: *Automatic Recognition of Sheet Music*, Sc. D. Diss. (Massachusetts Institute of Technology, Cambridge 1966)

16.4 D. Prerau: *Computer Pattern Recognition of Standard Engraved Music Notation*, PhD Diss. (Massachusetts Institute of Technology, Cambridge 1970)

16.5 A. Samuel: The banishment of paper-work, New Sci. **21**(380), 529–530 (1964)

16.6 S. Mori, C. Suen, K. Yamamoto: Historical review of OCR research and development, Proc. IEEE **80**(7), 1029–1058 (1992)

16.7 D.S. Prerau: Computer pattern recognition of printed music. In: *Fall Joint Computer Conference 1971*, AFIP Conf. Proc., Vol. 39 (1971) pp. 153–162

16.8 D. Blostein, H.S. Baird: A critical survey of music image analysis. In: *Structured Document Image Analysis*, ed. by H.S. Baird, H. Bunke, K. Yamamoto (Springer, Berlin 1992) pp. 405–434

16.9 C. Dalitz, T. Karsten: Using the Gamera framework for building a lute tablature recognition system. In: *6th Int. Soc. Music Inf. Retr. Conf. (ISMIR)* (2005)

pp. 478–481
16.10 L.L. Wei, Q.A. Salih, H.S. Hock: Optical tablature recognition (OTR) system: Using Fourier descriptors as a recognition tool. In: *2008 International Conference on Audio, Language and Image Processing, Shanghai* (2008) pp. 1532–1539, https://doi.org/10.1109/ICALIP.2008.4590235
16.11 C. Dalitz, C. Pranzas: German lute tablature recognition. In: *Int. Conf. Document Anal. Recognit. (ICDAR)* (2009) pp. 371–375
16.12 V.G. Gezerlis, S. Theodoridis: Optical character recognition of the orthodox hellenic byzantine music notation, Pattern Recognit. **35**(4), 895–914 (2002)
16.13 C. Dalitz, G.K. Michalakis, C. Pranzas: Optical recognition of psaltic Byzantine chant notation, Int. J. Doc. Anal. Recognit. (IJDAR) **11**(3), 143–158 (2008)
16.14 L. Pugin: Optical music recognition of early typographic prints using hidden Markov models. In: *7th Int. Conf. Music Inf. Retr. (ISMIR)* (2006) pp. 53–56
16.15 L. Tardón, S. Sammartino, I. Barbancho, V. Gómez, A. Oliver: Optical music recognition for scores written in white mensural notation, EURASIP J. Image Video Process. **2009**, 843401 (2009), https://doi.org/10.1155/2009/843401
16.16 D. Bainbridge: *Extensible Optical Music Recognition*, PhD Diss. (University of Canterbury, Canterbury 1997)
16.17 K. MacMillan, M. Droettboom, I. Fujinaga: Gamera: Optical music recognition in a new shell. In: *Proc. Int. Comput. Music Conf.* (2002) pp. 482–485
16.18 D. Marr: *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information* (Freeman, New York 1982)
16.19 T. Pun: C. De. Garrini: Cybernétique et vision par ordinateur. In: *Le déficit visuel, de la neurophysiologie à la pratique de la réadaptation*, ed. by A.B. Safran, A. Assimacopoulos (Masson, Paris 2014) pp. 213–224
16.20 R. Bruyer: *Le Cerveau Qui Voit* (Editions Odile Jacob, Paris 2000)
16.21 A. Rebelo, I. Fujinaga, F. Paszkiewicz, A.R.S. Marcal, C. Guedes, J.S. Cardoso: Optical music recognition: State-of-the-art and open issues, Int. J. Multimed. Inf. Retr. **1**(3), 173–190 (2012)
16.22 K.M. Sayre: Machine recognition of handwritten words: A project report, Pattern Recognit. **5**, 213–228 (1973)
16.23 T. Plötz, G. Fink: Markov models for offline handwriting recognition: A survey, Int. J. Document Anal. Recognit. **12**, 269 (2009)
16.24 K.C. Ng, R.D. Boyle: Recognition and reconstruction of primitives in music scores, Image Vis. Comput. **14**(1), 39–46 (1996)
16.25 I. Fujinaga, J. Riley: Recommended best practices for digital image capture of musical scores. In: *3rd Int. Conf. Music Inf. Retr. (ISMIR)* (2002) pp. 261–263
16.26 W. Koseluk: Digitalization of musical sources: An overview. In: *The Virtual Score: Representation, Retrieval, Restoration, Computing in Musicology*, Vol. 12, ed. by W.B. Hewlett, E. Selfridge-Field (MIT Press, Cambridge 2001) pp. 219–226
16.27 D. Bainbridge, T. Bell: The challenge of optical music recognition, Comput. Humanit. **35**, 95–121 (2001)
16.28 E. Selfridge-Field: Optical recognition of musical notation: A survey of current work. In: *Computational Musicology: An International Directory of Applications*, Vol. 9, ed. by W.B. Hewlett, E. Selfridge-Field (1993) pp. 109–146
16.29 P. Martin, C. Bellissant: Low-level analysis of music drawings. In: 1st Int, Conf. Doc. Anal. Recognit., ICDAR **pp**, 417–425 (1991)
16.30 H. Fahmy, D. Blostein: A graph grammar programming style for recognition of music notation, Mach. Vis. Appl. **6**, 83–99 (1993)
16.31 D. Bainbridge, T. Bell: A music notation construction engine for optical music recognition, Softw. Pract. Exp. **33**(2), 173–200 (2003)
16.32 K. MacMillan, M. Droettboom, I. Fujinaga: Gamera: A structured document recognition application development environment. In: *2nd Int. Symp. Music Inf. Retr. ISMIR* (2001) pp. 173–178
16.33 K.C. Ng: Music manuscript tracing. In: *4th Int. Workshop, Graphics Recognit.: Algorithms and Applications (GREC)* (2001) pp. 322–334
16.34 J. Burgoyne, L. Pugin, G. Eustace, I. Fujinaga: A comparative survey of image binarisation algorithms for optical recognition on degraded musical sources. In: *8th Int. Conf. Music Inf. Retr. (ISMIR)* (2007) pp. 509–512
16.35 C. Dalitz, M. Droettboom, B. Pranzas, I. Fujinaga: A comparative study of staff removal algorithms, IEEE Trans. Pattern Anal. Mach. Intell. **30**(5), 753–766 (2008)
16.36 H. Miyao: Stave extraction for printed music scores. In: *3rd Int. Conf. Intell. Data Eng. Automated Learning (IDEAL)* (2002) pp. 562–568
16.37 F. Rossant: A global method for music symbol recognition in typeset music sheets, Pattern Recognit. Lett. **23**(10), 1129–1141 (2002)
16.38 I. Fujinaga: Exemplar-based learning in adaptive optical music recognition system. In: *Int. Comput. Music Conf* (1996) pp. 55–60
16.39 H. Kato, S. Inokuchi: A recognition system for printed piano music using musical knowledge and constraints. In: *Int. Assoc. Pattern Recognit. Workshop on Syntactic and Struct. Pattern Recognit* (1990) pp. 231–248
16.40 B. Coüasnon: Formalisation grammaticale de la connaissance a priori pour l'analyse de documents: Application aux partitions d'orchestre. In: *Actes du dixième congrès Reconnaissance des Formes et Intelligence Artificielle, Rennes* (1996) pp. 465–474
16.41 I. Knopke, D. Byrd: Towards musicdiff: A foundation for improved optical music recognition using multiple recognizers. In: *8th Int. Conf. Music Inf. Retr. (ISMIR)* (2007) pp. 123–126
16.42 E.P. Bugge, K.L. Juncher, B.S. Mathiesen, J.G. Simonsen: Using sequence alignment and voting to improve optical music recognition from multiple recognizers. In: *12th Int. Soc. Music Inf. Retr. Conf. (ISMIR)* (2011) pp. 405–410
16.43 M. Church, M.S. Cuthbert: Improving rhythmic transcriptions via probability models applied post-

OMR. In: *15th Int. Soc. Music Inf. Retr. Conf. (ISMIR)* (2014) pp. 643–648
16.44 H.E. Poole: Music printing. In: *Music Printing and Publishing*, ed. by D.W. Krummel, S. Sadie (Norton, New York 1990) pp. 3–78
16.45 R. Rasch (Ed.): *Music Publishing in Europe 1600–1900 Concepts and Issues, Bibliography* (Berliner Wissenschafts, Berlin 2005)
16.46 F. Rossant, I. Bloch: Robust and adaptive OMR system including Fuzzy modeling, Fusion of musical rules, and possible error detection, EURASIP J. Adv. Signal Process. **2007**, 81541 (2007)
16.47 L. Pugin, J.A. Burgoyne, I. Fujinaga: MAP adaptation to improve optical music recognition of early music documents using hidden Markov models. In: *8th Int. Conf. Music Inf. Retr. (ISMIR)* (2007) pp. 513–516
16.48 E. Selfridge-Field: *Beyond MIDI: The Handbook of Musical Codes* (MIT Press, Cambridge 1997)
16.49 Makemusic Inc.: musicXML, http://www.musicxml.com (2017)
16.50 WG_1599 – Working Group for XML Musical Application: 1599-2008 – IEEE Recommended Practice for Defining a Commonly Acceptable Musical Application Using XML, http://standards.ieee.org/findstds/standard/1599-2008.html (2017)
16.51 Music Encoding Initiative: http://www.music-encoding.org
16.52 A. Hankinson, P. Roland, I. Fujinaga: The music encoding initiative as a document-encoding framework. In: *12th Int. Soc. Music Inf. Retr. Conf. (ISMIR)* (2011) pp. 293–298
16.53 A. Hankinson, L. Pugin, I. Fujinaga: An interchange format for optical music recognition applications. In: *11th Conf. Int. Soc. Music Inf. Retr. (ISMIR)* (2010) pp. 51–56
16.54 T.M. Breuel, U. Kaiserslautern: The hOCR microformat for OCR workflow and results. In: *Int. Conf. Document Anal. Recognit. (ICDAR)* (2007) pp. 1063–1067
16.55 S. George: Evaluation in the visual perception of music. In: *Visual Perception of Music Notation: Online and Offline Recognition*, ed. by S. George (IRM, Hershey 2004) p. 308
16.56 M. Dawe: About Neuratron, http://www.neuratron.com (2015)
16.57 capella-software AG: Products, http://www.capella.de/us/index.cfm/products (2017)
16.58 Visiv Ltd: User comments, reviews, etc., http://www.visiv.co.uk/quote.htm (2006)
16.59 Visiv Ltd: Version History, http://www.visiv.co.uk/vershv2.htm (2006)
16.60 Graham Jones: http://www.indriid.com/grahamjones.html
16.61 Wikipedia: Audiveris, https://en.wikipedia.org/wiki/Audiveris (2017)
16.62 Laurent Pugin: Aruspix, http://www.aruspix.net
16.63 Christoph Dalitz: GAMERA Project, http://gamera.informatik.hsnr.de
16.64 G. Vigliensoni, J.A. Burgoyne, A. Hankinson, I. Fujinaga: Automatic pitch detection in printed square notation. In: *Proc. Int. Soc. Music Inf. Retr. Conf., Miami* (2011) pp. 423–428
16.65 L. Pugin, J. Hockman, J.A. Burgoyne, I. Fujinaga: Gamera versus Aruspix: Two optical music recognition approaches. In: *9th Int. Conf. Music Inf. Retr. (ISMIR)* (2008) pp. 419–424
16.66 J. Cardoso, A. Capela, A. Rebelo, C. Guedes: A connected path approach for staff detection on a music score. In: *Proc. 15th IEEE Int. Conf. Image Process* (2008) pp. 1005–1008
16.67 A. Dutta, U. Pal, A. Fornés, J. Lladós: An Efficient Staff Removal Approach from Printed Musical Documents. In: *Proc. 2010 20th Int. Conf. Pattern Recognit* (2010) pp. 1965–1968
16.68 A. Fornés, V.C. Kieu, M. Visani, N. Journet, A. Dutta: The ICDAR/GREC 2013 Music Scores Competition: Staff removal, Lect. Notes Comput. Sci. **8746**, 207–220 (2014)
16.69 Laurent Pugin: Verovio, http://www.verovio.org
16.70 L. Pugin, R. Zitellini, P. Roland: Verovio: A library for engraving MEI music notation into SVG. In: *15th Int. Conf. Music Inf. Retr. (ISMIR)* (2014) pp. 107–112
16.71 McGill University: http://ddmal.github.io/diva.js (2016)
16.72 A. Hankinson, W. Liu, L. Pugin, I. Fujinaga: Diva: A web-based document image viewer. In: *Conf. Theory Prac. Digital Libraries* (2011)