AUTOMATIC HANDWRITTEN MENSURAL NOTATION INTERPRETER: FROM MANUSCRIPT TO MIDI PERFORMANCE

Yu-Hui Huang[◇⋆], Xuanli Chen[◇⋆], Serafina Beck[†], David Burn[†], and Luc Van Gool[◇∓]

[◇]ESAT-PSI, iMinds, KU Leuven [†]Department of Musicology, KU Leuven [∓]D-ITET, ETH Zürich

{yu-hui.huang, xuanli.chen, luc.vangool}@esat.kuleuven.be, {serafina.beck, david.burn}@art.kuleuven.be

[⋆]equal contribution

ABSTRACT

This paper presents a novel automatic recognition framework for hand-written mensural music. It takes a scanned manuscript as input and yields as output modern music scores. Compared to the previous mensural Optical Music Recognition (OMR) systems, ours shows not only promising performance in music recognition, but also works as a complete pipeline which integrates both recognition and transcription.

There are three main parts in this pipeline: i) region-of-interest detection, ii) music symbol detection and classification, and iii) transcription to modern music. In addition to the output in modern notation, our system can generate a MIDI file as well. It provides an easy platform for the musicologists to analyze old manuscripts. Moreover, it renders these valuable cultural heritage resources available to non-specialists as well, as they can now access such ancient music in a better understandable form.

1. INTRODUCTION

Cultural heritage has become an important issue nowadays. In the recent decades, old manuscripts and books have been digitalized around the world. As more and more libraries are carrying out digitalization projects, the number of manuscripts increases exponentially every day. The texts in these manuscripts can be further processed using Optical Character Recognition (OCR) techniques while the music notes can be processed by Optical Music Recognition (OMR) techniques. However, due to the nature of the manuscript, the challenges of OMR and OCR have to be addressed differently. For example, OMR has to deal with different types of notations from different time periods, such as Chant notation used throughout the medieval and the Renaissance periods while white mensural notation used during the Renaissance. Even within the same period, music symbols vary in different geographical areas [13]. In addition to the semantic characteristic, OMR has the additional problem as OCR of having to cope with the physical condition of historical documents [15].

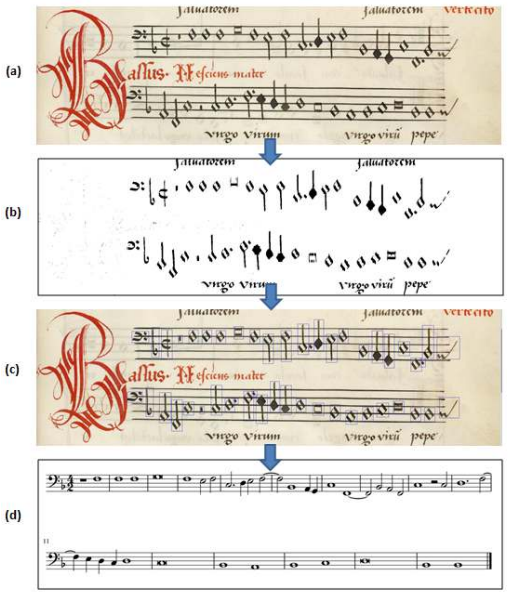

Figure 1: The overview of our framework. (a) Original image after ROI selection. (b) After preprocessing. (c) Symbol segmentation. (d) Transcription results.

While several OMR systems exist for ancient music scores in white mensural notation, most of them target at printed scores. To name a few, Aruspix [3] is an open source OMR software targeting those ancient printed scores; Pugin et al. utilized the Hidden Markov Models to recognize the music symbols and to incorporate the pitch information simultaneously. A comparative study made by Pugin et al. [13] shows that Aruspix has better performance on selected printed books than Gamut [11], which is another OMR software based on the Gamera [9] open-source document analysis framework. Gamut first segments the symbols based on the result after staff lines removal, and classifies it using kNN classifier.

Calvo-Zaragoza et al. [5] proposed an OMR system without removing the staff lines. They utilized histogram analysis to segment the staves as well as different music symbols, and classified by cross-correlating templates. Their method achieves averagely an extraction rate of 96%

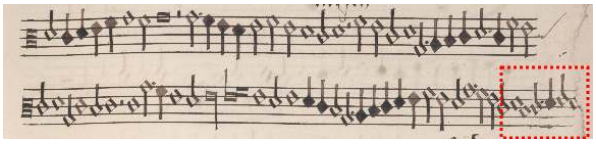

Figure 2: A regular case happens at the end of each voice: due to a lack of space, the writer extends the staff lines a little bit (red dashed box) and squeezes the remaining symbols on.

on the Archivo de la Catedral de Malaga collection which has a certain printing style.

In addition to the physical condition of the manuscripts, the substantial difference in style between writers renders OMR challenge. One and the same symbol can appear quite differently, depending on the writer. Moreover, the symbols sometimes are written too close to each other which increases the difficulty of symbol segmentation. This usually happens at the end of each voice as the writer wants to finish on the same line instead of adding a new one. In such cases, they usually elongate the staff lines manually in order to add more symbols, see e.g. Figure 2. Such cases increase the difficulty to apply OMR on these handwritten manuscripts in a systematic and consistent manner.

Similar to Gamut, we remove staff lines to detect the symbols, but differently, we employ the Fisher Vector [12] representation to describe images and Support Vector Machines (SVM) to classify them. With relatively less training data compared to others, our OMR system is able to recognize the symbols from different writers with high accuracy.

In contrast to the modern music (the music from the so-called Common practice period), the music notation up to the Renaissance is much different in appearance. Therefore, transcription from an expert is required to further process the data. Our goal therefore was to design and implement a system that automatically transcribes such music for users who lack the expert knowledge about these early manuscripts. In particular, our system is able to automatically transcribe most of contents in mensural music pieces as shown in Figure 1. We propose a new OMR system which not only recognizes the handwritten music scores but also transcribes it from white mensural notation to the modern notation. The modern notation is then encoded into MIDI files. The overall pipeline is described in Figure 3. In addition to provide a user friendly platform for the musicologists to analyze the music from old manuscripts, our system renders these valuable cultural heritage resources to non-specialists as well. Compared to most OMR system, the playable MIDI files in our system help people without any music knowledge access those ancient music.

The remaining of this paper is structured as the followings. Section 2 describes the image preprocessing steps. In Section 3, we introduce the core part of the OMR system, music symbol recognition. The transcription to modern notation is explained in Section 4. Experimental settings

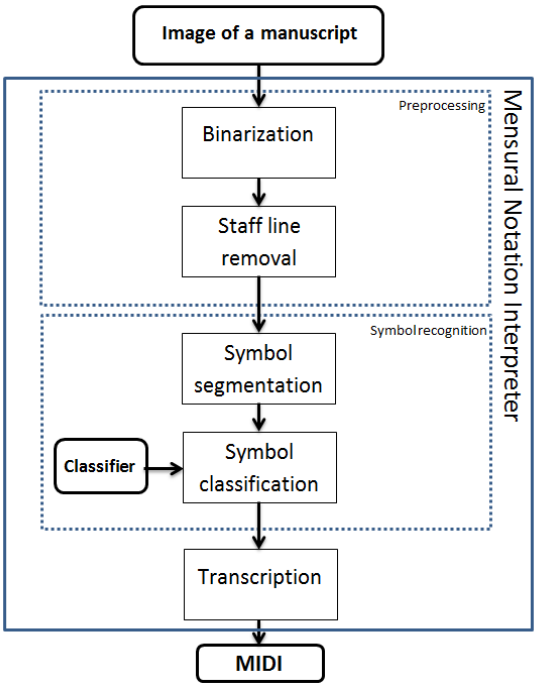

Figure 3: The overall scheme of our framework.

and results are shown in Section 5. Section 6 concludes the paper.

2. PREPROCESSING

Following typical OMR pipelines, we start from a preprocessing step. In consists of two parts, namely binarization and stave detection.

2.1 Binarization

In some collections of manuscripts, each scanned image comes up with a color check and a ruler aside the main manuscript. In order to achieve a good quality, the non-music parts need to be removed during the binarization. Given a high resolution scanned image of a music manuscript, the boundaries of the page are first detected by histogram analysis of pixel intensity in gray-scaled image. Thresholds are set to the horizontal and vertical histograms to segment the x- or y-axis into two parts: the page part containing the staves and the background parts together with the color check and the ruler. Because the Region of Interest (ROI) refers to the page part here, which contains much higher intensity of grey values compared to the black background. Based on this fact, with properly chosen threshold, ROI could be well selected. The result is shown in Figure 1a.

For those manuscripts containing colored initials or decorations, we apply K-means clustering under the Lab color space in order to filter out some colored non-music elements after cropping out the color check and the ruler. In the experiments we put K to the value 2, and successfully cluster the manuscript into two groups, the elements

with red color and the others containing the stave. We select the red group to build a mask to remove those non-music regions from the manuscript. After that, we then apply Otsu threshold to do the binarization. For simplicity, we will focus on a specific style, generating to other styles of manuscripts will be considered in the future work. Figure 1b shows an example result after these preprocessing steps applied.

2.2 Stave detection and staff lines removal

The stave in mensural notation are mostly composed of five lines. Based on this assumption, we use the stave detection program from [16]. Timofte et al. utilized dynamic programming to retrieve the patterns of five lines in order to detect the stave. While detecting the staff lines, the parameters of staff line thickness and space between two staff lines are optimized at the same time. Figure 1b shows the result after staff removal.

3. SEGMENTATION AND CLASSIFICATION

With the preprocessing steps of the previous section having been completed, we obtain binarized images without staff lines. In this section, we first describe how the symbols are segmented and then how the classification of the individual, segmented symbols works.

3.1 Segmentation

Given a binarized image without staff lines (Figure 4a), we employ the connected component analysis to separate different symbols. However, the symbols touching the staff line in the original manuscript may become separate after staff removal. As the Figure 4b shows, a semibreve or a minim may be separated into two parts.To solve this problem, we set up several heuristic rules to combine the parts of such broken symbols. For example, we observe that some overlapping or close neighbouring boxes detected with similar width could be merged into one individual symbol. Therefore we merge neighbouring boxes in this case. Yet, this procedure might be risky in that two close parts coming from different symbols may get erroneously merged as well. To tackle that, we set up a width threshold for merging boxes, i.e. if the box width is more than two times of the space between two staff lines, the two boxes will not be merged. The final result is shown in Figure 4c.

Moreover, in order to distinguish the lyrics from the music symbols, we use the stave region detected from the previous section as a mask to filter out those non-music symbols.

3.2 Classification

In order to train the classifier, we manually annotate the image of each music symbol by drawing the bounding box around it using the image annotation tool [4] from the original manuscript. Because the bounding boxes may differ from each other in size, for each cropped symbol I, we first normalize the image to the same height, which is determined by ensuring enough SIFT [10] features could be extracted to form the Fisher Vector representation [12].

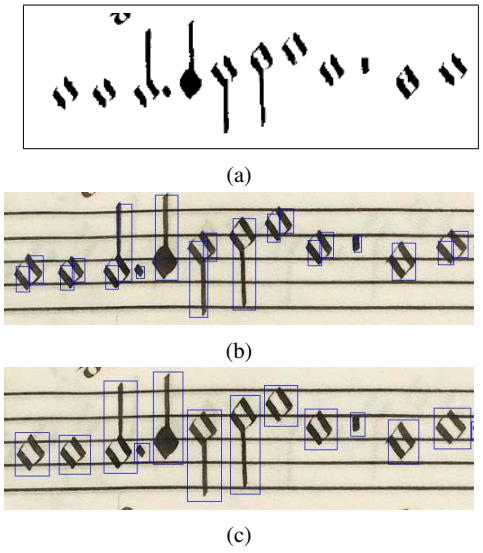

Figure 4: (a) After staff removal, some symbols become separated because some strokes touching the staff line are removed as well. (b) The result of applying connected component analysis on Figure 4a. (c) The result after applying heuristic rules to combine the broken symbols from Figure 4b.

For training, we use a Gaussian Mixture Model (GMM) with $K = 128$ components as the generative model for the patch descriptors. To estimate the parameters of the GMM, we obtain 10000 sample descriptors by applying dense SIFT in all the images from the training data, and reduce them from $D = 128$-d to $D = 64$-d using PCA. Then, the mean, variance and weight of each Gaussian component are estimated using the Expectation Maximization algorithm. The final fisher vector for an image I has dimension of $2KD = 16384$. This vector is then signed-squared-rooted and l_2 normalized. We follow the procedures in [7] to obtain the improved fisher vector, that is after pooling all the vectors of the training data, we apply the square rooting again to normalize. With bunch of vectors from training data, we train the classifier using linear SVM. We train the multi-class classifier with one versus one strategy and select the class with the highest probability.

During testing, we already obtained the bounding box information of each music symbol following the previous steps. To avoid the effect caused by binarization and staff removal, we extract the symbol again directly from the original colored image using the same coordinates given by the bounding box. Hereby, we would like to remind that the preprocessing steps are for symbol segmentation, while both training and testing patches are extracted from the original manuscript. Each segmented symbol is described in Fisher Vector representation in a similar way as we did for the training data. Then we use the trained multi-

class classifier to predict its class.

In our implementation, we used the VLFeat library [17] for the Fisher Vector and SIFT implementations, and libsvm [6] with linear kernel and default settings for the Support Vector Machine.

3.3 Pitch detection and channel separation

The pitch information is essential for transcription, and the pitch level is determined by the relative position of the note and the clef to the stave. After the music symbol is extracted and classified, we follow the post-processing steps described in [14] to retrieve the pitch level.

We divide the group of notes into two groups according to their stems. For the group of notes with stems, we perform histogram analysis to extract the y position of the stem, so that this can be used to localize the center of the note head. The detail of the histogram analysis is as following: we first project all the horizontal pixels onto the y-axis and then set up a threshold to separate the stem and the note head, by employing the fact that note head part has higher intensity of pixels than the stem. For the group of none-stem notes, we simply compute the middle point, departing from the highest and the lowest points of the symbol.

For clefs, the point of relevance is much easier to locate, since they can only be situated on staff lines. We simply determine the middle point of two squares from clef c and of the two dots or blobs from the right part of clef f, while we locate the center of the blob for clef g. For key signatures, we only encounter the case of flat, as the sharp is rare in the dataset we use. We adopt a similar strategy to that of the notes to locate the center of the blob for the flat symbol. With the relative position of the extracted symbol calculated, we connect this information to the staff line position in order to determine the pitch level of the corresponding symbol.

In the case of choirbooks, there are always several voices within one page of a manuscript in our dataset. Thus, in order to transcribe the music correctly, we need to recognize these different voices. As each voice ends with barlines, we use this as a criterion to separate different voices. After a barline is detected, we switch the notes detected afterwards to another channel.

4. TRANSCRIPTION

We aim at transcribing mensural music scores into modern notations. This will render the music accessible to a far larger group of people, also because much of this music has not even been published. The tool is also valuable for musicologists, because it takes over the time consuming manual transcription work. Instead, they can spend their time on the actual music analysis. With the vast digital manuscript collections of libraries that are being made available daily, the transcription tool makes it a lot easier to establish concordances. Also, printed and often not published transcriptions are sometimes hard to get by, so this tool means generally a big improvement of accessibility of transcriptions. Moreover, with all the available software libraries nowadays, such as *music21* [8] which is also used in our work, MIDI files could be generated directly from modern notation scores. Therefore the mensural script could be more easily accessed by general public.

4.1 Transcription rules

There are several difficulties in transcribing mensural music. Apart from notational challenges like ligatures and coloration, which are not supported yet, the main challenge of mensural music transcription is how to translate the mensuration, or time signature. In contrast to modern music, the time signature defines not only how long one measure is, but also defines how to divide a certain note.

There are four kinds of notes that can be divided in different ways. The division of maxima into longa is called *modus maximarum* or *modus maior*. From longa into breves, it is called *modus*. From breves into semibreves, it is called *tempus*. And from semibreves into minims, it is called *prolatio*. For all of these four divisions, depending on whether they are *perfect* or not, either a *ternary* or *binary* division is possible. If a note is divided in a *perfect*, i.e. *ternary* way, it will be divided into three sub-class notes. If one note however is divided in an *imperfect*, i.e. *binary* way, it will be divided into two sub-class notes. For example, in a case of *perfectum*, a longa will be divided into three breves, while in a case of *imperfectum*, the modus specifies that one longa has to consist of two breves. This rule also applies to the other three transcription pairs.

The temporal length of one breve in mensural music defines the length of one measure in modern music. In the normal case (i.e. without scaling of temporal length), the length of a semiminim equals that of a modern quarter note. Because a semiminim cannot be affected by the rules of *perfect / imperfect* for its division into its sub-class fusa (i.e. a quaver in modern notation), we are able to calculate the actual length of a breve, by treating semiminims as a unit. For instance, if *tempus* and *prolatio* (which refer to the semibreve-minim division and breve-semibreve, respectively), are both *perfect*, then one breve will be divided into three semibreves, and each semibreve will in turn be divided into three minims. As a result, one breve is divided into nine semiminims. If we treat one semiminim as a beat, then the corresponding time signature would be 9/4.

In addition, there is a variant version of mensuration symbols, these are the time signatures with a vertical line through the original symbol, usually called *cut-signs*. They imply a reduction of all the temporal values, of notes and rests, by a factor of two. In other words, with *cut-sign*, the playing speed of the music will be twice as faster. Note that most mensural music is rather slow compared to contemporary music. Beside the *cut-signs*, we also provide a parameter to artificially scale the speed of playing. In order to achieve that, we only need to change the mapping relationship between the mensural notes and the modern ones. For instance, in no-scaling cases, a semiminim is mapped to a quarter. If we speed up the music by two times, we just need to map semiminim to a quaver, which is half the

length of a quarter. In this case, one should adapt the time signature accordingly.

4.2 Implementation details

Given the aforementioned observations,the analysed mensural music can be encoded into modern music. In our pipeline, we first check the mensuration of the music piece. Taking into consideration that the mensuration might change at any time during the piece, this step should be repeated any time during the process. If there have been any changes, we apply the reduction ratio to the music afterwards. After this, we can determine the mapping relation between the semiminim and modern music notes and calculate the modern time signature according to the duration of one breve in the transcription. With determined time signatures and basic mapping relationships established, we can transcribe each element into modern musical notation, note by note and rest by rest. If the division is only binary or imperfect, we can directly transcribe the mensural music to modern music. We are still working on the transcription techniques for the perfect divisions, which include a lot more exceptions that can still present challenges. Once musical symbol recognition are ready, all we need to do is to carefully encode these symbols. One should be especially aware of the possibility that clefs and/or time signatures change in the middle of a piece. For this step, we chose the framework offered by *music21* [8] to encode the music information, because it offers an automatic parsing library and APIs towards visualization and MIDI output. The different voices in the original music sheet are encoded into different 'part objects' in this framework, while the whole piece is treated as a 'stream' object. Another thing that needs to be taken care of is the *punctus divisionis* sometimes appearing in a *perfect* division, which looks exactly like a normal dot with the function of prolonging note values, but instead of prolonging, the *punctus divisionis* functions as a kind of barline. Whether or not we are dealing with this kind of dot, should be established from the note durations directly preceding and following it.

5. EXPERIMENTAL RESULTS

5.1 Dataset and evaluation

We evaluate our pipeline on the Alamire collection which includes manuscripts of various writers in several books. Depending on the sources, those manuscripts are in high resolution from 7200x5400 to 10500x7400 pixels. For training, we randomly select the manuscripts from the following books: *Vienna, Österreichische Nationalbibliothek (VienNB), MS Mus. 15495, 15497, 15941, 18746; Brussels, Koninklijke Bibliotheek (BrusBR) Ms. 228, and IV.922* [1]. We use the image annotation tool made by Kläser [4] to manually draw the bounding box around each symbol and to annotate the corresponding information. In total we have about 2800 samples for training over 33 classes. The classes include the notes, rests, key signature (flat), most of the frequent time signatures and other symbols such as barlines and custos. The testing data comes from different books, without any overlap with the training data: *Munich, Bayerische Staatsbibliothek, Mus. MS. F (MunBS F) [2]; London, British Library MS Royal 8G.vii (LonBLR), and 's-Hertogenbosch, Archief van de Illustre Lieve Vrouwe Broederschap, MS. 72A (MS 72A)* [1]. In total, there are about 3700 samples for testing. In our evaluation, we report the result of classification and segmentation separately.

Book	MunBS F	LonBLR	MS 72A
N	839	1313	1636
R_{ext}	85.73%	94.36%	90.25%

Table 1: Symbol extraction result on three books.

5.2 Symbol segmentation

We follow the evaluation process in [5]. The extraction rate is defined as $R_{ext} = \frac{M_e}{T}$, where M_e is the number of music symbols extracted and T is the total number of music symbols within the manuscript. Table 1 shows the symbol segmentation results on three collections where N is the total number of symbols per book. Most false negatives of detection come from custodies, as they are often over segmented into several parts after staff removal. Some of the other false negatives come from the symbols on the sixth staff line, below or above the stave, causing the symbols above or below the stave not correctly extracted. Moreover, the ornate capitals in front of the piece may distract the detection especially on the MunBS F collection. Unlike the colored initials in LonBLR, the black initial makes the separation of symbols more difficult. These issues are being solved and will be addressed in the future work.

5.3 Symbol classification

To evaluate the classification step, we first correct the segmentation errors from the last step as Figure 1c shows, and then use prediction accuracy to evaluate the classification. Table 2 presents the classification result on the same collections. The accuracy reaches 98 % on the LonBLR and the MS 72A collections, and 95 % on the MunBS F collection. After analysis, we found the typical error for the MS 72A collection is the misclassification of a breve rest as a colored breve. In MunBS F, most of the classification errors are from the semibreve notes which are mistakenly classified as points. Some incidents are caused by similar symbols, such as the note fusa recognized as semiminim and the note maxima classified as longa. The reason might be found in the imbalanced training samples in our training set. As some symbols do not happen appear so often such as the note maxima and time signatures, they are less present in the set. It makes the training collection more challenging if one wants to avoid this issue.

With limited training data, the use of the Fisher Vectors and SVMs yields a promising classification perfor-

Book	MunBS F	LonBLR	MS 72A
Accuracy	95.52%	98.83%	98.94%

Table 2: Classification result on three books.

mance on handwritten symbols from different writers. As the manually annotated training data is hard to obtain, our method shows an obvious advantage compared to earlier alternatives.

6. CONCLUSION

In this paper, we presented a framework to automatically analyse and transcribe handwritten mensural music manuscripts. The inclusion of the transcription part not only provides the musicologists with a simple platform to more efficiently study those manuscripts, but also assists music amateurs to explore and enjoy this ancient music. Moreover, the MIDI-output feature offers the public at large easy and convenient access to these musical treasures.

We have collected a dataset of handwritten mensural notation symbols from different books for evaluation. We believe it is fair to claim that our symbol segmentation attains good performance. The classification based on the Fisher Vector representation and SVMs achieves very high classification rate on handwritten symbols. Furthermore, we implemented an accurate transcription mechanism which embeds musicological information.

We plan to extend this work by enabling counterpoint checking so that mistakes in original music manuscripts can be pointed out to the musicologists easily. In addition, we intend to implement scribe identification in our system (an early module for that is ready) to assist authorship identification.

7. ACKNOWLEDGMENTS

We are grateful to Alamire Foundation for their support and we would like to thank Lieselotte Bijnens, Lonne Maris, Karen Schets and Tim Van Thuyne for their help on symbol annotations. The work is funded by the Flemish IWT/SBO project: New Perspectives on Polyphony, Alamire's musical legacy through high-technology research tools.

8. REFERENCES

[1] IDEM. http://elise.arts.kuleuven.be/alamire/. Accessed: 2015-04-23.

[2] Munich, Bayerische Staatsbibliothek, Handschriften-Inkunabelsammlung, Musica MS F. http://www.digitale-sammlungen.de/. Accessed: 2015-04-23.

[3] Aruspix project. http://www.aruspix.net/, 2008. Accessed: 2015-04-23.

[4] Image annotation tool with bounding boxes. http://lear.inrialpes.fr/people/klaeser/software, 2010. Accessed: 2015-04-23.

[5] J. Calvo-Zaragoza, I. Barbancho, L. J. Tardn, and A. M. Barbancho. Avoiding staff removal stage in optical music recognition: application to scores written in white mensural notation. *Pattern Analysis and Applications*, pages 1433–7541, 2014.

[6] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[7] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *British Machine Vision Conference*, 2011.

[8] M. S. Cuthbert and C. Ariza. music21: A toolkit for computer-aided musicology andsymbolic music data. In *Proceedings of the ISMIR 2010 Conference*, 2010.

[9] M. Droettboom, G. S. Chouhury, and T. Anderson. Using the gamera framework for the recognition of cultural heritage materials. In *Joint Conference on Digital Libraries : Association for Computing Machinery*, 2002.

[10] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2004.

[11] K. MacMillan, M. Droettboom, and I. Fujinaga. Gamera: Optical music recognition in a new shell. In *Proceedings of the International Computer Music Conference*, 2002.

[12] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *Proceedings of the 11th European Conference on Computer Vision*, 2010.

[13] L. Pugin and T. Crawford. Evaluating omr on the early music online collection. In *Proceedings of the ISMIR 2013 Conference*, 2013.

[14] L. Pugin, J. Hockman, J. A. Burgoyne, and I. Fujinaga. Gamera versus aruspix – two optical music recognition approaches. In *Proceedings of the ISMIR 2008 Conference*, 2008.

[15] C. Ramirez and J. Ohya. Symbol classfication approach for omr of square notation manuscripts. In *Proceedings of the ISMIR 2010 Conference*, 2010.

[16] R. Timofte and L. Van Gool. Automatic stave discovery for musical facsimiles. In *Asian Conference on Computer Vision*, 2012.

[17] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.