MuRET as a software for the transcription of historical archives

José M. Iñesta, David Rizo, and Jorge Calvo Zaragoza
Dpto. de Lenguajes y Sistemas Informáticos
Universidad de Alicante (Spain)
email: inesta@dlsi.ua.es

***Abstract*—The transcription process from historical handwritten music manuscripts to a structured digital encoding has been traditionally performed following a fully manual workflow. At most, it has received some technological support in particular stages, like optical music recognition (OMR) of the source images, or transcription to modern notation with music edition applications. Currently, there is no mature and stable enough solution for the OMR problem, and the most used music editors do not support early notations, such as the mensural notation. A new tool called MUsic Recognition, Encoding, and Transcription (MuRET) has been developed, which covers all transcription phases, from the manuscript image to the encoded digital content. MuRET is designed as a machine-learning based research tool, allowing different processing approaches to be used, and producing both the expected transcribed contents in standard encodings and data for the study of the transcription process.**

***Keywords*-Optical music recognition, Early music notation, Handwritten manuscripts, Software tools.**

I. Introduction

Optical music recognition (OMR) systems represent key tools for the publication of music score collections that are currently found only on paper. The main goal of an OMR system is to convert music score images to a digital structured format, like XML-based ones [4], [6]. Under the perspective of cultural heritage preservation, many of these collections are handwritten material, and some of them can be found in early notation systems.

In particular, Spanish white mensural notation was the dominant code for writing music in Spain during the 16th to 18th centuries, producing large collections of documents yet to be made accessible to the public. In all these cases, just the scanned or photographed images of the scores are accessible. The OMR techniques available today are far from ready to be applied to this kind of handwritten documents [1].

Existing OMR systems such as Audiveris[1] or Photoscore[2] are mainly devised to extract the musical content from printed or manuscript sheet music in order to edit them further in music edition applications. Aruspix [10] goes a step further and allows the superimposition and the compilation of early music prints. However, none of them is devised for recognizing the content of printed or manuscript scores from different notations, allowing the manual introduction of new materials, the encoding into all current standards, and the assisted transcription into an edited version ready for preparing a critical edition.

In this context, the Hispamus project [7] has been proposed with two main objectives:

- From a scientific point of view: to carry out research on pattern recognition and machine learning applied to Optical Music Recognition.
- From a more operational point of view: to provide tools to editors for editing handwritten music collections, specifically with the goal of accessing collections of early Spanish music.

A software system named MuRET (*MUsic Recognition, Encoding, and Transcription*) is under development in the Hispamus project, intended to produce both a transcribed copy of the original (*diplomatic* version) and the material for rendering a *critical* edition, where possible mistakes, inkblots, or even missing paper parts that may affect the information in the original work can be compensated.

An important design feature of MuRET is the possibility to also render a translation to modern notation when the original works are written in early notations, like handwritten or printed mensural. This way, the work can be readable by a contemporary musician, also allowing the public to enjoy and search into the digital contents of these works, either as a musicologist or as a performer.

Finally, it has to be taken into account that the MuRET software is conceived not as a publishing software, but as a tool for research, where one can plug-in algorithms, and test and compare them by using proper metrics. No batch or massive document processing are implemented.

II. Methodology

As mentioned above, one of the main goals of the developed software is to process historical documents, covering the workflow from the digitized images to the production of the structured symbolic format (Fig. 1). It is important to point out that the digitization itself is not an objective of the project, so we assume that others have scanned or photographed the documents and we have the images already. On the other end of the pipeline, the software is not oriented to the final publication, so a structured XML-based format is output by the system that can be the input of the final rendering software for publication if needed.

[1]https://github.com/Audiveris
[2]https://www.neuratron.com/photoscore.htm

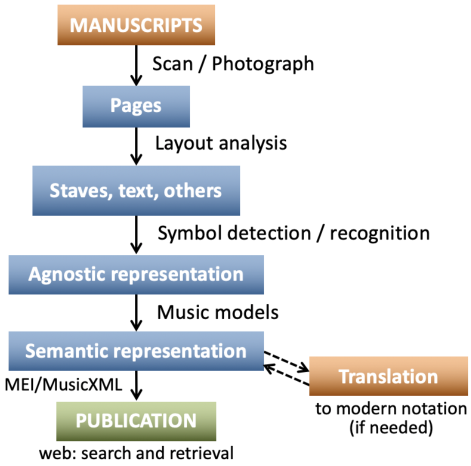


Figure 1. The workflow pipeline in MuRET. The first (manuscript scanning) and last (score publication) stages are out of the scope of this software.

From a technical point of view, it is important to note that all the recognition and processing stages of the system are designed using Machine Learning (ML) techniques. Most available OMR systems are based on heuristic methods, but one of their problems is that they are based on too many repertoire-dependent context rules [1], [11]. Therefore, it is hard to extend them to recognize early music notations, that are sometimes very different from the modern one.

By applying innovative ML technologies we can build models based on *training pairs*: sets of images that are presented together with ground-truth labels. This is a key point of this approach, because it opens up the possibility of adapting the system, in principle, to any music notation, provided that labelled data is available for the system to learn from. In fact, the MuRET system is being developed with labeled data from two different notations of handwritten music scores: vocal music manuscripts written in Spanish mensural notation and Spanish traditional songs manually transcribed in modern music notation by ethno-musicologists.

MuRET has been designed to support the whole process of transcription of a manuscript source keeping the traceability of each step. The main stages shown in Figure 1 will be presented next, pointing out the main technical features.

A. Layout analysis

The system input is a set of digitized image files of the manuscripts. Each page is processed independently, although the page order may affect the process of adapting the recognition process to the collection of documents.

In the first stage, each page is segmented into meaningful regions by using a layout analysis system. For that, a number of collections have been labeled manually with their ground-truth regions, hierarchically classified in texts regions: title, author, lyrics, multi-lyrics (when more than one lyric line is sung for the same staff), staves: staff or empty staff, and others, that can include defects in the paper or drawings, for example (see Fig. 2).

Figure 2. Examples of different ground-truth annotated regions in an image, like title, staves and lyrics.

The ground-truth areas have been used to train a region-oriented detection network (in particular, a RetinaNet [8], usually employed for object detection). This system has been used for detecting these kinds of regions in new images, obtaining precisions around 99% for staff detection and 80% for lyrics in preliminary experiments.

Lyrics are not subjected to further processing yet, but the staff regions are the different areas to apply the next stages of MuRET.

B. Symbol detection and recognition

After testing a number of approaches, at this point, the MuRET system performs OMR in an end-to-end (*holistic*) fashion, staff by staff, and different methods have been implemented for error corrections.

Holistic staff recognition: in the first stage of recognition, each staff is processed in just one step (or *end to end*), that is, without any previous symbol segmentation or staff removal procedure (see Fig. 3). This can be achieved by using recurrent convolutional neural networks (see [3]) that output a sequence of probabilities for each symbol that is then processed with a connectionist temporal classification method (or CTC optimization [5]) yielding the sequence of symbols that maximizes the probability.

The advantage of this approach is that the ground-truth data consists simply of pairs of staff images and their corresponding ground-truth sequence of music symbols. No need for segmentation or preprocessing of the image is needed because the model is able to figure out what is in

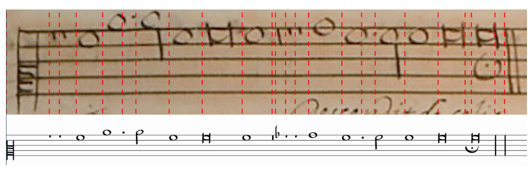


Figure 3. Staff-based end-to-end approach. The model receives an entire staff region and yields a sequence of symbols, including the approximate positions where they have been detected in the staff (red dashed lines).

the staff if enough training data was provided. Therefore, the need of well-annotated data is of extreme importance for this algorithm to perform properly. On the other hand, it provides a way to learn virtually any kind of notations if enough data is available, at least in the context of monophonic scores, like those studied in this project.

User-driven symbol recognition: in this case, the user locates the symbols manually. The use of a digital pen [2] results in a more ergonomic interaction (see Fig. 4), especially for making corrections to system errors. The user can either trace with the e-pen the shape of a symbol or draw a rectangle around the target symbol. In both cases, this procedure yields a bounding box for each symbol that is received by a classifier as input. At the current state, the classifier is a convolutional neural network (CNN) that, from a 40×40 resize of the bounding box, predicts the class of the symbol therein.

Another CNN uses an extension of the bounding box, from the bottom to the top of the staff area, to estimate the vertical position of the symbol in the staff.

The advantage of this approach with respect to the end-to-end method is that the classifiers need less ground-truth data to obtain good results and it provides a comfortable and intuitive way for the user to make corrections to the end-to-end output.

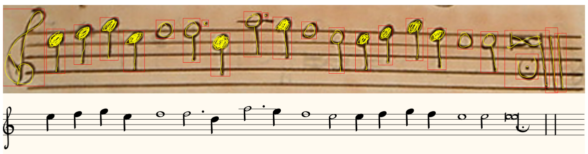


Figure 4. User-driven symbol recognition. Classifiers deal with manually located symbol bounding boxes. In this figure, the bounding boxes and traces for all the symbols in the staff are displayed, as an example.

C. Agnostic vs semantic encoding

A novel approach to the OMR operations is to generate the output sequence of symbols in two stages. The first sequence is named *agnostic* [12], which contains symbols characterized by their graphic value and position in the staff. This means that their role in the score is not considered. For example, a $\sharp$ symbol is tagged as *sharp*, regardless of whether it locally modifies a note or is part of a key signature.

The second encoding is the *semantic* one, where meaningful musical information is encoded in any standard music format. In this case, a group of two adjacent sharp symbols at the beginning of a staff is given the meaning of a D Major key signature. This also means that, for example, a note in the fifth line (coded this way in the agnostic representation) is converted into a F$\sharp$ if the G clef and the D major signature were detected at the beginning of the staff.

At this moment, the conversion from the agnostic encoding to the semantic one is implemented using a grammar-driven approach that models a set of hand-crafted conversion rules. However, this strategy is not scalable in terms of the different notations and situations that may appear. Therefore, we are studying how to learn this conversion process by using pairs of agnostic and semantic sequences, approaching it as a machine translation problem.

The advantages of dividing the recognition process into these two phases are twofold: first, removing the musical knowledge needed to correctly understand the meaning of the score permits to turn the OMR task into a standard artificial vision problem. The objects in the image are recognized just by their shapes and positions in it. Second, an error in the recognition of a single symbol can cause a chain of other errors in pitch and rhythm. If all the symbols are just shapes without other meaning than its name, the errors that may happen can be corrected by the expert before converting the correct output to the semantic encoding.

D. Output representation

Once the system has produced a semantic output of the transcribed score, it seems reasonable to code it in a structured format using a standard language. The choice we have made for that is the use of MEI (Music Encoding Initiative [6]). This way, we can use different tools to convert the output to a printed score by using Verovio[3] [9] (see Fig. 5) or to other formats like MusicXML (only for modern notation), MIDI, Humdrum, PDF, etc.

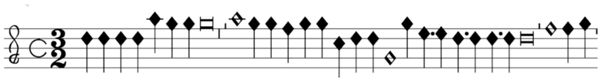


Figure 5. Conversion from agnostic to semantic representation. After the symbols have been assigned their semantic meaning, the staff can be rendered using Verovio.

The MEI encoding from the semantic representation is straightforward since the tokens represented in both languages are the same. So one-to-one translation, plus the required language formats, were only needed.

III. Conclusion

The current goal of the works in the Hispamus project is not to generate final edited preprints, but to produce contents to be sent to online services or publishers. The MuRET system is being designed to export in any interchange format that will be eventually edited by the publishers to fulfill their publishing workflow.

The subsystems integrated into MuRET are based on state-of-the-art machine learning techniques that permit us to adapt the recognition and transcription capabilities of

[3] http://www.verovio.org

the system to virtually any notation or collection if enough annotated data are found.

The agnostic symbol recognition phase makes it easier to perform, but a translation into a semantic transcription of the score has to be made after that. Machine translation techniques are being studied to achieve that goal.

Acknowledgements

This work is supported by the Spanish Ministry HISPAMUS project TIN2017-86576-R, partially funded by the EU.

References

[1] Donald Byrd and Jakob Grue Simonsen. Towards a standard testbed for optical music recognition: Definitions, metrics, and page images. *Journal of New Music Research*, 44(3):169–195, 2015.

[2] Jorge Calvo-Zaragoza, David Rizo, and José Manuel Iñesta. Two (note) heads are better than one: Pen-based multimodal interaction with music scores. In *17th Int. Society for Music Information Retrieval Conference*, pages 509–514, 2016.

[3] Jorge Calvo-Zaragoza, Alejandro H. Toselli, and Enrique Vidal. Handwritten music recognition for mensural notation with convolutional recurrent neural networks. *Pattern Recognition Letters*, 128:115 – 121, 2019.

[4] Michael Good. MusicXML: An internet-friendly format for sheet music. In *XML Conference and Expo*, 2001.

[5] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 369–376, New York, NY, USA, 2006. ACM.

[6] Andrew Hankinson, Perry Roland, and Ichiro Fujinaga. The music encoding initiative as a document-encoding framework. In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, 2011.

[7] Jose M. Iñesta, Pedro J. Ponce de León, David Rizo, José Oncina, Luisa Micó, Juan Ramóon Rico-Juan, Carlos Pérez-Sancho, and Antonio Pertusa. HISPAMUS: Handwritten spanish music heritage preservation by automatic transcription. In *Proceedings of the 1st International Workshop on Reading Music Systems*, 2018.

[8] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. *arXiv e-prints*, page arXiv:1708.02002, Aug 2017.

[9] L. Pugin, R. Zitellini, and P. Roland. Verovio—a library for engraving MEI music notation into SVG. In *Proceedings of the 15th International Conference for Music Information Retrieval (ISMIR 2014)*, October 2014.

[10] Laurent Pugin. Editing Renaissance Music: The Aruspix Project. *Internationales Jahrbuch für Editionswissenschaften*, pages 94–103, 2009.

[11] A. Rebelo, I. Fujinaga, F. Paszkiewicz, A.R.S. Marçal, C. Guedes, and J. S. Cardoso. Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval*, 1(3):173–190, 2012.

[12] D. Rizo, J. Calvo-Zaragoza, J.M. Iñesta, and I. Fujinaga. About agnostic representation of musical documents for optical music recognition. In *Music Encoding Conference, Tours, 2017*, May 2017.

Incremental Supervised Staff Detection

Alexander Pacha
Institute of Information Systems Engineering
TU Wien, Austria
alexander.pacha@tuwien.ac.at

***Abstract*—Music scores written in modern notation use staves as a reference system for assigning semantics to the individual symbols that appear in the score. Detecting this structural element is, therefore, a natural step in most Optical Music Recognition systems. However, many systems struggle to reliably detect staves. This paper investigates whether computers can learn to detect staves with a convolutional neural network given only a small set of images for which annotation are available. After an initial training phase, the network is asked to make prediction on a larger test dataset. A human annotator reviews the predictions and approves or rejects samples. Approved samples will be added to the training set for the next iteration to incrementally expand the training set and allow the network to operate well on a variety of music scores.**

After four iterations, we were able to obtain staff bounding box annotations for 14,000 out of 20,000 scores in our dataset. Although the evaluated approach has structural flaws that lead to imprecise results and deficits when detecting non-straight staves, it can serve as a viable starting point for future staff detection systems.

***Index Terms*—Optical Music Recognition, Music Staff, Object Detection**

I. Introduction

Music scores written in modern notation use staves - typically five parallel lines - as a reference system for notes. Staves are further divided into individual measures by bar lines to provide visual guidance for the reader. Detecting these structural elements is of fundamental importance to virtually every Optical Music Recognition (OMR) system. Most systems use staff detection as a preprocessing step to break the image down into meaningful sub-regions that can be processed individually [1]. A variety of methods have been proposed to robustly detect staves, or more precisely, individual staff lines before removing them [2]. This was a prerequisite for techniques such as connected-component analysis in the symbol detection stage. However, in the last few years, new approaches were proposed that are capable of detecting music objects without the need for staff removal [3], [4], but still require segmentation of the image into individual staves to overcome computational restrictions.

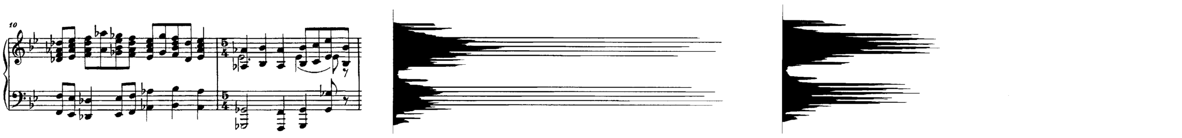
If measures can robustly be detected in music scores it is only a small step towards doing the same with staves. A universal staff detector in combination with a robust measure detector provides a vital framework for subsequent steps and is by itself already a useful tool, allowing users to quickly navigate through scores. One could also envision a collaborative tool where humans manually transcribe a music piece measure by measure, filling in an empty scaffolding of the score that was automatically generated. Such a tool could facilitate the crowd-sourced transcription of a large body of music, like the IMSLP [5]. Modern machine learning approaches with convolutional neural networks can reliably solve object detection tasks given a sufficient amount of annotated data. At the same time, they promise that the trained model will generalize to new data if the training data contains a sufficient amount of variation. To put this claim to the test and to avoid manually labeling thousands of images, this paper reports on experiments for training a staff detector with limited ground truth data.

II. Related Work

Under ideal conditions, staff lines in music scores are straight, parallel lines than span a large portion of the image. A simple, yet effective way to detect those lines is to perform a projection along the x-axis and look for very high peaks [6]. But this method deteriorates quickly if the image is slightly rotated or exhibits other distortions (see Fig. 1).

More robust methods were developed in the last 30 years including scan lines, Hough transformations and stable paths [7], [8]. The most notable effort on comparing these methods was the ICDAR Music Scores Competition [9]–[11], which produced the CVC-MUSCIMA dataset [12] and showed that several algorithms operate well even at high levels of synthetic degradation. So, is staff detection a solved problem? It is genuinely hard to answer because music scores can be extremely diverse. Even if an algorithm works well on a certain dataset, claiming it solves staff detection would require a thorough evaluation on a very large dataset such as the IMSLP [5] which contains nearly 500,000 scores and over 10 million pages. To the best of our knowledge, no one ever attempted such an evaluation. A recent test of commercial OMR applications indicates that many systems already struggle at this early stage if the input score is just 2° rotated [13], leaving plenty of room for improvement.

In recent years, the research interest shifted to machine learning methods which use convolutional neural networks to detect and remove staff lines [14]–[16]. Similarly, the detection of measures was also recently attempted with machine learning methods [17] and shows promising results. The main drawbacks of these machine learning methods are that they require a large amount of annotated data and while they do operate well within the boundaries of what data they saw during the training, it is not guaranteed that they work satisfactorily on new data.



(a) (b) (c)

Fig. 1. Primitive staff detection by projection from Bainbridge et al. The input score (a) with its projection profile under ideal conditions (b) and its projection profile after slightly rotating the image which reflects more realistic conditions (c).

III. Self-Learning Staff Detection

Ideally, we would like to have a versatile staff detector that can robustly detect staff lines under various conditions even at the presence of distortions. A state-of-the-art object detector [18] can robustly detect objects in images. Simple objects like staves and measures are no exception. It feels natural to further examine this avenue. However, due to the lack of a large annotated dataset (and out of sheer curiosity), an interactive approach is devised to build that staff detector. The idea is that you start with a small amount of annotated data, then you train on that data and perform inference on a much larger dataset. The detector will hopefully produce correct results for at least a few previously unseen pages. A human reviewer then examines the results. Those images with correctly detected staves are added to the set of scores for which we have correct annotations and used as training data for the next iteration. In theory, this process can be repeated until we know the staff positions of every music score in our database. Figure 2 illustrates this procedure.

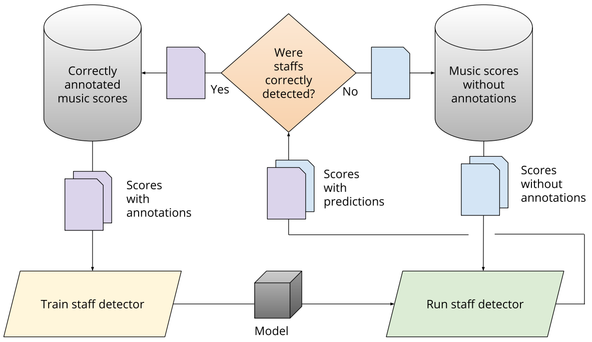


Fig. 2. The iterative workflow to train the self-learning staff detector.

For detecting bounding boxes, a Faster R-CNN [18] detector was used. The training regime is equivalent to the measure detector as described by Waloschek et al. [17] and the source code is publicly available[1].

The initial dataset for which ground truth annotations exist is the MUSCIMA++ dataset [19]. The target dataset for which no annotations were available is a large collection of multiple datasets containing approximately 20,000 images, ranging from typeset, born-digital images taken from the DeepScores dataset [20] to manuscripts with substantial degradation from the IMSLP. The curriculum for the first iteration was to generalize from the initial 100 pages to the entire CVC-MUSCIMA dataset of 10,000 images. After that first warm-up round, consecutive iterations evaluated the entire dataset for which the correct bounding boxes are not yet known.

[1] https://github.com/OMR-Research/MeasureDetector/tree/master/StaveDetector

IV. Results

After four iterations, we obtained reasonable bounding box annotations for 14,000 out of 20,000 scores. For the remaining 6,000 images, the detector was not able to produce acceptable predictions. While 14,000 seems high, it should be noted that the first 10,000 were just the CVC-MUSCIMA dataset for which correct predictions were easily obtained during the first iteration. Each consecutive iteration yielded a higher number of correct results. However, the idea that simply continuing for another few iterations will produce accurate results for the entire dataset, unfortunately, turned out to be unfeasible because of two reasons:

- The used detector operates on rectangular regions only. Real scores, however, are often bent or rotated, therefore a larger target bounding box is needed to cover the rotated staff. The model then learns to also use this larger bounding box for staves that are not rotated, causing increasingly inaccurate predictions.
- Prediction are rarely as precise as if the bounding boxes were annotated manually. If the annotator accepts a sample with such slight imperfections, he introduces a small error into the training set. These errors accumulate from iteration to iteration, leading to increasingly inaccurate results. This problem can best be observed with staves that should be smaller, because they are preceded by instrument names that we do not consider to belong to the staff. Therefore, these staves should be smaller. Given the lack of specific training data that exhibit this property, the detected regions frequently exceed the actual staff and do include the instrument name (see Fig. 3).

Reviewing the 14,000 images for which the neural network produced acceptable bounding boxes revealed that most of these images contain straight staves. Unsurprisingly, many of the bounding boxes exceed the boundaries of the contained staff quite significantly. Manuscripts with skewed staves were the hardest to predict correctly. A few examples of the results are given in the Appendix.

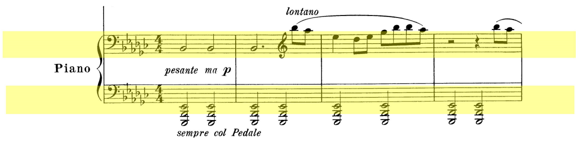


Fig. 3. A staff that is preceded by an instrument name with it's predicted bounding boxes shown as transparent yellow overlay.

V. Discussion and Conclusion

Reviewing images whether they contain appropriate bounding box predictions can be done much faster than actually annotating each image individually. Checking an individual image took less than a second, making this process very efficient. While the process is laborious at the beginning where less than 100 out of 1000 images can be moved from the pool of images without annotations to the pool with annotations, this procedure eventually gets even more efficient, as fewer images have to be reviewed each iteration. However, very dense scores, bent scores, or rotated scores still pose a major challenge to the detection model. The generalizability of the described approach to these scores remains limited.

The downside of this workflow is the decreasing quality of the bounding box predictions. In some scenarios, it can be acceptable to only have a coarse approximation, e.g., for cropping the image into sub-regions that are further processed, but it might not be suitable in other cases, e.g., if the staff line distance is estimated from the bounding box. Finally, in future work it would be interesting to compare this method to similar methods like Mask R-CNN [21] that perform instance segmentation and promise to also work on more irregular shapes. To allow for a better comparison, the best model will be made publicly available on Github. Large parts of the dataset can also be shared upon request.

References

[1] J. Calvo-Zaragoza, J. Hajič jr., and A. Pacha, "Understanding optical music recognition," *Computing Research Repository*, 2019. [Online]. Available: https://arxiv.org/abs/1908.03608

[2] A. Rebelo, I. Fujinaga, F. Paszkiewicz, A. R. Marcal, C. Guedes, and J. d. S. Cardoso, "Optical music recognition: state-of-the-art and open issues," *International Journal of Multimedia Information Retrieval*, vol. 1, no. 3, pp. 173–190, 2012.

[3] A. Pacha and H. Eidenberger, "Towards a universal music symbol classifier," in *14th International Conference on Document Analysis and Recognition*, IAPR TC10 (Technical Committee on Graphics Recognition). Kyoto, Japan: IEEE Computer Society, 2017, pp. 35–36.

[4] A. Pacha, J. Hajič jr., and J. Calvo-Zaragoza, "A baseline for general music object detection with deep learning," *Applied Sciences*, vol. 8, no. 9, pp. 1488–1508, 2018. [Online]. Available: http://www.mdpi.com/2076-3417/8/9/1488

[5] Project Petrucci LLC, "International music score library project," http://imslp.org, 2006.

[6] D. Bainbridge and T. Bell, "The challenge of optical music recognition," *Computers and the Humanities*, vol. 35, no. 2, pp. 95–121, 2001.

[7] C. Dalitz, M. Droettboom, B. Pranzas, and I. Fujinaga, "A comparative study of staff removal algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 5, pp. 753–766, 2008.

[8] J. d. S. Cardoso, A. Capela, A. Rebelo, C. Guedes, and J. Pinto da Costa, "Staff detection with stable paths," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 6, pp. 1134–1139, 2009.

[9] A. Fornés, A. Dutta, A. Gordo, and J. Llados, "The ICDAR 2011 music scores competition: Staff removal and writer identification," in *International Conference on Document Analysis and Recognition*, 2011, pp. 1511–1515.

[10] A. Fornés, A. Dutta, A. Gordo, and J. Lladós, "The 2012 music scores competitions: Staff removal and writer identification," in *Graphics Recognition. New Trends and Challenges*, Y.-B. Kwon and J.-M. Ogier, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 173–186.

[11] A. Fornés, V. C. Kieu, M. Visani, N. Journet, and A. Dutta, "The ICDAR/GREC 2013 music scores competition: Staff removal," in *Graphics Recognition. Current Trends and Challenges*, B. Lamiroy and J.-M. Ogier, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 207–220.

[12] A. Fornés, A. Dutta, A. Gordo, and J. Lladós, "CVC-MUSCIMA: A ground-truth of handwritten music score images for writer identification and staff removal," *International Journal on Document Analysis and Recognition*, vol. 15, no. 3, pp. 243–251, 2012.

[13] J. Noll, "Intelligentes notenlesen," *c't*, vol. 18, pp. 122–126, 2019.

[14] J. Calvo-Zaragoza, L. Micó, and J. Oncina, "Music staff removal with supervised pixel classification," *International Journal on Document Analysis and Recognition*, vol. 19, no. 3, pp. 211–219, 2016.

[15] J. Calvo-Zaragoza, A. Pertusa, and J. Oncina, "Staff-line detection and removal using a convolutional neural network," *Machine Vision and Applications*, pp. 1–10, 2017.

[16] A.-J. Gallego and J. Calvo-Zaragoza, "Staff-line removal with selectional auto-encoders," *Expert Systems with Applications*, vol. 89, pp. 138–148, 2017.

[17] S. Waloschek, A. Hadjakos, and A. Pacha, "Identification and cross-document alignment of measures in music score images," in *20th International Society for Music Information Retrieval Conference*, 2019.

[18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 91–99.

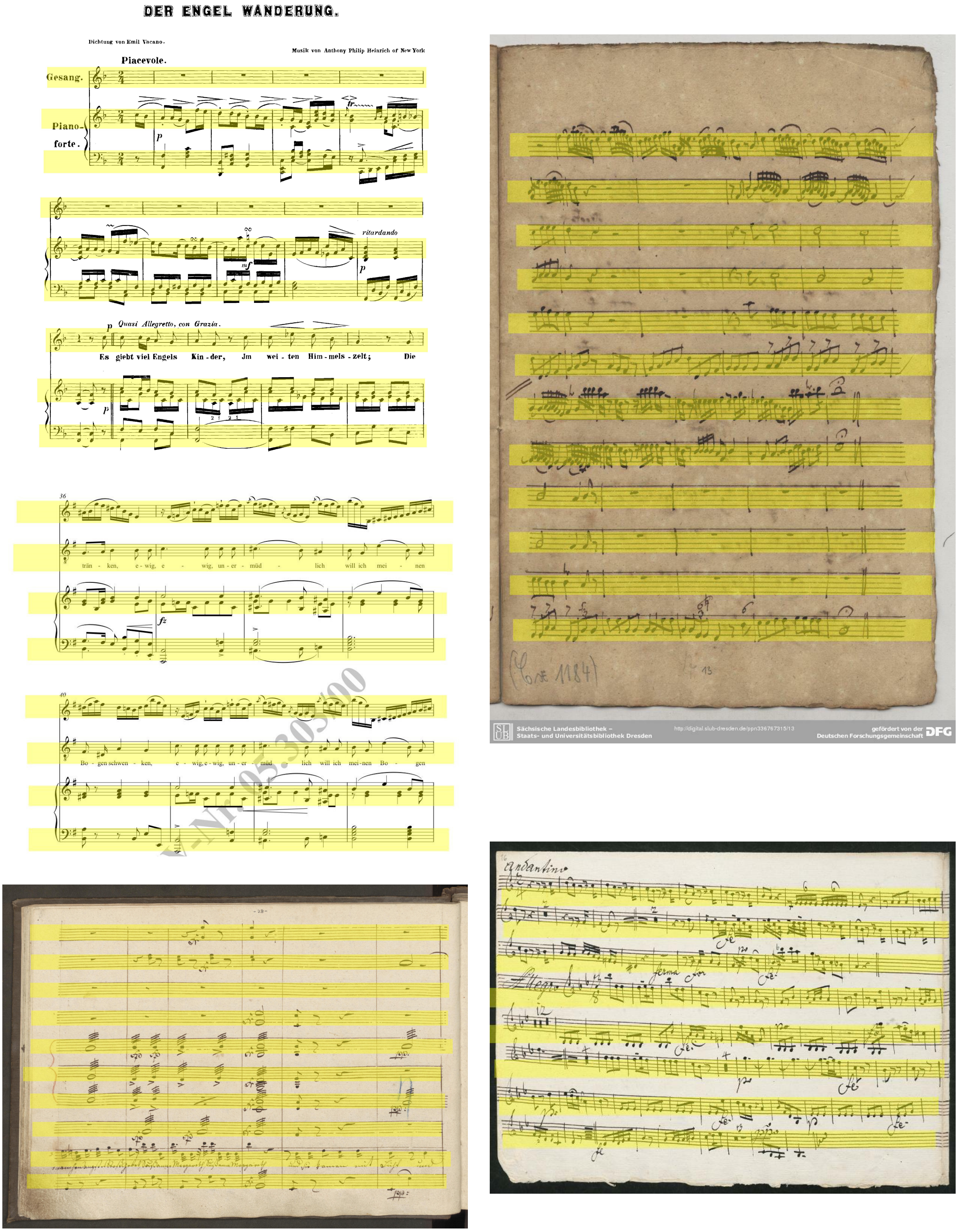
[19] J. Hajič jr. and P. Pecina, "The MUSCIMA++ dataset for handwritten optical music recognition," in *14th International Conference on Document Analysis and Recognition*, Kyoto, Japan, 2017, pp. 39–46.

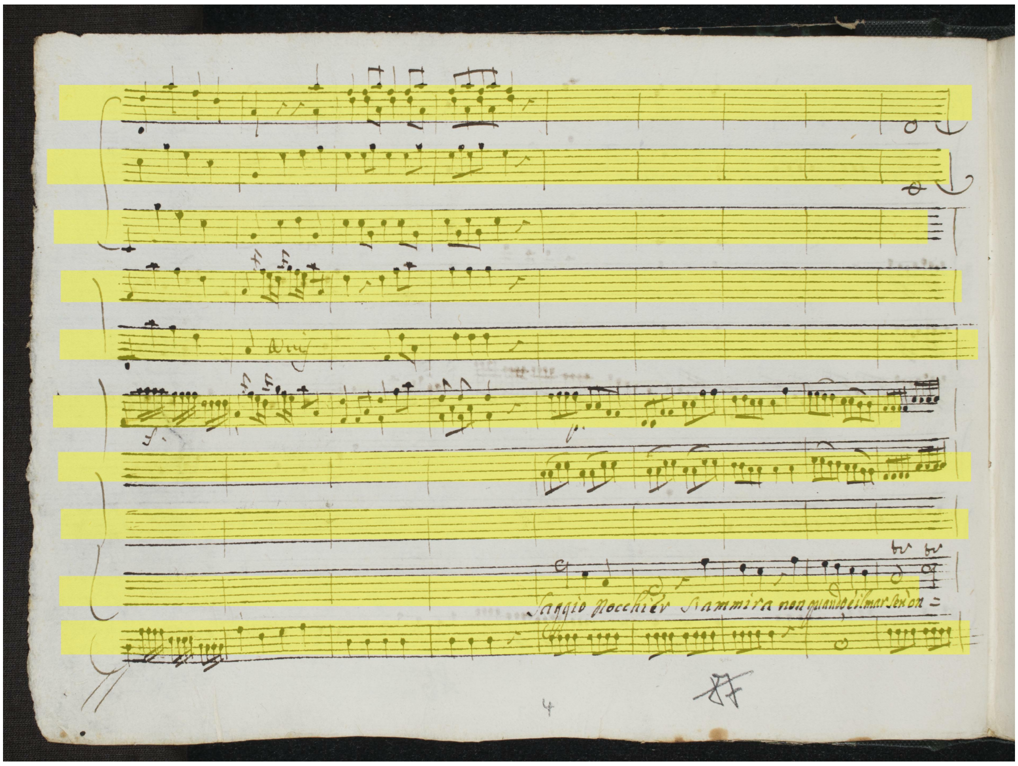
[20] L. Tuggener, I. Elezi, J. Schmidhuber, M. Pelillo, and T. Stadelmann, "Deepscores - a dataset for segmentation, detection and classification of tiny objects," in *24th International Conference on Pattern Recognition*, Beijing, China, 2018.

[21] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask r-cnn," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

Appendix

The following appendix contains a small selection of scores to demonstrate the variety of the data and the performance of the detector. The predicted bounding boxes are superimposed as transparent yellow overlay.


DER ENGEL WANDERUNG.
Dichtung von Emil Vacano.
Musik von Anthony Philip Heinrich of New York
Piacevole.
Gesang.
Piano-forte.
ritardando
Quasi Allegretto, con Grazia.
Es giebt viel Engels Kin-der, Jm wei-ten Him-mels-zelt; Die
trän - ken, e - wig, e - wig, un - er - müd - lich will ich mei - nen
Bo - gen schwen - ken, e - wig, e - wig, un - er - müd - lich will ich mei-nen Bo - gen
Sächsische Landesbibliothek – Staats- und Universitätsbibliothek Dresden
gefördert von der Deutschen Forschungsgemeinschaft DFG
Andantino
Allegro



Robust Transcript Alignment on Medieval Chant Manuscripts

Timothy de Reuse, Ichiro Fujinaga
Center for Interdisciplinary Research in Music Media and Technology
McGill University
Montreal, Canada
{timothy.dereuse, ichiro.fujinaga}@mcgill.ca

***Abstract*—We present a generalizable method of performing transcript alignment on the lyrics of medieval chant manuscripts. We use optical character recognition to generate a preliminary transcript of each page and then use a global sequence alignment method to match it up to the known correct transcript, combining the two incomplete sources of information into a high-quality alignment. We demonstrate this approach on manuscript pages using two different script styles from four different sources. This method requires little training data and works even when the transcript and the page itself have differing textual content, achieving per-syllable accuracies of 80–90% across the four sources.**

***Index Terms*—Optical Music Recognition, Transcript Alignment, Sequence Alignment, Historical Document Analysis**

I. Introduction

Western music notations has its origins in neume notation, which began as marks placed above syllables of text to denote their melodic contour. Accordingly, most notated Western European music up until the 15th century was for voice, and included lyrics [1]; however, little research in the field of Optical Music Recognition (OMR) has addressed the textual component of music notation. In order to encode early notated vocal music into a symbolic format, it is necessary to perform Optical Character Recognition (OCR) on the handwritten lyrics, which is not trivial and requires a large amount of training data to achieve a practical level of accuracy [2].

In some cases, we have access to a high-quality transcript of the lyrics of a manuscript. This tells us *what* text is on each page, but not *where* the text lies. In order to graphically display the location of text or associate musical content with each syllable, we need to ascertain where each syllable of the ground truth appears on the page; this task is called *transcript alignment*. Assigning musical notation to the aligned text is itself nontrivial, but for lack of space this issue is not addressed here. We focus on the lyrics of medieval chant manuscripts, which have many idiosyncrasies that make them difficult to analyze. To our knowledge, there is no previous research on transcript alignment on this type of manuscript.

The method we demonstrate here uses an OCR system to produce a preliminary transcription of each manuscript image, which misidentifies many characters but has estimated positions for all of them. We then use a dynamic programming-based sequence alignment algorithm to align this OCR transcript to an existing, correct transcript, combining the two sources of information into a high-quality alignment. Since chant manuscripts exist in a wide variety of notations and script styles, we focus on developing a method that can be adapted to other manuscripts; existing OCR models could be used with minimal effort spent on training and data preparation, since there is no requirement that the OCR transcript be highly accurate.

This research has been supported by the Social Sciences and Humanities Research Council of Canada (SSHRC) and the Fonds de Recherche du Québec – Société et Culture (FRQSC).

II. Related Work

Only a handful of published works in OMR address text. Most research focuses on separating lyrical and musical content, using traditional document analysis techniques [3]–[5] or deep learning [6]. George [7] discusses using OCR on extracted lyrics and aligning identified words with musical symbols on the page. Hankinson et al. [8] incorporate an existing, pre-trained OCR system as a step into an OMR system intended for printed square-note neume notation, but note that the resulting text still has many errors.

On historical handwritten documents, transcript alignment is generally performed using Hidden Markov Models (HMMs) [9]–[11] or with dynamic programming methods such as dynamic time warping [12], [13]. A common paradigm with these methods is the exploitation of *anchor words*, which are words that appear only once in a given transcript and so can be located in the image with a higher degree of certainty, so that alignment can be performed on strings of text that lie between anchor words. It is also possible to render a transcript as an image file and directly map between regions of the manuscript image and regions of the synthesized image, though this requires the availability of a font that bears resemblance to the handwriting used in the manuscript [14].

Aligning the inaccurate output of an OCR system to a known transcript is not a novel technique, but it has most often been done as a step in training or evaluating an OCR system. Feng and Manmatha [15] use HMM-based sequence alignment to assign a score to the similarity between an OCR system's output and a ground-truth transcript. Romero-Gomez et al. [9] use HMM-based sequence alignment to automate the generation of OCR training data, though their method operates on individual text lines rather than whole pages.

III. Alignment Method

We operate on four manuscripts available in the Cantus manuscript database [16]: the Salzinnes Antiphonal (CDN-Hsmu M2149.14),[1] Einsiedeln (Stiftsbibliothek, Codex 611(89)),[2] St. Gallen 388 (CH-SGs 388), [3] and St. Gallen 390 (CH-SGs 390).[4] The Salzinnes and Einsiedeln manuscripts are written in Gothic script (Figure 1), while the two from St. Gallen are written in Carolingian minuscules (Figure 2). The Cantus database[5] contains plain text transcripts for each chant in these manuscripts. Any text that is not part of a chant is not transcribed. Chants often begin on one page and end on the succeeding page, but the Cantus database lists each chant as occurring on the page where it begins. So, given a folio number, we can retrieve a string of transcribed chants that mostly correspond to the text on the page of interest, but may contain text *not* on the page (the beginning and endings of chants on neighboring pages), and may exclude some text that *is* on the page (non-chant text). The goal of this method is to correctly align all textual content that lies in the intersection between the Cantus transcript and the manuscript page, while ignoring content that only appears in one or the other.

A. Pre-Processing

For the Salzinnes manuscript, we use a set of pages where text layers have already been extracted using the pixelwise classification method developed by Calvo-Zaragoza et al. [6]; on the other three manuscripts we use a set of pages where the text layer has been isolated from the background manually. After this step, the text layer is deskewed to straighten the text lines. Lines are identified by finding prominent peaks on the horizontal projection profile, and splitting into strips at local minima.

We assemble training data by manually transcribing manuscript pages, and train two OCR models using the OCRopus open-source OCR system [17]. In principle, we could use any OCR system capable of outputting per-character horizontal positions, but of the open-source options available we found OCRopus to be both accurate in its character segmentation and convenient in its utilities for generating training data. The first model is trained on forty pages of Gothic script from the Salzinnes manuscript, which comprises a total of 2302 words. The second is trained on Carolingian minuscule script from St. Gallen 390 (five pages) and St. Gallen 388 (two pages) comprising a total of 1140 words. Both of these models are trained until their output no longer seems to improve, which took about eight hours on an ordinary desktop PC for each of them. The character error rate of the OCR results against the training data is 0.127 for the Gothic script model and 0.125 for the Carolingian script model.

The scribes who wrote these manuscripts often abbreviated commonly occurring lyrics to save space on expensive parchment. A word may not always appear as an abbreviation, but there are rarely two different abbreviations that represent the same word. `dñs` → `dominus`, `dñe` → `domine`, `&` → `et`, and `alla` → `alleluia`. We also add an `m` after vowels that have bars over them, which is the most common meaning of that symbol (e.g., `ū` → `um`). This treats abbreviations as if written out in full, but with the letters of each syllable overlapping, allowing us to handle them normally when grouping characters into syllables. The assumption made here is that abbreviations never collapse more than syllable into a single letter, which is true for those that we handle.

[1]`cantus.simssa.ca/manuscript/133/`
[2]`www.e-codices.unifr.ch/en/list/one/sbe/0611`
[3]`www.e-codices.unifr.ch/en/csg/0388/`
[4]`www.e-codices.unifr.ch/en/csg/0390`
[5]`cantus.uwaterloo.ca/`

Fig. 1. A section from folio 042r in the Salzinnes Manuscript. Note abbreviations, non-chant text, and the large ornamental letter.

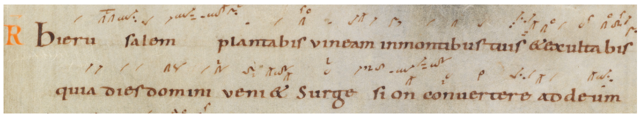


Fig. 2. A section from folio 23 in the St. Gallen 390 Manuscript.

B. Sequence Alignment

After running the trained OCR models on each text line in a page, we retrieve a transcript that is inaccurate, but contains correct positions for most of the characters. We use the Needleman-Wunsch algorithm [18], a global sequence alignment method, to match these sequences together. This algorithm lines up both transcripts alongside each other, and tries to make the sequences match in as many positions as possible by adding *gaps* to both sequences, which let the alignment skip over characters that are only in one of the sequences. More formally, a gap represents a position where one would have to insert or delete a character in order to transform one sequence into the other. Where there is non-chant text in the OCR transcript, the algorithm tends to insert gaps into the Cantus transcript; where there is text in the Cantus transcript that is not on the page being processed, the algorithm tends to insert gaps into the OCR transcript. Our implementation uses affine gap penalties, which encourages the algorithm to use fewer long gaps rather than many short ones, encouraging matched regions to be unbroken, contiguous strings.

Table I shows the result of this sequence alignment on an excerpt from the top of a page of the Salzinnes manuscript (Figure 1), where the text starts in the middle of the chant:

TABLE I
AN EXCERPT FROM THE CANTUS TRANSCRIPT OF THE PAGE SHOWN IN FIGURE 1, AN EXCERPT FROM THE OCR TRANSCRIPT OF THE SAME PAGE, AND THE RESULTS OF ALIGNING THESE TWO STRINGS WITH THE NEEDLEMAN-WUNSCH ALGORITHM.

Cantus Transcript	OCR Transcript
`bethleem et videamus hoc verbum quod fa`	`ctū est qd ds ostendit notbis alla Euouae.`
`ctum est quod dominus ostendit nobis alleluya euouae`	`vus perbum`
`et venerunt festinantes et`	`t venerūt festinātes et m Ad.x. Antipl.`
`invenerunt mariam et joseph et infantem positum`	`uenerūt mariā et ioseph et infantē positū`

↓

Global Alignment

```
---------------------------------------ctum est q--d d-----s ostendit notbis alleluia Euouae.
bethleem et videamus hoc verbum quod factum est quod dominus ostendit no-bis alleluya euouae--

vus perbumt venerumt festinamtes et m Ad.x. Antipl.uen erumt mariam et ioseph et infantem positums
--- -e----t venerunt festinantes et -----------i--nven erunt mariam et joseph et infantem positum-
```

"`-ctum est quod dominus...`" Because of this, the transcript here includes a portion of the text from the previous page, which is necessary in practice because we have no information as to where a chant crosses a page break. Wherever two characters are matched up in this alignment, we can assume that both characters refer to the same textual material.

C. Grouping into Syllables

Chant is more naturally segmented into syllables than into words, since each syllable is sung to one or more neumes. The last stage of the alignment involves splitting the Cantus transcript into syllables and analyzing which syllables have been matched to which characters in the OCR transcript. If the syllable is aligned with any characters, whether or not the characters themselves match, the syllable is assigned the union of those characters' bounding boxes on the page. This also applies if a syllable is assigned to some characters and some gaps. In Table I, the second instance of `quod` in the transcript, assigned to `q--d` in the alignment, will take the bounding box of `qd` in the original OCR. If a syllable is aligned only with gaps, then it is assumed to refer to text not on the page, and is ignored. Any OCR characters that are not aligned with any syllable of the transcript are assumed to be non-chant text and are ignored. Finally, we reposition each bounding box to compensate for the deskewing of the whole page in the preprocessing step, so that they correctly match up with their content (syllables) in the original manuscript.

IV. RESULTS

We evaluated this method against manually assembled ground-truth annotations of alignment on five pages from the manuscripts under examination. The annotations consisted of a list of text syllables, each associated with a single bounding box that fits the characters on that page. We calculated the intersection-over-union (IoU) of each syllable's bounding box in the ground truth and its corresponding bounding box from our alignment. However, the size of our bounding boxes is influenced by the precision of our text line segmentation method. To ensure that we were not also implicitly evaluating how well we segment text lines, we binarized the text layer first, and ignored white pixels in the IoU calculation. This means that two bounding boxes were given an IoU score of 1 only if all of the black pixels in one are also in the other, regardless of their total size. We place each syllable into one of three categories based on its IoU under this scoring method:

- Matches: IoU > 0.5
- Partial matches: IoU < 0.5
- Misses: IoU = 0, or the corresponding bounding box is missing from our alignment altogether.

The results of this evaluation are in Table II, where the Accuracy column denotes $\frac{\text{Matches}}{\text{Total}}$. Figure 3 shows the result of the alignment from the same excerpt shown as an example in Table I. Larger images illustrating the results of the alignment on pages from each manuscript are available in the appendix (Section VI). In each of these images, syllables of text are highlighted with yellow if they correspond to a partial match, and red where they correspond to a miss. It is difficult to directly compare these results to others in the literature, since transcript alignment on chant manuscripts has (to our knowledge) not been previously addressed; however, Fischer et al. [11] achieved word-label accuracies of 92% with their HMM-based alignment algorithm on a Latin manuscript with a similar style as the two St. Gallen manuscripts we used here.

The method performed best on long, unbroken strings of chant text, with few abbreviations. Most errors were a result of non-chant text, occurring frequently at the beginning or end of larger segments of non-chant text that are bookended by musical text; the sequence alignment algorithm can "figure out" that there exists text in the OCR transcript to skip, but it begins or ends the gap too early or too late. Rubrics[6] or other isolated markings can also cause similar off-by-one errors, especially when an adjacent character is transcribed incorrectly by the OCR model; in that case, the sequence alignment has no cost incentive to align a gap to one over the other. Uncommon abbreviations also cause errors, since the sequence alignment would have to insert a pattern of several gaps to correctly align only the characters in the abbreviation.

[6]Short instructions or descriptions related to the liturgical process, commonly inserted between chants.

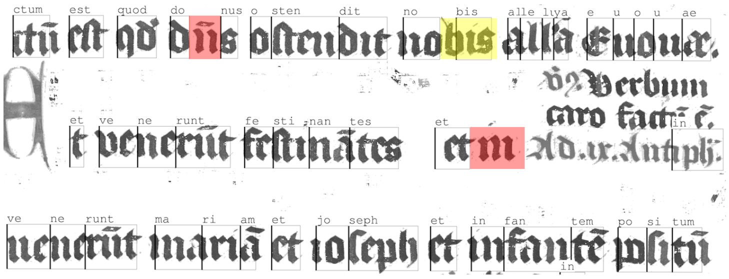


Fig. 3. The results of the transcript alignment performed on the image in Figure 1 using the sequence alignment results shown in Table I. The yellow highlight marks a partial match, while the two red highlights mark misses.

TABLE II
THE PER-SYLLABLE ACCURACY OF OUR ALIGNMENT METHOD, TESTED AGAINST MANUALLY CONSTRUCTED GROUND TRUTH.

Folio	Matches	Partial	Misses	Total	Accuracy
Salzinnes, 013r	157	10	5	172	91.2%
Salzinnes, 020v	151	9	3	163	92.9%
St. Gallen 390, 23	99	12	15	126	78.6%
St. Gallen 388, 28	310	21	20	351	88.3%
Einsiedeln, 004v	211	23	17	251	84.1%

More often, abbreviations resulted in the alignment attempting to cram extra characters into neighbouring words.

V. Conclusion

We have demonstrated a generalizable method of performing transcript alignment on medieval chant manuscripts. Generalizing this approach to other manuscripts requires only an OCR model that can achieve adequate per-character accuracy on the script under consideration. Compared to HMM-based models that require manual creation of ground-truth alignment data for each manuscript under consideration, this approach has the potential to save a significant amount of time in the end-to-end OMR process, especially when alignments on multiple manuscripts are desired. Future work on this method may focus on testing how well the OCR models need to perform to achieve optimal alignment results, and searching for optimal sets of parameters to fine-tune the behavior of the global sequence alignment.

References

[1] H. Strayer, "From Neumes to Notes: The Evolution of Music Notation," *Musical Offerings*, vol. 4, no. 1, pp. 1–14, 2013.

[2] F. Cloppet, V. Églin, V. C. Kieu, D. Stutzmann, and N. Vincent, "ICFHR2016 Competition on the Classification of Medieval Handwritings in Latin Script," in *Proceedings of the 15th International Conference on Frontiers in Handwriting Recognition*, Oct. 2016, pp. 590–595.

[3] J. A. Burgoyne, Y. Ouyang, and T. Himmelman, "Lyric Extraction and Recognition on Digital Images of Early Music Sources," in *Proceedings of the 10th International Society for Music Information Retrieval Conference*, 2009, p. 5.

[4] C. Dalitz, G. K. Michalakis, and C. Pranzas, "Optical Recognition of Psaltic Byzantine Chant Notation," *International Journal of Document Analysis and Recognition*, vol. 11, no. 3, pp. 143–158, 2008.

[5] C. M. Dinh, H. J. Yang, G. S. Lee, and S. H. Kim, "Fast Lyric Area Extraction from Images of Printed Korean Music Scores," *IEICE Transactions on Information and Systems*, vol. E99.D, no. 6, pp. 1576–1584, 2016.

[6] J. Calvo-Zaragoza, F. Castellanos, G. Vigliensoni, and I. Fujinaga, "Deep Neural Networks for Document Processing of Music Score Images," *Applied Sciences*, vol. 8, no. 654, 2018.

[7] S. E. George, "Lyric Recognition and Christian Music," in *Visual Perception of Music Notation: On-Line and Off-Line Recognition*, S. E. George, Ed. IRM Press, 2004, pp. 198–226.

[8] A. Hankinson, J. A. Burgoyne, G. Vigliensoni, A. Porter, J. Thompson, W. Liu, R. Chiu, and I. Fujinaga, "Digital Document Image Retrieval Using Optical Music Recognition," in *Proceedings of the 13th International Society for Music Information Retrieval Conference*, 2012.

[9] V. Romero-Gómez, A. H. Toselli, V. Bosch, J. A. Sánchez, and E. Vidal, "Automatic Alignment of Handwritten Images and Transcripts for Training Handwritten Text Recognition Systems," in *Proceedings of the 13th IAPR International Workshop on Document Analysis Systems*, 2018, pp. 328–333.

[10] J. Rothfeder, R. Manmatha, and T. M. Rath, "Aligning Transcripts to Automatically Segmented Handwritten Manuscripts," in *Document Analysis Systems VII*, ser. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2006, pp. 84–95.

[11] A. Fischer, V. Frinken, A. Fornés, and H. Bunke, "Transcription Alignment of Latin Manuscripts Using Hidden Markov Models," in *Proceedings of the 2011 Workshop on Historical Document Imaging and Processing*, 2011, pp. 29–36.

[12] R. Cohen, I. Rabaev, J. El-Sana, K. Kedem, and I. Dinstein, "Aligning Transcripts of Historical Documents Using Energy Minimization," in *Proceedings of the 13th International Conference on Document Analysis and Recognition*, 2015, pp. 266–270.

[13] A. Kumar and C. V. Jawahar, "Content-Level Annotation of Large Collection of Printed Document Images," in *Proceedings of the Ninth International Conference on Document Analysis and Recognition*, 2007, pp. 799–803.

[14] G. Sadeh, L. Wolf, T. Hassner, N. Dershowitz, and D. S. Ben-Ezra, "Viral Transcript Alignment," in *13th International Conference on Document Analysis and Recognition*, 2015, pp. 711–715.

[15] S. Feng and R. Manmatha, "A Hierarchical, HMM-based Automatic Evaluation of OCR Accuracy for a Digital Library of Books," in *Proceedings of the Joint Conference on Digital Libraries*, Chapel Hill, NC, 2006.

[16] *Cantus: A Database for Latin Ecclesiastical Chant – Inventories of Chant Sources*. Directed by Debra Lacoste (2011-), Terence Bailey (1997-2010), and Ruth Steiner (1987-1996). Web developer, Jan Koláček (2011-). [Online]. Available: http://cantus.uwaterloo.ca/

[17] T. M. Breuel, "The OCRopus Open Source OCR System," in *Proceedings of the Society of Photo-Optical Instrumentation Engineers*, 2008.

[18] S. B. Needleman and C. D. Wunsch, "A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins," *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, 1970.

VI. Appendix

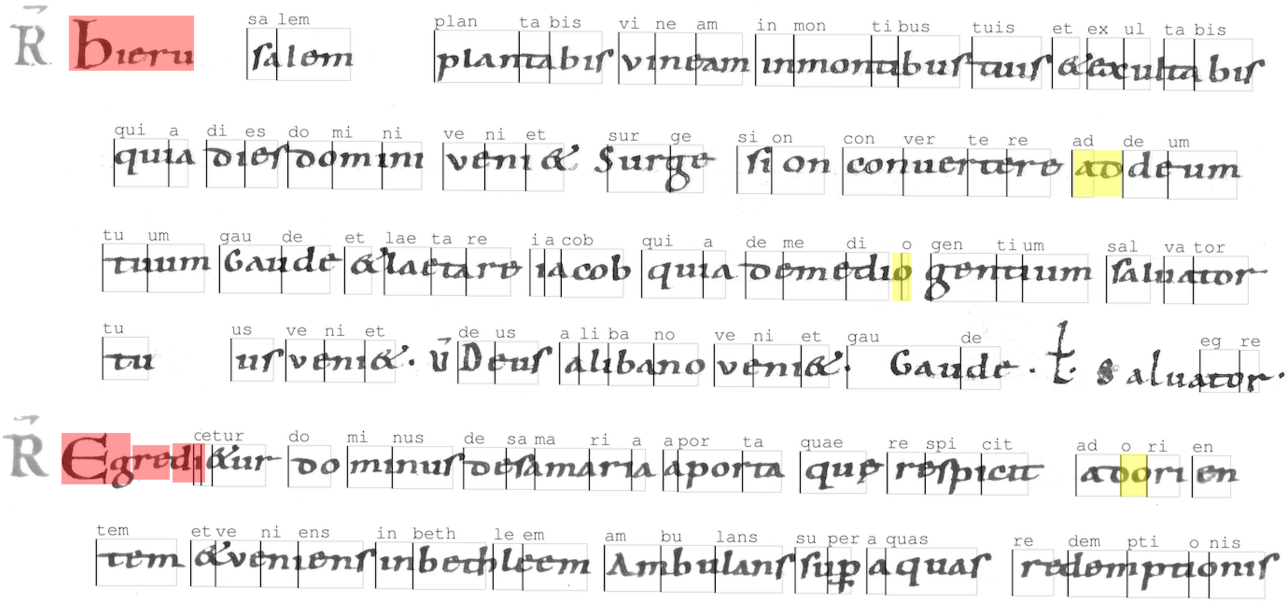


Fig. 4. An excerpt from the results of our alignment method on St. Gallen 390, folio 23. Yellow highlights on syllables mark partial matches, and red highlights on syllables mark misses.

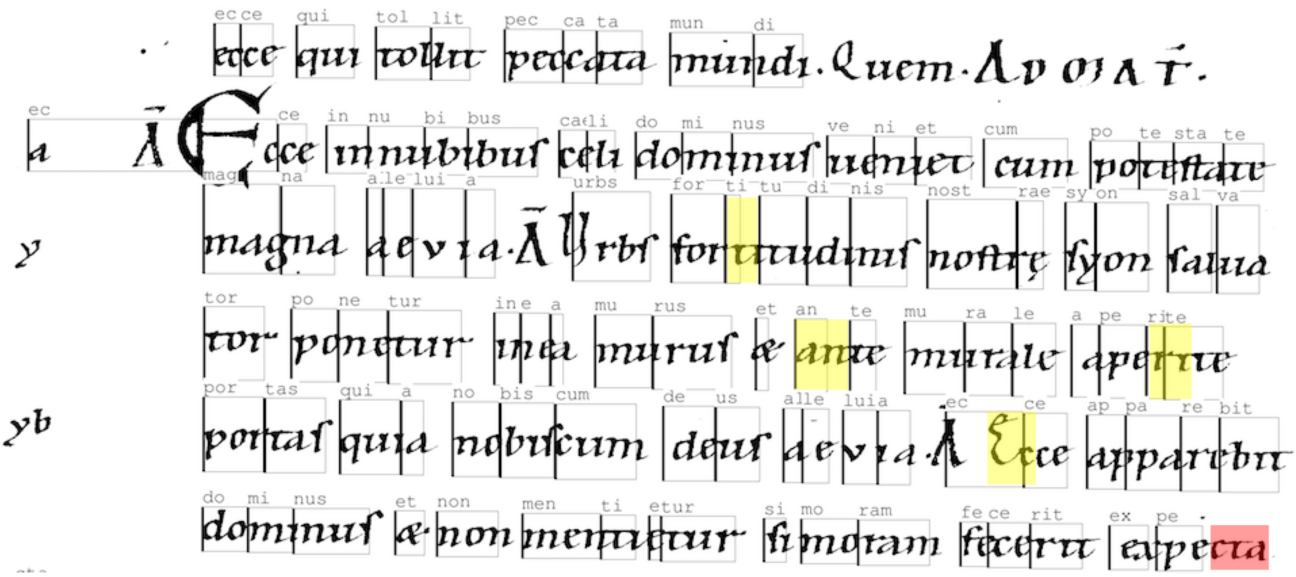


Fig. 5. An excerpt from the results of our alignment method on St. Gallen 388, folio 28. Yellow highlights on syllables mark partial matches, and red highlights on syllables mark misses.

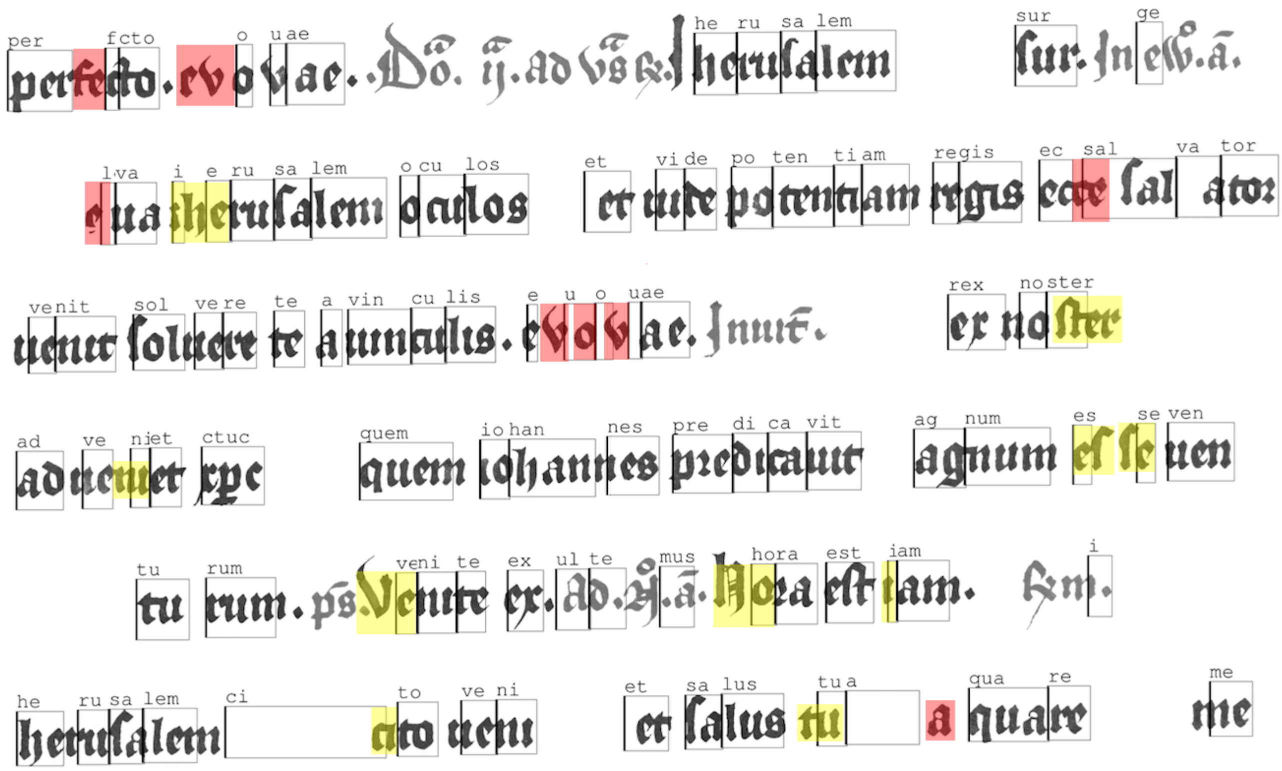


Fig. 6. An excerpt from the results of our alignment method on Einsiedeln, folio 004v. Yellow highlights on syllables mark partial matches, and red highlights on syllables mark misses.

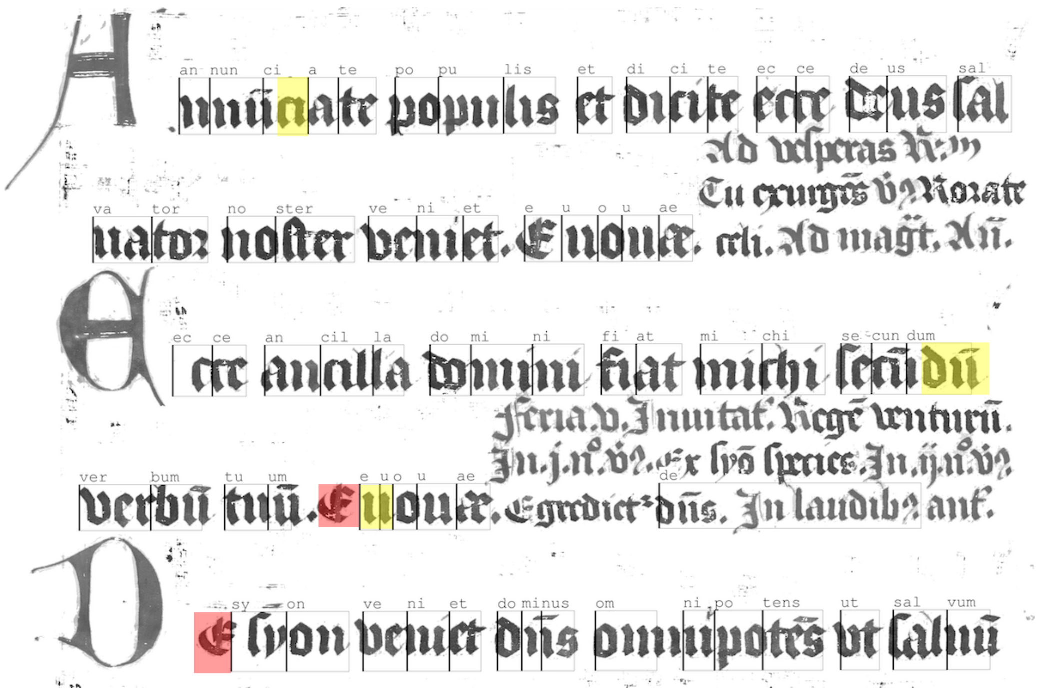


Fig. 7. An excerpt from the results of our alignment method on Salzinnes, folio 020r. Yellow highlights on syllables mark partial matches, and red highlights on syllables mark misses.

ReadSco: An Open-Source Web-Based Optical Music Recognition Tool

Antonio Ríos-Vila, Jorge Calvo-Zaragoza, David Rizo, Jose M. Iñesta
Department of Software and Computing Systems
University of Alicante, Alicante, Spain
{arios,jcalvo,drizo,inesta}@dlsi.ua.es

***Abstract*—We introduce READSCO, an open-source web-based community tool for Optical Music Recognition. READSCO aims to serve as a connection between research results and practical use. We describe the design decisions considered to both favors a rapid integration of new advances from the research field and facilitate the community's participation in its development. The project is still in its planning phase, so this work is a good opportunity to present the main idea and get direct feedback from other researchers.**

***Index Terms*—Optical Music Recognition, Open-Source Software, Web-based Application**

I. INTRODUCTION

Optical Music Recognition (OMR) [1] is still considered an open problem, especially in the most complex contexts. However, recent efforts, supported by modern machine learning techniques, have moved the research field from dealing with sub-problems (such as staff-line removal [2] or symbol classification [3]), to a state in which complete results are attainable [4]–[6].

Despite the above, many issues hinder taking OMR results from research to practice, such as the file management system, the development of appropriate visual interfaces, or exporting the recognition results to standard formats. In this work we introduce READSCO project, which aims to fill that gap from a community and scientific perspective. READSCO is a web-based open-source cross-platform OMR system, which covers the engineering processes required, while also seeking to be considered as a tool on which to integrate recent results of OMR research.

II. BACKGROUND

Since the rise of general-purpose computing, there have been efforts to make computers capable of reading music. It is obvious that the industry is aware of this, and there exist commercial applications that attempt to provide this service as SmartScore [7] or PhotoScore [8]. However, these applications are proprietary and therefore do not pursue the philosophy of scientific and community development.

There also exists Audiveris [9], an open-source desktop-based system for OMR. This system is the one that most closely resembles our intention; however, it has not established itself as a community item. We aim to build an open-source tool that can be shared by the research community and enables a rapid integration of recent developments. At the same time, we want to provide a useful tool for practitioners.

We take inspiration from two recent web-based applications: Rodan [10] and MuRET [11]. Rodan is a rather generic system, where users can specify their own OMR workflow from pre-defined steps. Our application architecture and OMR pipeline conformation are inspired by how it defines a "process" ("jobs" in Rodan). MuRET is a tool specifically designed for scholars, which covers all transcription phases from the manuscript source to the encoded digital content. It is designed as a technology-focused research tool, allowing different approaches to be used, as well as producing both the transcribed contents in standard formats and data for the study of the transcription process itself. In this case, we took its technology as a reference for developing the project, as well as an example of a user interface for this kind of applications.

III. READSCO

In this section, we describe the technologies that build up READSCO. More details are provided in its public repository, available at https://github.com/OMR-Research/ReadSco.

A. System Architecture

From an architectural point of view, all technical requirements of the application must be considered:

- Offer both fast analyses without user intervention and step by step user-guided analyses
- Easy scalability
- Easy code maintenance and fast error detection
- Low coupling among developed services and independence among the technologies used to develop the application's functionalities
- Warrant the easy and secure community contribution and support

A microservices architecture seems to be the most suitable architecture to achieve those requirements. This architecture splits the application's logic in a set of programs, known as microservices, which focus on a specific task [12]. They are completely independent of one another. However, combining them allows the development and fulfillment of complex functionalities in a distributed way. One practical example of what microservices would be in a common application would be database access, file storage, or, in our case, a specific task or process which takes part in the OMR workflow. This architecture simplifies considerably the application scalability.

In the case of READSCO, the microservice architecture allows the fulfillment of our primary objectives. It makes it possible to separate OMR tasks in detached programs. Splitting up OMR into microservices by its function and coordinate them with an event management system allows us to easily control and manage them with a user interface. It also enables us to handle the substitution and exchange of them, as it is supposed in a collaborative model. That is, microservices architecture supports the pursued community contribution model. Contributions will be made to a particular set of microservices, and not to the whole application. Further complex errors and code conflicts in contributions will also be easier to handle in this architecture, which will be an enormous advantage in the future.

Once has been determined the convenience of this architectural model, the first blueprint of the application structure was designed, as represented in Figure 1.

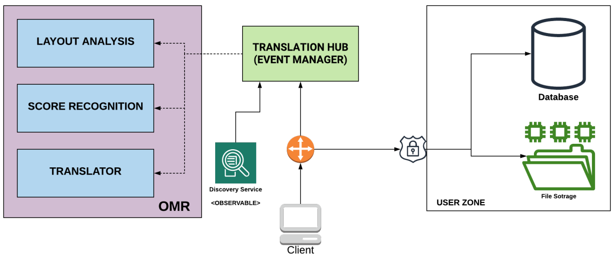


Fig. 1. Application structure blueprint

This blueprint is an ideal representation of the minimum services required for the application. There are two main areas: the first one is the OMR pipeline, which contains all the system's detection and recognition layers, as well as an encoding unit, called translator, between our internal format and common music representation formats, such as MEI, MusicXML or `**kern`. Each of these layers is represented by a microservice. These, in turn, are connected to an event manager. It is responsible for sending instructions to all of the OMR layers to do their tasks and send back the results for the user. To access this area there is an intermediate service, called Translation Hub, which receives all user's requests and processes them to prevent OMR layers from taking responsibilities they should not handle. It also allows asynchronous task handling without losing connection with the user.

The second zone corresponds to the user's data management. In this current state, there will be only a simple user account management, with its corresponding database, and a file system to store rendered scores and its corresponding OMR result. This zone may suffer several changes, as it hasn't been a priority during development and needs thorough research on document indexation and storage.

Generally speaking, access to all back-end services there is within an external application. Namely with a web browser, a mobile application or a command-line client that will be developed for advanced functionalities and uses. Access to microservices will only depend on the user's requests. So, all workload will be distributed in the application. Users will only use the required microservices for their specific requests. This strategy will avoid further overload issues in our servers.

Once the system is reviewed, there will be only one final issue to tackle: how to organize the application to support different technologies and programming languages. As the intention is to be a community-supported project, it will be desirable to not require a specific language knowledge for our contributors to their further software development or services for the application. Our proposal to alleviate this issue is the use of Docker [1] and Ansible [2] as deployment technologies.

Docker is a virtualization software that allows packaging applications for easy deployment on the Internet. The most remarkable advantage to READSCO is the possibility of virtualizing microservices developed in different programming languages and execute them, on a deployment scenario as in a development one, without the necessity of making compiler and interpreter installations or additional configurations. In turn, it exists the possibility to share all of these virtual systems, called images, publicly. Hence, a community user who wants to contribute to a specific service or to develop its own does not need to make any software configurations to set the system on his other development environment. He or she just needs Docker to execute the public image or locally build our system with few Docker commands.

Unfortunately, to coordinate numerous containers with the Docker API may result uncomfortable and tedious in both software development and application deployment scenarios. As our objective is to make these tasks as easy as possible to engage our contributors, it is required to count with some automation in what is called "container orchestration". For this issue, we chose Ansible as our orchestrator. Ansible is an IT automation tool developed by RedHat which allows full automation of application configuring and running. In our case, it will take the responsibility of pulling updated Docker images, verify any changes and automatically kill and restart all affected microservices. It works with configuration files, called playbooks, that execute a user-defined set of tasks. It is compatible with all operative systems and does not require any specific server architecture to work, which makes it a powerful tool to use in development and deployment workflows. Community contribution will be even easier: the user only needs to download our default Ansible configuration files and update them with the specific Docker images he wants to work with. It is not needed to rebuild and run manually any container, the tool will do it for him or her. In any case, to provide a comfortable developing experience and avoid common network troubles, READSCO organization will leave detailed instructions on how it is recommended to work with its technologies in the official application documentation. Furthermore, it is not necessary to push code, even it is

[1] https://www.docker.com
[2] https://www.ansible.com

suggested for error debugging and community updates. It will only be needed for an upgraded system or the new microservice Docker image to be pushed into the official READSCO Docker repositories.[3] Any contribution will be reviewed by the official READSCO organization, which will have the Ansible playbooks to easily include new updates and services into the official application servers.

B. Interface

READSCO is also an application allocated for practical user interaction. Therefore, we also focused our attention on bringing a comfortable User Experience. After analyzing the different user profiles which may interact with our application, we decided to guide its design on usability. The application is devised to be simple, direct and easy to use. Another feature taken into account in this process is the responsive layout, as it will be displayed on both computer screens and mobile devices. It should be emphasized that READSCO is intended to be deployed as a web software and as a native Android and iOS application. Therefore we are currently developing it with Ionic 4 and Angular 2 [13], which can fulfill this requirement.

In the current state of the application, users upload score images, select the area in which they are interested to be analyzed and receive the score in a standard coding format that can be rendered with Verovio [14].

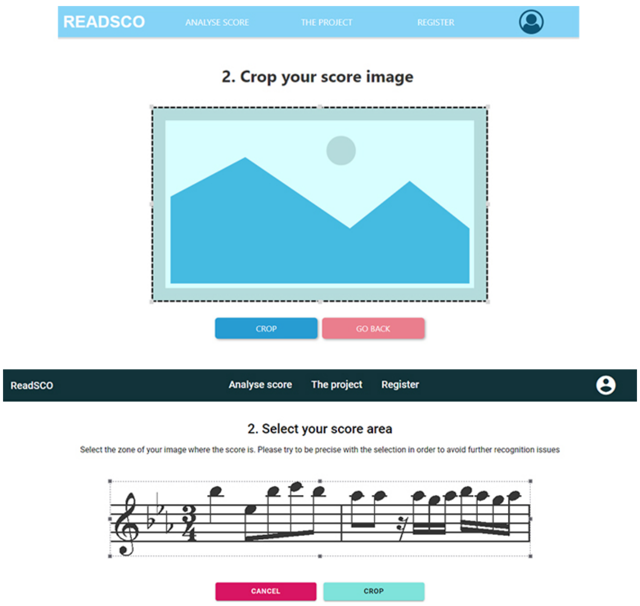


Fig. 2. READSCOHigh-Fidelity prototype for image selection functionality (above) and final interface implementation (below)

We are aware that OMR technologies do not provide perfect results, and they probably never will. However, in the current state of our project, we do not find it interesting to develop a correction interface but we rather follow an "I'm Feeling Lucky" approach. The user can download the recognition result in a standard format that can be opened by any score edition software, to correct the remaining errors. In the future, we will consider the possibility that these corrections can be done online, although a thorough investigation about the best interface to do this is still required.

C. Optical Music Recognition

Our optical processing does not treat the document globally but instead divides it into bounded regions. In our case, the main recognition core works at a single-staff level. This is analogous to what happens in text recognition algorithms, where single-line level recognition is assumed [15].

Thus, we use the optical recognition model proposed by Alfaro-Contreras et al. [16]. This work is based on the use of deep neural networks that process a single-staff image and yields a sequence of tokens as a hypothesis. These tokens include both the recognized musical symbols and some information about the geometric relationship among them. This allows recognizing non-sequential structures such as chords or ligatures. In addition to its excellent performance, this approach is *holistic*, that is, the recognition is performed in a single step, thus facilitating its integration into the system. In our case, during the training of the neural network, we consider a strong data augmentation process to provide robustness against possible alterations of the input image [17].

To complete the recognition, we simply add two algorithms that operate before and after the staff-level recognition:

- *Staff segmentation*: there are several algorithms to divide a score into isolated staves, given the excellent reference that the staff lines represent. Specifically, we currently use the idea proposed by [18].
- *Notation Reconstruction*: the result provided by the optical recognition needs to be parsed to retrieve the actual music semantics. We convert the sequence of tokens to a sequence-based semantic representation (similar to `**kern`). This process is currently performed by a rule-based algorithm. We are however willing to explore other approaches such as those used in machine translation.
- *Encoding*: once the music content is captured, there is still the need for encoding it into a standard format. For this purpose, we built a hand-crafted parser that processes semantic sequences and exports them into MEI, MusicXML, and MIDI. The former is automatically retrieved for the recognition to be rendered by Verovio (see the previous section) and the rest of them are provided when requested.

At present, we leave the most complex levels of music notation structure (such as pianoform or polyphony) out of our targeted domain since it is not clear how to deal with it with state-of-the-art end-to-end technologies. However, the system is constructed to allow for an easy replacement of the underlying OMR operation. Therefore, any advance in the research field should be easy to integrate into READSCO.

[3] https://hub.docker.com/orgs/readsco/repositories

As a summary, our current OMR pipeline is depicted in Figure 3. It should be noted that this is the pipeline that we initially assume in READSCO. As mentioned above, we intend that the tool allows integrating different approaches to OMR through a standard communication protocol among the different agents involved within the system.

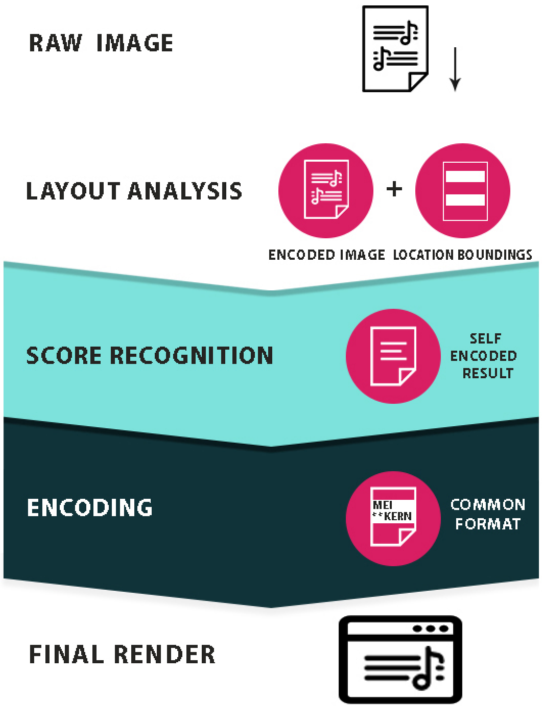


Fig. 3. ReadSco recognition pipeline from image upload to end-user visualization. Results produced by one system come as input data to the next one.

IV. CONCLUSIONS

The latest results in the OMR field make us optimistic about developing effective technology to solve the problem of automatically encoding the content of a music score image into a structured representation.

READSCO intends to establish itself as an open-source tool for which the community can be involved with the least possible effort. The main objective of our initiative is to provide a link between research and practice, as well as identifying new challenges and opportunities in this research field.

ACKNOWLEDGMENT

This work is supported by the Spanish Ministry HISPAMUS project TIN2017-86576-R, partially funded by the EU.

REFERENCES

[1] Jorge Calvo-Zaragoza, Jan Hajic Jr., and Alexander Pacha. Understanding optical music recognition. *Computer Research Repository*, abs/1908.03608, 2019.

[2] Antonio-Javier Gallego and Jorge Calvo-Zaragoza. Staff-line removal with selectional auto-encoders. *Expert Systems with Applications*, 89:138–148, 2017.

[3] Alexander Pacha and Horst Eidenberger. Towards a universal music symbol classifier. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 2, pages 35–36. IEEE, 2017.

[4] Jorge Calvo-Zaragoza and David Rizo. End-to-end neural optical music recognition of monophonic scores. *Applied Sciences*, 8(4), 2018.

[5] Arnau Baró, Pau Riba, Jorge Calvo-Zaragoza, and Alicia Fornés. From optical music recognition to handwritten music recognition: A baseline. *Pattern Recognition Letters*, 123:1–8, 2019.

[6] Alexander Pacha, Jorge Calvo-Zaragoza, and Jan Hajič jr. Learning notation graph construction for full-pipeline optical music recognition. In *20th International Society for Music Information Retrieval Conference (in press)*, 2019.

[7] Musitek. Smartscore x2. http://www.musitek.com/smartscore-pro.html, 2017.

[8] Neuratron. Photoscore 2018. http://www.neuratron.com/photoscore.htm, 2018.

[9] Hervé Bitteur. Audiveris. https://github.com/audiveris, 2004.

[10] Andrew Hankinson. Optical music recognition infrastructure for large-scale music document analysis. McGill University, 2015.

[11] David Rizo, Jorge Calvo-Zaragoza, and José M. Iñesta. Muret: A music recognition, encoding, and transcription tool. In *5th International Conference on Digital Libraries for Musicology*, pages 52–56, Paris, France, 2018. ACM.

[12] Nicola Dragoni, Ivan Lanese, Stephan Thordal Larsen, Manuel Mazzara, Ruslan Mustafin, and Larisa Safina. Microservices: How to make your application scale. In Alexander K. Petrenko and Andrei Voronkov, editors, *Perspectives of System Informatics*, pages 95–104, Cham, 2018. Springer International Publishing.

[13] Ionic framework official web. https://ionicframework.com/.

[14] Laurent Pugin, Rodolfo Zitellini, and Perry Roland. Verovio: A library for engraving mei music notation into svg. In *ISMIR*, pages 107–112, 2014.

[15] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2298–2304, 2016.

[16] Mara Alfaro-Contreras, Jorge Calvo-Zaragoza, and Jose M. Iñesta. Approaching end-to-end optical music recognition for homophonic scores. In *9th Iberian Conference on Pattern Recognition and Image Analysis*, 2019.

[17] Jorge Calvo-Zaragoza and David Rizo. Camera-PrIMuS: Neural End-to-End Optical Music Recognition on Realistic Monophonic Scores. In *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, September 23-27*, pages 248–255, 2018.

[18] Lorenzo Quirós, Alejandro Toselli, and Enrique Vidal. Multi-task layout analysis of handwritten musical scores. In *9th Iberian Conference on Pattern Recognition and Image Analysis*, 2019.

OMMR4all – a Semiautomatic Online Editor for Medieval Music Notations

Christoph Wick
Chair for AI and Applied Computer Science
University of Würzburg
Würzburg, Germany
christoph.wick@uni-wuerzburg.de

Frank Puppe
Chair for AI and Applied Computer Science
University of Würzburg
Würzburg, Germany
frank.puppe@uni-wuerzburg.de

***Abstract*—This paper presents *OMMR4all*, an online optical music recognition (OMR) and correction framework for Medieval neume notations. On the one hand it provides OMR algorithms to automatically capture staff lines, layout, and music symbols which use state-of-the-art Deep Learning models. On the other hand it provides a web application including an overlay editor to correct errors at any stage during the automatic processing. Since the notation styles between books can show a high variance, the default models provided by *OMMR4all* might not be well-suited for the actual material at hand. Therefore, new models can be trained based on manually corrected material to improve the automatic recognition of further pages. Experiments show, that only a few pages (about 5-10) are required to obtain a robust model, however an iterative training approach steadily improves the models by adding newly annotated scores. The goal of *OMMR4all* is to provide an easy to use tool targeting music scientists that aim to build up large-scale collections of encoded historical material by minimising the human effort.**

***Index Terms*—optical music recognition, web app, medieval manuscripts, neume notation, user interface**

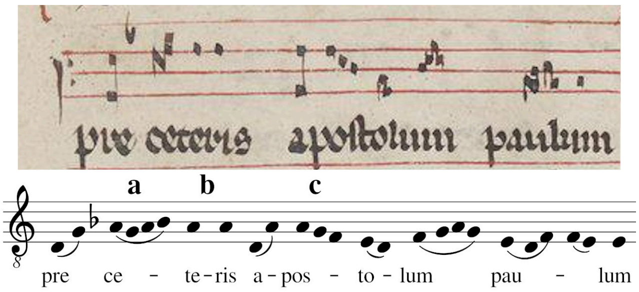


Fig. 1. Rendered example transcription which is encoded in MEI. A neume consisting of looped note components (NC) is visualised by a slur (a), each new neume starts with a large space (b), gapped NCs are notated with a small space (c).

I. Introduction

A still present desideratum for music research especially regarding historical manuscripts is a library storing machine-readable information, for example MEI, of the vast amount of available material. This digital form of music can then be used for large-scale music research such as similarity detection of melodies or comparisons of different version of the same piece of music. However, the encoding of the historical manuscripts is quite cumbersome because a lot of human effort is required. The current rise of artificial intelligence is a new hope to solve this task automatically, however, the current algorithms for optical music recognition (OMR) are not perfect which is why human knowledge and work is yet required for the transcription process. Therefore, the main goal of OMR is to minimise to human effort.

This paper presents our novel software Optical Medieval Music Recognition For All[1] (*OMMR4all*) which tackles the transcription of Medieval manuscripts that are written in different neume notations such as the square notation. *OMMR4all* implements a semi-automatic workflow starting from a single scanned page and outputs the encoded music for example as MEI. To process the music, we embed existing tools for OMR combined with an overlay editor to correct errors. Neumes are represented according to the current MEI standard (4.0.1) which stores syllables with their corresponding neumes that are denoted as single note components (NCs). Each NC stores its connection, gapped or looped, to its predecessor. An example transcript of a single line can be seen in Figure 1 which is manually rendered in a modern style. The aim of our software is to capture all information about the melody and its corresponding lyrics. Additionally, we store all positional information of each single symbol which can be used for a lookup in the original manuscript. This also allows to train new models for notation styles that are not known yet. The required ground truth can be created by utilising the overlay editor.

II. Related Work

In [5] Vigliensoni et al. present an OMR workflow embedded in the SIMSSA project [2] which targets Medieval and Renaissance music. The main stages are the analysis of the document, the reconstruction and encoding of its music, and finally the generation and correction of the score. Their document analysis relies on pixel-wise labelling which is automatically generated by the convolutional neuronal network presented in [1] and can manually be corrected by Pixel.js [4]. After a succeeding symbol classification based on the resulting layers, the music can be reconstructed by finding the pitch of neumes and the music is saved as MEI. Finally, a

[1]https://ommr4all.informatik.uni-wuerzburg.de

superimposition of the original image and the OMR are shown in the overlay editor Neon2 [3] which allows for a manual post-correction. Their workflow presents a promising approach of a semi-supervised OMR task on historical manuscripts. It shares many similarities with *OMMR4all* but also has its limitations. *OMMR4all* does not rely on a pixel-wise labelling of the original document which can be quite cumbersome in many cases if it is manually corrected. Also it cannot be guaranteed that a perfect labelling yields substantially better results in proceeding steps. In contrast, the output of our OMR approaches are polygons (e.g. staff lines) or individual symbols which can be easily corrected in the provided overlay editor. The idea of Neon.js is an overlay editor that allows humans to easily inspect differences of the OMR results and the underlied original image. Our editor picks up this idea but solves a fundamental shortcoming of the current version[2]: Neon.js expects straight staff lines with a fixed line distance for each staff, which is rarely the case in actual manuscripts. The major problem is that in various staves the overlayed staff lines mismatch partially more than one line which introduces high problems for an easy readability and comparison. The staff line detection algorithm of *OMMR4all* allows to draw exact staves whereby the staff lines and all of its symbols can be drawn at its actual positions in the manuscript.

III. OMMR4ALL

A. Workflow

The proposed workflow of *OMMR4all* is shown in Figure 2. The expected input is a high-resolution scan of a single page and the prepared lyrics. The lyrics must be written in a text file (e.g. in Word or a simple text editor) and use a "-" to separate syllables and "—" to denote staff breaks. Line breaks "\n" or multiple white-spaces can be used at will to enable a better readability of the lyrics in plain text.

The image processing starts with an image deskewing and binarisation, afterwards staff lines and staves are detected. The staves help to define and detect the layout in the next step. Then, the symbol detection to identify neumes or clefs is applied. Finally, the syllables of the prepared text are assigned to neumes and the encoding can be exported as MEI.

At each step of the workflow a human can interfere to guarantee a correct input for the subsequent steps. But also, after an application of the complete workflow all elements can be changed. This is enabled by the integrated overlay editor which will be briefly introduced in the next sections among a specification of the individual steps of the workflow.

1) Preprocessing: The expected input of the *OMMR4all*-workflow is a single sheet of music, a double page must be manually split. The preprocessing step applies OCRopus'[3] binarisation and deskewing algorithms to obtain an image with on average horizontally oriented staff lines and creates a gray-scale and binarised version of the deskewed page. These images serve as input for the automatic algorithms.

[2]https://ddmal.music.mcgill.ca/Neon/ (accessed August 2019)
[3]https://github.com/tmbdev/ocropy

2) Staff line detection: The next step is to identify staff lines and their corresponding staves. Here, we apply our algorithm presented in [6] which uses a Fully Convolutional Network (FCN) to identify pixels belonging to a staff line The pixels are then combined to a polyline which represents a single staff line. Based on their relative distances the detected staff lines are combined into staves and pruned by taking only staves that match the allowed number of staff lines per staff in the material at hand (e.g. four). In [6], we show that the algorithm detects approximately 99% of all staves on manuscripts written in square notation and is robust to new layouts.

3) Layout analysis: A further step is the layout analysis aiming to segment the input image into regions representing different types of text such as lyrics, or denoting the actual boundaries of a staff including all adjacent notes or clefs. For the current workflow, that targets solely the transcription of the music and its lyrics, an accurate layout analysis is optional since no algorithm currently expects exact boundaries of the respective elements. Instead, *OMMR4all* assumes bounding boxes each containing all staff lines of each respective single staff. The area between two staves defines the rough location for bounding boxes for lyrics. Despite the fact that an accurate layout analysis is optional, *OMMR4all* offers an automatic algorithm that can detect text (for example lyrics or page-numbers), music, or drop capital regions based on the relative coordinates of connected components to the detected staves. Depending on the material at hand this simple algorithm yields reliable results.

4) Symbol detection: The symbol detection which is based on another FCN (see [6]) acts on a single staff, and locates and identifies individual NCs and their connections, accidentals, or clefs. The FCN produces a pixel-wise label map for each symbol whose connected components represent single symbols. Our symbol detection presented in [6] shows that the transcription of a line yields an accuracy of about 87% on manuscripts written in square notation with most common errors being missing or additional notes. The current algorithm does not detect rare symbols (compared to notes or clefs) such as accidentals or liquescents which hence must be manually inserted.

5) Syllable assignment: The last step is to assign the prepared text to individual neumes. The automatic syllable assignment algorithm matches neumes and syllables one after the other in reading direction. Therefore, the algorithm is only successful if there is one neume per syllable. A completely new algorithm which includes Calamari [7] for an Optical Character Recognition (OCR) of the actual depicted text is currently in progress (see section V).

6) Iterative training approach: OMMR4all allows to train individual models used in the staff line and symbol detection to tackle a specific still unknown notation style of a book. Training requires a few pages of ground truth which has to be created based on similar models.

In general, the staff line models are expected to generalise well among different notation styles since lines are very similar across many notations. The symbol notations show a

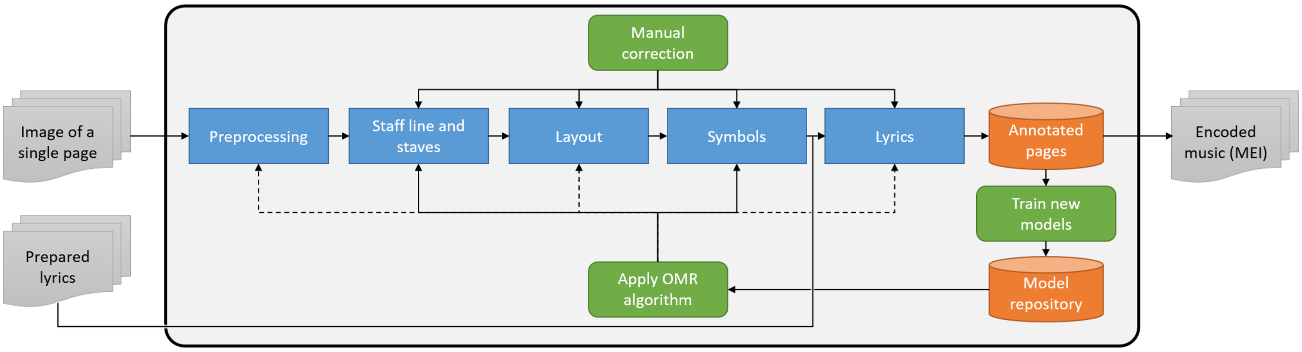


Fig. 2. The proposed workflow of OMMR4all. Documents serving as input or output are shown in gray. The steps of our workflow are shown in blue. Human (inter)actions are drawn in green. The orange elements show the storage for the annotated pages or the trained models. Dashed arrows indicate that these algorithm do currently not rely on a model from the repository, however they can be extended to use them in the future.

higher variance, for instance when comparing Gothic or square notations.

B. Software Architecture

OMMR4all[4] is an open source software which implements a client-server-model based on a REST API including user authentication. This allows for a low barrier for musicologists to our software in their research, because no installation is required, and the web app is fully platform independent. Furthermore, the heavy computational loads for training of new models are outsourced to a server that can host expensive GPUs for a reduction of processing time. Thus, a simple laptop or desktop PC is sufficient as an access-point to *OMMR4all*. Finally, the data is stored centralised which allows a word-wide access from any internet-ready working place.

1) Server: The backend server of *OMMR4all* is implemented in Python 3 running Django[5]. By default the server provides algorithms for the staff line, layout, symbol and text analysis as shown in section III-A. To extend existing or to add new algorithms the server allows an easy extension to other algorithms for these specific tasks. New algorithms can be written in Python using an existing algorithm API, but also an integration of tools or frameworks implemented in different languages is feasible.

2) Client: The client application is implemented in TypeScript using Angular[6]. The web-app is split into several views allowing to work and process a full book or a single page. The editor of a single page is an overlay editor providing tools to manually correct or create annotations. This tool is presented separately in the next section. The book view allows to run all algorithms in a fully automatic way for a complete book which is applicable if appropriate models are already available. Naturally, the client provides interfaces to create and upload new books, to download or export the annotations, or to manage user permissions if several users are working with the same material.

[4]https://github.com/ommr4all
[5]https://www.djangoproject.com/
[6]https://angular.io/

TABLE I
EVALUATION OF THE TRANSCRIPTION TIMES IN MINUTES. WE LIST THE NUMBER OF SYMBOLS, THE REQUIRED TIMES FOR CORRECTING THE STAFF LINES (SL.), SYMBOLS, SYLLABES, AND THE TOTAL TIME WHEN USING *OMMR4all* AND COMPUTE THE SPEED-UP (SU) COMPARED TO MONODI+. ALL VALUES ARE AVERAGES AND RELATIVE TO A PAGE.

Notation	#Symb.	*OMMR4all*				Monodi	SU
		SL.	Symb.	Syll.	Tot.		
Gothic	158	0.3	2.3	1.8	4.5	5.6	1.3
Square	267	0.6	3.3	2.9	6.9	8.5	1.2

3) Overlay-Editor: Since the automatic tools provided by *OMMR4all* are not expected to be perfect, their results must be manually corrected in an elegant and user friendly way. The integrated overlay editor tackles this task by providing a view that superimposes the annotations on the page image. The editor uses the exact positions of the staff lines or symbols to enable an easy-to-read way to detect and correct mistakes. Furthermore, the editor allows to move and edit the pasted syllables to the correct neumes.

The editor allows to create individual comments regarding for example disambiguities during the annotation process. These comments can then be integrated in a critical apparatus.

OMMR4all is designed to be easy to use without a steep learning curve because the ergonomic editor mainly relies on mouse interactions for selection, moving, dragging, or inserting musical symbols. More experienced users can use short cuts to speed-up the editing process.

IV. EVALUATION

To evaluate the transcription time, we compare *OMMR4all* to Monodi+[7] which is a sophisticated tool specifically designed to input plain chant via the keyboard. We chose five pages written in Gothic or square notation (a page is edited in Figure 3), respectively, and measured and averaged the times to obtain the transcript consisting of syllables and neumes. The models used for square notation were trained on 49 pages of ground

[7]Submitted to this workshop: Eipert et al., Editor Support for Digital Editions of Medieval Monophonic Music

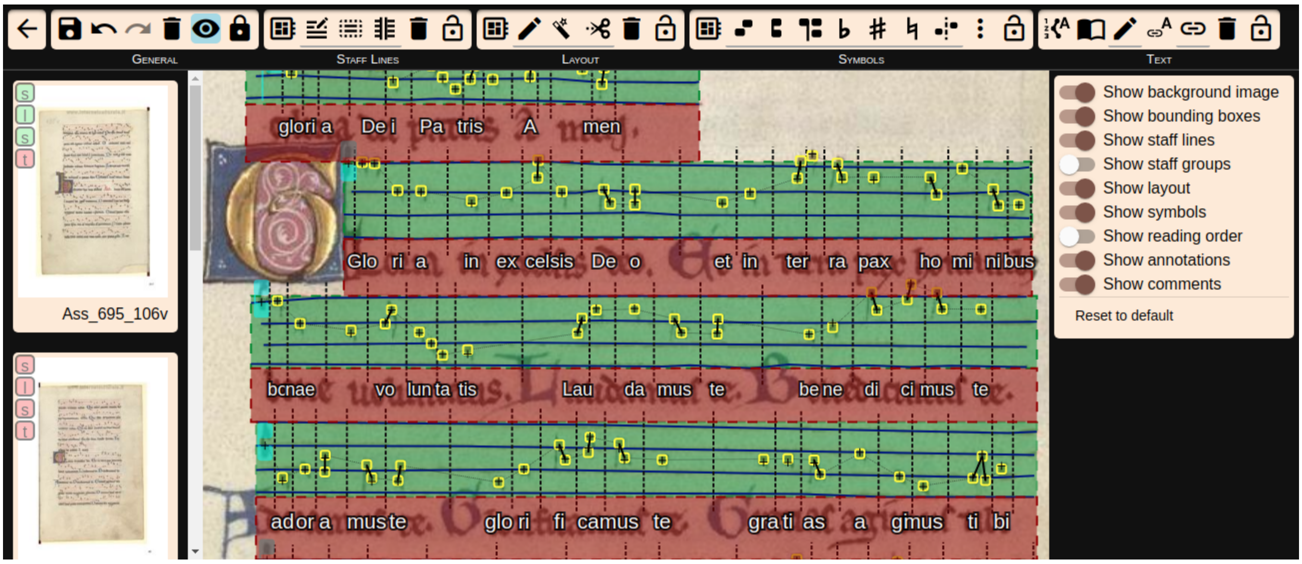


Fig. 3. Screen-shot of the overlay editor. In this example an simple layout is used: Staves (green), lyrics (red). Individual note components are rendered as yellow boxes, graphical connections of neumes are drawn as a solid line connecting two notes, while the dashed vertical lines indicate the start of a new neume. Clefs are drawn in cyan. The syllables of the lyrics are aligned below the corresponding neume within the respective text region. The various buttons of the tool bar define tools to correct the annotations or to launch the automated tools. Not shown are the reading order and comments.

truth chosen from a different book. To train the Gothic model, we selected another four pages of the same book. Both tools allowed to paste the prepared lyrics, therefore, only the NCs must be corrected or written. Table I shows that we achieved a speed-up of 1.3 and 1.2. The transcription time using Monodi+ in these experiments is already at its limit because every NC must be manually created. However, since the achieved accuracies of the symbol detection are still less than 90%, it can be expected that if more ground truth of the book at hand is available, working with *OMMR4all* will further reduce the transcription time. Furthermore, an automatic algorithm to minimise the effort to assign syllables (41% time, syll. / Tot. in Table I) is missing. Naturally, if the models can achieve human accuracy, books can be processed fully automatically. A principal difference of *OMMR4all* is, that its annotations yield an inherent explanation component for the origin of each symbol, which is very useful for example in a critical apparatus for borderline cases that need to be commented.

V. Future Work

Despite the many features of the *OMMR4all* framework there are many possible improvements or extensions. Some pending tasks will be presented in this section.

First, several tools and algorithms are planned to tackle the acquisition and encoding of text. The main problem is that currently no OCR engine can reliably deal with handwritten text of the targeted material without specific training. Therefore, in a first stage, we still rely on prepared text but we try to improve the automatic mapping of syllables to neumes by inclusion of the erroneous results of Calamari. Preliminary results showed that even if the OCR result of a lyrics line contains many mismatches of characters, the predicted character positions are mainly correct. Therefore, by aligning the actual syllables of the pasted text line with the OCR result, a rough estimation of the actual syllable position is feasible.

Other plans tackle further monophonic notation styles such as the later mensural notations or even older neume notations without staff lines. Hereby, the overlay editor requires only smaller cosmetic changes to store and display the directional notation, while new algorithms must be integrated or developed to capture the actual content automatically.

References

[1] Jorge Calvo-Zaragoza, Francisco Castellanos, Gabriel Vigliensoni, and Ichiro Fujinaga. Deep neural networks for document processing of music score images. *Applied Sciences*, 8(5):654, 2018.

[2] Ichiro Fujinaga, Andrew Hankinson, and Julie E. Cumming. Introduction to SIMSSA (Single Interface for Music Score Searching and Analysis). In *Proceedings of the 1st International Workshop on Digital Libraries for Musicology*, DLfM '14, pages 1–3, New York, NY, USA, 2014. ACM.

[3] Juliette Regimbal, Zoé McLennan, Gabriel Vigliensoni, Andrew Tran, and Ichiro Fujinaga. Neon2: A Verovio-based square-notation editor. In *Music Encoding Conference*, 2019.

[4] Zeyad Saleh, Ké Zhang, Jorge Calvo-Zaragoza, Gabriel Vigliensoni, and Ichiro Fujinaga. Pixel.js: Web-based pixel classification correction platform for ground truth creation. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 2, pages 39–40. IEEE, 2017.

[5] Gabriel Vigliensoni, Alex Daigle, Eric Lui, Jorge Calvo-Zaragoza, Juliette Regimbal, Minh Anh Nguyen, Noah Baxter, Zoe McLennan, and Ichiro Fujinaga. Overcoming the challenges of optical music recognition of early music with machine learning. *DH 2019*, 2019.

[6] Christoph Wick, Alexander Hartelt, and Frank Puppe. Staff, symbol and melody detection of medieval manuscripts written in square notation using deep fully convolutional networks. *Applied Sciences*, 9(13), 2019.

[7] Christoph Wick, Christian Reul, and Frank Puppe. Comparison of OCR accuracy on early printed books using the open source engines Calamari and OCRopus. *JLCL: Special Issue on Automatic Text and Layout Recognition*, 33(1):79–96, 2018.