HUMAN-DIRECTED OPTICAL MUSIC RECOGNITION

ABSTRACT

We propose a scheme for optical music recognition that puts the human in the loop. Starting from the results of our recognition engine, we pose the problem as one of constrained optimization, where the human can specify various pixel labels, while our recognition engine seeks an optimal explanation subject to the human-supplied constraints. In this way we enable an interactive approach, with a uniform communication channel from human to machine, where both iterate their roles until the desired end is achieved. Pixel constraints may be added to various stages, including staff finding, system identification, and measure recognition. Results on a test show show significant speed up when compared to purely human-driven correction.

1. INTRODUCTION

Optical Music Recognition (OMR) holds potential to transform score images into symbolic music libraries, thus enabling search, categorization, and retrieval by symbolic content — a cornerstone of ISMIR goals. Such symbolic libraries would serve as the foundation for the emerging field of computational musicology, and provide data for a wide variety of fusions between music, computer science, and statistics. Equally exciting are the commercial possibilities, such as the digital music stand, and systems that support practice and learning through objective analysis of rhythm and pitch.

In spite of this promise, as well as OMR's central place in the early years of ISMIR, progress has been slow. Even the best systems, which appear to be commercial, leave much to be desired [5]. In many cases the effort needed to correct OMR output may be more than that of entering the music data from scratch [3].

The reason for these disappointing results is simply that OMR is *hard*. Bainbridge [2] discusses some the challenges of OMR that impede its development. One central challenge is that music notation contains a large variety of *somewhat*-rare musical symbols and conventions [9], such as articulations, bowings, tremolos, fingerings, accents, harmonics, stops, repeat marks, 1st and 2nd endings, *dal segno* and *da capo* markings, trills, mordants, turns, breath marks, etc. While one can easily build recognizers that accommodate these unusual symbols and special notational cases, the false positive detections that result often outweigh the additional correct detections they produce. Under some circumstances, some not-so-rare symbols fall into this better-not-to-recognize category, such as augmentation dots, double sharps, and partial beams. Other difficulties stem from the kinds of image degradation often encountered, including poor or variable contrast, skew and warping of an image caused when the document is not aligned or flat in the scanner bed, hand-written marks, damage to pages, as well as other sources.

Some recent research has been dedicated to the improvement of fully automated OMR systems in post-process fasion, or other ways that leaves the core recognition engine intact. These efforts either tried to create systems that could adapt automatically [7, 16], or added musically meaningful constraints for recognition [11, 14], or combined multiple recognizers to achieve better accuracy [5, 12]. However, we have not yet reached our shared goal of creating large scale symbolic music databases with OMR. Hankinson *et al.* [10] provided a compelling model for *distributed* large-scale for OMR, however the approach still requires a large amount of careful proofreading and correction.

In light of these many obstacles and our collective past history, it seems unwise to bet on fully automated OMR systems that will produce high-quality results with any consistency. Instead we favor casting the problem as an *interactive* one, thus putting the human in the computational loop. In this case the essential challenge becomes one of minimizing the user's effort, putting as much burden as possible on the computer, (but no more). There are many creative ways to integrate a person into the recognition pipeline, allowing her to correct, give hints, or direct the computation. This work constitutes an effort in this direction.

Our first attempt to bring the human into OMR pipeline built a user interface allowing the correction of individual primitives: stem, beam, single flag, etc. Thus the user's task was simply to cover the image ink by adding and deleting appropriate primitives. A benefit of this approach is that it presents the user with a clearly-defined task that doesn't require knowledge of the system's inner workings. There are, however, several weaknesses to this approach: the human tagging process is laborious; it requires the person to precisely *register* the primitive with the image; and it allows the person to create uninterpretable configurations of primitives (say a stem with no note head) creating havoc further down the OMR pipeline. Our aim here is to improve on all these weaknesses while still presenting a simple task to the user.

Our current approach first presents the user with the original recognition results, obtained through fully automatic means. The user may then label any individual pixel according to the recognition task at hand. For instance, during system recognition the user may label a pixel as *white space* or *bar line*, while during measure recognition we use a richer collection of labels including, *closed note head*, *stem*, *ledger line*, *beam*, *sharp*, etc. The system then re-recognizes subject to the user-imposed constraint. Since our recognizers embed highly restrictive assumptions on the primitives they assemble, a single correction often fixes a number of problems at once. Human and machine then iterate the process of providing and synthesizing human-supplied constraints into recognized results.

This approach leaves the registration problem in the hands of the machine, where we believe it belongs. Furthermore, since our system can only recognize meaningful configurations of symbols, we avoid the problem of trying to assemble human-tagged composite symbols that may not make sense. While the resulting process may still be laborious, our results indicate that the human burden can be reduced considerably by employing this strategy. Furthermore, there are many other ways of introducing human-specified constraints into the recognition process, thus the current effort is something of a *proof of concept* for a longer-term goal.

2. INTERACTIVE OMR

Rebelo [15] suggested that an interactive OMR system could be a realistic solution to the problem, though the central challenge of fusing the human and machine contributions still remains open. Human-in-the-loop computation has received considerable attention recently [13]. It has been applied to a wide variety of areas, such as retrieval systems [17], object classification [4], character recognition [19], image labeling [18] and fined-grained visual categorization [8, 20]. The success of all these different applications is manifest in the statement in von Ahn [19]: “Human processing power can be harnessed to solve problems that computer cannot yet solve.”

There have already been several OMR systems taking into account human-in-the-loop computation. For instance, Fujinaga [9] proposed an adaptive system that could incrementally improve its symbol classifiers based on human feedback. Church [6] implemented an interface accepting user feedback to guide misrecognized measures toward similar correct measures found elsewhere in the score. Our system uses human feedback in an entirely different manner — as a means of *constraining* the recognition process in a user-specified manner, thus leveraging the user’s input in the heart of the system. It also constitutes a general framework for posing human-in-the-loop recognition as constrained optimization.

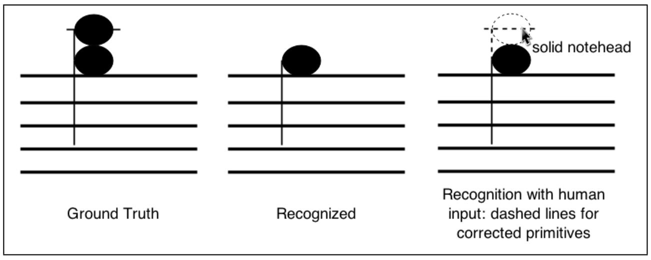

solid notehead
Ground Truth
Recognized
Recognition with human input: dashed lines for corrected primitives

Figure 1. Symbol re-recognition with user input

3. HUMAN-DIRECTED RECOGNITION

As motivation consider the example given in Figure 1. Suppose our recognition misses the upper note head of the chord. Then suppose the user labels a single pixel that belongs to the missing note head. When the system re-recognize subject to this constraint, the note head, its associated ledger line and its associated stem portion may all be recognized correctly, with the extra objects resulting from inherent constraints in the recognizer. We strive for results in which multiple recognition errors are fixed with a single piece of user input, thus making good use of the user’s time.

For all recognition components of our system, including staff finding, system identification, and symbol recognition, we formulate the essential tasks as optimization problems. Letting x denote a pixel location in the image, and $I(x)$ the grey level intensity at x, we have four types of probability models for these intensities indexed by $\mathcal{M} = \{b, w, t, n\}$. These correspond to pixels we believe to be *black*, *white transitional*, and *null* [14], with the probability models denoted by p_b, p_w, p_t, p_n. For instance, a solid note head could be modeled as an ellipsoidal region of pixels labeled *black*, surrounded by a *transitional* region, surrounded again by a region labeled as *white*. All other pixels in the image would be labeled as *null*.

For a possible image interpretation, H, we assign each image pixel to one of the four models through the function $M_H(x)$. For instance, the interpretation may be a particular configuration of staff lines tracked throughout the image, while $M_H(x)$ could label these lines as *black* with the remainder as *null*. We compute the score of a particular hypothesis as

$$S_H = \sum_x \log \frac{p_{M_H(x)}(I(x))}{p_n(I(x))} \qquad (1)$$

In theory, the sum extends over the entire image, though hypotheses generally label many pixels as *null*, in which case they only contributes 0’s to the sum of Eqn. 1. Our approach for all phases of recognition begins by optimizing S_H subject to the inherent grammatical constraints on the hypothesis, [14].

In formulating human-directed recognition, we allow the user to introduce various constraints by labeling individual pixels. For instance, the user may specify that a certain pixel must be labeled as a *staff line*, *bar line*, *note head*, *stem*, *sharp*, *beam*, etc. Thus, at any point in

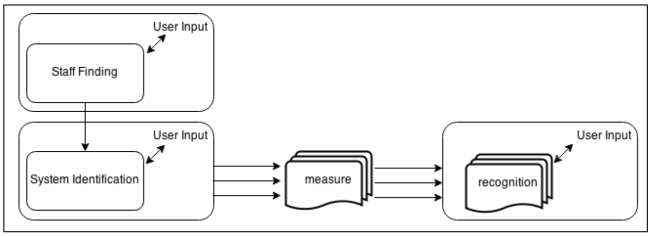


Figure 2. Workflow of human-directed OMR system

our interactive computation we have a collection of user-supplied constraints, $C = \{(x_i, l_i)\}$ for $i = 1, \ldots, n$, meaning that the user forces the pixel at x_i to be labeled as l_i. From these constraints we develop an additional term to our objective function

$$T_H = \sum_x t(x, P_H(x))$$

where $P_H(x)$ is the label of location x according to the hypothesis, H, and

$$t(x, P_H(x)) = \begin{cases} C & x = x_i, P_H(x) = l_i \text{ some } i \\ -C & x = x_i, P_H(x) \neq l_i \text{ some } i \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Thus the objective function gives a *bonus* of C whenever the user-specified constraint is satisfied, and a *penalty* of $-C$ if it is not. While the negative constraints (uses of $-C$) are redundant in theory, they allow us to compute hypothesis scores by summing Eqn. 1 only over the relevant (not *null*) pixels. Our constrained objective function is then

$$Q_H = S_H + T_H. \tag{3}$$

We choose C large enough so that only hypotheses respecting the constraints can be found by the recognition engine.

The work flow of our system is illustrated in Figure 2. As with most other OMR systems, we identify the staves and system structure first, thus segmenting the whole page into systems and measures, and then we recognize the music symbols in each measure. We've included human intervention in all three phases: staff finding, system identification, and symbol recognition.

3.1 Staff Finding and System Identification

Staff identification provides an illuminating example of our human-in-the-loop strategy. First, we summarize briefly the non-interactive version that finds the initial staves for the user.

The first phase of this basic algorithm chooses a collection of full-page-width overlapping image slices so that each staff line must be associated with at least one image slice that completely contains the staff and no other staves. For each such slice we track the vertical movement of the 5 parallel staff lines, allowing the vertical positions to vary gradually over the width of the image, assuming that such a staff exists (it may not). Each such trace can be described by a collection of *black* pixels that mark the staff position. We score each trace using Eqn. 1 applied only to these pixels using the *black* model and seek the optimal trace through DP. Similar algorithms are found throughout the OMR literature. An additional DP algorithm then seeks the optimal partition of the entire image into slices, where each slice can be labeled as either a staff line (scored under the data model above), or as blank space (not contributing to the score). We run the algorithm at a variety of different staff spacings and choose the optimally scoring configuration.

The important observation is that the permissible labelings of *black* pixels are highly constrained: these pixels occur in a structure of 5 parallel lines of known spacing whose height varies gradually if at all. Consider the case in which the algorithm fails to identify a staff line, for instance, by mistaking a flurry of ledger lines as a staff line. In such a case, a single hand-labeled pixel on a correct staff line position will constrain the recognition engine to find an global interpretation consistent with the constraint. Similarly, we may choose to label a rectangle containing a "false positive" staff line pixels as *white space*, thus creating a different type of constraint that achieves the same result. While errors of the staff finding algorithm are comparatively rare, this approach easily fixes the few errors we do observe, usually with a single iteration of labeling followed by re-recognition.

The case of system recognition proceeds similarly to staff finding. The primary feature identifying a system is that the staves in the system all share the same horizontal bar line position. Thus, bar line identification and the partition of staves into systems are inextricably linked. For this reason we estimate systems and bar lines *simultaneously*. We consider every collection of consecutive staves as a possible system, seeking the optimal configuration of bar lines for each. For a candidate system, we seek the best configuration of bar lines subject to a minimal separation constraint. We formulate this problem as optimizing Eqn. 1 using the candidate bar line locations, finding the globally best configuration using DP. As with staff finding, we label the bar line pixels as *black* and use only these sites in scoring a hypothesis. Having scored each candidate system, we can easily recover the optimal partition of staves into systems by choosing the partition of staves giving the maximal sum of data scores, again using DP.

Again we have a problem where the permissible configurations of *black* pixels are highly constrained. Here a single pixel hand-labeled as a bar line or a single rectangle labeled as *white space* will cause global changes to the recognized systems. An example will be presented in our experiments.

3.2 Measure Recognition

Measure recognition forms the heart of our system, in which we seek a collection of non-overlapping hypotheses that explain the contents of a measure. There are two phases to this process. First we begin by launching dedicated recognizers for beamed groups, isolated chords and notes, clef-key signature groups, whole notes, isolated symbols such as rests, text dynamics, slurs, and hairpin crescendos. These model-based recognizers look for grammatically-constrained hypotheses that restrict the shape, as with slurs and hairpins, or configuration of primitives, as with beam groups and isolated notes. Then we allow the identified hypotheses to *compete* for regions of overlap by "auctioning off" the contested regions. While the details are many, at the highest level measure recognition simply optimizes the data likelihood function of Eqn. 1 where H represents the collection of possible measure symbols and all their parameters — in essence all the information needed to *draw* the result.

The interactive phase of measure recognition begins by presenting the user with our recognizer's result for the measure, coloring the image with the recognized results to facilitate comparisons with the original. The user then labels single pixels or entire rectangles with descriptive tags such as *stem*, *solid note head*, *sharp*, *ledger line*, *slur* etc. Some of the labels may provide additional information, such as *3-beam* or *2-flag* to give the recognizer more precise information. The system then re-recognizes the measure subject to the constraints imposed by the user labels by optimizing Eqn. 3. taking into account the user-imposed constraints. User and computer alternate their contributions of supplying constraints and constrained recognition until the user is satisfied with the result.

Most of our recognizers work by first identifying possible candidate locations, then by employing the appropriate dedicated recognizer at the location. For example, beamed groups, isolated notes and chords, slurs, and hairpins all work this way. The interactive interface allows the user to add and delete candidates as well, thus ensuring that all necessary candidates are present, and that the recognition is not burdened by false positive candidates. The latter lead to unnecessary computation, but more importantly, may produce unwanted measure symbols that "win out" over the ones we seek. Thus editing the candidates gives the user a way to nip these unwanted results in the bud. In the iterative process the user may edit candidates and label pixels in any order desired.

4. EXPERIMENTS

As staff finding errors are comparatively rare, we focus our experiments on the remaining two recognition steps of system identification and symbol recognition.

Our system identification test set consists of a collection of 55 pages coming from 20 randomly selected IMSLP [1] scores (16 out of them don't have any system or bar line errors), as described in Table 1. In system identification we simultaneously group the staves into systems and recognize bar lines for each system. The partition of staves into systems is equivalent to a binary decision for each of the "gaps" between staffs, identifying whether or not the system continues through the gap. Thus when there are n staves on a page there are $n - 1$ such binary decisions, hence $n - 1$ possible errors. It doesn't seem possible to evaluate bar line errors meaningfully unless the containing system is correctly identified. For this reason we divided the process into two phases: first we take user input to correct the systems; once these are correct we allow further input to correct the bar lines. We only count the bar lines errors from the point where the systems are correct, and do so by including both false negative and false positive bar lines as single errors.

For system identification the user is first presented with the results from the fully automatic recognition process, then allowed to correct interactively by labeling individual pixels or rectangular regions as *bar line* or *white space*. After each such user action our algorithm re-recognizes subject to the new constraint as well as past constraints. Table 1 tallies the results of this process. The table indicates, for instance, that we were able to correct the 10 system errors with only 4 user actions, while the 192 remaining bar line errors were corrected with 133 actions. In both cases we arrive at the desired result with significantly less effort than hand correction of each change, partly because we employ fewer actions, but also because the actions require less of the user.

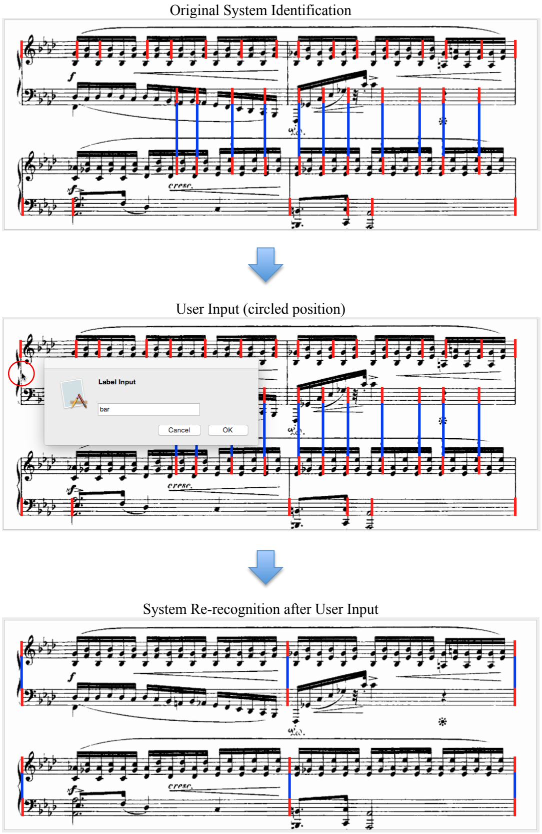
Figure 3 shows an illuminating example of how the system works. In the top panel we see an incorrectly recognized system with a collection of mostly incorrect bar lines shown in red with gaps between bar lines of a system shown in blue. Thus the system interprets this portion of the image as three systems, the first and last containing only a single staff, while the middle system constains two staves. The middle panel shows the user identifying a single pixel as "bar line", which allows the system to fix a large number of related errors, as shown in the bottom panel.

For measure recognition we tested on the 3rd movement, *Notturno*, from *Borodin's Second String Quartet*. For this movement we have hand-labeled ground truth consisting of notation primitives. The ground truth was created beginning with fully automatic recognition and then laboriously adding and deleting various primitives. Our system performs symbol recognition at the staff measure level. In keeping with this, we also evaluate at the staff measure level by counting (automatically) the number of false positive and false negative primitives. In doing so we only count errors for the types of primitives that our system tries to identify, not including, for instance, "wrong side" note heads and grace notes.

In the user correction phase we iterate between accepting a human-supplied pixel label and re-recognizing the measure subject to all current constraints. There are 36 possible labels the user can assign to a pixel including the 3 types of note head (whole, half, and solid), 3 types of flag (1 flag, 2 flags , 3 flags), stems, sharp, flat, double sharp, ledger line, augmentation dots, staccato, accent, slur, hairpin (cresc. and dim.), etc., and, of course, white space. We assume that the user tries to minimize the amount of work in achieving the results, as was the case with the particular user who performed these tests (the first author). More specifically, certain types of user labels, such as beams and stems, often fix a number of problems at once due to the highly constrained relations between beamed group and

IMSLP ID	Pages	System Errors	System Corrections	Bar line Errors	Bar line Corrections
11741	21	0	0	77	49
86550	12	1	1	27	17
113998	10	9	3	19	15
114193	12	0	0	69	52
total	55	10	**4**	192	**133**

Table 1. Evaluation of System Identification with user input


Original System Identification

User Input (circled position)

System Re-recognition after User Input

Figure 3. System Identification with user input (experiment on the sixth page of Chopin's Nocturnes, Op.15)

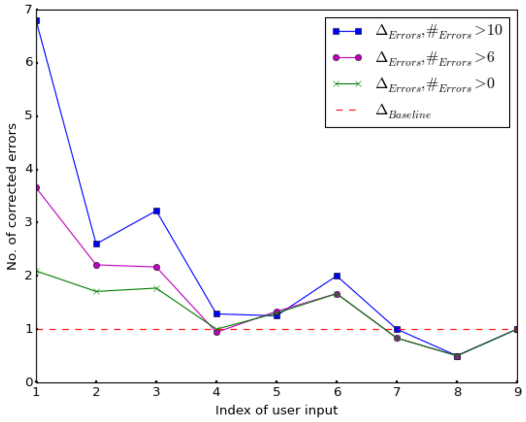


Figure 4. Error decrease at the i-th input

Measures	Primitive Errors	Corrections	Remaining Errors
119	423	235	25

Table 2. Evaluation of Measure Recognition with user input

chord primitives. Thus we assume our users will add these labels first, when needed.

Figure 4 shows us the average number of corrections made by the i-th ($i = 1, 2, \ldots, 9$) user input. The dotted line in the figure at 1 represents our baseline, since if primitives were corrected one-by-one there would be 1 user action for each corrected error. The figure separates the measures by the number of errors incurred in the initial recognition. From these graphs one can see that the benefit for our approach is greatest in the hard measures (with many errors), which is where the majority of the user's time is spent. The figure also shows that in each category the first several user actions give the most benefit while the incremental benefit of the later user actions decreases. Table 2 aggregates over the entire experiment showing a total of 423 corrections resulting from 235 clicks. 25 errors remain uncorrected through this process through various issues with our recognition engine. We give a view of the way the process evolves over a measure in the website below [1].

While these experiments show that this approach decreases the user's correction effort considerably, there are

[1] www.googledrive.com/host/0B5VHUijvJ3tmSUE1OWVVYVoyYjg

additional benefits. For one, the tedious and sensitive process of registration is relegated to the computer, shifting this burden away from the user and resulting in greater accuracy. In addition, we are guaranteed that all resulting composite structures (beamed groups, chords, clef-key-signature structures) are composed of meaningful configurations of primitives.

5. DISCUSSION

Given the enormous challenge of fully automatic optical music recognition, it seems reasonable to cast the problem in terms of human-in-the-loop computing, where both machine and person contribute what they do best. We have proposed a simple template for all human input — the labeling of individual or collections of pixels — thus providing a uniform format for providing information to the system, while not requiring any knowledge of the system's inner workings. The evaluation shows that the system improves significantly on one-by-one correction of primitives, while relieving the person of the difficult registration task, and guaranteeing that the eventual results are grammatically meaningful.

We see this as the first step in a move toward user-directed recognition, with several interesting and unexplored variations that are already apparent. The current work involves making constraints on the *image labeling*, though another interesting class of constraints could be placed in the recognition *models* themselves. One simple example involves the many objects that orbit around note heads, such as articulations, accidentals, and augmentation dots. For a given measure, beamed group, or note, any combination of these could be "switched" on or off, thus constraining the search space of the recognition process.

Equally promising is allowing the user to edit the model used for a particular note, beamed group, or measure, requiring some number of notes in a chord, some number of chords in a beamed group, or some collection of symbols in a measure. As we further constrain our model in this way the problem begins to resemble the *registration* problem, rather than *recognition*. We envision allowing the user access to this entire continuum, though specifying only what is needed to get the correct result.

This work treats the measure as the unit of analysis. This makes sense given the current state of our system which also uses the measure in this way, though this probably isn't the right unit for human-directed recognition. It occasionally takes 30 seconds or so to recognize a complicated measure, which is not consistent with the interactive needs of the system we envision. A better idea would be to allow the user to build up the final measure, symbol-by-symbol, treating past symbols as regions that cannot be violated. This avoids the time-consuming phase of our system that must find non-overlapping variants of the original hypotheses, and would allow faster response to the user's requests.

Finally, it is worth noting that we have proposed a general paradigm for human-in-the-loop computing, as one of constrained optimization that may apply to a wide variety of recognition problems that require human assistance.

6. REFERENCES

[1] IMSLP/Petrucci Music Library. `http://imslp.org`.

[2] David Bainbridge and Tim Bell. The challenge of optical music recognition. *Computers and the Humanities*, 35(2):95–121, 2001.

[3] Pierfrancesco Bellini, Ivan Bruno, and Paolo Nesi. Assessing optical music recognition tools. *Computer Music Journal*, 31(1):68–93, 2007.

[4] Steve Branson, Catherine Wah, Florian Schroff, Boris Babenko, Peter Welinder, Pietro Perona, and Serge Belongie. Visual recognition with humans in the loop. In *Computer Vision–ECCV 2010*, pages 438–451. Springer, 2010.

[5] Donald Byrd and Megan Schindele. Prospects for improving omr with multiple recognizers. In *ISMIR*, pages 41–46, 2006.

[6] Maura Church and Michael Scott Cuthbert. Improving rhythmic transcriptions via probability models applied post-omr. In *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014*, pages 643–648, 2014.

[7] Michael Droettboom, Karl Macmillan, and Ichiro Fujinaga. The gamera framework for building custom recognition systems. In *Proceedings of the Symposium on Document Image Understanding Technologies*, pages 275–286, 2003.

[8] Kun Duan, Devi Parikh, David Crandall, and Kristen Grauman. Discovering localized attributes for fine-grained recognition. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3474–3481. IEEE, 2012.

[9] Ichiro Fujinaga. *Adaptive optical music recognition*. PhD thesis, McGill University Montréal, Canada, 1996.

[10] Andrew Hankinson, John Ashley Burgoyne, Gabriel Vigliensoni, and Ichiro Fujinaga. Creating a large-scale searchable digital collection from printed music materials. In *Proceedings of the 21st international conference companion on World Wide Web*, pages 903–908. ACM, 2012.

[11] Rong Jin and Christopher Raphael. Interpreting rhythm in optical music recognition. In *ISMIR*, pages 151–156. Citeseer, 2012.

[12] Ian Knopke and Donald Byrd. Towards musicdiff: A foundation for improved optical music recognition using multiple recognizers. *dynamics*, 85(165):121, 2007.

[13] Alexander J Quinn and Benjamin B Bederson. Human computation: a survey and taxonomy of a growing field. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1403–1412. ACM, 2011.

[14] Christopher Raphael and Jingya Wang. New approaches to optical music recognition. In *ISMIR*, pages 305–310, 2011.

[15] Ana Rebelo, Ichiro Fujinaga, Filipe Paszkiewicz, Andre RS Marcal, Carlos Guedes, and Jaime S Cardoso. Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval*, 1(3):173–190, 2012.

[16] Florence Rossant and Isabelle Bloch. Robust and adaptive omr system including fuzzy modeling, fusion of musical rules, and possible error detection. *EURASIP Journal on Applied Signal Processing*, 2007(1):160–160, 2007.

[17] Chi-Ren Shyu, Carla E Brodley, Avinash C Kak, Akio Kosaka, Alex M Aisen, and Lynn S Broderick. Assert: A physician-in-the-loop content-based retrieval system for hrct image databases. *Computer Vision and Image Understanding*, 75(1):111–132, 1999.

[18] Luis von Ahn and Laura Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326. ACM, 2004.

[19] Luis von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. recaptcha: Human-based character recognition via web security measureschurchimproving. *Science*, 321(5895):1465–1468, 2008.

[20] Catherine Wah, Steve Branson, Pietro Perona, and Serge Belongie. Multiclass recognition and part localization with humans in the loop. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2524–2531. IEEE, 2011.