Human-Directed Optical Music Recognition

Liang Chen; Indiana University Bloomington; Bloomington, USA
Christopher Raphael; Indiana University Bloomington; Bloomington, USA

Abstract

We propose a human-in-the-loop scheme for optical music recognition. Starting from the results of our recognition engine, we pose the problem as one of constrained optimization, in which 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 significant speed up when compared to purely human-driven correction.

Introduction

Optical Music Recognition (OMR) holds potential to transform score images into symbolic music libraries, thus enabling search, categorization, and retrieval by symbolic content, as we now take for granted with text. 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 applications 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, progress in OMR has been slow; even the best systems, both commercial and academic, leave much to be desired[7]. In many cases the effort needed to correct OMR output may be more than that of entering the music data from scratch[8]. In such cases OMR systems fail to make any meaningful contribution at all.

The reason for these disappointing results is simply that OMR is *hard*. Bainbridge [17] discusses some the challenges of OMR that impede its development. One central problem is that music notation contains a large variety of *somewhat*-rare musical symbols and conventions [4], 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 somewhat-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.

Another issue arises from the difficulty in describing the high-level structure of music notation. Objects such as chords, beamed groups, and clef-key-signatures, are highly structured and lend themselves naturally to grammatical representation, however, the overall organization of symbols within a measure is far less constrained. The OMR literature contains several efforts to formulate a *unified* grammar for music notation [10, 11]. These approaches represent grammars of *primitive* symbols (beams, flags, note heads, stems, etc.) and begin by assuming a collection of segmented primitives. While our grammars have significant overlap with these approaches, one of our primary uses for the grammar is the segmentation of the symbols into primitives — we do not believe it is realistic to identify the primitives without understanding the larger structures that contain them. Kopec [12] describes a compelling Markov Source Model for music recognition that simultaneously segments and recognizes. However, the approach addresses a small subset of music notation and does not generalize in any obvious way. In particular, our primary focus is on the International Music Score Library Project (IMSLP), while Kopec's model covers a small minority of the examples encountered there.

Other difficulties stem from the kinds of image degradation 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 fashion, or other ways that leave the core recognition engine intact. These efforts either create systems that adapt automatically [16, 24], add musically meaningful constraints for recognition [1, 5], or combine multiple recognizers to achieve better accuracy [9, 7]. However, OMR research is still a long way from our shared goal of creating large scale symbolic music databases. Hankinson *et al.* [15] created a prototype system for *distributed* large-scale OMR, which converts a collection of Gregorian chant scores into symbolic files to facilitate their *in situ* content-based retrieval, though 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, note head, single flag, sharp, augmentation dot, 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, how-

ever, several weaknesses to this approach: the human tagging process is laborious; it fails to provide important syntactic relations between primitives; 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/half/whole note head*, *stem*, *ledger line*, *beam*, *sharp*, *single flag*, 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 — the precise location of primitives — 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 constitutes an initial exploration of a longer-term goal.

Interactive OMR

Various authors, such as Rebelo [13], suggest that 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 [23]. It has been applied to a wide variety of areas, such as retrieval systems [19], object classification [20], character recognition [18], document indexing [25], image labeling [22] and fined-grained visual categorization [21]. Romero [26] proposed a Hidden Markov Model (HMM) for computer-assisted text transcription, in which the user-imposed prefix is used to constrain both the sequence decoding and language priors. The potential of all these different applications is summarized in von Ahn's statement [18]: "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 [4] 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 is worth noting that our approach constitutes a generic framework that poses human-in-the-loop recognition as constrained optimization, applicable beyond the specific confines of OMR.

Human-Directed Recognition

As motivation consider the example given in Figure 1. Suppose our recognition misses the upper note head of the chord (Figure 1b). Then suppose the user labels a single pixel that belongs to the missing note head as *solid head* (Figure 1c). When the system re-recognizes subject to this constraint, the note head, its associated ledger line, accidental and 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.

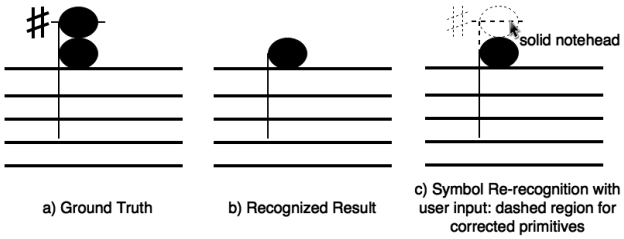

Figure 1: Symbol re-recognition with user input

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 $\mathscr{M} = \{b, w, t, n\}$. These correspond to pixels we believe to be *black*, *white transitional*, and *null*[1], 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

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

In theory, the sum extends over the entire image, though hypotheses generally label many pixels as *null* which only contributes 0's to the sum of Eqn. 1, thus we confine the sum to the region of pixels not labeled as *null*. Our approach for all phases of recognition begins by optimizing S_H subject to the inherent grammatical constraints on the hypothesis [1].

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 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

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

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. Our constrained objective function is then

$$Q_H = S_H + T_H. \quad (3)$$

We choose C large enough so that satisfying the constraints is the first priority of the recognizer, essentially disallowing hypotheses that do not respect the constraints. The work flow of our system is illustrated in Figure 2. At the highest level the user is required to follow a specific order of three sequential steps:

1. The system begins with staff finding. Here the user must correct any misrecognized staves before moving on the the next step, though they are rare.
2. Systems, subsystems, and bar lines are recognized. Again the user must correct errors resulting from this phase before moving on.
3. The contents of the measures are recognized, one by one. The basic process begins with the automatically recognized results. Then user and machine iterate between offering human-supplied constraints and automatic re-recognition subject to these constraints. The user creates constraints by clicking on individual pixels and labeling the pixels with an appropriate tag. The recognized results will be automatically saved.

Users correct errors by clicking on any incorrect recognized pixel and giving the desired label for the location. The system accepts these locations and labels and use them as constraints in the recognition process.

Staff Finding

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

The first phase of our staff finding 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 as in Figure 3. For each such slice we track the height 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 in some slices). 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. A complementary DP algorithm then seeks the optimal partition of the vertical dimension of the image into slices, where each slice can be labeled as either a staff line (scored under the data model above), or as blank space (scored under the null model). 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 non-intersecting structures of 5 parallel lines of fixed spacing whose height varies gradually if at all. Consider the case in which our 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 a 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.

We will reuse this basic channel of information flow between human and computer in other aspects of our system. A virtue of this approach is that the user employs a uniform method of communication regardless of the intricacies of the recognition algorithms.

System Identification

In a musical score staves are grouped in *systems* that identify staves that are played at the same time. The primary feature identifying a system is that the staves in the system share the same horizontal bar line positions, optionally extending these bar lines between the staves. 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. Thus, if there are N staves there are $\frac{N(N-1)}{2}$ possible systems to be considered. For a candidate system, we seek the best configuration of bar lines subject to a minimal separation constraint between bar lines. 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*, all all other sites are labeled *null*.

Having scored each candidate system we seek the best partition of the staves into systems, where the score of a particular grouping into staves into systems is the sum of the data model scores for each system. This optimal partition is easily identified with DP, once again nesting one DP problem inside another.

Again we have a problem where the permissible configurations of *black* pixels are highly constrained since each system must place the bar lines in identical horizontal positions. The algorithm has an incentive for grouping staves together, even though this decreases the degrees of freedom in placing bar lines, since this allows the algorithm to correctly explain bar lines that extend between adjacent staves, as they often do. For system identification a single pixel hand-labeled as a bar line or a single rectangle labeled as *white space* causes global changes to the recognized systems. An example will be presented in our experiments.

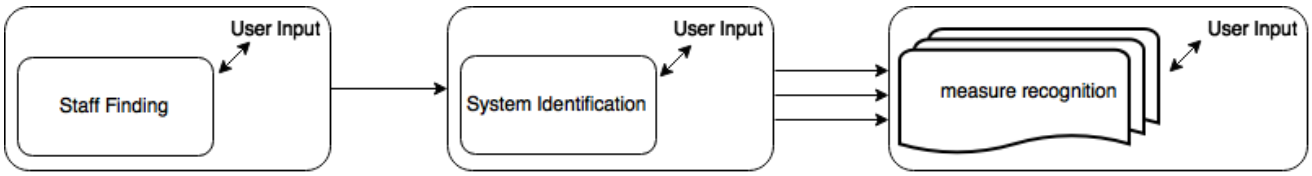

Figure 2: Workflow of human-directed OMR system

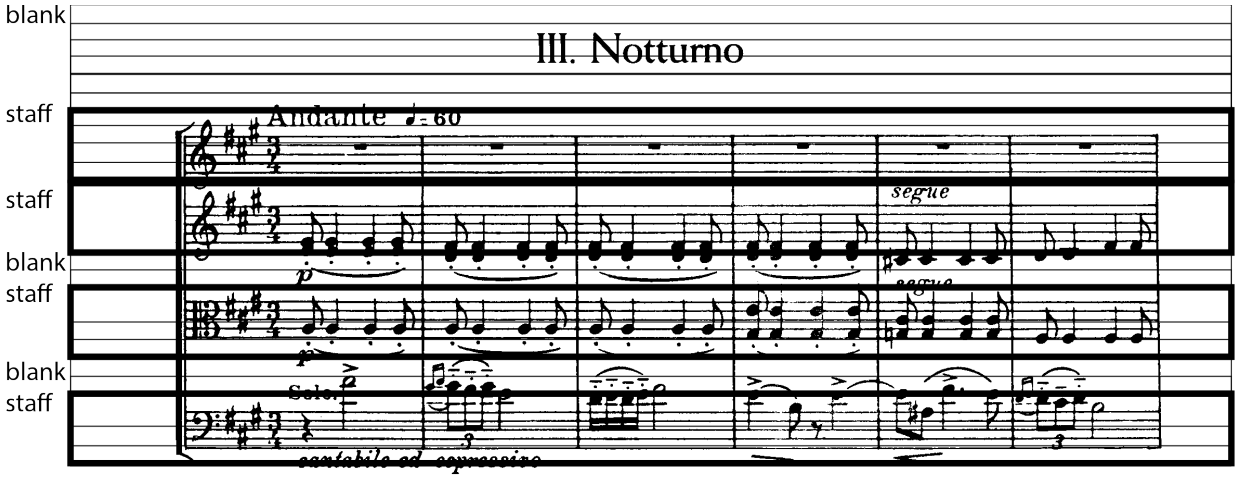

Figure 3: Staff-line recognition. The top-down sliding windows contain either **no** staff or a single staff. We simultaneously decode the page staff structure and recognize the precise location of staves in the staff-containing windows.

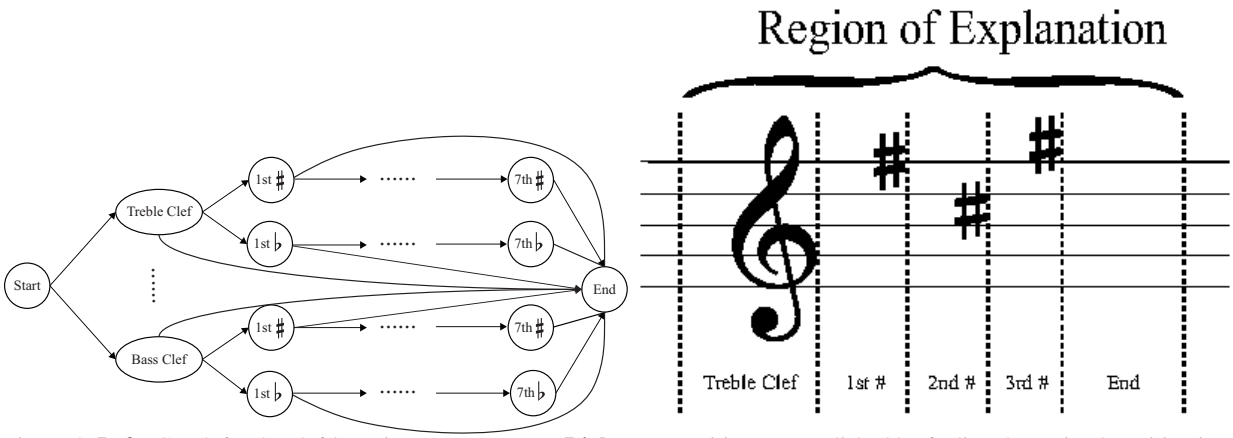

Figure 4: **Left:** Graph for the clef-key-signature structure. **Right:** Recognition accomplished by finding the optimal partition into labeled regions where the labeling corresponds to a path through the graph.

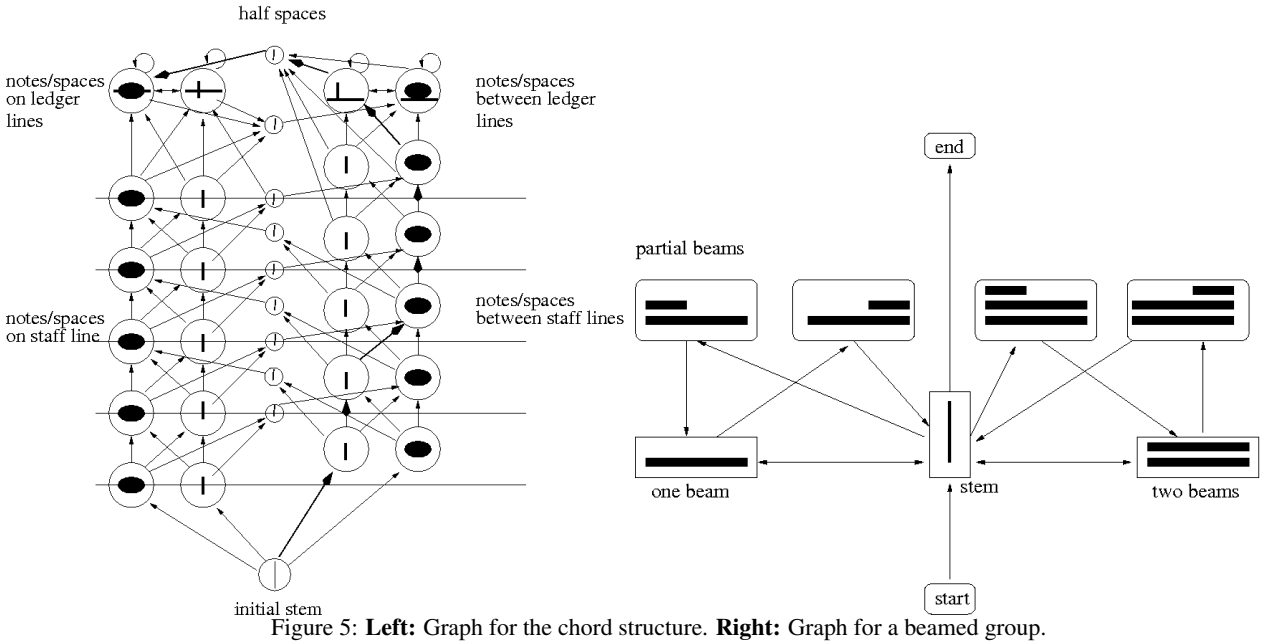

Figure 5: **Left:** Graph for the chord structure. **Right:** Graph for a beamed group.

Measure-Level Symbol Recognition

Measure recognition forms the heart of our system. In this process where we seek a collection of non-overlapping symbol hypotheses that explain the contents of a staff measure. There are two phases to this process: finding potential symbols and resolving the overlap between these symbols.

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. In all cases except the isolated symbols, we employ model-based recognizers that look for grammatically-constrained hypotheses that restrict the shape, as with slurs and hairpins, or restrict the possible configuration of primitives, as with beam groups and isolated notes. For instance, the monotonic nature of the tangent angle in a slur provides a grammatical constraint on the (height,angle) "states" that can be used in recognition. A simple grammatical example can be seen in Figure 4 which describes a model for the clef-key-signature complex found at the left edge of a staff. The graph in the left panel of the figure expresses the notion that the structure begins with one of several possible clefs, followed by up to 7 possible sharps or flats, whose vertical positions are known given the clef — each legitimate interpretation corresponds to a path through this graph. The right panel of the figure shows the goal of our recognition: we seek a labeled partition of the target region in which the segment labels come from a path through the graph. Given a partition and a labeling, the positions of the clef and accidentals are determined, thus we can compute the data score, Eqn. 1, for this interpretation. We compute globally optimally scoring interpretation (partition and labeling) using DP. This basic recognition paradigm is a common one found throughout the document recognition literature and elsewhere.

This essential recognition approach is generic and is used for various other symbol recognizers. For instance, the left panel of Figure 5 describes a simplification of the grammar we use for isolated chords, involving constrained configurations of stems, note heads, and ledger lines. Similarly, the right panel of the same figure shows are grammar for beamed groups. In both of these cases we create regular grammars that, in essence, draw all possible configurations of the primitives and seek the globally best explanation of the data. In the case of the chord grammar the partition of the image is along the vertical dimension, while for the beamed group it is along the direction of the beams.

After we identify our possible hypotheses we allow them to *compete* for overlapping regions by "auctioning off" the contested image portions. We do this by allocating the contested region to both objects, recognizing the other subject to this allocation constraint, taking the best scoring joint (and non-overlapping) interpretation. These processes are described in more detail in [2].

The interactive portion of measure recognition begins by presenting the user with our recognizer's result for the measure, superimposing the recognized objects in color over 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 entire measure subject to the user-supplied constraints. The constraints are easy to impose in the individual recognizers simply by adding a bonus when a constrained pixels is labeled correctly, or imposing the bonus if not. User and computer alternate their contributions of supplying constraints and constrained recognition until the user is satisfied with the result.

An illustrative example is shown in Figure 6. The original incorrect interpretation provided by the recognizer is shown using the dashed path. After the user clicks on the 4th flat and labels the pixel as *flat*, the constrained recognizer finds the bold path which correctly interprets the clef-key-signature complex.

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 in-

teractive 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.

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 55 pages coming from 20 randomly selected IMSLP [3] scores (16 of the 20 don't have any system or bar line errors), as described in Table 1. As discussed in Section 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 making 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 $n-1$ pos-

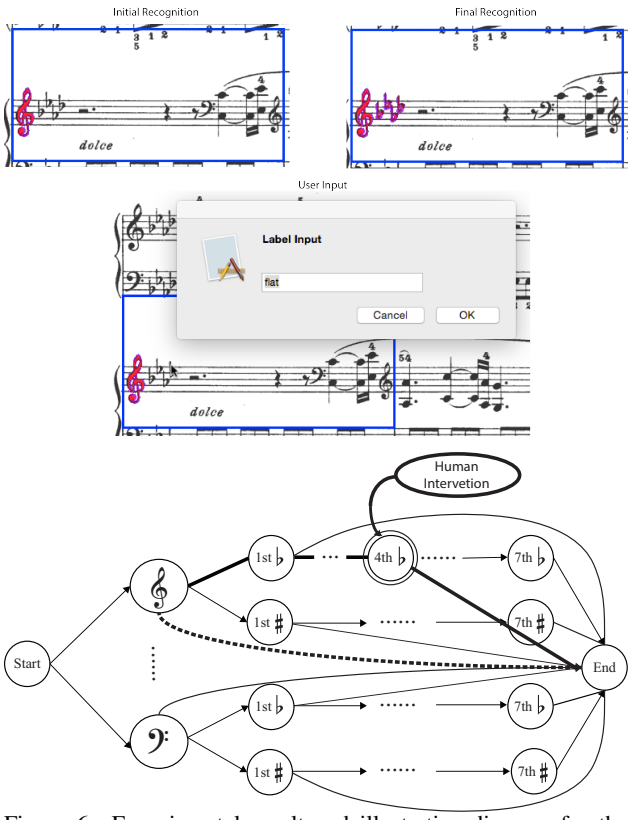

Figure 6: Experimental result and illustrative diagram for the user-supplied constraint used in recognition. **Top Left:** Initial recognition without any user-supplied constraint; **Top Right:** Final recognition with a single user-supplied constraint; **Mid:** User place a label on the last flat key; **Bot:** Explanation of the human-directed symbol recognition: dashed path for the original recognition and bold path for the optimal one subject to user-supplied constraint.

sible errors. It doesn't seem possible to evaluate bar line errors meaningfully unless the containing system is correctly identified (see the top panel of Figure 7). 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. With both systems and bar lines we see that the constraints in our models allow the process to fix errors that were not explicitly identified.

Figure 7 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 contains two staves. According to our error tally, this counts as 3 system gap errors. The middle panel shows the user identifying a single pixel, circled in red, as "bar line", thus allowing the algorithm 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*. Figure 8 presents one example of how human-directed symbol recognition works. The upper-left panel shows the original measure, while the upper-middle panel shows our system's initial recognition of the chord which misses a half note head as well as the associated natural sign and augmentation dot. The user clicks on a single pixel, as in the bottom panel, identifying the pixel as part of a half note head using the *open* tag. Using this information the system is able to correctly identify the note head as well as its associated accidental and augmentation dot. While not represented in the figure, the grammatical relations between all of these symbols are understood as well. We created hand-labeled ground truth consisting of notation primitives for this movement by using fully automatic recognition and then laboriously adding and deleting various primitives. Needless to say, such ground truth requires a great deal of effort to create, and is difficult to do at a large scale. Since our system performs symbol recognition at the staff measure level, we also evaluate at the staff measure level by counting (automatically) the number of false positive and false negative primitives for each such measure. 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 sub-

Table 1: Evaluation of System Identification with user input

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**

ject to all current constraints. There are 36 possible labels the user can assign to a pixel at present, 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 these experiments. 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 chord primitives. Thus we assume our users will add these labels first, when needed.

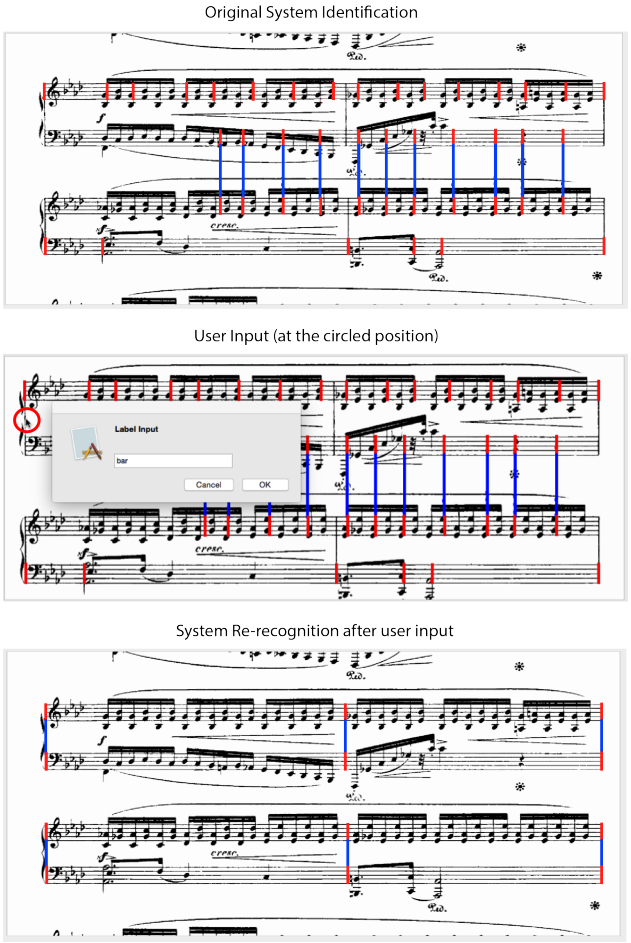

Figure 7: System Identification with user input (experiment on the sixth page of Chopin's Nocturnes, Op.15). **Top:** The original recognition. **Middle:** User places identifies a single bar line pixel circled in red. **Bot:** The result subject to the user-specified constraint.

Table 2: Evaluation of Measure Recognition with user input

Measures	Primitive Errors	Corrections	Remaining Errors
119	423	235	25

Figure 9 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 height 1 represents our baseline, since if primitives were corrected one-by-one there would be 1 user action for each corrected error. The figure partitions 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 ben-

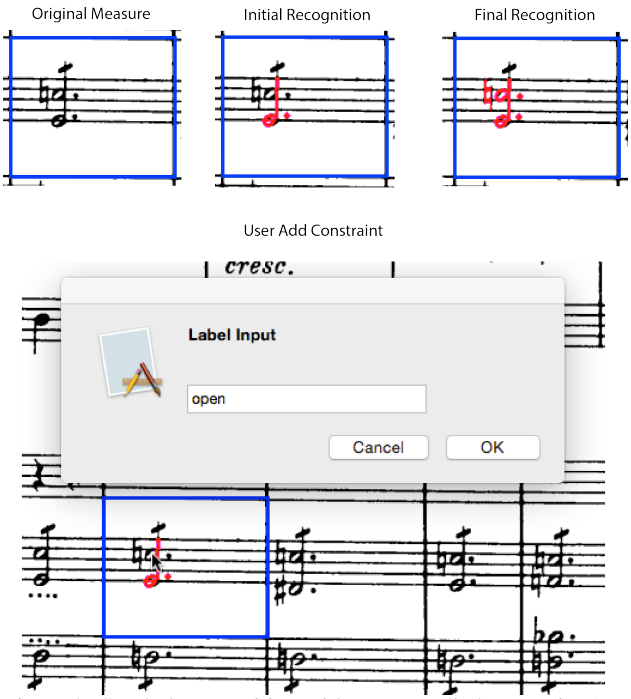

Figure 8: Symbol Recognition with user-supplied constraint (experiment on the third page of the third movement, Notturno, of the second Borodin String Quartet). **Top Left:** The original measure. **Top Middle:** The initial recognized result. **Top Right:** The second recognized result with one extra user input. **Bot:** The place where user added constraint.

efit 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 limitations of recognition engine. For instance, the current parameter settings may make it impossible to recognize certain configurations. We show the evolving recognition process over a measure in the website below[1].

Our symbol representations have tree-like structures. For example, any number of stems may belong to a beam, while a several note heads may belong to the stem, while accidentals and other modifying symbols may belong to the note head. When the user corrects one of the leaf nodes of a tree, the benefit is restricted to the symbol in question since there are no symbols "downstream" to benefit from the added information. For example, if the system misses an accidental, the user-supplied constraint can provide no benefit beyond recognizing and precisely locating the symbol. While we didn't distingushing between "leaf node" and "non leaf node" corrections in our evaluation, it is clear that the benefit, as measured in primitive detections, will be distinctly different in these two cases.

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

Discussion

Given the difficulty of optical music recognition, as well as the great deal of domain knowledge necessary to correctly understand and interpret musical scores, it seems reasonable to cast the problem in terms of human-in-the-loop computing thus giving both machine and person contribute what they do best. We have proposed a simple template for human input — the labeling of individual or collections of pixels — that provides a uniform format for human-machine communication, without 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.

[1] `http://music.informatics.indiana.edu/papers/drr16/`

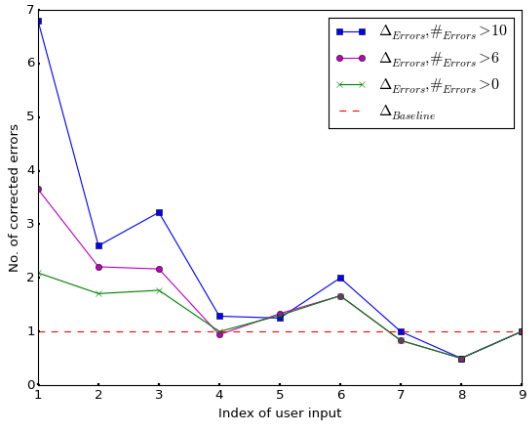

Figure 9: Error decrease at the i-th input: **Blue:** average error decrease in the measures having more than 10 errors; **Magenta:** error decrease in the measures having more than 6 errors; **Green:** average error decrease in the measures having errors.

We see this as the first step in a move toward user-directed recognition, with several promising unexplored variations 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.

Another promising direction allows 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 the model the problem begins to resemble the *registration* problem, rather than *recognition*. Similar kinds of model restrictions apply to system identification, for instance, allowing the user to specify the maximum or minimum number of staves per system. Such a constraint would be helpful, for instance, in piano music where nearly all systems have two staves. As above, these constraints could be switched on and off as needed.

The current 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.

We continue to improve the user interface so as to save more time for the users. A message box with auto-competition or drop-down selection will facilitate label selection. It is also possible to let the system actively predict the user-supplied labelings so that the user can frequently skip the tedious input step. The system can also offer different options to the user, for example simple/complex beaming, beamed groups with one-way/two-way stems, thus allowing the user to limit the application of more permissive and error-prone symbol models.

Finally, it is worth noting that we have proposed a general paradigm for human-in-the-loop recognition in terms of constrained optimization that may apply to a wide variety of problems that require human assistance. Such recognition problems arise in many domains, such as computer vision, natural language processing, and machine listening, where considerable domain knowledge is needed to guide and interpret data analysis, though this knowledge is hard to incorporate directly into the recognition models.

References

[1] Christopher Raphael, Jingya Wang, New Approaches to Optical Music Recognition, ISMIR, pg. 305-310. (2011).

[2] Christopher Raphael, Rong Jin, Optical music recognition on the International Music Score Library Project, IS&T/SPIE Electronic Imaging, pg. 90210F-90210F (2014).

[3] IMSLP: `http://imslp.org`.

[4] Ichiro Fujinaga, Adaptive optical music recognition, PhD diss., McGill University Montral, Canada, 1996.

[5] Rong Jin, Christopher Raphael, Interpreting Rhythm in Optical Music Recognition, ISMIR, pg. 151-156. (2012).

[6] Maura Church, Michael Scott Cuthbert, Improving Rhythmic transcriptions via probability models applied Post-OMR, ISMIR, pg. 643-648. (2014).

[7] Donald Byrd, Megan Schindele, Prospects for Improving OMR with Multiple Recognizers, ISMIR, pg. 41-46. (2006).

[8] Pierfrancesco Bellini, Ivan Bruno, and Paolo Nesi, Assessing optical music recognition tools, Computer Music Journal, pg. 68–93. (2007).

[9] Ian Knopke, Donald Byrd, Towards Musicdiff: A Foundation for Improved Optical Music Recognition Using Multiple Recognizers, ISMIR, pg.123-126. (2007).

[10] Hoda Fahmy, Dorothea Blostein, A graph-rewriting paradigm for discrete relaxation: application to sheet-music recognition, International Journal of Pattern Recognition and Artificial Intelligence 12, no. 06, pg. 763-799.(1998).

[11] Bertrand Coasnon, Pascal Brisset, Igor Stephan, and Couasnon Pascal Brisset, Using logic programming languages for optical music recognition, Proc. of the Third International Conference on The Practical Application of Prolog, (1995).

[12] Gary E. Kopec, Philip A. Chou, and David A. Maltz, Markov source model for printed music decoding, IS&T/SPIE's Symposium on Electronic Imaging: Science &Technology, pg. 115-125. (1995).

[13] Ana Rebelo, Ichiro Fujinaga, Filipe Paszkiewicz, Andr R. S. Maral, Carlos Guedes, Jaime S. Cardoso, Optical music recognition: state-of-the-art and open issues, IJMIR 1(3): pg. 173-190. (2012).

[14] Charalampos Saitis, Andrew Hankinson, and Ichiro Fujinaga, Correcting Large-Scale OMR Data with Crowdsourcing, Proc. of the 1st International Workshop on Digital Libraries for Musicology, pg. 1-3. (2014).

[15] Andrew Hankinson, John Ashley Burgoyne, Gabriel Vigliensoni, and Ichiro Fujinaga, Creating a large-scale searchable digital collection from printed music materials, Proc. of the 21st international conference companion on World Wide Web, pg. 903-908. (2012).

[16] Michael Droettboom, Karl MacMillan, and Ichiro Fujinaga, The Gamera framework for building custom recognition systems, Symposium on Document Image Understanding Technologies, pg. 275-286. (2003).

[17] David Bainbridge, Tim Bell, The Challenge of Optical Music Recognition, Computers and the Humanities 35(2): 95-121 (2001).

[18] Luis von Ahn , Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum, recaptcha: Human-based character recognition via web security measures, Science 321, no. 5895, pg. 1465-1468. (2008).

[19] 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, no. 1, pg. 111-132. (1999).

[20] Steve Branson, Catherine Wah, Florian Schroff, Boris Babenko, Peter Welinder, Pietro Perona, and Serge Belongie, Visual recognition with humans in the loop, ECCV, pg. 438-451. (2010).

[21] Kun Duan, Devi Parikh, David Crandall, and Kristen Grauman, Discovering localized attributes for fine-grained recognition, CVPR, pg. 3474-3481. (2012).

[22] Luis von Ahn, and Laura Dabbish, Labeling images with a computer game, Proc. SIGCHI, pg. 319–326. (2004).

[23] Alexander J. Quinn, and Benjamin B. Bederson, Human computation: a survey and taxonomy of a growing field, Proc. SIGCHI, pg. 1403-1412. (2011).

[24] 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, pg. 160–160. (2007).

[25] Robert Clawson, and William Barrett, Intelligent indexing: A semi-automated, trainable system for field labeling, IS&TSPIE Electronic Imaging, pp. 94020A-94020A. International Society for Optics and Photonics. (2015).

[26] Vernica Romero, Alejandro H. Toselli, Luis Rodrguez, and Enrique Vidal., Computer assisted transcription for ancient text images, Proceedings of the 4th international conference on Image Analysis and Recognition, pp. 1182-1193. (2007).

Author Biography

Liang Chen received his BS in microelectronics from Shanghai Jiao Tong University (2009) and his MS in circuit and system from Institute of Semiconductors, Chinese Academy of Sciences (2012). He is currently a third-year PhD student in Music Informatics at Indiana University, Bloomington. His work has focused on Optical Music Recognition (OMR) and computer-based music notation.

Christopher Raphael is on the faculty in the School of Informatics and Computing at Indiana University, Bloomington where he leads the Music Informatics group. His research interests include recognition and many applications to the musical domain. He received his PhD in Applied Mathematics from Brown University in 1991 and his BS in Computer and Information Science from the University of California at Santa Cruz in 1984.