Renotation of Optical Music Recognition Data

Liang Chen, Christopher Raphael
School of Informatics and Computing, Indiana University Bloomington
chen348@indiana.edu, craphael@indiana.edu

ABSTRACT

We present the problem of music renotation, in which the results of optical music recognition are rendered in image format, while changing various parameters of the notation, such as the size of the display rectangle or transposition. We cast the problem as one of quadratic programming. We construct parameterizations of each composite symbol expressing the degrees of freedom in its rendering, and relate all the symbols through a connected graph. Some of the edges in this graph become terms in the quadratic cost function expressing a desire for spacing similar to that in the original document. Some of the edges express hard linear constraints between symbols expressing relations, such as alignments, that must be preserved in the renotated version. The remaining edges represent linear inequality constraints, used to resolve overlapping symbols. The optimization is solved through generic techniques. We demonstrate renotation on several examples of piano music.

1. INTRODUCTION

A symbolic music encoding, such as MEI [1], MusicXML [2], and many others, represents music in a hierarchical format, naturally suited for algorithmic manipulation. One of the most important applications of symbolic music encodings will likely be what we call *music renotation* [3], which renders an image of the music notation, perhaps modified in various ways. For instance, music renotation may produce a collection of parts from a score, show the music in transposed form, display notation expressing performance timing, or simply display the music in an arbitrarily-sized rectangle. Music renotation may someday be the basis for digital music stands, which will offer a range of possibilities for musicians that paper scores do not, such as instant and universal access to music, page turning, search, and performance feedback. While these ideas could apply to any type of music notation, our focus here is on common Western notation.

As long as there has been interest in symbolic music representations, there has been a parallel interest in rendering these encodings as readable music, e.g. [4], [5]. There is a well-developed craft surrounding this engraving problem, predating algorithmic efforts by centuries. The essential task is to produce music documents that are easy to read and make efficient use of space. Often the engraver must choose between many possible equivalent representations of the music, while considering the placing of symbols, avoiding unnecessary overlap, preserving important alignments between symbols, and leading to a pleasing document overall.

Our essential idea is to leverage the choices made in a previous engraving to guide our renotated rendition. This includes the many *hard* or categorical decisions made in the original document, such as what notes should be beamed, what stem directions to use, as well as the many other choices that lead to equivalent notation in terms of pitch and rhythm. However, we also seek to leverage the specific *soft* choices of layout and spacing that give the music the desired density of information while leading to high readability. In essence, our renotated music seeks to copy as much of the original layout as possible, while simultaneously satisfying the constraints imposed by our renotated format, such as the dimensions of the display rectangle.

Our formulation of the renotation problems ties in naturally to our work in optical music recognition (OMR). Our *Ceres* system [6–8] is a longstanding research interest that seeks to recognize the contents of a music document. From the standpoint of recognition, it is almost impossible to identify a musical symbol in an image without also knowing the precise location and parametrization of the symbol, thus accurately registering the recognized symbol with the image data. Therefore, a welcome consequence of OMR is a precise understanding of the music layout, including both the *hard* and *soft* choices, leaving our current effort poised for high-quality renotation. This is the goal we address here.

Our essential approach is to cast the renotation problem as one of *optimization*, expressed in terms of a graph that connects interdependent symbols. This view is much like the spring embedding approach to graph layout, so popular in recent years [9–12]. Spring embeddings seek to depict the nodes and edges of a graph in a plane, with minimal conflict between symbols, while clustering nearby nodes. Unlike the graph layout problem, there is no single correct graph for music renotation. Rather, the construction of an appropriate graph lies at the heart of the modelling challenge. Our work bears some resemblance to the work of Renz [13], who takes a spring embedding view of one-dimensional music layout.

2. MODELING THE NOTATION

At present, we do not model the entire collection of possible musical symbols, limiting ourselves to the most com-

mon ones, though providing generic mechanisms to extend to larger collections. This restriction of attention parallels the state of our OMR system, which recognizes only a subset of possible notation, at present. It seems that any music notation effort must eventually come into contact with the "heavy tail" problem — there are a great many unusual symbols, special cases, and exceptions to familiar rules which, collectively, pose a formidable modeling challenge [14]. To avoid getting buried in the many details of real-life music notation we focus here on the *core* of musical symbols, by which we mean those involved in conveying pitch and rhythm (beams, flags, stems, note heads, ledger lines, accidentals, clefs, augmentation dots, time signatures, key signatures, rests, etc.), with a few additions to this core.

We divide the world of music symbols into *composite* and *isolated* symbols. The isolated symbols, (clefs, accidentals, etc.), are stand-alone symbols, represented by single characters in a music font. The composite symbols, including beams and chords, are formed by constrained configurations of *primitives*, such as beams, stems, heads, ledger lines, flags, etc. For purposes of renotation we view the *satellites*, (accidentals, articulations, augmentation dots, etc.) as isolated symbols, rather than as parts of a composite symbol.

When drawing a composite symbol one must obey basic constraints between the constituent primitives. For instance, the vertical positions of the note heads are fixed in relation to the staff, while their horizontal positions are fixed in relation to the note stem. These relations hold with both regular and "wrong-side" note heads. After accounting for these constraints, the only degrees of freedom in drawing a chord (not including its satellites) are the horizontal and vertical locations of the stem endpoint. The locations of all other primitives follow deterministically once the stem endpoint is fixed. Beamed groups have more degrees of freedom in their rendering. After fixing the corners of the beamed group one must also decide upon the horizontal positions of the stems, leading to $n + 2$ degrees of freedom for an n-note beamed group. With these parametrizations in mind, we can think of all symbols, composite and isolated, as parametrized objects, where an isolated symbol is parametrized simply by its two-dimensional location. In some cases, one coordinate of an isolated symbol may be fixed, as with the vertical coordinate of an accidental or clef. In such cases the symbol has a one-dimensional parameter.

2.1 Overview of Approach

We cast the renotation problem as constrained optimization, in particular, quadratic programming (QP) [15], in which one minimizes a quadratic function subject to linear equality and inequality constraints. To this end we let $v_1, \ldots, v_N$ be the musical symbols, isolated *and* composite, and let $x = (x_1, \ldots, x_N)$ be the parameters for these symbols. Thus the notation is completely determined by x. We define a quadratic objective function, $Q(x)$, measuring the desirability of the parameter x. Q penalizes the degree to which the renonated symbols' relative positions differ from those in the original notation — we simply try to copy the original notation as much as possible. To capture hard constraints, such as the essential symbol alignments between rhythmically coincident notes, we include a collection of linear equality constraints $\{a_m^t x = b_m\}_{m=1}^{M}$, which we will summarize as $Ax = b$. To capture the non-overlapping constraints, we include inequality constraints of the form $\{a_{m'}^t x \leq b_{m'}\}_{m'=1}^{M'}$ summarized as $A'x \leq b'$. Thus we define our desired solution as

$$\hat{x} = \arg \min_{x: Ax=b, A'x \leq b'} Q(x) \tag{1}$$

QP gives us a formulaic way of solving an optimization problem, once posed in this manner.

To be more specific, we define our objective function and constraints in terms of a graph, $G = (E, V)$. In our graph, the parametrized symbols become the vertex set, $V = \{v_1, \ldots, v_N\}$. We define an edge set composed of three types of edges between these vertices, *soft*, *hard* and *conflict*, denoted by $E = E_s \cup E_h \cup E_c$. Our graph will be *connected* by the $E_s \cup E_h$ edges, thus ensuring that there are no notational "islands" that function independently of the rest. For every edge, $e = (v_i, v_j) \in E_s$ we define a quadratic term, $q_e(x_i, x_j)$, so that our objective function, Q, decomposes as

$$Q(x) = \sum_{e=(v_i,v_j) \in E_s} q_e(x_i, x_j)$$

Similarly, for each edge $e = (v_i, v_j) \in E_h$ we define a constraint, $a_e^t x = b_e$, where the vector, a_e, is non-zero only for the components corresponding to x_i and x_j. Analogously we have a constraint, $a_e^t x \leq b_e$ for each edge $e \in E_c$. Thus A and b from Eqn. 1 become

$$A = \begin{pmatrix} a_{e_1}^t \\ a_{e_2}^t \\ \vdots \\ a_{e_M}^t \end{pmatrix} \quad b = \begin{pmatrix} b_{e_1} \\ b_{e_2} \\ \vdots \\ b_{e_M} \end{pmatrix}$$

where $e_1, \ldots, e_M$ is an arbitrary ordering of the edges of E_h, with a similar definition for A' and b'.

2.2 A More Detailed Look: Beamed Groups

We examine here the case of a beamed group in more detail, so as to flesh-out our ideas. As already described, the beamed group itself (without its satellites) is parametrized by an $(n + 2)$-dimensional parameter, x_b. Each note on each chord of the beamed group may have an associated accidental, while the heights of these accidentals are fixed at the appropriate staff position. Thus we can write $\{x_i^{\text{acc}}\}$ for the the horizontal positions of these accidentals, indexed by i. Absent other considerations, there is an ideal horizontal distance between a note head and its accidental. However, in some cases, particularly with a densely-voiced chord, the accidentals may be placed far to the left of this ideal to avoid overlap. For each accidental we introduce a quadratic term, $q_i^{\text{acc}}(x_b, x_i^{\text{acc}})$, that penalizes the degree that the accidental displacement differs from that in

the original image. We also must deal with potential overlap between the accidentals, and will return to this issue in what follows.

Augmentation dots are more variable in their vertical placement than are accidentals. In part, this results from the need to avoid conflicts with the staff lines, however, densely-voiced chords often require that augmentation dots be notated far away from their nominal vertical positions to avoid conflicts. We introduce quadratic terms $q_i^{k,\mathrm{aug}}(x_b, x_i^{k,\mathrm{aug}})$ for $k = 1, 2$ where $x_i^{k,\mathrm{aug}}$ is the kth coordinate of the ith augmentation dot position. Again, this term penalizes the difference between the actual displacement from the note head and that observed in the original image. If each member of a chord has a single augmentation dot, those augmentation dots are constrained to share a single horizontal position by adding appropriate linear equality constraints, proceeding analogously with additional augmentation dots.

Note heads may have one or several symbols above (stem-up) or below (stem-down) the note head, such as articulations, ornaments, fingerings, etc. We will refer to these symbols collectively as "markups." The case of a single markup is easy to handle, as such a symbol is usually centered over the note head, thus constraining its horizontal position. This situation is handled in analogy with the accidental, in which we seek to replicate the vertical separation found in the original image while constraining the relative horizontal position to the note head. Occasionally a note head has several associated markup symbols, requiring that we *infer* the desired constraints. Often markups are stacked, which calls for a shared horizontal position with all of the symbols (equality constraints). Though sometimes one sees more elaborate configurations calling for different kinds of alignment. For instance, one may have a trill with two numbers above the trill indicating the fingers to be used. Typically the pair of finger numbers would be centered over the trill, calling for a different collection of constraints. Suffice it to say that when multiple markups are present we must infer the desired equality constraints before we impose these.

2.3 Conflict Resolution

When we view a beamed group in isolation we would expect no conflicts between the various satellites surrounding the note heads: their preferred relative positions are those from the original image, which presumably do not overlap. However, we now view a beamed group as a *flexible* object having desired relationships with the *other* symbols composing the notation. These other symbols exert influence on the beamed group. As we allow the beam group to be deformed away from its original presentation, we should expect conflicts to develop between the symbols as they inadvertently overlap one another.

We make two observations regarding symbol conflicts that will guide our process for resolving them, not just with beamed groups, but all conflicts that can arise. Consider, first, the case of two rectangular bounding-box regions for symbols v_i and v_j, $(a_l^i, a_h^i)\times(b_l^i, b_h^i)$ and $(a_l^j, a_h^j)\times(b_l^j, b_h^j)$, both expressed as the cross product of horizontal and vertical intervals, as in Figure 1. There are four relevant linear inequality constraints, the satisfaction of any one of these ensuring the regions do not overlap:

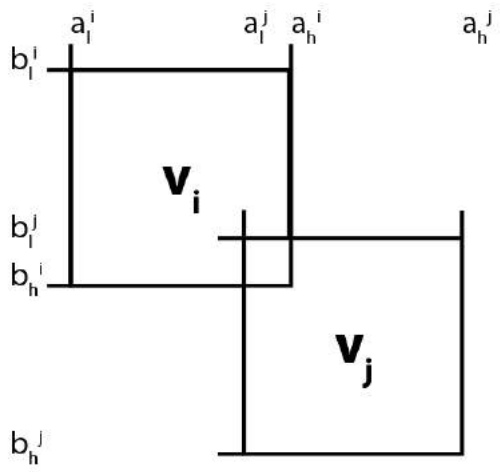

Figure 1. A conflict between two symbols, v_i and v_j, can be resolved by moving the relative positions of their bounding boxes in one of four ways, illustrated by the four dotted rectangles.

$$\begin{aligned} a_l^i &\geq a_h^j \\ a_h^i &\leq a_l^j \\ b_l^i &\geq b_h^j \\ b_h^i &\leq b_l^j \end{aligned}$$

These correspond to moving v_j left, right, up, or down, relative to v_i. There is no general way to express a disjunction of linear inequality constraints as a conjunction of linear inequality constraints, thus we cannot, with QP, perfectly model the concept that a pair of symbols cannot overlap. We resolve overlap between a symbol pair by selecting *one* of the above four constraints to include in the QP formulation.

The second observation is that it isn't necessary to include all inequality constraints in the initial statement of our optimization problem. That is, suppose we minimize the objective function, $Q(x)$, subject only to the equality constraints. If the resulting configuration happens to satisfy the inequality constraints, then we have found the fully constrained (equality and inequality) optimum as well.

We extend this reasoning for our problem. While there are, in principle, non-overlapping (inequality) constraints between any pair of symbols in our notation, we do not impose these at first, instead solving a simpler QP problem that allows the symbols to overlap. If the solution of this problem contains overlap, we impose additional constraints to alleviate this problem. In particular, for each overlapping symbol pair, we choose one of the inequality constraint from the four constraints enumerated above that is near to being satisfied. We further restrict our choice of constraint to be *consistent* with the original notation. For instance, if two symbols, v_i and v_j overlap but v_j lies to

the right of v_i in the original notation, we could choose the constraint $a_h^i \leq a_l^j$. This constraint would be expressed in terms of the parameters x_i, x_j, and included into the set of inequality constraints. This approach guarantees that our QP problem has a non-trivial feasible solution, since the layout of the original image is a feasible point. If v_i and v_j overlap in the *original* image, we cannot resolve this situation, though such situations are uncommon and usually occur in layout situations without obvious solutions.

We iterate the process of solving a QP problem, and imposing new constraints to resolve any newly-discovered overlapping symbols. As all of the newly-added inequality constraints are consistent with the original notation, the QP problem has a feasible region, thus remains well-posed. Our final result is a solution to the QP problem with equality constraints, plus the additional constraints imposed through the iterative process. In this way we *approximate* the notion of non-overlapping constraints.

Our approach constrains us to solve layout problems in a way that is similar to what has been done in the original notation, rather than seeking novel ways of resolving conflicts. In essence, this is consistent with the heart of our proposed approach that leverages the existing notation by deferring to the original notation. The problem of constructing good notation from a symbolic music representation containing no positional information is considerably more challenging, and, perhaps, unnecessary.

2.4 Constructing the Symbol Graph

We have reduced the modeling problem to constructing a graph on the symbol vertices with edges labeled as either *soft*, *hard* or *conflict*, corresponding to quadratic terms, equality constraints, and inequality constraints. The graph for each system first begins by ordering the *basic* symbols according to their left-to-right presentation in the original image. These *basic* symbols include all notes, rests, clefs, key signatures symbols, and bar lines. These are connected in left-to-right manner with *soft* edges derived from the original image. While a beamed group is composed of a collection of notes, the edges are between the notes themselves, reflecting our desire to space approximately as done in the original. These edges are only concerned with horizontal distance. The other type of soft edges are used to express the desired vertical distance such as the stem length and the distance between markups and their associated note heads.

In addition, we introduce *hard* edges between various pairs of symbols. There are several situations that give rise to these equality constraints. For instance, "opposing chords" are pairs of "stacked" chords with a stem-up chord on top and a stem-down chord on bottom. These are recognized as single symbols by our OMR system, thus our representation implicitly understands the need for the alignment constraint. We add hard edges between such chord pairs to force their horizontal alignment. Other symbol alignments are detected by thresholding the actual positions of symbols, while these are also modeled with hard edges that force their horizontal alignment. Our eventual goal is to have the process entirely determined through interpretation of the symbols' meaning, for instance rhythm recognition to determine simultaneous notes, though we approximate this goal, at present, through thresholding.

While our analysis is presently limited to these basic symbols, we can extend to a greater variety of symbols, including slurs, text, dynamics, etc., by "anchoring" to the basic symbols. In the most common case, such a symbol is owned or related to a basic symbol. For instance, a dynamic symbol or text may be associated with a certain note, requring its horizontal placement to align with the note. A great variety of symbols can be included into the present context by aligning them to the basic symbols with hard edges that force the appropriate horizontal alignment while using a quadratic term to encourage the vertical spacing in the original image.

Figure 2 shows a graph that is constructed from the notation, expressing the optimization problem we solve. Different colored edges in this figure correspond to different types of quadratic penalty, alignment, and overlap constraint. As outlined above, we proceed by solving the optimization problem defined by the hard and soft edges first. In doing so, conflicts will arise between symbols that overlap in the "optimal" result. We then iterate the process of imposing conflict edges (inequality constraints) between the overlapped symbols until none remain. These conflict edges, as well as the other types of edge are indicated in the figure.

3. RENOTATION EXPERIMENTS

Here we present preliminary experiments demonstrating our proposed ideas in several renotation scenarios. All the image data, graphical models and generated notations in our experiments can be accessed from the website: `http://music.informatics.indiana.edu/papers/smc17/`. As mentioned before, we treat only a subset of the possible realm of music notation, in an effort to develop an overall approach without being hampered by the many complexities of real-life notation. In particular, we restrict our attention to the rhythm and pitch bearing symbols. We believe the basic framework we have established extends naturally to more complex notation.

Our first experiment displays music notation in *panorama mode*, in which the image is composed of a single long rectangle containing only a single system. This kind of display avoids line breaks, so it is easier to produce as it doesn't need to accommodate the notion of a *page*. Some people may prefer the simplicity of panorama notation in some situations, though the strengths and weaknesses of this notation mode are beyond the scope addressed here.

Panorama mode requires that we delete clef-key-signatures that usually begin each line of music, as the single display line makes these redundant. Otherwise, the display layout follows the approach described above. As the bounding box for the notation doesn't lend itself to the pages of a conference proceedings paper, we include links to two examples. The first example shows a page of the first movement of Mozart's Piano Sonata, K. 333, with the notation recognized and renotated in panorama mode from the

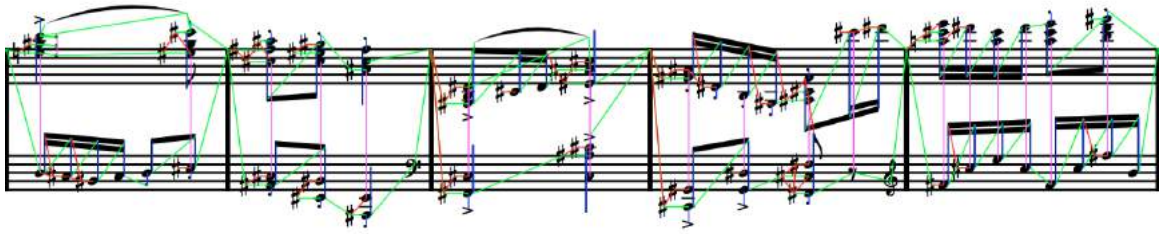

Figure 2. Graph showing the types of edges between the vertices. *Green:* Soft horizontal edges (quadratic term); *Blue:* Soft vertical edges (quadratic term); *Magenta:* Hard Edges (equality constraint); *Red:* Conflict edges (inequality constraint).

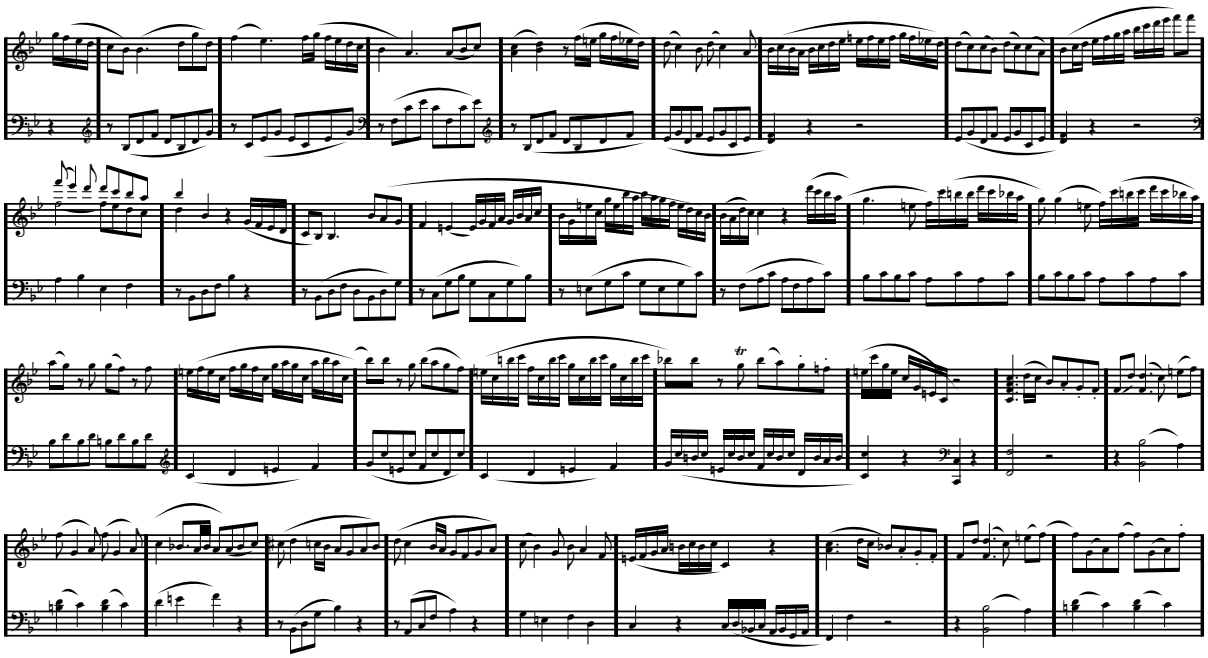

Figure 3. A section from the first movement of Mozart's Piano Sonata, K. 333 (Breitkopf and Härtel 1878 edition)

1878 Breitkopf and Härtel edition [1]. As with this example, many of the most useful public domain editions are quite old, though these editions are still excellent sources for renotation, with a significant degree of expertise and care supporting the engravings. The second shows a page of the Rigaudon from Ravel's *Le Tombeau de Couperin*, using the Durand 1918 edition [2].

Obviously the notation for these examples appears spare without the additional symbols that describe style, dynamics, articulations, etc., though one can still appreciate the value of the proposed ideas based on these examples. While some spacing issues are not resolved in the most readable manner, the overall notation manages to preserve the appropriate rhythmic alignment and produce somewhat natural-looking notation in a *fully automatic* way.

[1] http://music.informatics.indiana.edu/papers/smc17/Mozart_panorama.pdf

[2] http://music.informatics.indiana.edu/papers/smc17/Ravel_panorama.pdf

The second pair of experiments show the same two pieces, now notated in page mode, in Figures 3 and 4. The page sizes were chosen to be considerably wider than the original, thus forcing our algorithm to adjust the spacing to achieve the customary alignment of the rightmost bar line of each system. In addition to removing the original clef-key-signatures at the start of each original staff line, we added the appropriate key-clef-signatures at the beginnings of our renotated staff lines.

4. CONCLUSIONS AND FUTURE WORK

Our renotation examples demonstrate work in progress, thus there are still some issues that need to be resolved in the definition of our objective function before the notation has a professional, easy-to-read look. For instance, smaller symbols, such as accidentals and articulations have different requirements for spacing, and are not ideally han-

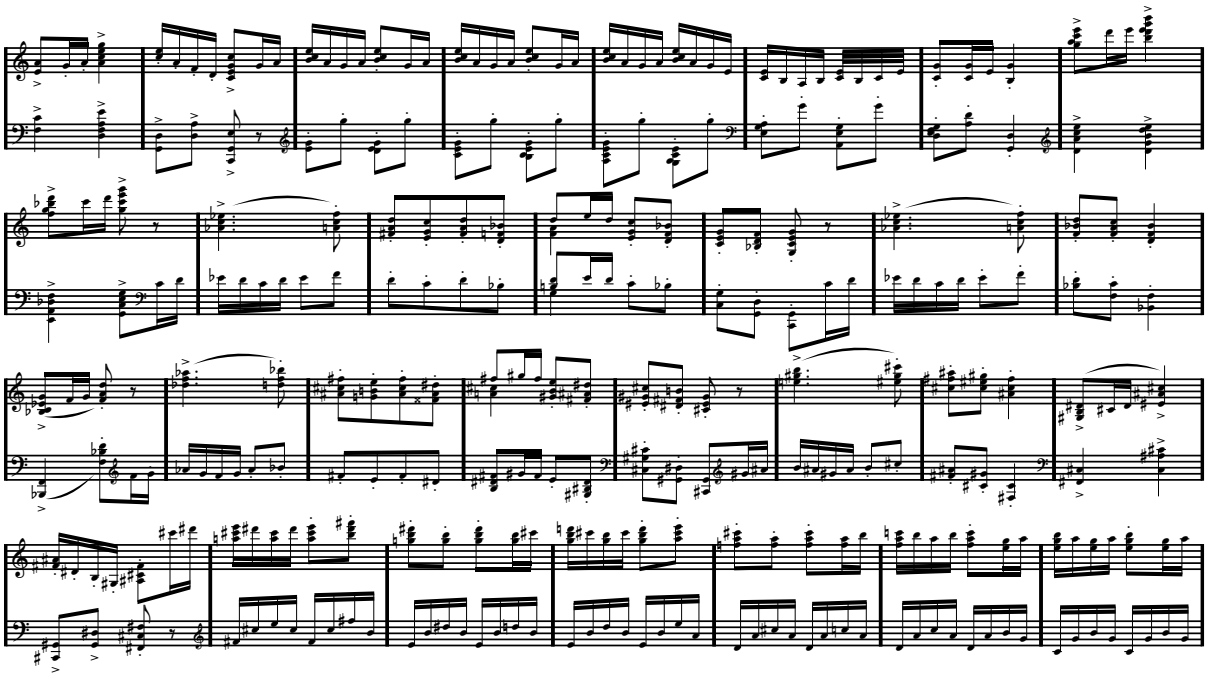

Figure 4. A section from the Rigaudon from Ravel's *Le Tombeau de Couperin*, (Durand and Cie. 1918 edition)

dled by the more generic approach to spacing we take at present.

There are some aspects of this approach that are novel and worth emphasizing. First of all, two-dimensional layout of nearly anything can be a challenging problem, though especially difficult in a domain such as music engraving, where there is a well-developed, if not clearly formulated, standard practice. Rather than try to explicitly formulate the many aspects of good music notation, we seek to leverage existing examples, using these as a guide that will pull our results toward reasonable layout. This includes both the inter-symbol spacing as well as the categorical or discrete decisions of music notation (stem direction, beaming, etc.).

We have formulated the problem as a constrained optimization, specifically as quadratic programming, thus allowing us to incorporate many different objectives into our final result. Furthermore, the approach is fully automatic. It is worth noting, however, that no fully automatic approach will likely lead to the high quality notation we wish to produce. The optimization approach we present can easily be extended to handle explicit human guidance in the form of hand-specified constraints or additional terms for the quadratic objective function. When these are added, we still compute a global optimum that takes into account all other aspects of the objective function, as well as other constraints. Thus our approach provides a natural framework for including human-supplied guidance.

There are additional applications of the proposed techniques we have not presented within, but are still pursuing currently, which we hope will further support the overall framework we present. For instance, our symbolic representation of the music data easily allows for small transpositions in which notes are shifted a few staff positions up or down, while the transposed accidentals are handled in a formulaic (and correct) way. It is worth noting that this recipe for transposition runs into difficulty with larger transpositions that call for changes with stem directions and, perhaps, other notational choices.

Finally, we hope to soon pursue uses of notation designed to convey performance characteristics, rather than readability. For instance, we can present the horizontal position of each note as proportional to its actual onset time, thus allowing one to *see* the expressive use of timing, embedded in the notation itself. Additional variations would allow the visualization of fine-grained pitch, of interest for both tuning and vibrato, as well as dynamics. Our work with score alignment allows for estimation of the appropriate performance parameters, while the renotation ideas may prove useful for conveying this information in a way that is easy for the practicing musician to assimilate.

5. REFERENCES

[1] A. Hankinson, P. Roland, and I. Fujinaga, "The music encoding initiative as a document-encoding framework," in *ISMIR*, 2011, pp. 293–298.

[2] M. Good, "Musicxml for notation and analysis," *The virtual score: representation, retrieval, restoration*, vol. 12, pp. 113–124, 2001.

[3] L. Chen, R. Jin, and C. Raphael, "Renotation from optical music recognition," in *International Conference on Mathematics and Computation in Music*, 2015, pp. 16–26.

[4] L. Pugin, R. Zitellini, and P. Roland, "Verovio: A library for engraving mei music notation into svg," in *ISMIR*, 2014, pp. 107–112.

[5] H. W. Nienhuys and J. Nieuwenhuizen, "Lilypond, a system for automated music engraving," in *Proceedings of the XIV Colloquium on Musical Informatics (XIV CIM 2003)*, 2003, pp. 167–72.

[6] C. Raphael and J. Wang, "New approaches to optical music recognition." in *ISMIR*, 2011, pp. 305–310.

[7] R. Jin and C. Raphael, "Interpreting rhythm in optical music recognition." in *ISMIR*, 2012, pp. 151–156.

[8] L. Chen, E. Stolterman, and C. Raphael, "Human-interactive optical music recognition," in *ISMIR*, 2016, pp. 647–653.

[9] S. G. Kobourov, "Spring embedders and force directed graph drawing algorithms," *arXiv preprint arXiv:1201.3011*, 2012.

[10] T. M. Fruchterman and E. M. Reingold, "Graph drawing by force-directed placement," *Software: Practice and experience*, vol. 21, no. 11, pp. 1129–1164, 1991.

[11] P. Eades, "A heuristics for graph drawing," *Congressus numerantium*, vol. 42, pp. 146–160, 1984.

[12] T. Kamada and S. Kawai, "An algorithm for drawing general undirected graphs," *Information processing letters*, vol. 31, no. 1, pp. 7–15, 1989.

[13] R. K., "Algorithms and data structure for a music notation system based on guido notation," Ph.D. dissertation, Technischen Universität Darmstadt, 2002.

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

[15] D. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1995.