Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2007, Article ID 81541, 25 pages
doi:10.1155/2007/81541

Research Article

Robust and Adaptive OMR System Including Fuzzy Modeling, Fusion of Musical Rules, and Possible Error Detection

Florence Rossant[1] and Isabelle Bloch[2]

[1] *Telecom, Signal, and Image Department, Institut Supérieur d'Electronique de Paris (ISEP), 21 Rue d'Assas, 75006 Paris, France*
[2] *Signal and Image Processing Department, ENST, CNRS UMR 5141, 46 Rue Barrault, 75634 Paris Cedex 13, France*

Received 1 December 2005; Revised 28 August 2006; Accepted 28 August 2006

Recommended by Ichiro Fujinaga

This paper describes a system for optical music recognition (OMR) in case of monophonic typeset scores. After clarifying the difficulties specific to this domain, we propose appropriate solutions at both image analysis level and high-level interpretation. Thus, a recognition and segmentation method is designed, that allows dealing with common printing defects and numerous symbol interconnections. Then, musical rules are modeled and integrated, in order to make a consistent decision. This high-level interpretation step relies on the fuzzy sets and possibility framework, since it allows dealing with symbol variability, flexibility, and imprecision of music rules, and merging all these heterogeneous pieces of information. Other innovative features are the indication of potential errors and the possibility of applying learning procedures, in order to gain in robustness. Experiments conducted on a large data base show that the proposed method constitutes an interesting contribution to OMR.

1. INTRODUCTION

This paper proposes improvements and extensions to our earlier work on optical music recognition (OMR) [1]. OMR aims at automatically reading scanned scores in order to convert them into an electronic format, such as an MIDI file, or an audio waveform. This conversion requires a symbolic representation of the score content, achieved through recognition of its individual components and their structure. The motivation for OMR is manifold, and possible applications cover several topics addressed in this special issue, including automatic transcription, editing, transposition and arrangement, semantic analysis, fingerprinting (which is facilitated by the symbolic representation), feature extraction, indexing and mining, which are important components of query systems, and can all benefit from symbolic representations.

The literature acknowledges active research in the 1970's and 1980's, see, for example, the reviews in [2, 3], until the first commercial products in the early 1990's. The success of these works relies heavily on available knowledge (as opposed to other document analysis problems): reasonable number of symbols, strict location of the staff lines, strong rules of music writing. But still, the problem remains difficult and solutions are generally computationally expensive, even in cases of typeset music.

Despite the advances in the field and the available softwares, there are still some unsolved problems, and recognition is not error or ambiguity free. As already mentioned in [2, 5–8], major problems result from the difficulty to obtain an accurate segmentation into individual meaningful entities. This is due to the printing and digitalization as well as to the numerous interconnections between musical symbols. The complexity of primitive arrangements (e.g., grouped notes) [8], the considerable symbol variability and the continuous evolution of the notation system [9, 10] are other features of the music notation that make it difficult to recognize.

The research in this domain reached a noticeable level, since commercial softwares could be developed and are now widely used. Despite their efficiency, they still failed in a number of configurations. As an illustration, Figure 1 shows examples where the recognition performed by such a software [4] leads to an inconsistent metric in a bar. These errors are due to primitive recognition failures, but they are also probably due to the lack of constraints related to musical rules in the recognition method.

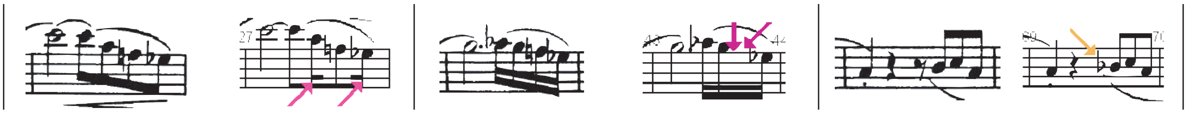


Figure 1: *Examples of recognition errors provided by a commercial software [4]*. On the left-hand side of each column, the original bar, on the right-hand side, the recognition results.

The aim of this paper is to propose a symbol segmentation and analysis method that attempts to overcome the difficulties summarized above, and to show how musical rules can be modeled and introduced at a higher level in order to obtain consistent and reliable results, with better performances.

Numbers of methods have been proposed in order to improve primitive segmentation and recognition (e.g., [5–8, 10, 11]). More particularly, an interesting idea to deal with symbol variability is to propose extensible and adaptive recognition systems [9, 10] that allow supervised learning of symbols. In this paper, we propose a fuzzy model that relies on a robust symbol detection and template matching results. The proposed method allows adapting automatically the class model to the processed score, modeling explicitly the symbol variability within the score, and, as an option, refining automatically the symbol models and the related parameters from a manually corrected excerpt of the music score.

The musical rules are characterized by the particular structure they impose to the scores, but also by their flexibility, either in their parameters (relative position of symbols, e.g.), or in their application modes (redundancy of symbols, e.g.). Recent developments acknowledge the necessity of introducing rules in recognition methods. Most methods introduce structural information at the symbol level, by constraining the spatial arrangement of primitives (note head, note tail, etc.) [7, 8, 12–15]. Syntactic and semantic rules have to be introduced at higher level since they involve several symbols in a bar or in several bars. All methods deal with such information in order to retrieve the semantic content of the recognized symbols (e.g., the pitch of a note, considering its position on the staff but also the clef, the key signature and the accidentals) (e.g., [12–15]). But very few methods model and integrate the syntactic rules in the recognition process itself. They generally restrict to the local graphical rules [7, 14, 15], and to the metric criterion (number of beats per measure) in order to correct some errors [13, 15–18], and consequently are still incomplete with respect to the set of usual musical rules. As for the flexibility aspects, it has not really been addressed until now. A few methods deal with uncertainty and fuzziness at symbol level [19, 20], but not at the rule level. The first real attempt to model explicitly this flexibility was proposed in [1].

Based on this short literature overview, it appears that a number of problems remain unsolved by existing methods. Most of them are linked to the specificities of the musical writing and will be detailed in Section 2. The aim of this paper is to propose a better modeling of available knowledge (in particular musical rules and their flexibility or imprecision), to improve decision making, and to gain in robustness by indicating possible errors. This last feature is an original point that was not addressed before and that allows an interaction with the user, from which learning and correction procedures can be very easily applied. One of the original aspects of our approach is to develop a fuzzy formalism, which allows us to propose a consistent modeling of heterogeneous knowledge, and to exploit the richness of the fuzzy sets theory in terms of knowledge representation, fusion, and decision making [21–24]. Some ideas have already been developed in our earlier work. The aim of this paper is to describe the complete system, including significant additions and improvements with respect to previous work [1, 25, 26].

The paper is organized as follows. Section 2 describes the specificities of musical scores and the main difficulties for recognition. Section 3 provides an overview of the method. The next sections describe each step in more detail. Section 4 deals with the preprocessing steps and the extraction of primitive symbols. In Section 5, we describe how each detected symbol is analyzed in order to generate recognition hypotheses. In Section 6, structural information is used to guide the recognition. This information consists of graphical and syntactical rules that are usually respected in musical writing. In Section 7, the fusion of all available information is performed, leading to the final decision making. Section 8 aims at improving the robustness of the whole system by indicating possible errors and introducing a learning procedure. Experimental results are presented and discussed in Section 9. Section 10 contains concluding remarks and some proposals for future work.

2. SPECIFICITIES OF MUSICAL SCORES

Before presenting our work, we first define precisely which musical scores are processed by our system, and recall the main concepts and definitions used in musical notation, in order to clarify the terminology used all through this paper. Then we summarize the main difficulties in OMR. The aim of our approach is to provide answers to these open questions.

2.1. *Musical notation framework*

The music scores processed by our system are written in the usual Western classical notation. This notation relies on a set of music symbols, codified by structural, graphical, and

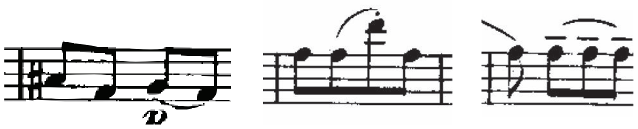


Figure 2: Different note groups, corresponding to the same rhythm.

syntactic rules: for example, the structure of the staves, consisting of five parallel and equally spaced lines, the structure of the notes, composed of assembled primitives (note head, stem, tail, etc.), the relative position of symbols, the use of metric and tonal rules, and so forth. We only consider this notation in our work, and we restrict to monophonic scores (one musical voice per staff). Chords (notes played together) are also not processed, nor polyphonic music. Although the main ideas proposed in this paper are general and can probably be extended or adapted to other kinds of scores, it should be noticed that the actual program is designed under the assumptions presented above, and that the related a priori knowledge is hard coded in it. The last assumption concerns the symbols currently handled, which are restricted to the main ones (notes, rests, accidentals, etc.), that are mandatory for playing the music. Ornaments, for example, are not yet considered.

A very important component of musical writing consists of a set of rules. As they are at the core of our high-level interpretation approach, we review them in this section. They can be divided into two classes: graphical rules, describing the relative positions of symbols, and syntactic rules, involving tonality and metric. Let us summarize the ones that apply in monophonic classical music. They are numbered for later use in this paper.

Graphical rules

(1) An accidental should be placed before the note it modifies and at the same height (vertical position) on the staff.
(2) A duration dot should be placed right after the note head.
(3) A staccato dot should be placed just above (or below) the note head.

Syntactic rules

(4) The number of beats per bar should always correspond to the global metric indication.
(5) Notes are generally grouped into beats, multiple of beats, or beat fraction, so as to facilitate the rhythm and beat structure understanding. This is illustrated in Figure 2. Groups can include silences as well.
(6) The accidentals of the key signature are applied implicitly to each note having the same name (same pitch up to octave shifts).
(7) An accidental applies to the next note, but also to all notes at the same height in the same bar. It does not apply anymore in the next bar if it is not explicitly repeated.
(8) A duration dot modifies the length of a note by a factor 1.5.

A noticeable feature of musical writing is that these rules have different strengths and can be applied with more or less flexibility. Let us take a few examples.

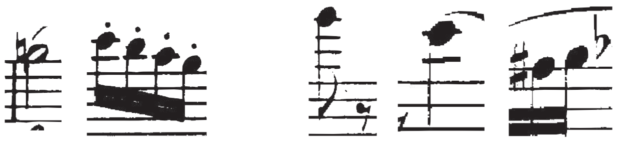
(i) Rule (4) about metric is strict for almost all bars of a score. Exceptions concern only the possible first notes before the first bar (upbeat), and partial bar related to repeat signs. In this case, the sum of the length of the bar preceding the repeat sign and the length of the bar to which the repeat starts should match the metric.
(ii) Rule (7) states that it is theoretically not useful to repeat an accidental in the same bar. However this is by no means forbidden and numerous examples can be exhibited where this redundant repetition is done, for the sake of readability. Similarly, although it is not necessary to cancel the effect of an accidental in the next bar, this often occurs, for similar reasons.
(iii) Rule (5) is by essence very flexible, since several groups can be consistent with usual beat subdivision or grouping. It can be even more relaxed, by allowing unusual groups for interpretation sake (see, e.g., the last example in Figure 2).
(iv) Graphical rules are imprecise: they impose a relative position between symbols, but this position cannot be given in a precise way. A lot of variations occur within a score depending on symbol density, or between different scores (from different editors typically).

These examples show that rules are usually flexible, or contain imprecise parameters. They are usually satisfied, but can also be relaxed, or applied in different ways.

The main originality of our approach lies in the modeling of these rules, along with their flexibility and imprecision, and in their introduction in the recognition procedure. These features of our approach allow us to provide original and efficient answers to the open questions listed in the next section.

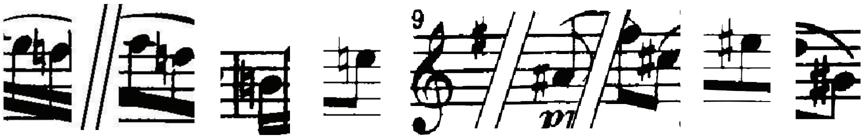
2.2. Main difficulties in OMR

Although OMR has a number of common features with text recognition (limited number of symbols, writing codes), both domains are actually quite different and OMR raises specific problems. Although most of these problems have already been mentioned in the literature (e.g., [2, 5, 8]), several of them are not yet properly addressed by existing methods,



(a) (b)

Figure 3: Examples of common printing defects: symbols touching (a), fragmented or damaged symbols (b).



(a) (b)

Figure 4: Typesetting variability between different publishings and also within the score.

or only partial solutions are proposed, addressing only a part of them. Here we take them all into account, and we propose global solutions.

Most systems, including ours, rely on a first segmentation step, aiming at localizing and isolating individual symbols, before recognition. However, symbols are usually connected by staff lines and beams, making this step a difficult one [5, 11]. The problem is even more complex due to the limited quality of original scores and their scanned version (Figure 3). Although a lot of effort was dedicated to this step [5, 6, 8, 11, 27], it is clear that not all problems can be solved at this level, and the imperfections of the segmentation have to be taken into account during the next processing steps.

Another difficulty is related to the variability of symbols. This is obvious when considering different scores, in particular from different editors, but variability can even be high in the same score (Figure 4). This results in an additional level of ambiguity, which will strongly influence the individual symbol recognition step. While learning procedures can be considered to cope with the first type of variability [9, 28], its efficiency will reach an intrinsic limit due to the intrascore variability, which has therefore to be addressed at a different level.

The types of rules used in musical writing carry intrinsic ambiguity, due to their flexibility, as explained above. Although it is clear that their modeling should be of great help in the recognition procedure, it is clear as well that these characteristics constitute an additional difficulty. Moreover, the interpretation and recognition of a symbol usually requires the use of several rules, acting either at the level of neighbor symbols, or at higher level. A fusion step is therefore necessary to guarantee both a good individual recognition and a good global consistency. This fusion has to deal with heterogeneous pieces of information and knowledge (local information issued from segmentation, interpretation of previous symbols, syntactic rules, etc.).

The problem can be expressed as follows: detect, segment, and recognize basic primitives as reliably as possible, model the available knowledge, at structural and syntactic levels, and exploit this information to disambiguate between possible interpretations of primitives and to provide a high-level interpretation. This task is complex for several reasons, as explained above. The main ones can be summarized as follows.

(1) Ambiguity is important, because of the printing imperfections, the difficulty to segment the score into meaningful entities, and the variability of primitives.

(2) Ambiguity is difficult to solve, because the number of possible arrangements of primitives is very high, although the number of symbols is restricted.

(3) Notation rules express flexible constraints or may be valid up to different precision degrees.

(4) These rules involve a large number of symbols, which can be spatially far from each other in the score.

(5) These rules have different characteristics. They apply at different levels. But since they are highly interdependent, they should be used together to lead to a consistent interpretation.

We show in the next sections how we can solve these problems by developing an adequate modeling of the information (of all types) along with its specificities and imperfections. In our model, these specificities and imperfections are considered as a piece of information as well. Their explicit modeling avoids the development of complex repairing methods, which would be necessary if this type of information would be ignored.

3. GENERAL STRUCTURE OF THE PROPOSED METHOD

An overview of the proposed method is illustrated in Figure 5. The input is decomposed into two types of information:

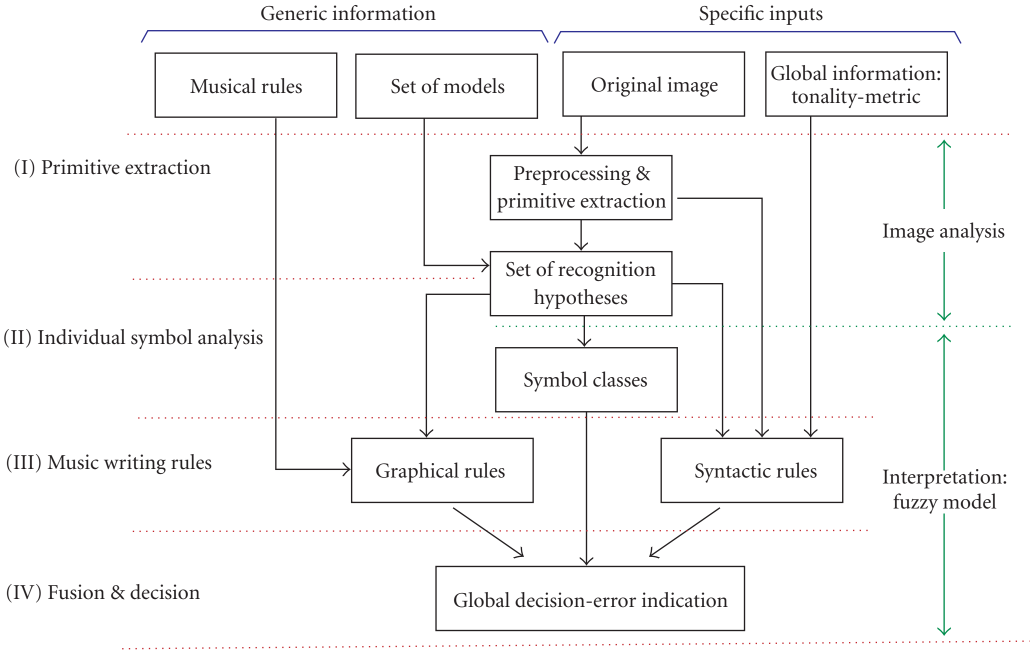

Generic information
Specific inputs
Musical rules
Set of models
Original image
Global information: tonality-metric
(I) Primitive extraction
Preprocessing & primitive extraction
Image analysis
Set of recognition hypotheses
(II) Individual symbol analysis
Symbol classes
(III) Music writing rules
Graphical rules
Syntactic rules
Interpretation: fuzzy model
(IV) Fusion & decision
Global decision-error indication

Figure 5: Overview of the proposed method.

(i) generic information, consisting of (i) a set of reference models of symbols, which will be used mainly in the first phases, dedicated to individual symbol extraction and analysis, and (ii) musical rules, used in the higher-level interpretation phases;

(ii) specific information related to the score to be analyzed, consisting of (i) the scanned page, and (ii) global information on tonality and metric (provided by the user).

The first phase consists of preprocessing steps and segmentation. First, the orientation of the score is estimated. This allows us to realign the staff lines along the horizontal axis and to improve their detection [25]. Segmentation of symbols is then performed, based on the accurate removal of staff lines that isolates most symbols, and the detection of vertical segments that feature all the others. Detection of beams connecting stems is also considered. This phase is described in Section 4.

In the next step, segmented objects are analyzed based on their correlation with the reference models (generic information). From correlation coefficients, at most three hypotheses are generated for each object. A hypothesis is an assignment of a segmented object to a symbol class, to some degree (modeled using possibility distributions). The possibility that the object is not a musical symbol is considered as well. This processing is detailed in Section 5.

The next phases deal with higher-level interpretation. It uses intensively the musical rules. This type of knowledge is adequately represented in the possibility theory because of the required flexibility mentioned in Section 2. After the first phase, these rules can be instantiated according to the analyzed score. This concerns, in particular, the tuning of parameters of the possibility distributions, based on parameters extracted from the score such as the interval between two staff lines. Then the recognition hypotheses generated previously are reconsidered and their mutual consistency according to the rules and to the specific global information (tonality and metric) is checked. This is presented in Section 6.

The best combination of recognition hypotheses after the fusion of all rules is chosen. This decision is made globally for all symbols in a bar. These fusion and decision steps are described in Section 7.

In order to simplify the scheme, it is presented as a unidirectional process in Figure 5. However, backwards procedures have been implemented as well, in order to correct errors and gain in robustness. We propose a method for indicating possible sources of errors which is an innovation with respect to existing systems. This allows an easy interaction with the user, who can correct these errors, for instance on the first page of the score. Based on these corrections, symbol models or specificities of the score can be more precisely learned. The learning procedure is expected to lead to fewer errors in the next pages of the score. This reduced interaction with the user is not tedious, likely to be well accepted if limited to one page for instance, and may be even required by the user in order to guarantee better results and fewer errors on the whole score. This learning should however not lead to very restricted models and flexibility should still be allowed, in order to deal with intra-score variability, as mentioned above.

Although the general system architecture, including successively preprocessing, symbol recognition, syntactic and semantic analysis, is similar to existing ones (e.g., [9, 13, 15, 18]), its individual components contain innovative features that allow overcoming the limitations of previous systems and answering in an elegant way the questions raised in Section 2. This concerns in particular the modeling of musical rules and their rigorous structured organization in the system, which avoids ad hoc uses of these rules spread all over the processing. A main feature of the proposed method is that no definite decision is made before the whole bar is analyzed, both at symbol level and at higher interpretation level using musical rules. This process manages ambiguity and imprecision and takes all the context into account, unlike existing methods. Another important feature is adaptability: possibility distributions are defined generically and their parameters are learnt on the specific analyzed score. This applies in particular to models and graphical rules. The proposed system goes one step further in this direction since models can be further refined and precisely adapted after one step of recognition-correction on a few bars or a page. Finally, the proposed fuzzy modeling exploits the advantages of fuzzy sets and possibility theory in terms of modeling and fusion of heterogeneous information and knowledge prone to imperfections, variability, and flexibility.

4. PREPROCESSING AND PRIMITIVE DETECTION

The music sheets (in A4 format) have been scanned at the resolution of 300 dpi and binarized to provide an image $I(x, y)$, where $I(x, y)$ at point (x, y) can take values 0 (white pixels in the following figures) or 1 (black pixels). Our coordinate system is defined by the origin, at the upper-left corner of the image, the vertical axis from top to bottom (x coordinate), the horizontal axis from left to right (y coordinate).

The preprocessing method concerning the staff detection and skew recognition, and the segmentation step are based on [25], with significant improvements, in order to better overcome the segmentation difficulties due to printing imperfections. Then, the musical symbols we want to recognize can be classified into two different types: (i) the symbols that are featured by a vertical segment, obviously the notes through their stem, but also the accidentals (flat, sharp, and natural), and the appoggiatura; (ii) the rests, the whole notes and the dots. The symbols of the second type can be easily isolated through a staff line removal algorithm. The problem is more complicated for the symbols of the first type, since notes are often beamed together, and accidentals often connected to note heads because of printing defects (Figure 3). But once they are detected through their vertical segment, a template matching procedure can be applied because this recognition method does not require precise prior knowledge about the pattern location.

Thus, our symbol segmentation method relies on two fundamental steps: the staff lines removal and the detection of vertical segments. These two steps must deal with common printing defects: locally warped staff lines, broken segments, skewed vertical segments. We propose in the next sections an accurate staff lines detection and removal algorithm, and a robust detector of vertical segments. The last section concerns the detection of beams connecting stems together, that also has to cope with variability and defects.

4.1. *Staff-line detection and removal*

The challenge is to realize an accurate removal of staff lines, capable of disconnecting nearby symbols that are syntactically separated, while avoiding suppressing pixels belonging to symbol primitives. This requires knowing the exact location of the staff lines at each horizontal coordinate. The skew angle is first calculated and corrected, the staff lines are coarsely detected and the staff spacing (S_I) is computed [25]. The staff line thickness (e) is also estimated, by detecting the maximum peak in the histogram of the black run-lengths [15]. Based on these parameters, we define the H_p coefficients of a correlation mask, that represents the cross-section of the staff, and that will be used in a horizontally tracking filtering, for the purpose of locating precisely the staff position:

$$\begin{gathered} M_p(x) = \begin{cases} 1 & \text{for } x = \dfrac{H_p}{2} + k^* S_I + i, \\ & \quad k \in [-2, 2],\ i \in [-\Delta_b, \Delta_h], \\ -1 & \text{otherwise,} \end{cases} \\ H_p = 2\lfloor 2.5 S_I \rfloor, \quad 0 \leq x < H_p, \\ \Delta_b = \left\lfloor \frac{e}{2} \right\rfloor, \quad \Delta_b + \Delta_h + 1 = e, \end{gathered} \tag{1}$$

with $\lfloor x \rfloor$ denoting the integer part of x.

We compute then the correlation between this mask and the image. Let us denote by $C(x, y)$ the result obtained at point (x, y):

$$C(x, y) = \frac{1}{H_p} \sum_{i=0}^{H_p - 1} \left(M_p(i) \cdot I'\left(x + i - \frac{H_p}{2}, y \right) \right), \tag{2}$$

where

$$I'(x, y) = \begin{cases} -1 & \text{if } I(x, y) = 0, \\ 1 & \text{if } I(x, y) = 1, \end{cases} \quad 0 \leq x < H,\ 0 \leq y < W. \tag{3}$$

The correlation score is computed at each y coordinate for several x around the average position of the third staff line. Without symbol superimposed on the staff, the maximum value $C(x_{\text{opt}}, y)$ leads to the exact position x_{opt} of the third staff line at the y coordinate. To deal with symbol interference, we use a filter technique that integrates continuously the previous correlation scores in order to be insensitive to superimposed symbols and just track the slow variations of the staff line position. This algorithm can be formalized as follows:

$$\begin{gathered} \text{filt\_staff}(x, y) = \alpha^* \text{filt\_staff}(x, y - 1) + (1 - \alpha)^* C(x, y), \\ \max_x \left(\text{filt\_staff}(x, y) \right) = \text{filt\_staff}(x_{\text{opt}}, y). \end{gathered} \tag{4}$$

The filter is initialized on the left-hand side, at the starting position of the staff, and the filter is applied at each increasing y, leading to the accurate vertical position x_{opt} at each horizontal coordinate. The parameter α is set to 0.98.

Figure 6 shows an example of result. The black lines represent the average position of the staff lines, extracted in the first step, while the red lines follow exactly the warped staff lines.

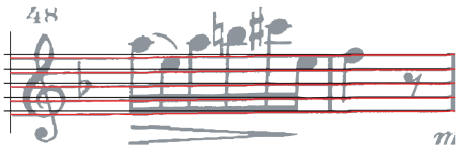


Figure 6: Accurate detection of warped staff lines.

The proposed method provides reliable results. It has also been successfully tested on polyphonic music scores that present higher symbol density. Compared to other tracking algorithms presented in the literature, the strong point is the continuity of the analysis, that probably ensures, as in [27], a better robustness towards interfering symbols than more local methods [5, 6].

In the next step, the staff lines are simply removed by considering the length of the black vertical runs intersecting the staff lines. A run is a set of connected pixels of the same color within a column or a line. Let us consider the staff line number k, and a black run located between x_1 and x_2 at the y coordinate. This run is deleted if the following conditions are simultaneously verified:

(1) $(x_2 - x_1 + 1) \leq e + 2$,
(2) $x_1 < (x_{\text{opt}} + kS_I)$,
(3) $x_2 > (x_{\text{opt}} + kS_I)$.

Figure 7 shows two excerpts of a score after staff line removal. The objects are properly disconnected, while the beamed notes remain well connected, and so the first step of the segmentation process succeeds. Nevertheless, we can notice some imprecision on the symbol boundaries: some of them, especially the whole ones, may be erased on their thin part tangent to the staff lines, while some small parts of the staff lines may be left attached to the symbol, since they are connected to it. These imperfections increase the symbol variability, since the obtained shape may vary for a given symbol class, depending on its position on the staff. The problem of erroneous removing was partially addressed in [5, 11, 29]. In our method, the resulting imprecision on the symbol shape will be formalized at a later stage, in the fuzzy symbol model part.

4.2. *Vertical segment detection*

The vertical segment detection must overcome two major difficulties due to bad printing quality: skew and interruptions. The latter are difficult to solve because the number of consecutive white pixels interrupting the segment is generally in the same range as the number of white pixels separating symbols that are close together. Another difficulty is that vertical segments may be part of a symbol (case of accidentals), or are connected to another primitive (note head or beam). We define the geometrical and topological features of the vertical segments we want to extract, at the considered image size and resolution:

(1) length greater than 1.5 staff space,
(2) typical thickness from 1 to 5 pixels, on the linear parts,
(3) separation with nearby symbol greater than 2 pixels,
(4) separation between two nearby segments greater than 2/5 staff space,

and the defects that frequently arise

(5) small interruption, from 1 to 2 pixels.
(6) slight skew.

This analysis leads us to propose to compute three images, from which the segments can be extracted: a label image I_v of the vertical runs, where the value at point (x, y) is equal to the length of the black run passing through this point, a label image I_h of the horizontal black run-lengths, and a filtered image I_l extracting the pixels of horizontal runs that meet items (2) and (3):

$$I_l(x,y) = I(x,y)\sum_{j=-4}^{4}\left(\overline{I(x,y+j)}N_l(j)\right), \qquad (5)$$
$$N_l = \frac{1}{4}\begin{bmatrix}1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1\end{bmatrix}.$$

The segment extraction process is carried out as follows:

(i) vertical closing, up to 2 pixels, of the vertical black runs whose nearby extremities are both maxima in I_l. This step addresses item (5), while avoiding connecting objects that are effectively syntactically disconnected. The label images I_v and I_h are then updated,

(ii) search for the longest vertical run in I_v at every y coordinate. Let us denote by x_h and x_b the coordinates of the extremities. The run is retained if it satisfies simultaneously criteria 1 to 3, expressed as

$$l = x_b - x_h + 1,$$
$$l > 1.5S_I, \qquad \frac{\sum_{x=xh}^{xb} I_l(x,y)}{l} > \frac{1}{4}. \qquad (6)$$

The second condition is purposely loose because we search for segments that are generally included in a wider shape and we do not want any significant segment to be oversight. But it is sufficient to discard black runs that belong to a very thick component, such as noisy interconnected beams of note group,

(iii) horizontal analysis of the remaining adjacent black runs by the mean of a horizontally narrow sliding window ($2/5S_I$ wide), that keeps only the longest run and discard all the others. This step addresses item (4). In this way, just one black run is retained per vertical segment.

(iv) the last step solves the problem of skew (item (6)). When a thin vertical segment is skewed, the detected

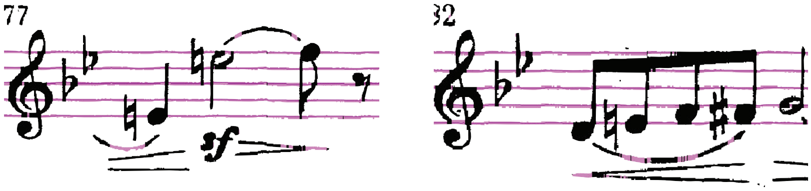


Figure 7: Examples of staff lines removal.

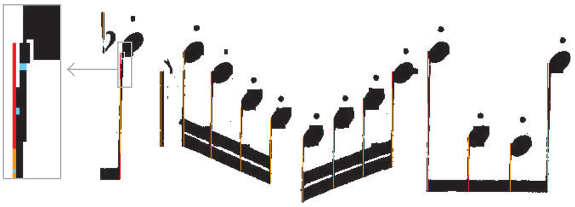


Figure 8: *Vertical segment detection*. In light blue the black pixels added through vertical closing, in orange the main black run and in red the run extensions corresponding to the merged black runs.

black run (at coordinate y) does not reveal the entire segment, and it is necessary to reconsider the nearby runs that are connected to the extremities of the detected one. The average thickness e_p of the segment is estimated, using the labelling image I_h, by considering only the pixels that are maxima (1.0) in image I_l:

$$e_p = \frac{1}{N} \sum_{\substack{x_h \leq x \leq x_b, \\ I_l(x,y)=1.0}} I_h(x, y), \tag{7}$$

$$N = \text{Card}\left\{(x, y)/x_h \leq x \leq x_b,\ I_l(x, y) = 1.0\right\}.$$

A run is considered as an extension of the detected one, if it is connected to it, and if the average thickness of the corresponding segment, computed also by (7), is almost equal to e_p. The process is iterated until no new run can be merged.

Figure 8 shows an example of the results we obtain on degraded images with skew and breaks. We can see that the vertical segments are properly detected despite these defects.

Compared to [27], the proposed method is simple and not computationally expensive, but it has proved to be accurate and robust enough, in the sense that all significant vertical segments are properly detected, except in very bad printing conditions with wide interruptions. In this case, we could relax item (5). The imperfections are the double detection of symbols, obviously in the case of sharp or natural, and sometimes, in the case of a very thick segment; typically some end bar lines are detected twice. This problem is generally solved at the next step of the segmentation. Indeed, the vertical segments are used as seeds of a region growing and image-labeling algorithm, and the relevant symbols are located by a bounding box. When two vertical black runs lead to identical bounding boxes, and if their width is less that the typical size of a symbol (less than 1.5 space), they are merged. When the bounding boxes are wider, it may be a case of bad connection between nearby symbols, and the ambiguity has to be solved at higher-level stages. The most important point is that no vertical tall symbol is overlooked, since the corresponding symbol would not be analyzed, and this error would never be corrected. Figure 9 shows some results. The detection of the vertical segment is precise; the bounding boxes are also generally correct, but some imprecision cannot be avoided, because of symbol fragmentation or symbols touching (see, e.g., the slur crossing the bar line, the forte symbol touching the beam, or the flat that is fragmented and is also touching the next note head). Therefore, bounding boxes are not strongly involved in the recognition of the symbols featured by a vertical segment, however they can be used to indicate the free spaces left between them, where rests can be found [25].

4.3. Beams detection

Beams are very difficult to detect and classify as primitive, since they dramatically vary in shape and size and are part of composed symbols. They are also prone to printing defects (inconsistent connections between them, as illustrated in Figure 3, variable thickness, disconnection from stems), and interfere with staff lines. They also may be assembled in different ways with the stems. Thus, rather than trying to segment and classify beams, we propose a solution that just checks the presence of a segment of adequate thickness, that connects the extremities of two note stems. Consequently, the method is applied after the classification hypotheses have been generated, for any pair of objects assumed to be both black notes (quaver, semiquaver, etc.). The proposed method is based on region growing and on a modified Hough transform. It has been designed to overcome the defects mentioned above.

Let us consider two vertical segments, assumed to be stems, and the corresponding black note heads. The position of the stems is known accurately (results of the vertical

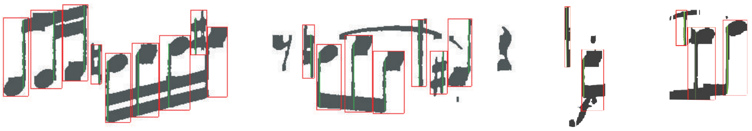
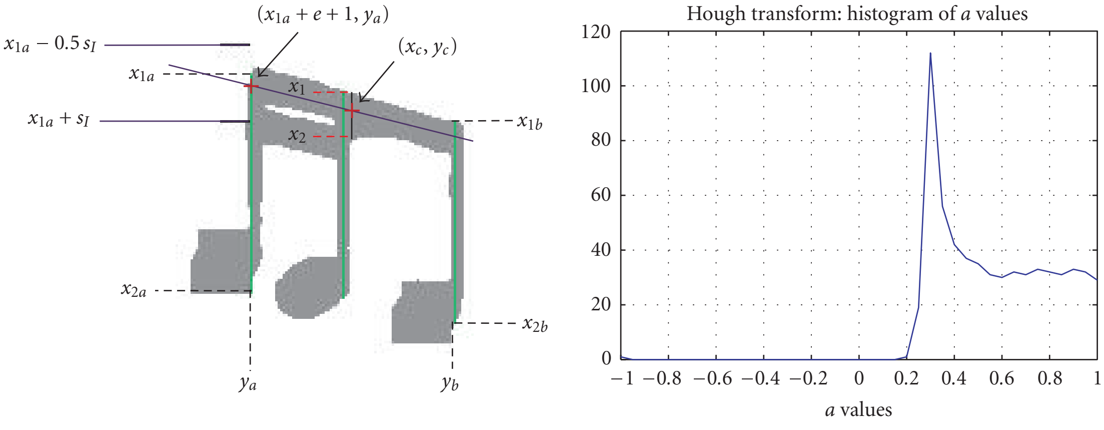


Figure 9: Segmentation of symbols featured by a vertical segment.



(a) (b)

Figure 10: *Detection of a beam connecting a stem in y_a and a stem in y_b*. (a) Parameters involved in the region growing process and modified Hough transform. (b) Hough accumulator leading to the beam slope value.

segment detector) and the coordinates are denoted by y_a, x_{1a}, x_{2a}, and y_b, x_{1b}, x_{2b} (Figure 10(a)). The position of the note heads is also known, as a result of the individual symbol analysis step (described below in Section 5.1). Based on this information, it follows that the beam has to be searched near the stem end, opposite to the note heads.

A region growing algorithm is applied from left to right, for increasing y coordinates. Let us consider the case where the note heads are both down. The seed is defined as the small pixel column at the y_a coordinate, between $x_{1a} - S_I/2$ and $x_{1a} + S_I$. It corresponds to the range where the most extreme beam is expected and it allows some breaks between the stem and the beam. Let us denote by x_1 and x_2 the extremities of the current pixel column at the y_c coordinate ($y_a < y_c < y_b$), that has been aggregated to the region. We compute (8) for each black point (x_c, y_c) ($x_1 < x_c < x_2$), the slope a of the line passing though this point and the extremity of the first stem $(x_{1a}+e+1, y_a)$. The parameter e represents the half minimum thickness of the beam, and is set to $0.25S_I$,

$$x = ay + b,$$

$$a = \frac{x_c - (x_{1a} + e + 1)}{y_c - y_a}, \tag{8}$$

$$b = x_{1a} + e + 1 - ay_a.$$

The values found for a are approximated to discrete values ranged from −1 to +1 (−45° to 45°). They increment the Hough accumulator (actually the histogram of a values, see Figure 10(b)) if the black run centred on (x_c, y_c) is longer than the minimum thickness $(2e + 1)$. At the end of the region growing process, and if the process has reached the y_b coordinate, the maximum value of the accumulator is searched for, leading to the optimal parameters a_{opt} (Figure 10(b)) and b_{opt} (8) of the beam. The last step consists in validating these parameters. The number of black pixels, located on the segment centred on the line $x = a_{\text{opt}}y + b_{\text{opt}}$ and of thickness $2e+1$, is counted. If the ratio expressed in (9) is greater than 0.8, then the presence of a beam connecting the two stems is validated. This threshold is not a sensitive parameter; it allows to deal with some beam defects compared with the theoretical model,

$$q = \frac{\text{number\_of\_black\_pixels}}{(y_b - y_a + 1)(2e + 1)}. \tag{9}$$

In order to increase the reliability of the method, two accumulators are in fact handled, corresponding to the line passing though the extremity of the first stem (y_a) as explained above, and in the same way, to the line passing through the extremity of the second stem (y_b). The one optimizing the ratio (9) is preferred and its parameters are stored in the data fields of the analyzed objects, indicating that both objects, with the assumptions that they are both black note heads, can be connected by a beam centred on this line. Thus, the method provides accurate results, once at most one stem is almost connected to the beam. Note that the detected beam is the most extreme one, and that other beams may exist but are not checked. These results will be further refined in order to compose groups of more than two notes and deduce the duration of each note (see Section 6.2.2).

5. INDIVIDUAL SYMBOL ANALYSIS AND GENERATION OF RECOGNITION HYPOTHESES

5.1. Template matching

Symbol analysis is mainly based on template matching. A set of models is used to compute the correlation between the class models and the segmented symbols. The models (Figure 11) are designed for the typical score size and the chosen scanning resolution, to avoid preliminary scaling. Other sets of models could also be integrated in the system, and selected depending on the staff space S_I. Let us define the correlation between the model k, M^k, of size $d_x^k \cdot d_y^k$ and origin (i_k, j_k), and the tested shape S, at the (x, y) position in the image I:

$$C_S^k(x,y) = \frac{1}{d_x^k \cdot d_y^k} \sum_{i=0}^{d_x^k-1} \sum_{j=0}^{d_y^k-1} M^k(i,j) \cdot I'\left(x+i-i_k, y+j-j_k\right) \tag{10}$$

with

$$M^k(i,j) = \begin{cases} -1 & \text{for a white pixel} \\ 1 & \text{for a black pixel} \end{cases}, \quad 0 \le i < d_x^k,\ 0 \le j < d_y^k,$$

$$I'(i,j) = \begin{cases} -1 & \text{if } I(i,j) = 0, \\ 1 & \text{if } I(i,j) = 1. \end{cases} \tag{11}$$

In the case of perfect superimposition between shape and model, the result will reach the maximum score of 1.0. The score decreases with the number of pixels that differ from the model. In template matching, the correlation is computed for several (x, y) coordinates. So, the highest score $C^k(S)$, obtained at (x_k, y_k), is a measure of similarity and of localization:

$$C^k(S) = C_S^k(x_k, y_k) = \max_{(x,y)} C_S^k(x,y). \tag{12}$$

Numbers of classification methods may be used, and the solutions proposed in the OMR literature are numerous. Most of them rely on subsegmentation of the composed symbols (the note groups), recognition of the segmented primitives, followed by a reassembly phase based on rules expressing structural criteria (relative position of primitives) (e.g., [7, 8, 12–15]). The primitives themselves (note head, vertical segment, tail, beams) and the other types of symbols (accidentals, rests, dot) can be classified in various ways. We can mention again structural methods based on the extraction of geometric or topologic features [8, 10, 11, 15, 18, 30] or projection profile analysis [9, 31, 32], skeleton extraction and analysis [29, 33], and methods that directly compare the unknown shape to models, typically neural networks [6, 19], or template matching [9, 32]. The structural methods need to locate precisely the shape, making them very sensitive to segmentation defects that result from undesired connections between distinct symbols, or symbol fragmentation. These segmentation problems are impossible to solve at this stage of the image analysis without any contextual information. That is why some authors need to imbricate segmentation and classification in a very complex process [7, 8] or introduce feedback in order to review some recognized symbols based on inconsistency detection at the semantic analysis level [15, 18, 34]. The main problem with these methods is that they are fundamentally based on local interpretations, and thus, do not take into account all the contextual information.

Because of these problems, we prefer the second type of method (template matching), combined with the robust symbol detection presented above. This methodology presents some powerful advantages.

(i) It does not need to locate precisely the analyzed shape. In particular, it can deal with fragmentation or undesired symbol connections, and provide meaningful results as long as the symbol is approximately localized, which is guaranteed in our case. The correlation score is simply computed on a small area deduced from the vertical segment, and the position where the maximum correlation score is obtained provides the exact position of the symbol. For the note groups, we can search for the note head at the extremities of the detected vertical segment, with some tolerance to deal with stem/note head disconnection cases. So, we can avoid complex methods of subsegmentation and reconstruction. Moreover, the computing area can be more restricted by considering some structural rules. For example, note heads and accidentals are placed on a staff line or on a staff space.

(ii) The class model is very easy to adapt to the analyzed score. A simple basic idea is to extract the different models from the analyzed score itself. It is also possible to store and test different models per class, corresponding to various publishings (Figure 11) [25]. So, the method is easily adaptable. Moreover, we will propose in the next section an automatic adaptation of the class models, based on the correlation scores obtained on the whole music sheet, as well as, in Section 8, a learning method allowing a better processing of a specific score. These proposals really take advantage of template matching.

(iii) The direct comparison of correlation scores can be used to extract several recognition hypotheses, with the position obtained for each one. All these pieces of information are used in the fuzzy model of symbol classes, as mentioned above, and in the evaluation of graphical rules related to the relative position of symbols.

Template matching is applied for the symbol featured by a vertical segment on a small area deduced from the position of the vertical segment. For all the other classes, except augmentation dots, the correlation is computed on the free space between the bounding boxes. In both cases, a priori knowledge about the possible symbol positions according to the tested class is used in order to increase the reliability and the speed of the analysis. The last case concerns the

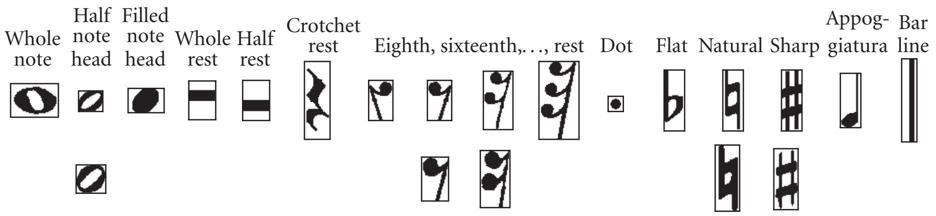


Figure 11: *Reference models M^k*. Up to two models are used for each class k.

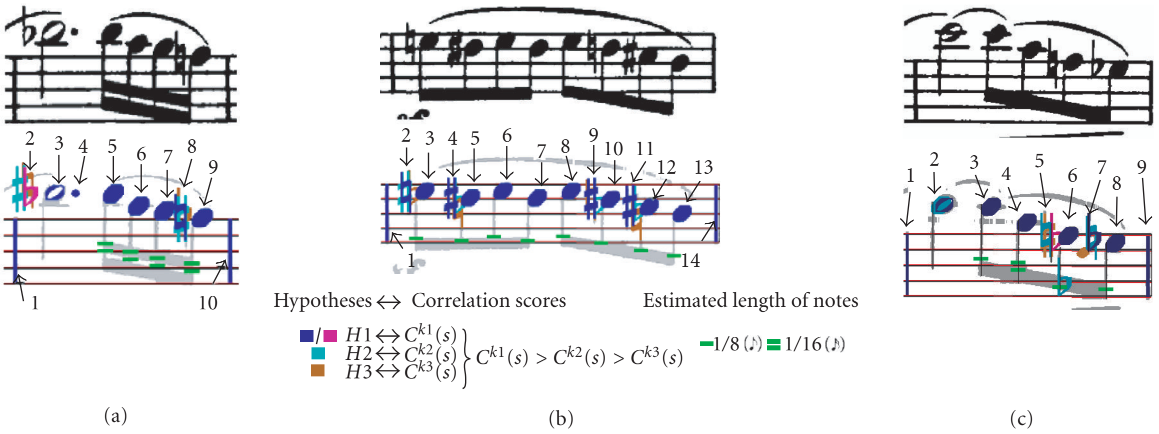


Figure 12: Original images and recognition hypotheses.

augmentation dots: the correlation is computed in a small searching area after a possible note head (Figure 15) or a rest.

Then, at most three recognition hypotheses (H1, H2, H3) are selected, based on the obtained correlation scores, each hypothesis assigning the pattern to a possible class. When the highest correlation score is less than the decision threshold $t_d(k)$, the possibility that there is no symbol ((−) in Tables 1 and 2) is also added as H0 hypothesis. The decision threshold values $t_d(k)$ are defined for each class k. They allow taking into account that some classes are more sensitive to typesetting variations (small value for $t_d(k)$) or have a higher probability of false detection (large value).

Figure 12 shows three bars, (a), (b), and (c), and the recognition hypotheses superimposed on the original images. Table 1 indicates some of the associated correlation scores.

These examples (Table 1) show the limits of the individual analysis: the correlation scores may be very ambiguous, and the highest one does not always correspond to the right hypothesis. The ambiguity results from symbol typesetting variability, printing, and segmentation defects, as illustrated above (Figures 3, 4, and 7); it is obvious that no individual decision can be taken at this stage. But we can notice that the correct solution is included in the set of hypotheses. So, the next stages aim at disambiguating these primary results by modeling all the information extracted from the score and the knowledge about music writing.

5.2. *Fuzzy model of symbol classes*

This section addresses specifically the problem of symbol variability and shape imprecision [1]. As the correlation scores provide similarities between each analyzed symbol and models, we define the degree of possibility $\pi_k(S)$ that S belongs to class k as an increasing function of $C^k(S)$:

$$\pi_k(S) = f_k(C^k(S)). \tag{13}$$

The shape of the possibility distribution for class k (Figure 13) is defined by two parameters, S_k and D. The first one is learnt from the correlation scores obtained on the whole score:

$$S_k = \frac{t_d(k) + D/2 + n(k)m(k)}{n(k)+1}, \tag{14}$$

where $n(k)$ is the number of objects having their highest correlation score with model M^k, this correlation score being larger than the threshold value $t_d(k)$. The average value $m(k)$ is computed from these scores, and represents a mean similarity degree between the objects of class k in the score and the reference model M^k of the program. So, the parameter S_k takes a large value when M^k matches closely the objects of the processed score, a small one otherwise. The second parameter, D (always 0.4), defines the width of the uncertainty area around S_k, in which the possibility degrees are strictly

Table 1: Some hypotheses and correlation scores.

	2	3	8
H0	(−)		
H1	♭ 0.66	𝅝 0.60	♮ 0.58
H2	♯ 0.43		♯ 0.48
H3	♮ 0.43		♭ 0.39

(a)

	2	9	11
H0			
H1	♮ 0.62	♯ 0.58	♯ 0.59
H2	♯ 0.52	♭ 0.58	♭ 0.49
H3	♭ 0.50	♮ 0.55	♮ 0.45

(b)

	2	5	7
H0		(−)	
H1	𝅝 0.59	♭ 0.56	♭ 0.72
H2	● 0.38	♮ 0.49	♮ 0.54
H3		♯ 0.42	♩ 0.43

(c)

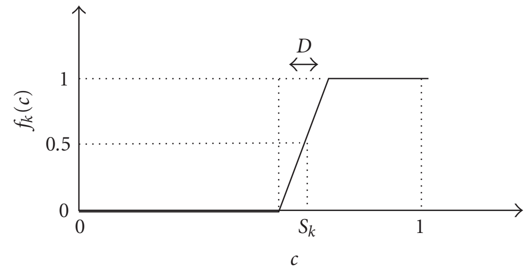


Figure 13: Possibility distribution of class k.

between 0 and 1. It models the range of the admissible variations of symbols of class k within the score. Thus, the possibility distributions allow modeling both inter and intra score symbol variability.

The shape of the distribution π_k does not need to be estimated very precisely [23]. It has experimentally proved to be robust. The most important is that it is not a binary function (there is no crisp threshold) and that it is increasing: the higher is the correlation score, the higher is the degree of possibility.

Table 2 shows the results obtained for the three bars of Figure 12. Compared to Table 1, the possibility degrees present less ambiguity. In particular, many hypotheses receive a zero possibility degree. It should also be noticed that the classification rank may change: see, for example, object 5 in bar (c), for which the possibility of a flat is now eliminated.

6. INTEGRATION OF GRAPHICAL AND SYNTACTICAL RULES

Until now, each object was processed individually. In this section, we introduce the graphical and syntactic relationships that are imposed by the notation (Section 2.1). As most of the music writing rules are flexible or contain imprecise parameters, they cannot be expressed in a crisp way, and are rather a matter of degree. Therefore, we define compatibility degrees to express these consistency rules.

6.1. Graphical consistency

Given a set of hypotheses, the aim is to compute the compatibility degree between each object and all the surrounding objects in the bar, according to their classes. Horizontal and vertical position criteria are first expressed separately and then combined together to form graphical compatibility degrees between two symbols (binary interaction). Positions extracted by template matching are intensively used for this aim. Then the results are merged to get graphical consistency degrees of higher orders, that is, involving more than two symbols. This hierarchical method allows comparing the global graphical consistency of different hypothesis combinations, which involve any number of symbols, which are more or less distant from each other.

All the graphical rules indicated in Section 2.1 have been modeled: the position of an accidental or an appoggiatura before a note head [1], the position of a point after a note head or above a note head, in order to disambiguate duration dots and staccato dots [26], the relative position of any other pair of symbols [26]. For the sake of clarity, we briefly summarize and illustrate how the first graphical rule is modeled, and how the final graphical compatibility of each object S_n is obtained, for each hypothesis configuration.

Rule (1) expresses that an accidental should be placed before a note and at the same height. The possibility degree that the object S_n is an accidental of class k_n, and that a following nearby object S_m $(m > n)$ is a note of class k_m, is a function of the compatibility degree $C_p(S_n^{k_n}, S_m^{k_m})$ between both symbols:

$$C_p(S_n^{k_n}, S_m^{k_m}) = \begin{cases} \alpha_l f_l(\Delta l) + \alpha_h f_h(\Delta h) & \text{if } f_l(\Delta l) > 0,\ f_h(\Delta h) > 0, \\ 0 & \text{otherwise,} \end{cases} \tag{15}$$

where f_l and f_h are two functions defining the admissible values of Δl and Δh, respectively, the differences in horizontal and vertical positions between S_n and S_m (Figure 14). This combination is a compromise between two criteria, excluding the cases where at least one is not satisfied at all. The chosen coefficients $\alpha_l = 0.2$ and $\alpha_h = 0.8$ express their relative importance. Using a degree between 0 and 1 instead of a crisp threshold on each criterion allows us not to discard completely an accidental which is not exactly at the theoretically expected position.

The compatibility degrees help to compare two competing hypotheses. For object 2 in bar (a) in Figure 12, we obtain

Table 2: Some hypotheses and possibility degrees.

	2	3	8
H0	(−)		
H1	♭ 0.17	𝅝 0.44	♮ 0.30
H2	♯ 0.00		♯ 0.07
H3	♮ 0.00		♭ 0.00

(a)

	2	9	11
H0			
H1	♮ 0.43	♯ 0.40	♯ 0.43
H2	♯ 0.20	♭ 0.00	♭ 0.00
H3	♭ 0.00	♮ 0.20	♮ 0.00

(b)

	2	5	7
H0		(−)	
H1	𝅝 0.35	♭ 0.00	♭ 0.40
H2	● 0.00	♮ 0.13	♮ 0.25
H3		♯ 0.00	♩ 0.00

(c)

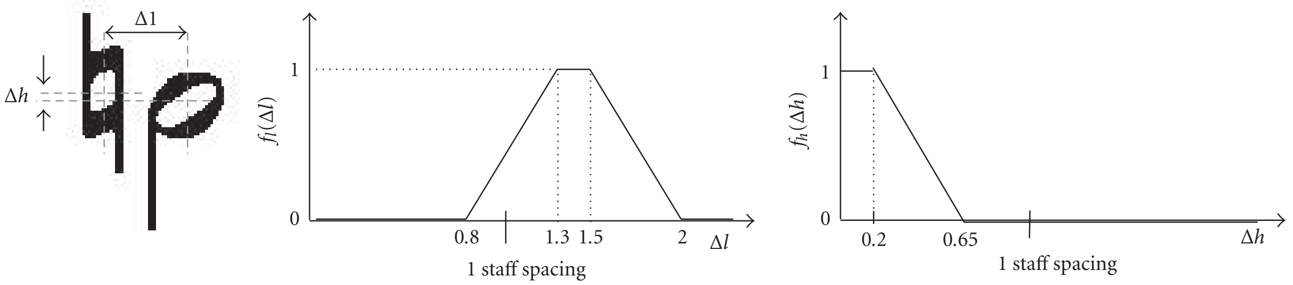


Figure 14: Accidental/note graphical compatibility.

a compatibility degree with the next note of 1.0 in the hypothesis of a flat, and of 0.77 in the hypothesis of a sharp. These results rank correctly the flat before the sharp and strengthen the right hypothesis H1. Most of the wrong accidental hypotheses are in this way discarded, but the graphical compatibility degree is not always sufficient. For object 9 in bar (b), the compatibility degree is equal to 0.0 for a flat, and this correctly discards this hypothesis, but it is equal to 1.0 for both sharp and natural. This second example shows that additional criteria have to be considered in order to decide between two equally possible and still ambiguous hypotheses.

Compatibility degrees are calculated for all pairs of nearby symbols, according to their classes [1, 26]. As there may be more than one nearby symbol before or after S_n, especially in case of high symbol density, a global graphical compatibility degree for each object S_n classified in class k_n, is computed:

$$C_p\left(S_n^{k_n}\right) = \left[\min_{j<n} C_p\left(S_j^{k_j}, S_n^{k_n}\right)\right] \cdot \left[\min_{l<n} C_p\left(S_n^{k_n}, S_l^{k_l}\right)\right]. \tag{16}$$

It is a conjunctive combination expressing that symbol S_n must be graphically compatible with all the previous and the next symbols. The use of the min t-norm operator, in each of both factors, allows to get graphical compatibility degrees that are comparable whatever the number of involved symbols. Then, the product t-norm provides more discriminating results than the min operator does.

This combination has proved to be very efficient, leading to very representative results on the global graphical consistency of the different hypothesis configurations, thus contributing to disambiguate them.

6.2. *Syntactic consistency*

In this section, we introduce the syntactic rules related to tonality, accidentals, and metric. As underlined in Section 2, these rules are very flexible and involve many symbols that may be far from each other. We propose to define compatibility degrees that evaluate each symbol or each group of symbols against the considered rule. The proposed method overcomes both difficulties stated above, and it is a very original and innovative point with respect to systems found in the literature.

6.2.1. *Tonality and accidentals*

The key signature is an input parameter of the recognition program. It is indicated in the score as an ordered sequence of accidentals (flats and sharps) placed just after the clef. This is a strict rule that always applies. So, we assign binary syntactic compatibility coefficients $C_S(S_n^k)$ to the accidentals found after the clef, equal to 1.0 when satisfying the rule, 0.0 otherwise.

The second rule concerns all the other accidentals in the score. If a symbol belongs to one of the accidental classes (flat, sharp, natural), it must be consistent with the key signature (tonality) and the other accidentals in the score. According to rules (6) and (7), and their flexible application, we have to consider the consistency between accidentals of same height (up to octave shifts): the accidental in the key signature, the immediately previous accidental in the same bar, and, if the latter does not exist, in the previous nearby bars. A syntactic compatibility degree is assigned to every accidental S_n, based on the configuration evaluation [23].

Table 3 provides the defined coefficients in the case where there is no accidental of same height (up to octave shifts)

Table 3: Syntactic compatibility coefficient between two accidentals in a bar, with no accidental of the same height in the key signature.

	S_n = sharp	S_n = natural	S_n = flat
S_m = void	0.75	0.5	0.75
S_m = sharp	0.5	1.0	0.0
S_m = natural	1.0	0.5	1.0
S_m = flat	0.0	1.0	0.5

in the key signature. They have been defined as follows: the most common configurations are when a sharp or a flat occurs for the first time in the bar, or when a natural cancels a previous flat or sharp. The first configuration gets a possibility degree equal to 0.75, so over the middle value, while the second configuration gets the maximum possibility degree, 1.0, in order to strengthen any consistent interaction bringing new information inside a bar. But it is also possible that the second accidental recalls the first one in order to make the reading easier. That is why the corresponding degree takes the middle value 0.5 reflecting that this configuration is possible but corresponds to usual but nonmandatory practice. The last possibility degree is 0.0 when a flat occurs after a sharp, or vice versa, because this configuration should not occur, at least in tonal music.

Let us now take an example, with accidentals 4 and 9 in bar (b) (see Figure 12(b)). Nine different combinations are possible and evaluated, based on Table 3, each of them leading to a syntactic compatibility coefficient for both objects (see Table 4).

This example shows how distant objects interact. The better average syntactic configuration is obtained for sharp/natural and flat/natural (0.75/1.0). But as the flat hypothesis for object 4 gets a graphical compatibility with the following note head equal to 0.0, against 1.0 for the sharp hypothesis, we can guess that the fusion of both graphical and syntactic criteria will lead to the correct solution: sharp for object 4, natural for object 9.

Other accidental combinations have been defined, taking into account the presence of the accidental in the key signature or in previous bars (refer to [1] for a complete description). To our knowledge, no other method of the literature models and integrates the accidental consistency in their recognition algorithms. However, it has proved to increase significantly the accidental recognition, since it allows a global evaluation of their mutual consistency, and takes into account the notation flexibility.

6.2.2. Meter

Meter is generally considered at the end of the recognition process, in order to detect and correct errors [13, 15, 17, 18]. The number of beats per bar (Rule (4)) is checked for this goal, since it is a strict rule that has always to be satisfied (up-beat excepted). Then, additional criteria, such as vertical alignment may be used in a post correction process. We propose to integrate Rule (5) about note grouping in the recognition process itself, combined also with the strict Rule (4), in order to increase the reliability of note-length interpretation.

The method proposed in [1] has been modified in order to deal with groups including rests. Its reliability has also been improved, thanks to the accurate beam detection method, presented in Section 4.3. The parameters a and b of the thick line linking the extremities of two beamed stems are first found. These first results are used to compose larger groups. In bar (c), for example, (Figure 12(c)), objects 3, 4, 6, and 8 are all assumed to be filled notes, and are detected to be connected in pairs. Consequently, they must form a single group of four notes and this larger association is checked. The results obtained for each beam portion are at the same time refined, so that the most extreme beam is precisely located. Then, each note length can be obtained by simply counting the number of beams on the left-hand side and/or the right-hand side of the stem, the analyzed cross-section being determined by the detected beam (a, b), the note head position (x_0), and the stem position (y_0) ((17), Figure 15(a)). Some additional criteria are also defined to deal with connected beams: for example, when the thickness of the detected beam is greater than $1.2S_I$, it is counted as two beams,

$$\begin{aligned} y &= y_0 \pm 0.25S_I, \\ x_{11} &= a^* y + b + 0.3S_I, \\ x_{11} &= \max\left(x_0 + S_I, x_{l1} - 3S_I\right). \end{aligned} \tag{17}$$

It should be noticed that the method is applied for every combination of hypotheses found for a bar, but that all results obtained for a given note group are stored in the data fields, so that identical configurations are not processed twice.

Notes that are not connected to others are assumed to be crotchets or isolated quavers. No specific treatment is implemented in order to recognize flags. However the duration of isolated notes can also be found by counting the number of possible flags at the end of the note stem [25], as illustrated in Figure 15(b). The analysed cross-section is defined from the stem location and the note head position (x_{11} is then equal to the stem extremity coordinate in (17)).

Dots, which are searched for after every note head (Figure 15), are then attached to the note heads. Eighth rests and shorter rests have also to be considered, since they may be included in a note group. When a rest is inside a note group, it can be included in it without any doubt. But when it is just before it, we have to consider two cases: the one for which the rest is outside the group and the one for which it is part of the group. The total length of each formed group is then computed. If it does not reach a conventional value (Rule (5)), its rhythmical internal organization is compared with the usual ones, according to the time signature. At most two hypotheses are made, that increase or decrease the total length of the group, while changing the smallest number of note lengths and respecting the dots and the rests of the group. Altogether, there are at most 5 hypotheses H^l per group: the initial interpretation, two possible corrections for the group without any rest just before it, two possible

Table 4: *Examples of compatibility coefficients obtained for measure (b) and objects 4 and 9*. Shaded boxes correspond to accidental hypotheses with a graphical compatibility coefficient with the next note head equal to 0.0.

4	9	$C_s(s_4^k)$	$C_s(s_9^k)$
♯	♯	0.75	0.5
♯	♭	0.75	0.0
♯	♮	0.75	1.0

4	9	$C_s(s_4^k)$	$C_s(s_9^k)$
♭	♯	0.75	0.0
♭	♭	0.75	0.5
♭	♮	0.75	1.0

4	9	$C_s(s_4^k)$	$C_s(s_9^k)$
♮	♯	0.5	1.0
♮	♭	0.5	1.0
♮	♮	0.5	0.5

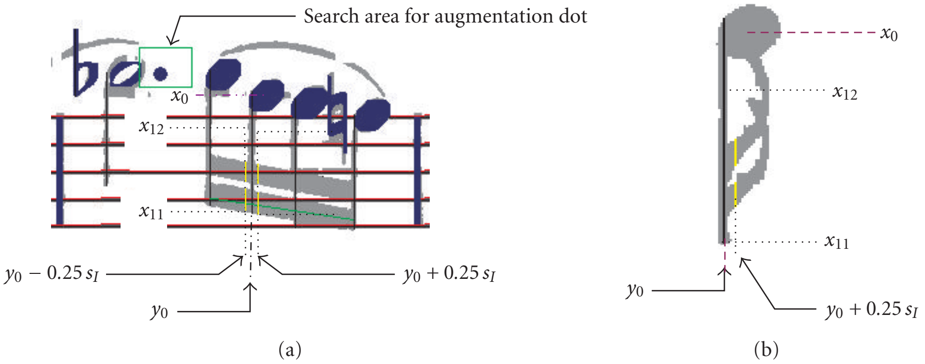


Figure 15: *Beam (a) or flag (b) detection for filled note length evaluation*. In yellow the detected vertical segments counted as beams or flags.

corrections for the group including the rest just before it, if existing. Let $L(g)$ be the number of notes and rests in the group g, and $l(g)$ the number of notes whose beam count is changed. The possibility degree assigned to each group hypothesis H^l is computed as

$$C_l^{H^l}(g) = \pi_l^{Hl}(g)\left(1.0 - \frac{l(g)}{L(g)}\right). \tag{18}$$

The first factor, $\pi_l^{Hl}(g)$, evaluates the group consistency: it is equal to 1.0 when the total group length is normal (so, for all proposed corrections), equal to 0.5 otherwise. When a rest is before a beamed note group, the initial interpretation gets a $\pi_l^{Hl}(g)$ coefficient equal to 0.5, if none of the possible groupings (with or without the rest) seems to be normal, 1.0 otherwise. In this last case, possible corrections are proposed for the unusual configuration. For example, $\pi_l^{Hl}(g)$ is equal to 1 for an initial interpretation such as 𝄾♬ (1/4), but other possibilities are proposed when considering just the 3 beamed notes ♬ (3/16). The second factor has another interpretation. It expresses that the more the new interpretation differs from the initial one, the more the possibility degree decreases. Thus, the product of both terms leads to prefer initial configurations that can be correct, or corrected groups with few corrections. Isolated notes are not affected by this process, and their possibility degree is always set to 1.

For example, in bar (c) of Figure 12, the duration of the second note is false, and the total length of the group is equal to the unusual value, 7/16 (1/8 + 1/16 + 1/8 + 1/8). The possibility degree of this initial configuration is 0.5 and the following corrections are proposed: (1/8 + 1/8 + 1/8 + 1/8) = 1/2, with a possibility degree equal to 0.75, and (1/16 + 1/16 + 1/16 + 1/16) = 1/4 with a possibility degree equal to 0.25. So, the first correction (1/8 + 1/8 + 1/8 + 1/8) = 1/2 (the correct solution) is preferred to the other configurations, and Rule (4) applied to the whole bar will also confirm this hypothesis in the next stage.

The proposed method has proved to be very efficient (Figure 16), especially for n-tuplets interpretation, more rhythm models being now integrated with respect to [1]. Its efficiency has been increased, due to the new beam detection and counting method, which provides reliable preliminary results, so that only isolated errors need to be corrected. Thus, corrections can generally be proposed without ambiguity. Groups of two notes, or more than 4 notes, with just one misinterpretation are generally free of ambiguity. Errors in three note groups are more difficult to solve. For example, (1/16 + 1/8 + 1/8) = 5/16 can be corrected in either (1/16 + 1/16 + 1/8) = 1/4, (1/16 + 1/8 + 1/16) = 1/4, or (1/12 + 1/12 + 1/12) = 1/4 [1], with a possibility degree equal to 2/3 for all. In such ambiguous situations, the most frequent group is preferred, that is, (1/16 + 1/16 + 1/8) = 1/4 in this example. Altogether, the number of erroneous corrections is insignificant compared with the number of good corrections.

7. FUSION AND DECISION MAKING

The fuzzy model described in Sections 5.2 and 6 leads to a set of possibility degrees evaluating recognition hypotheses and their mutual consistency according to musical rules. The next step consists in merging all these pieces of information and in searching for the optimal configuration according to all criteria, in order to take a decision that is consistent with the music writing. This global evaluation is an important and powerful feature of the proposed recognition method, since symbols are highly interdependent as explained in Section 2.2.

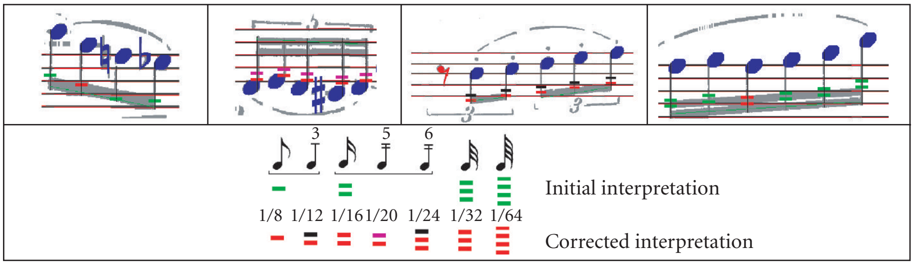


FIGURE 16: *Examples of groups with length correctly interpreted thanks to Rule (5). In red the lengths that were not correct after the beam counting process*. In the second bar, for example, the notes initially interpreted as semiquavers (1/16) are corrected into a quintuplet of semiquavers (1/20).

In order to decrease the complexity, the global optimization problem is divided into subproblems, where optimization is performed in each bar separately. This decomposition is very natural because symbols are interrelated by musical rules especially at the bar level. All configurations of recognition hypotheses are sequentially considered and evaluated. A configuration j is composed of a set of $N(j)$ objects S_n ($n = 1,\ldots,N(j)$) assigned to classes $k(n,j)$. For this configuration, several length hypotheses H^l are also made, combining together the different hypotheses made on each of the $N(j,H^l)$ groups g ($g = 1,\ldots,N(j,H^l)$) of notes and eventually rests. Such a configuration will be referenced as (j,H^l) in what follows. The decision process is divided into two steps:

(i) fusion of all possibility degrees and compatibility degrees,

(ii) decision by maximization of the resulting global function, and bar length checking.

7.1. Fusion

The general consistency of the configuration is first checked: every duration dot must be in the search area of a note, every accidental (tonality excepted) must be followed by a note, and every symbol must have a nonzero graphical compatibility degree. All the configurations that do not satisfy these preliminary criteria are inconsistent and therefore immediately discarded. For all the others, the possibility degrees and compatibility degrees are merged step by step to provide the final possibility degree $\mathrm{Conf}(j,H^l)$ of the configuration.

The global compatibility degree $C_t^{(j)}(S_n^{k(n,j)})$ of an object n, classified in class $k(n,j)$ in the configuration j, with all the other objects of the configuration, is deduced from its graphical compatibility coefficient $C_p(S_n^{k(n,j)})$ (16), and, for accidentals exclusively, from its syntactic compatibility degree $C_S(S_n^{k(n,j)})$ (Section 6.2.1). The global compatibility degree is then merged with the possibility degree of membership to a class $\pi_{k(n,j)}(S_n^{k(n,j)})$ (13), using a product, and all these results are averaged to provide the global possibility degree for the hypothesis combination j (19). This fusion method leads to a better discrimination than the one proposed in [1]:

$$\mathrm{Conf}_r(j) = \frac{1}{N(j)} \sum_{n=1}^{N(j)} \left[\pi_{k(n,j)}\left(S_n^{k(n,j)}\right) C_t^{(j)}\left(S_n^{k(n,j)}\right)\right]. \qquad (19)$$

Then we have to combine the hypotheses made on the length of the beamed notes, in order to express the global possibility degree $\mathrm{Conf}_l(j,H^l)$ of the $N(j,H^l)$ note/rests groupings. The fusion method relies again on an average, and we refer to [1] for a complete explanation.

The final function combining all the possibility degrees and compatibility coefficients for the configuration (j,H^l) is given by

$$\mathrm{Conf}\,(j,H^l) = \mathrm{Conf}_r(j)^*\,\mathrm{Conf}_l\,(j,H^l). \qquad (20)$$

It is the product of two factors, expressing that both criteria, the consistency of the recognition hypotheses and the possibility degree of the length hypotheses, have to be simultaneously satisfied. The use of the product t-norm instead of the minimum for instance makes this rule more severe.

7.2. Decision

The total length of the bar, denoted by $D(j,H^l)$, is the sum of the length of the note/rest groups in the configuration (j,H^l), and of the length of the isolated rests (only depending on j). The decision algorithm chooses the configuration (j,H^l) which meets at best two decision criteria, which are by priority order

(i) the total length $D(j,H^l)$ of the bar is correct,

(ii) the $\mathrm{Conf}(j,H^l)$ function is maximized.

This means that the algorithm chooses among the configurations matching the time signature the one which reaches the highest score $\mathrm{Conf}(j,H^l)$. If no configuration satisfies the strict length constraint, the algorithm retains the configuration maximizing $\mathrm{Conf}(j,H^l)$.

So the strict rule (Rule (4)) concerning the metric is expressed in the last step. It should be noticed that all rules mentioned in Section 2.1 have been integrated in the decision process. The fuzzy model allows merging both graphical and syntactic criteria, which involve different numbers of symbols, close or far from each other. Thus, the strong ambiguity on the symbol classes is reduced, leading to recognition results that are consistent with the notational syntax. The drawback of the method is its computational cost, especially for bars including many symbols, since the number of configurations grows exponentially (product of the number of hypotheses generated per symbol). It is statistically acceptable, around 350 per bar. When it is too high, we just discard the "less possible" class hypotheses. In the current version of the system, the average processing duration for one music sheet (10 staves) is around 5 seconds on a Pentium 4 3.2 GHz. But some graphical and syntactical criteria can be added in order to implement heuristics, that reduce more drastically and intelligently the computational cost of the decision making step.

8. IMPROVEMENTS AND GAIN IN ROBUSTNESS

We present in this section two improvements that allow an easier use and a better reliability of the system. The first one is the automatic indication of potential recognition errors, based on the results obtained during the analysis. This is an innovative feature that is very important since it is a very tedious work for the user to check one by one all the recognized symbols, even if the average recognition rate is good. The second improvement concerns adaptation procedures that allow refining the symbol models after a recognition/correction cycle made on a short excerpt of the score, so that the reliability is significantly increased on the rest of the score. As already mentioned, this option may be very interesting when large scores are processed, since the time spent for supervised learning compensates largely the time wasted for painful manual corrections. One could also imagine that the learnt features can be stored and reused when similar scores provided by the same publisher are processed. These two improvements constitute an important advanced step leading to an OMR system with enhanced practical use convenience.

8.1. Potential error indication

There are three types of errors: symbol missing, symbol confusion, added symbol. Results provided by the recognition process are reused to analyze the solution output by the decision process: the considered criteria are the class possibility degrees, the graphical compatibility degrees, and the rhythmical decomposition of the measure.

As most authors, we first check that the total length of the bar matches the time signature. All bars that do not satisfy Rule (4) are indicated as potentially false. The rhythmical decomposition of all the others is then studied. It is already nearly known, through the note group detection and their association with rests. Based on this information, we define a decomposition step, which is equal to 1/4 or 1/8 in a binary metric, 3/8 or 1/8 in a ternary metric. Isolated rests and notes are processed to complete the rhythmical decomposition of the whole bar, so that each association tends to a total length equal to a multiple of the step. All associations that cannot match this ideal decomposition are assumed to be impossible and indicated as potentially erroneous. The internal rhythmical structure of the others is compared with the usual groups. The associations that are possible but not usual are also indicated as potentially false, with another code to distinguish them from the impossible ones. The criteria are based on the time signature and the length repartition in the group. For example, a group equivalent to 3 quavers is common in a ternary metric, possible but unusual in a binary metric. A group such as (3/16, 1/16, 1/8, 1/8) is considered to be possible in a binary metric, but really unusual, and may result from the adding of an inexistent augmentation dot. So the indication of possible errors will be a great help to detect both error length and false augmentation dot.

Possibility degrees computed in the fuzzy modeling step are also reused as confidence measures. Let us consider again each symbol S_n^k, classified in class k, with a possibility degree $\pi_k(S_n^k)$ (13). Small values for $\pi_k(S_n^k)$ can reveal a misclassification. Consequently, the following rule is applied: if $\pi_k(S_n^k) < t_S^k$, then the symbol S_n is indicated as potentially incorrect. The thresholds t_S^k have been learnt, so that the correction rate is maximized and the "false alarm" rate is less than 1% of the total number of symbols. The learning base is made of half the total score database and is representative of the different publishings included in it.

The last rule is defined in order to indicate some missing symbols, from the hypotheses that have been left aside by the decision algorithm. If a symbol hypothesis gets a nonzero $C_p(S_n^k)$ compatibility degree (16) with the symbols of the chosen configuration, then it is indicated as potentially missing. Another criterion has been added, in order to decrease the number of false alarms, due to confusions with some inscriptions: the symbol must be located on the stave, within a margin of 2 staff spaces.

8.2. Supervised learning of symbol models

Symbols may vary from one publishing to another. Some classes, such as filled note heads or flat, are quite invariant, but others, for example hollow note heads or sixteenth rests, may strongly vary in shape. Two ways are proposed to deal with different fonts: firstly, up to two class models (Figure 11) are tested at the beginning of the individual symbol analysis step, and the most suitable is retained after some processing [25]; secondly, the fuzzy symbol class model allows to adapt the class model to the processed score (Section 5.2). Thus, the recognition process, in its standard application mode, is automatically adaptable and free of user interventions. But, these automatic procedures are not sufficient and cannot work properly when the reference class models are too different from the symbols in the analyzed score. Some learning procedures are in this case required.

The user is invited to select a set of staves, from which the symbol classes can be learnt, and to correct manually each mistake made by the standard recognition process. So, a set of known prototypes is available, with their coordinates in the music sheet, and this prototype list is then passed to an automatic learning procedure, whose purpose is to extract class

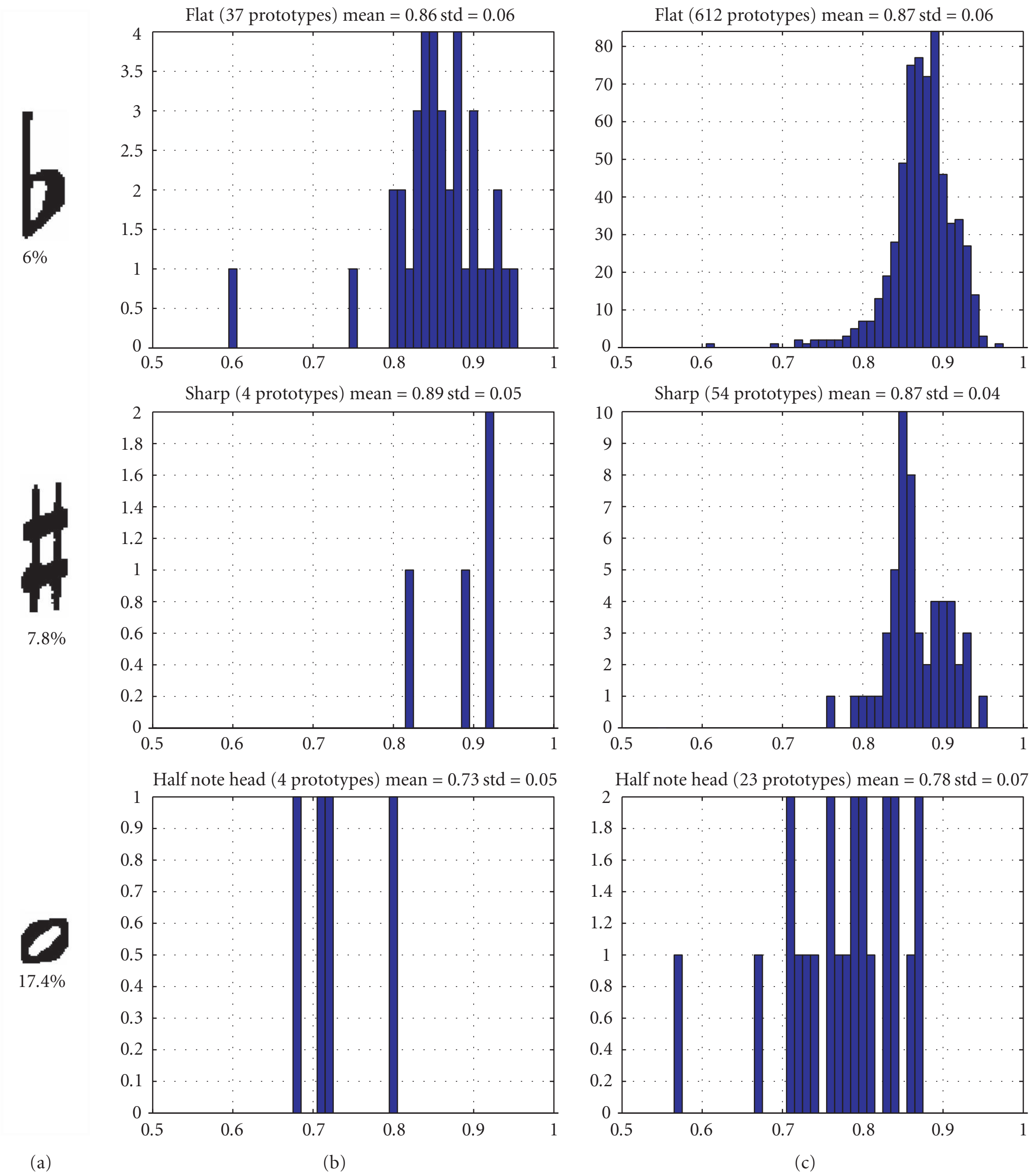


Figure 17: *Examples of symbol model learning.* (a) The learnt model M_a^k for class k and the proportion of symbols used as prototypes in the learning procedure. (b) The histogram of the correlation scores between M_a^k and the prototypes of the learning database. (c) The histogram of the correlation scores between M_a^k and the symbols in the whole score.

models, representative of the processed score, and to deduce some recognition algorithm parameters: the thresholds $t_d(k)$ used in the hypothesis generation (Section 5.1), and in the definition of the possibility distributions $\pi_k(S)$ (Section 5.2).

In a first step, the prototype list is scanned object per object, and a standard reference model M^k is correlated with the symbol in the image around the coordinates stored in the list (10). The image staff lines have been removed beforehand according to Section 4.1. A small image is extracted, based on the position where the maximum correlation score is found. For each class k, the average of all the extracted small images is computed and then binarized, using a threshold equal to 0.5. The resulting binary image M_a^k is assumed to be a representative model for class k (Figure 17).

Nevertheless, some precautions must be taken in order to reach the expected results. Firstly, the symmetry of note head (filled notes, half notes, whole notes) must be respected. Therefore a central symmetry relatively the image origin is performed on each extracted image and the symmetric is also integrated in the average. Thus, the method works perfectly, whatever the proportion of stems up and down. Secondly, some specific procedures must be applied for hollow objects, that may be damaged by the staff line removal process (half note heads, whole note heads, flats). For these classes, two average images are considered, the first one for the symbols put on a staff line, the second one for the symbols between two staff lines. Both images are independently binarized, and then combined together to produce the final class model: a logical AND is performed for the pixels that are never suppressed by the staff line removal process, while only the first image is used to provide the other pixels.

In the second step, the algorithm parameters that are related to the class models are tuned, that is, the decision thresholds $t_d(k)$ used in the generation of recognition hypotheses, and, as a consequence, the possibility distributions $\pi_k(S)$. The correlation is again computed between the new class models M_a^k and the image without staff lines. Let us denote by $C_k(S_n)$ the correlation score between the model M_a^k and the nth prototype ($0 \leq n < N_k$) of class k. Due to the symbol variability inside the image, some correlation score variations are still observed. The mean value C_k^m and the minimum value $\mathrm{Min}(C_k(S_n))$ obtained on the set of prototypes provide a good estimation of the variation range. The minimum correlation score $t_d(k)$ that has to be reached, in order to store class k as first hypothesis H1 without H0 hypothesis (Section 5.1), is set according to the following equations:

$$C_k^m = \frac{1}{N_k}\sum_{n=0}^{N_k-1} C_k(S_n),$$
$$t_d(k) = \mathrm{Max}\left(C_k^m - \frac{D}{2}, \mathrm{Min}\left(C_k(S_n)\right)\right). \tag{21}$$

The $t_d(k)$ parameters are clipped to $C_k^m - D/2$ ($D = 0.4$, see Section 5.2) in order to avoid setting too low thresholds due to strongly damaged isolated symbols, D representing the maximum correlation score variation range around the mean value that can be observed in practice.

In the generalization stage, the possibility distribution for class k will still be learnt from the individual symbol analysis made on the complete processed music sheet, according to (14) and Figure 13, but with the decision threshold $t_d(k)$ defined in (21).

The method leads to satisfactory results if enough symbols N_k per class are learnt. In our experiments, we can notice that $N_k = 5$ prototypes per class (symmetric prototypes included) are sufficient to get significant parameters. But obviously, the more the prototypes, the more reliable will be the results. Figure 17 illustrates the learning procedure applied to a large score (16 music sheets, 172 staves), where some symbols, especially the sharps, got unsatisfactory recognition rates with the recognition procedure without the learning phase. Eleven staves were used for the supervised learning. Comparison between the correlation score distributions for the learning data base and for the whole score shows that the symbol models are representative of the processed publishing. These distributions also confirm that symbol variability is still present in the score. Consequently, fuzzy modeling of the symbol classes and music rules are still necessary.

All other parameters of the recognition process do not depend directly on the class model and are left unchanged. The adapted recognition algorithm applied in generalization is exactly the same as the standard one, but uses the learnt set of models M_a^k instead of the default ones, and the related parameters $t_d(k)$. Recognition results, before and after supervised training, are provided in Section 9.

9. EXPERIMENTAL RESULTS

Tests of the proposed method have been conducted on a large music score database (100 music sheets, about 48000 symbols) by various composers and publishers. The base includes examples of various levels of difficulty in terms of symbol density, rhythmical complexity. No key change occurs in the tested sheets, since it is not yet handled by our system (clef, tonality, and metric are given as input parameters). Original scores were of good quality, without physical degradations, but many of them show printing defects and symbol variability such as the ones illustrated in Figures 3 and 4. The scanning has been performed carefully on three different scanning devices, and no specific processing was then applied to enhance the image quality. The tests have been performed without any parameter modification. So care has been taken not to train on specific cases, and we believe that this database is general enough to illustrate the strengths and the performances of our approach.

The evaluation of an OMR system is not straightforward. Indeed recognition rates may not be absolutely significant, since primitives may be correct while semantic is incorrect, and some proposals are currently made in order to define a standard evaluation method, dealing with this difficulty, and that allows comparing different systems. According to [35], our evaluation is made at both symbolic and semantic levels. We first calculate the recognition rates of all the recognized symbols: notes (composed of a note head, whose position relative to the staff lines has been calculated, a stem, and beams or tails), accidentals, whole notes, rests, and dots. This first evaluation is not far from the semantic level: the last step consists in attributing accidentals and dots to notes, dots to rests, in order to retrieve pitch and duration. This has been already realized in our system, since structural, graphical, and syntactic information have been integrated in the recognition process (Sections 6.1, 6.2). The ultimate step consists just in propagating some information over the bars (clef, meter, key, accidentals), and this is obvious in monophonic music. Thus, the semantic analysis is ambiguity free in our system, and the correct semantic can be regenerated by simply correcting one by one the following mistakes: symbol added, symbol missing, confusion, false note length interpretation, false interpretation of the note head position relative to the staff lines. These results are presented in Table 5 and commented below. They are completed by an estimation of the note identification rate, representative of the quality of the final interpretation.

Referred to the total number of symbols, the average recognition rate is now 99.2%. The symbol error rate (0.8%)

Table 5: Recognition rates per class.

Class	$r_k(k)$		$r'_k(k)$
|	99.60		0.33
♭	98.92	97.06$^{(1)}$	0.18
♮	97.65		0.08
♯	98.95	98.33$^{(1)}$	0.00
♩	64.55		8.68
•	97.48		0.70
𝅀	100.00		0.00
𝄿	90.67		0.00

Class	$r_k(k)$	$r'_k(k)$
𝄾	97.38	0.75
𝄽	–$^{(2)}$	0.48
𝄽	99.40	3.11
𝄼	100.00	4.05
𝄻	98.62	3.45
●	99.95	0.05
𝅗	98.00	0.52
𝅝	97.70	0.00

(1) The rate in the left column is obtained for all accidentals; the rate in the right column excludes key signature accidentals.

(2) The crotchet rest never occurs in the database.

is split into confusion (0.2%) and symbol missing (0.6%). There is also 0.3% of added symbols which generally result from confusion with inscriptions that have not to be recognized in our system (such as textual annotations or indications for the interpretation). The length of the filled notes is correctly interpreted for 99.3% of them, and the estimated position of the note heads on the staff is exact for 99.0% of the notes. Errors are due to the space variations between the additional lines placed above or below the staff: these defects are not yet handled by our method. Notes on the staff itself are correctly interpreted, thanks to the accurate staff line detection.

The recognition rate has been slightly increased with respect to [1, 26], although we have introduced more difficult scores. This is due to the improvements realized at the preprocessing stage, especially the ones described in this paper: staff lines tracking, robust detection of vertical segments. They both contribute to improve very significantly the segmentation and, as a consequence, the recognition reliability of the imperfectly printed music scores. The fuzzy model has also been completed and finalized with respect to [1, 26]. The graphical modeling part now involves all the symbols in the bar and it has proved to be very efficient especially in case of high symbol density. The note group model, that has been extended to rests, combined with the robust beam detection, has also contributed to reach better results, for both note length interpretation and symbol recognition. Figure 18 shows the histogram (normalized to 100%) of the recognition rates obtained per music sheet. One third of them get a global recognition rate higher than 99.75%, and all the recognition rates are larger than 91.0%, proving the reliability of the proposed method. When excluding the appoggiaturas, which have the worst recognition rate (64.55%, see Table 5) and are ornaments rather than mandatory musical symbols, the results are better: more than 40% of the music sheets obtain a global recognition rate above 99.75%, and the worst recognition rate is higher than 94.0%.

Let us now analyse in more detail the results. Table 5 provides the recognition rates obtained for each class of symbols and the rates of added symbols. All are normalized by the number of occurrences belonging to the class

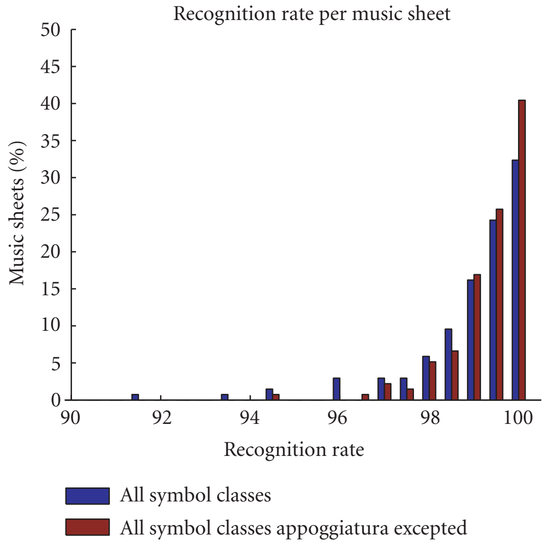


Figure 18: Histogram of the recognition rates obtained per music sheet.

$$\begin{aligned} r_k(k) &= \frac{\text{number of occurrences of class } k \text{ correctly recognized}}{\text{total number of occurrences of class } k} * 100, \\ r'_k(k) &= \frac{\text{number of added occurrences of class } k}{\text{total number of occurrences of class } k} * 100. \end{aligned} \tag{22}$$

All the recognition rates, sixteenth rest and appoggiatura excepted, are above 95%. Appoggiaturas are really prone to errors, since they show a high variability and they are involved in just one loose graphical rule; so disambiguating them is not straightforward. We can also observe high rates

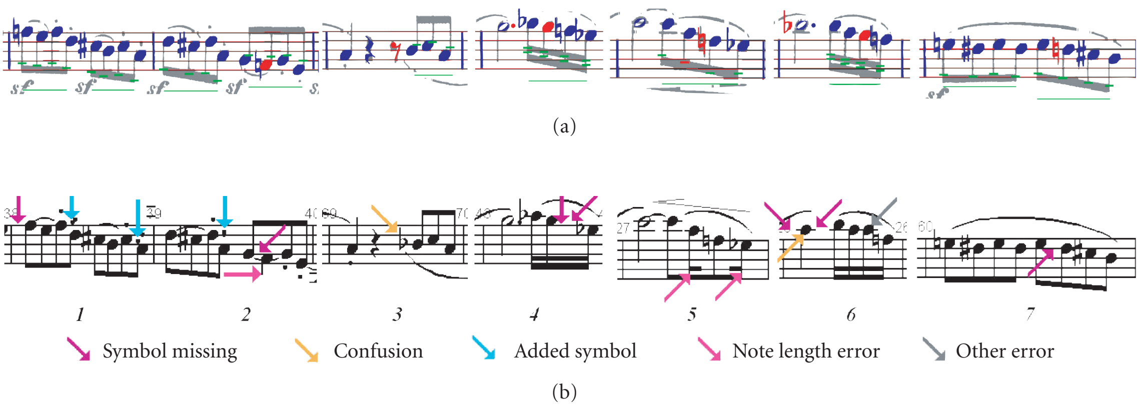


Figure 19: Comparison with the commercial software SmartScore [4]. (a) Results with our method: 0 error. (b) Results with SmartScore: 16 errors.

of added rests. This is generally a consequence of the strict metric rule applied to pick-up measures or upbeats, leading to confusion with some other inscription, in order to satisfy the rule.

The final interpretation of the notes is exact for 97.8% of them. The errors are due to the bad estimation of the position relative to the staff (1.0%), an indirect error, such as an accidental or a bar line missing (0.3%), a false flag or beam interpretation (0.7%), an augmentation dot error (0.2%).

Comparison between the results provided by our method (Figure 19(a)) and those output by SmartScore 3.2 Pro Demo [4] (Figure 19(b)), one of the most efficient commercial softwares, shows that the proposed method is able to solve problems for which SmartScore fails. For example, there are some confusions between staccato dots and duration dots with SmartScore (bars 1, 2), while we avoid this problem thanks to our graphical model. We can also point out that symbols touching each other are often not recognized with SmartScore (bars 2, 4), while our program models such configurations and performs well. Accidentals are better recognized thanks to the graphical and syntactical model (bars 2, 4, 6, 7). Especially, the second natural in bar 7 (symbol 9 in bar (b) of Figure 12) is correctly recognized, thanks to the syntactic rule that expresses its consistency with the previous sharp, while it is suppressed by SmartScore. Lastly, the symbol classes and syntactic consistency models allow avoiding the confusions made by SmartScore on the eight rest (bar 3), the half note (bar 6), or the filled note length (bar 5). Examples were extracted from two music sheets. The global symbol recognition rate is 92.0% for SmartScore, 98.7% for our method. 85.3% of quavers get the correct length with SmartScore, 99.3% with our method.

At this stage of the proposed method, one possible way that could improve the results could be adding some structural criteria in the symbol analysis process. Correlation scores provide a global similarity measure between two shapes. But results are ambiguous because of the symbol variability, and also because some musical symbols are highly correlated: for example natural and sharp, eighth rest and sixteenth rest, black filled note head and hollow note head. Symbol variability results in a relatively small correlation score between a processed symbol and the corresponding class model, while intercorrelation results in relatively high correlation scores between the processed symbol and other models of different classes. Adding some structural features could help to disambiguate between two hypotheses. For example, a hollow note head has white pixels inside it, while filled note head should not have any, and this simple criterion can help to prefer one class to another, and consequently decrease the confusion rate between both symbols. This proposal does reduce the interest of using template matching in any way, and all the arguments exposed in Section 5.1 still hold. The idea is just expressing explicitly useful information that is more or less hidden in the correlation scores. This structural information will have to be integrated in the fuzzy model in order to avoid crisp decision criteria that cannot deal with printing defects and segmentation imprecision.

The proposed method for potential error indication has also been evaluated: 84% of the detectable errors (symbol added, symbol missing, confusion, note length error) are well indicated, 52% directly, 32% indirectly (a correct error indication has been made within the same bar). The false alarms represent 2.5% of the total number of symbols, which is acceptable. From a practical point of view, the processing of a music sheet of 10 staves leads in average to more than 5 well indicated errors, 1 nonindicated error, and less than 10 false indications. Given that such a music score contains more than 400 symbols, we can consider that the proposed method is a real help, making the manual post-correction much easier. The detection of potential pitch errors should be added in order to complete the process.

Figure 20 gives some examples of error indications. We can see that confusions, missing or added symbols are correctly indicated (bars 1, 2, 3, 5), except one false alarm (bar 2). Length errors in bars 2 and 6 are also well indicated thanks to the indication of false group (bar 2), or potentially false group (bar 6). All other errors in these examples are easily corrected thanks to the indications made on surrounding

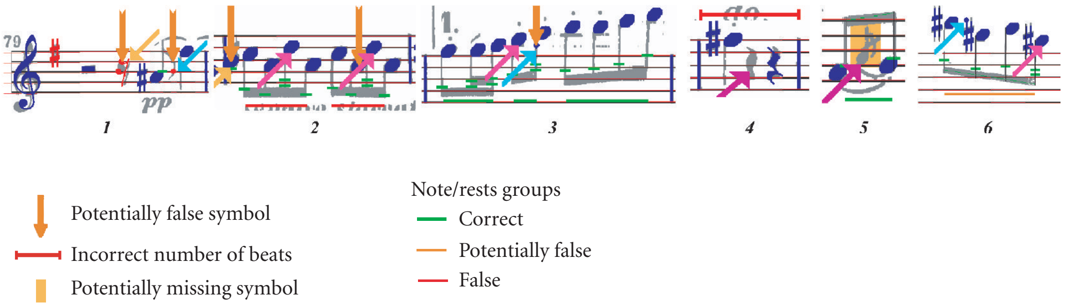


Figure 20: Potential error indications.

symbols: the added duration dot in bar 6 is quickly located through the indication of potentially false note group, the missing crotchet in bar 4 through the indication of false number of beats.

The average processing time needed for one music sheet of 10 staves is around 35 seconds on a Pentium 4 3.2 GHz. We can notice that preprocessing and individual symbol analysis steps take much more time than the global decision stage (5 seconds). One first improvement could consist in integrating a connected component analysis [10] in order to accurately segment symbols that are not featured by a vertical segment, and thus, decrease the amount of time spent for correlation with rest models. Moreover, at the time this article is written, research has been concentrated on the recognition method, and the actual code needs now to be restructured and then optimized. So, even if the run-time is quite high with respect to commercial software such as SmartScore [4], it is not prohibitive and it can certainly be dramatically reduced.

Finally, the experimental results of the supervised learning stage are presented. The method has been applied on scores for which one or more classes had bad recognition rates with the standard recognition program (without learning procedures). Some staves extracted from the score have been corrected and the class models have been learnt. Table 6 provides some results obtained on the large score whose learning has been illustrated in Section 8.2 (Figure 17). We can see that the average recognition rate is increased; especially sharps are now perfectly recognized. The second example presented in Table 7 shows similar results.

The results show that the learning stage may be able to lead to a 100% recognition of some symbols. However, when recognition rates are already quite satisfactory and when errors are mainly due to bad printing or other sources of ambiguity rather than inadequate class models, it is difficult to achieve a perfect recognition, and the recognition rates are only slightly improved. This is explained by the fact that the fuzzy model already manages symbol variability and allows disambiguating symbol by checking music writing rules. In this case, the general benefit of a correction/learning cycle is reduced. Overall, the adaptation method has always led to an increase in the general recognition rate in all our experimental tests. It has especially proved to be very efficient when some class symbols are really badly recognized, this problem arising when the internal class models of the standard recognition program are not suited to the processed score. It should be noticed that some other specificities of a music score could be learnt, for example some graphical parameters such as the typical distance between an accidental and a note head, or the thresholds used for potential error indication.

10. CONCLUSION

In this paper, we have described a complete optical music recognition system. After clarifying the music notation and the difficulties specific to this field, we tried to propose appropriate solutions to each issue. Efforts have been concentrated at both low-level and high-level analyses.

Low-level analysis includes preprocessing, segmentation and individual symbol analysis. Some robust algorithms were proposed to achieve a reliable segmentation: an accurate detection and removal of the staff lines, and a robust detection of vertical segments that feature all symbols that are not isolated through staff line removal. These algorithms can deal with the common printing defects, that is, skewed and warped staff lines, skewed and interrupted vertical segments, undesired connections. Template matching was then chosen as symbol analysis method, since it can deal with segmentation imprecision and printing defects, such as symbol fragmentation, bad connections, or primitive disconnections. The correlation scores are computed between the processed image and some generic class models, in small areas deduced from segmentation results and from the music rules related to possible symbol positions according to their class. They lead to a set of recognition hypotheses, each one assigning a detected object to a class. It should be noted that the vertical segment analysis, combined with template matching and the proposed method for beam detection and counting, allows retrieving compounded symbols (i.e., beamed note groups), while overcoming the common printing defects affecting them.

High-level analysis aims at disambiguating the recognition hypotheses, by analyzing the preliminary results (correlation scores and positions) and incorporating musical rules. Music notation is intrinsically ambiguous because of its variability and flexibility. It is also difficult to model since rules may involve more than two symbols which may be very distant, and since they apply at different levels. Three important features of the proposed method, that relies on the fuzzy sets

Table 6: *Recognition rates before and after supervised learning (example 1, 16 sheets)*. The other symbol recognition rates are identical. The global recognition rate is increased from 99.69% to 99.86%, and the added symbol rate is decreased from 0.53% to 0.11%.

Class	Without learning	Supervised learning	Class	Without learning	Supervised learning
♭	99.51	99.84	•	97.65	98.23
♮	99.43	100.00	𝄿	93.75	100.00
♯	90.77	100.00	●	99.94	99.98

Table 7: *Recognition rates before and after supervised learning (example 2, 2 sheets, 21 staves)*. The global recognition rate is increased from 95.64% to 98.09%, and the added symbol rate is decreased from 3.13% to 2.32%.

Class	No-learning	Supervised learning	Class	No-learning	Supervised learning
♯	91.67	100.00	𝄽	87.50	100.00
𝄿	0.00	71.43	•	95.24	100.00
𝄾	81.25	100.00			

and possibility theory, give some answers to these problems: the fuzzy model allows modeling symbol variability, segmentation imprecision, and the flexible and imprecise nature of the music notation; the main musical rules are all modeled and integrated in the decision process; the decision process is global in the sense that all possible combination of hypotheses provided by the low-level analysis for each bar are evaluated against all rules. So, the final decision is made by taking into account all criteria and it is thus consistent with the music writing.

Experiments conducted on a large database show good recognition rates and interesting results for both symbol recognition (99.2%) and note length interpretation (99.3%). The parameters of the system, which are either learnt on the score or set based on general music considerations, proved to be experimentally robust. The recognition rates have been increased with respect to [1, 26], thanks to the gain in robustness realized at the low level analysis, and the improvements realized in the fuzzy model. Comparison with SmartScore [4], a well-known commercial software, shows that the proposed method is an important contribution to OMR. It is obvious, when comparing recognition results, that the fuzzy modeling of musical rules and their integration in a global decision process contribute to a better interpretation.

Improvements can be made in handling pick-up measures that are responsible of many added symbols, and improving the pitch recognition of notes below or above the staff. Also adding some structural information about musical symbols could help to decrease the confusion rate between intercorrelated symbol classes. This idea raises two issues: which information is suitable and how it can be integrated in the fuzzy model, in order to avoid rigid criteria that cannot deal with symbol variability, printing and segmentation defects. Lastly, some improvements can be made in order to decrease the processing time: detecting more accurately the symbols that are not characterized by a vertical segment in order to better restrict the areas where correlation is computed; implementing some heuristics that reduce the computational cost required for decision making, especially in case of large bars involving many symbols.

Another innovative feature of the proposed method is the automatic indication of potential recognition errors that rely on the results obtained in the fuzzy modeling part. Most errors are directly or indirectly indicated to the user, and the manual correction is in this way facilitated. Some learning procedures are also proposed in order to adapt symbol models and related parameters to the processed score, so that the recognition is improved on the rest of the score. Experiments show very good results on scores, for which some classes got unsatisfactory recognition rates before learning, and almost perfect recognition rates after. These two features really allow a better reliability and a more convenient use of the system, since manual correction is facilitated and the time spent for this tedious task considerably reduced. This is especially important when processing large scores. Further improvements may include learning other specificities of the score, at the graphical level, for example, and refining the thresholds used for error indications.

REFERENCES

[1] F. Rossant and I. Bloch, “A fuzzy model for optical recognition of musical scores,” *Fuzzy Sets and Systems*, vol. 141, no. 2, pp. 165–201, 2004.

[2] D. Blostein and H. Baird, “A critical survey of music image analysis,” in *Structured Document Image Analysis*, H. S. Baird, H. Bunke, and K. Yamamoto, Eds., pp. 405–434, Springer, Berlin, Germany, 1992.

[3] N. Carter, R. Bacon, and T. Messenger, “The acquisition, representation and reconstruction of printed music by computer: a review,” *Computer and the Humanities*, vol. 22, no. 2, pp. 117–136, 1988.

[4] SmartScore 3.2 Pro Demo. http://www.musitek.com.

[5] D. Bainbridge and T. C. Bell, "Dealing with superimposed objects in optical music recognition," in *Proceedings of 6th International Conference on Image Processing and Its Applications*, vol. 2, pp. 756–760, Dublin, Ireland, July 1997.

[6] P. Bellini, I. Bruno, and P. Nesi, "Optical music sheet segmentation," in *Proceedings of 1st International Conference on Web Delivering of Music (WEDELMUSIC '01)*, pp. 183–190, Florence, Italy, November 2001.

[7] B. Coüasnon and J. Camillerapp, "Using grammars to segment and recognize music scores," in *International Association for Pattern Recognition Workshop on Document Analysis Systems (DAS '94)*, pp. 15–27, Kaiserslautern, Germany, October 1994.

[8] K. C. Ng and R. D. Boyle, "Recognition and reconstruction of primitives in music scores," *Image and Vision Computing*, vol. 14, no. 1, pp. 39–46, 1996.

[9] D. Bainbridge and T. Bell, "An extensible optical music recognition system," in *Proceedings of the 19th Australasian Computer Science Conference (ACSC '96)*, pp. 308–317, Melbourne, Australia, January-February 1996.

[10] I. Fujinaga, "Adaptive optical music recognition," Ph. D. dissertation, McGill University, Montreal, Calif, USA, 1997.

[11] N. Carter and R. Bacon, "Automatic recognition of printed music," in *Structured Document Image Analysis*, H. S. Baird, H. Bunke, and K. Yamamoto, Eds., pp. 456–465, Springer, Berlin, Germany, 1992.

[12] D. Bainbridge and T. Bell, "A music notation construction engine for optical music recognition," *Software: Practice and Experience*, vol. 33, no. 2, pp. 173–200, 2003.

[13] M. Droettboom, I. Fujinaga, and K. MacMillan, "Optical music interpretation," in *Proceedings of the International Workshops on the Statistical, Structural and Syntactic Pattern Recognition Conference (SSPR '02)*, pp. 378–386, Windsor, Canada, August 2002.

[14] H. Fahmy and D. Blostein, "A graph-rewriting paradigm for discrete relaxation: application to sheet-music recognition," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 12, no. 6, pp. 763–799, 1998.

[15] H. Kato and S. Inokuchi, "The recognition system of printed piano using musical knowledge and constraints," in *Proceedings of IAPR Workshop on Syntactic and Structured Pattern Recognition*, pp. 231–248, Murray Hill, NJ, USA, June 1990.

[16] D. Blostein and L. Haken, "Using diagram generation software to improve diagram recognition: a case study of music notation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 11, pp. 1121–1136, 1999.

[17] B. Coüasnon and B. Rétif, "Using a grammar for a reliable full score recognition system," in *Proceedings of International Computer Music Conference (ICMC '95)*, pp. 187–194, Banff, Canada, September 1995.

[18] M. Ferrand, J. A. Leite, and A. Cardoso, "Improving optical music recognition by means of abductive constraint logic programming," in *Proceedings of the 9th Portuguese Conference on Artificial Intelligence (EPIA '99)*, pp. 342–356, Évora, Portugal, September 1999.

[19] M.-C. Su, C.-Y. Tew, and H.-H. Chen, "Musical symbol recognition using SOM-based fuzzy systems," in *Proceedings of 20th International Conference of the North American Fuzzy Information Processing Society and the International Fuzzy Systems Association*, vol. 4, pp. 2150–2153, Vancouver, BC, Canada, July 2001.

[20] G. Watkins, "The use of fuzzy graph grammars for recognising noisy two-dimensional images," in *Proceedings of Biennial Conference of the North American Fuzzy Information Processing Society (NAFIPS '96)*, pp. 415–419, Berkeley, Calif, USA, June 1996.

[21] I. Bloch, "Information combination operators for data fusion: a comparative review with classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, vol. 26, no. 1, pp. 52–67, 1996.

[22] I. Bloch and H. Maître, "Fusion of image information under imprecision," in *Aggregation and Fusion of Imperfect Information*, B. Bouchon-Meunier, Ed., vol. 12 of *Studies in Fuzziness and Soft Computing*, pp. 189–213, Springer, Berlin, Germany, 1997.

[23] D. Dubois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, New York, NY, USA, 1980.

[24] D. Dubois, H. Prade, and R. R. Yager, "Merging fuzzy information," in *Approximate Reasoning and Information Systems*, J. C. Bezdek, D. Dubois, and H. Prade, Eds., Handbook of Fuzzy Sets, chapter 6, pp. 335–401, Kluwer Academic, Norwell, Mass, USA, 1999.

[25] F. Rossant, "A global method for music symbol recognition in typeset music sheets," *Pattern Recognition Letters*, vol. 23, no. 10, pp. 1129–1141, 2002.

[26] F. Rossant and I. Bloch, "Optical music recognition based on a fuzzy modeling of symbol classes and music writing rules," in *Proceedings of IEEE International Conference on Image Processing (ICIP '05)*, vol. 2, pp. 538–541, Genova, Italy, September 2005.

[27] V. P. d' Andecy, J. Camillerapp, and I. Leplumey, "Kalman filtering for segment detection: application to music scores analysis," in *Proceedings of the 12th International Conference on Pattern Recognition (ICPR '94)*, vol. 1, pp. 301–305, Jerusalem, Israel, October 1994.

[28] I. Fujinaga, S. Moore, and D. S. Sullivan, "Implementation of exemplar-based learning model for music cognition," in *Proceedings of the 5th International Conference on Music Perception and Cognition (ICMPC '98)*, pp. 171–179, Seoul, Korea, August 1998.

[29] P. Martin and C. Bellissant, "Low-level analysis of music drawings images," in *Proceedings of the 1st International Conference on Document Analysis and Recognition (ICDAR '91)*, pp. 417–425, Saint-Malo, France, 1991.

[30] J.-P. Armand, "Musical score recognition: a hierarchical and recursive approach," in *Proceedings of the 2nd International Conference on Document Analysis and Recognition (ICDAR '93)*, pp. 906–909, Tsukuba Science City, Japan, October 1993.

[31] I. Fujinaga, "Optical music recognition using projections," M.S. thesis, McGill University, Montreal, Calif, USA, 1988.

[32] K. T. Reed and J. R. Parker, "Automatic computer recognition of printed music," in *Proceedings of the 13th International Conference on Pattern Recognition (ICPR '96)*, vol. 3, pp. 803–807, Vienna, Austria, August 1996.

[33] R. Randriamahefa, J. P. Cocquerez, C. Fluhr, F. Pépin, and S. Philipp, "Printed music recognition," in *Proceedings of the 2nd International Conference on Document Analysis and Recognition (ICDAR '93)*, pp. 898–901, Tsukuba Science City, Japan, October 1993.

[34] J. R. McPherson and D. Bainbridge, "Coordinating knowledge within an optical music recognition system," in *The 4th New Zealand Computer Science Research Students' Conference (NZCSRSC '01)*, pp. 50–58, Christchurch, New Zealand, April 2001.

[35] http://www.interactivemusicnetwork.org/wg_imaging/upload/assessingopticalmusicrecognition_v1.0.doc.

Florence Rossant was graduated from ISEP (Institut Supérieur d'Electronique de Paris) where she obtained her Telecom Signal and Image Engineering Degree in 1992. Being today in charge of the Telecom and Signal Laboratory at ISEP, she assumes currently teaching and research activities. She has just completed her Ph.D. degree at ENST (École Nationale Supérieure des Telecommunications) in the field of optical music recognition and she is also currently involved in a biometric identification research project—especially the Iris Identification part.

Isabelle Bloch is a Professor at ENST (Signal and Image Processing Department). She obtained her Ph.D. degree in 1990 and the Habilitation diploma in 1995. Her research interests include 3D image and object processing, 3D and fuzzy mathematical morphology, decision theory, information fusion, fuzzy set theory, belief function theory, structural pattern recognition, spatial reasoning, and medical imaging.