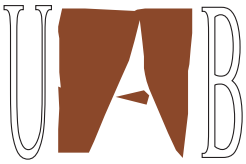


Escola Tècnica Superior d'Enginyeries

Departament de Ciències de la Computació

Analysis of Old Handwritten Musical Scores

Memòria del Treball Experimental presentat per **Alicia Fornés Bisquerra** dirigit per **Josep Lladós Canet** i codirigit per **Gemma Sánchez Albaladejo** dins del Programa de Doctorat en Informàtica, opció **Visió per Computador**, de la Universitat Autònoma de Barcelona.

el Dr. Josep Lladós Canet , i la Dra. Gemma Sánchez Albaladejo , del Departament de Ciències de la Computació de la Universitat Autònoma de Barcelona,

CERTIFIQUEN

que la present memòria ha estat realitzada sota la seva direcció per **Alicia Fornés Bisquerra** i constitueix el Treball Experimental del *Programa de Tercer Cicle d'Informàtica, opció Visió per Computador.*

Bellaterra, 7th November 2005

Josep Lladós Canet

Gemma Sánchez Albaladejo

Agraïments

Als meus directors, Josep Lladós i Gemma Sánchez, per la seva dedicació al llarg de tot el projecte.

Al Juanjo Villanueva i a la Universitat Autònoma de Barcelona, per donar-me l'oportunitat de realizar aquest treball de recerca.

Al Josep Maria Gregori Cifré del departament d'Art de la UAB, per la seva ajuda en l'accés a les partitures antigues del Seminari de Barcelona.

Als meus companys del CVC, per l'ajuda i suport: Anton, Ágata, Xevi, Misael, Xavier, David, Sergio, Oriol, Jaume, Juan, Aura, Agnès, Natxo,Dani, Anna, Mathias, Javier...

Als meus amics mallorquins, sempre disponibles quan soc a Mallorca: Noemí, Javi, Sebas, Cristina, Víctor, Sven, Tolo, Maite, Elena, MariMar, Jesús...

A les meves companyes de pis, per fer-me sentir com a casa: Bàrbara, Virginia, Rocio, Ester i Anna.

A la Mònica, per suportar amb paciència els meus diaris canvis d'humor.

A la meva germana, per ésser la meva gran confident i companya musical.

Al Ramon, pels seus constants i valuosos ànims durant aquests darrers tres anys.

Als meus pares, per sentir-los sempre propers a pesar de la distància.

A tots els que fan o han fet un treball de recerca, perquè saben la sang, suor i llàgrimes que comporta ;)

RESUM

El reconeixement de gràfics ha estat una àrea d'intensa recerca en el camp de l'anàlisi de documents. El reconeixement de gràfics es centra en la interpretació d'estructures gràfiques en imatges de documents. Entre les aplicacions més comunes es troba el reconeixement i la interpretació de diagrames de circuits lògics, plànols d'arquitectura, equacions matemàtiques, el reconeixement de logos i el reconeixement de partitures musicals.
En els darrers anys hi ha hagut un interès creixent en l'anàlisi de documents antics per a la preservació d'antigues col.leccions de documents existents en arxius històrics, i així convertir-les en llibreries digitals. El Reconeixement Òptic Musical consisteix en la identificació d'informació musical a partir d'imatges de partitures i la seva conversió en un format que pugui entendre l'ordinador. Aquests sistemes permetran un gran nombre d'aplicacions, incloent l'edició i renovació de partitures, la conversió al codi Braile, la creació de bases de dades per realitzar anàlisis musicològics o fins i tot la producció de fitxers d'àudio o de descripció musical.
Encara que el Reconeixement Òptic Musical es una àrea madura en partitures impreses, pocs treballs de recerca s'han fet sobre partitures manuscrites. En aquest treball de recerca, es proposa un mètode per a les etapes inicials del reconeixement: segmentació de les línies de pentagrama i de primitives gràfiques en partitures manuscrites. Després d'introduir la feina realitzada amb partitures modernes manuscrites, es descriu la metodologia aplicada per treballar amb partitures manuscrites antigues. En aquest tipus de partitures, les dificultats del reconeixement s'incrementen degut a la degradació del paper i la manca d'estàndard en la notació musical, per tant, es requereixen altres tipus de tècniques. El nostre mètode ha estat provat amb partitures del segle XIX amb un elevat percentatge de reconeixement.

Paraules clau: *Reconeixement Òptic Musical, Documents antics, Diagrames manuscrits, Reconeixement de documents gràfics, Reconeixement de símbols.*

ABSTRACT

Graphics recognition is an intensive research area within the document image analysis field. Graphics recognition is focused in the interpretation of graphical structures in document images.
Typical applications of graphics recognition are the interpretation of logic circuit diagrams, engineering drawings, maps, architectural drawings, mathematical equations, logo recognition and optical music recognition.
In the last years there is a growing interest in the analysis of ancient documents, in order to preserve old collections of documents existing in archives and to convert them to digital libraries.
Optical Music Recognition consists in the identification of music information from images of scores and

their conversion into a machine readable format. Such systems will allow a huge number of applications, including the edition and renewal of scores, the conversion into Braille code, the creation of collecting databases to perform musicological analysis, and the production of audio or musical description files.
Although Optical Musical Recognition is a mature area in printed scores, few research works have been done in handwritten ones. In this dissertation, we propose a method for the early stages of the recognition: segmentation of staff lines and graphical primitives in handwritten scores. After introducing our work with modern musical scores, an approach to deal with old handwritten scores is exposed. In such these old scores, difficulties in recognition are increased due to paper degradation and the lack of a standard in musical notation, so other techniques are required. Our method has been tested with several scores of XIX century with high performance rates.

Keywords: *Optical music recognition, Old documents, Handwritten diagrams, Graphics Recognition, Symbol Recognition.*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this section, a brief introduction to the field of Optical Music Recognition (OMR) is presented. Firstly, an initial location of this area in Pattern Recognition field, exposing the main applications and difficulties of OMR. Secondly, elements of musical notation, structure of scores and layers of the system are shown.

1.1 Framework

One of the major fields of research in Computer Vision is Pattern Recognition, consisting in the study of the operation and design of systems that recognize patterns in data. It encloses subdisciplines like discriminant analysis, feature extraction, error estimation, cluster analysis (together sometimes called statistical pattern recognition), grammatical inference and parsing (syntactical pattern recognition). Important application areas in image analysis are industrial inspection, person identification (including human faces recognition) and document analysis (see Fig.1.1), which includes the popular Optical Character Recognition and graphical analysis (where Optical Music Recognition belongs).

1.1.1 Document Image Analysis

Document image analysis, is the process that performs the overall interpretation of document images. Because of wide variability in the structure of documents, Document Image Analysis is divided in three major areas of research: Optical Character Recognition, Layout understanding(Structure of Documents) and Graphic Recognition (where Optical Music Recognition belongs).

Optical Character Recognition Optical Character Recognition (OCR) is focused on the recognition of text (printed and handwritten) in documents, and is one of the most classical and common areas of pattern recognition research. In [1] and [2], we can find extensive surveys of OCR research: the concept of Optical Character Recognition (optically because it deals with scanned-optically processed

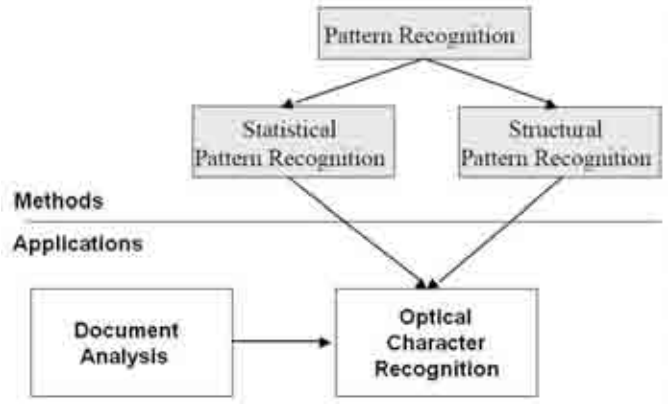


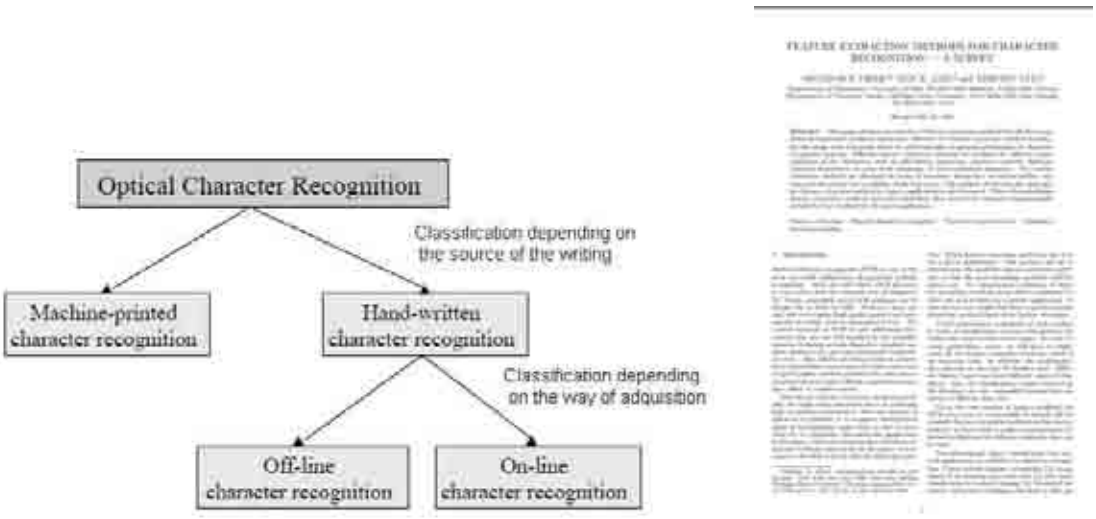
Figure 1.1: Pattern Recognition.

characters rather than magnetically processed ones) appeared in the 1940s with the development of digital computers. The principal motivation for the development of OCR systems is the need to cope with the enormous flood of paper such as bank cheques, commercial forms, government records, credit cards imprints and mail sorting. The research of OCR systems has been a subject of great interest, and a large number of papers and books have been published about this area. Presently, the methodologies in character recognition have advanced from the earlier use of primitive techniques for the recognition of printed numerals, to sophisticated techniques for the recognition of a wide variety of complex handwritten characters, symbols and chinese and japanese characters.

There are two major areas in OCR (see Fig.1.2a)): On-line and off-line character recognition. Off-line is performed after the writing or printing is completed. OCR deals with the recognition of optically processed characters (typically digitized by an optical scanner).

Layout understanding The aim of layout understanding is the analysis of data contained in image documents and the recognition of its structure: determine what region corresponds to text, graphics, tables and images. An interesting application of these systems is the possibility to perform the intelligent information retrieval from a digital library.

An approach of a layout document understanding system can be found in [3], where layout analysis starts with block segmentation which decomposes the digital image of a document page into regions. The process locates large streams of white space (background analysis) running horizontally or vertically. Streams that are considered region boundaries are used to partition the image into regions. A region must be bound by two horizontal and vertical background boundaries. Figure 1.3 shows the regions of a document after block segmentation. Then those regions are classified into one of the structural categories such as text, line drawings, tables and photographs. The classification of a region is performed by matching a set of features extracted from the region against the predefined reference features of a category.



(a) (b)

Figure 1.2: (a) OCR Classification. (b) Example of a Text document.

Graphics Recognition Graphics recognition of documents has been an area of intensive research in the field of pattern recognition and document analysis. Main applications are the interpretation of logic circuit diagrams, engineering drawings, maps, architectural drawings (see Fig.1.4), mathematical equations, logo recognition and **optical music recognition**.

The automatic interpretation of such documents, requires processes able to recognize the corresponding alphabets of symbols (see [4]). In a general way, a symbol can be defined as a graphical entity with a particular meaning in the context of an specific application domain. Each kind of graphic document own a characteristic set of symbols according to their visual properties (see Fig.1.5): in circuits and maps, symbols are simple 2D binary shapes composed of line segments, in logos there are complex gray level or color shapes whereas in musical scores symbols are a combination of line segments and solid shapes.

1.1.2 Optical Music Recognition

The advancement of computer technology has a great influence on the musical field. Whether in multimedia processing or in performing time consuming tasks (such as transposition), the computer offers accuracy. This musical information has to be represented in a machine readable format, and input methods become an important factor. Apart from the popular electronic keyboards, a good input method can be scanned images of musical scores (see Fig.1.6), due to the enormous quantities of paper-based music existing.

The aim of Optical Music Recognition (OMR) is the identification of music information from images of scores and its conversion into a machine legible format. This field was first attempted in the late 1960s and early 1970s, and research has been increasing over the last decade due to its potential benefits: The

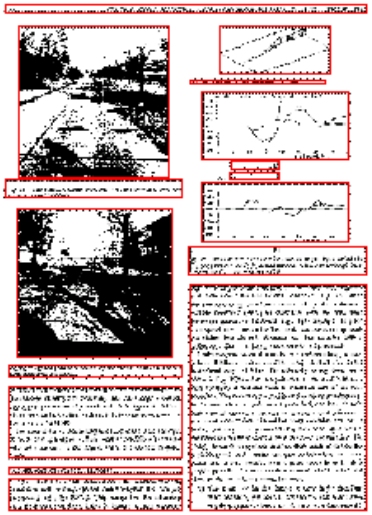


Figure 1.3: Regions segmented in a document.

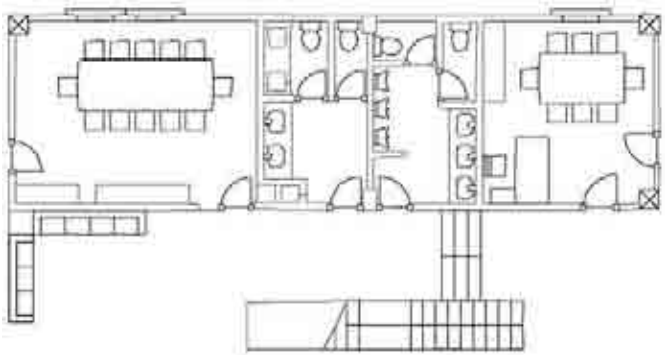


Figure 1.4: Architectural drawing.

existence of an automatic recognition system for musical scores can make practical the conversion of large quantities of scores into a computer-readable form. This is similar to the requirement for automatic entry of engineering drawings for existing documents.

Once music is stored (using some music representational language), it can be manipulated freely, enabling the development of a wide variety of applications (see [5],[6]):

- Edition and publication of scores never edited.
- Renewal of old scores.
- Conversion of scores into Braille code to aid blind musicians.
- Adaption of existing works to other instrumentations (such as the reduction of full scores to piano scores).

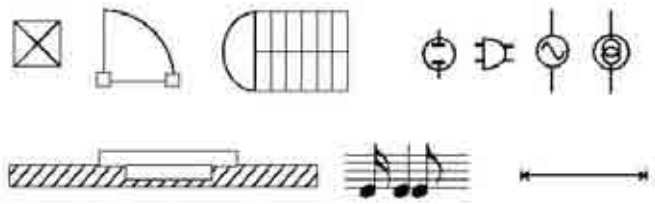


Figure 1.5: Some Graphic Symbols.

Figure 1.6: Example of a musical score.

- Making critical editions of musical compositions given different printed versions of the same composition.
- Transpose a music sample to some other clef or key signature.
- Produce parts from a given score or a full score from given parts.
- Read in a newly engraved piece of music and proofread it for syntactic and other errors.
- Print newly written music automatically (on-line recognition).
- Creation of collecting databases to perform musicological analysis
- Production of audio files (the computer becomes a musician).

- Production of musical description files: NIFF (Notation Interchange File Format) and MIDI (Music International Device Interface).
- Creation of collecting databases:
  - Creation of indices of themes and other music features.
  - Analysis of musical structure and style.
  - Automatic Harmonization of monophonic scores.
  - Testing theories of music.
  - Evaluation of algorithms for the automatic analysis or composition of music.

Optical Music Recognition has many similarities with Optical Character Recognition because whereas OCR recognizes characters in text, OMR recognizes musical symbols in scores, and some techniques of OCR can be effectively used in OMR. In fact OCR is a sub-task of OMR because lots of scores include text, that must be also recognized. It is nevertheless true that OMR belongs to graphical analysis because it requires the understanding of two-dimensional relationships (musical symbols have two-dimensional shapes).

Difficulties Main character recognition problems are: Shape discrimination (there is a lot of different fonts and styles in printed characters and thousands of handwriting shapes), deformation of the image caused by noise (disconnected line segments, holes and breaks in lines, isolated notes), translation (movement of a whole character or its components) and rotation (change in orientation); and variations in sizes and pitch. In addition, an OCR system must be able to distinguish figure from text, recognize touching characters, and be unaffected by proportional spacing and variable line spacing.

Main optical music recognition problems are:

- Musical symbols are written on the staff: For that reason, musical symbols are connected by staff lines making difficult the segmentation stage.
- Translation (movement of a whole symbol or its components) and rotation (change in orientation) of musical symbols.
- Variations in sizes: musical symbols have great variation in relative sizes (whereas in OCR, sizes of characters do not vary much, see Fig.1.7).

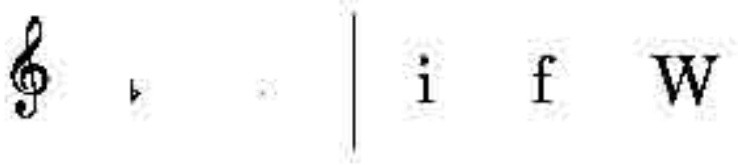


Figure 1.7: Sizes of musical symbols (treble clef, flat and duration dot) in front of sizes of text.

- Distinguish musical symbols from text-lyrics and dynamic marks.
- Touching objects: The scores to be recognized are complex and have a high density of symbols. As a result, connections are often found between musical objects that syntactically should not be touching, causing segmentation problems (see Fig.1.8).

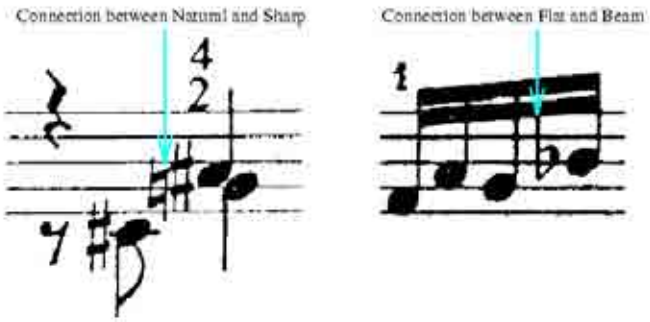


Figure 1.8: Touching musical symbols.

- Broken objects: one of the most common problems when scanning is the introduction of noise, which frequently causes broken objects: holes and breaks in lines, isolated notes. In addition, objects affected are small, making difficult their recognition.

Notice that it must be difficult to scale up a prototype into one capable of analyzing complex music (such as polyphonic music). Most existing OMR are adapted to simple scores, because grammar organization for complex scores could contain hundreds of production rules, becoming unmanageable.

Handwritten scores As commented in [7] and [8], although the introduction of new technologies (such as typewriter, computers and PDAs), handwriting persists as a means of communication and recording information for numerous day-to-day situations: postal addresses on envelopes, bank checks, handwritten fields in forms... For that reason, OCR systems for handwritten documents (see Fig.1.9) are required (also called Intelligent Character Recognition systems, ICR) in order to transform a language represented in its spatial form of graphical marks (handwritten documents) into its symbolic representation.

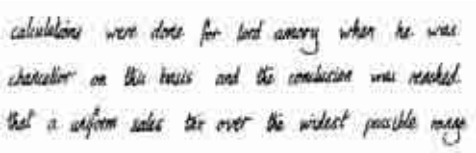


Figure 1.9: Example of a Handwriting text.

Concerning Optical Music Recognition, a system capable to recognize handwritten musical scores

(see Fig.1.10) is also required. In fact, OMR is a mature area for printed scores (there is a lot of literature about the recognition of printed scores) whereas few research works have been done in handwritten ones.

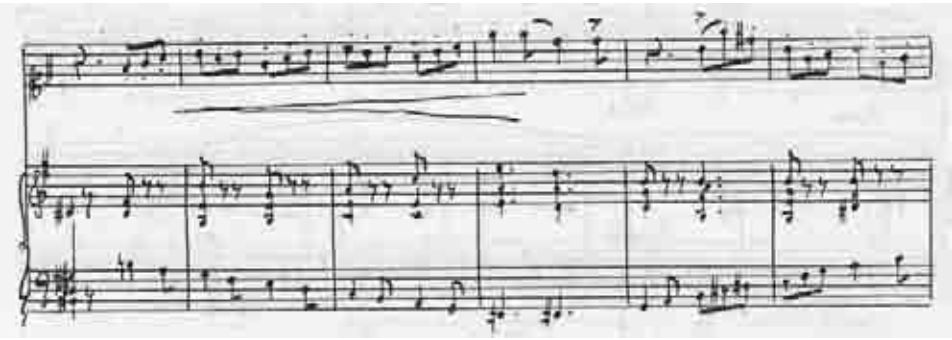


Figure 1.10: Example of an handwriting musical score.

When compared to classical OMR systems for printed scores, handwritten music recognition introduces additional difficulties in the segmentation and the recognition process. The most important are the following:

- Notation varies from writer to writer.
- Simple and large changes in notation can occur in the same score.
- Staff lines are often handwritten, so they mostly do not have the same height and are rarely straight.
- Symbols are written with different sizes, shapes and intensities.
- The relative size between different components of a musical symbol can vary.
- More symbols are superimposed in handwritten music than in printed music.
- Different symbols can appear connected to each other, and the same musical symbol can appear in separated components.

Old handwritten musical scores A growing interest in the Document Analysis area is the recognition of ancient manuscripts and their conversion to digital libraries, towards the preservation of cultural heritage. Recent works include the reconstruction of *Don Quixote* (see [9]), and the recognition of korean and greek handwritten documents (see [10], [11]). For those reasons, our work in handwritten scores has been conducted to the recognition of **old** handwritten scores (XVI-XIX centuries) of unknown composers, contributing to the diffusion of these scores never edited (see Fig.1.11).

Difficulties in the recognition of handwritten scores are increased when working with old documents: First of all, paper degradation requires specialized image-cleaning and binarization algorithms (see [12]). Secondly, there is a lack of standard notation, because notation differs from a century to another. In

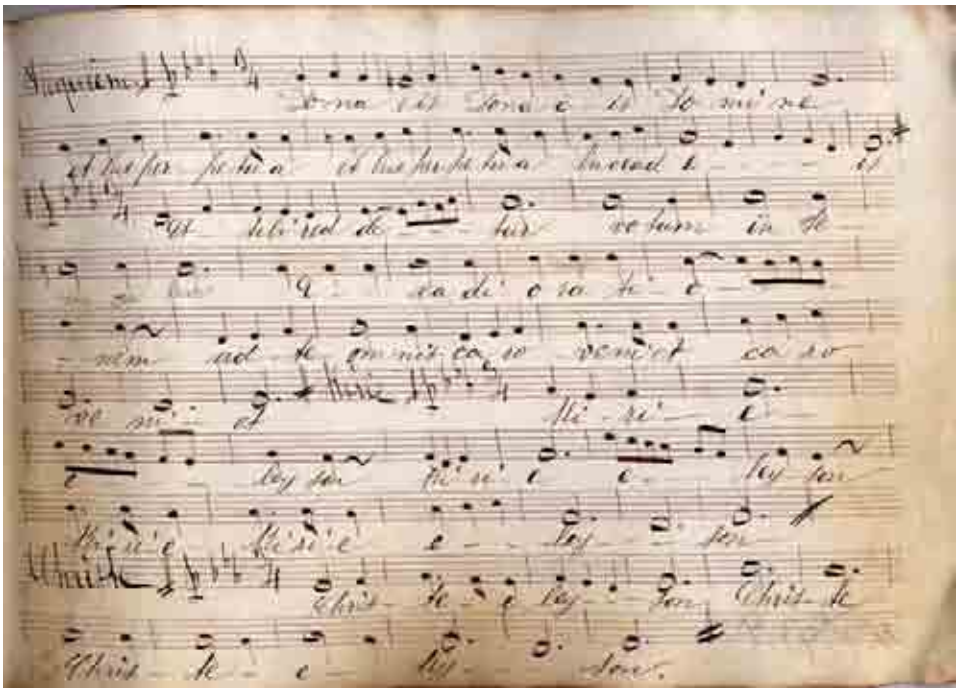


Figure 1.11: Example of an old handwriting musical score.

practice, in the XX century, composers often feel free to adapt notation to new uses (in fact, there are national *dialects* of music notation). This is obvious that this variability in music notation is increased when working with scores of XVI-XIX centuries.

To cope with these difficulties an expert system will be required to learn every new way of writing, and artificial intelligence based techniques will take advantage of higher level musical information.

1.2 Structure of a musical score

In this section, elements of musical scores are shown: a brief definition of Musical Notation is presented. After that, the graph grammars that formalize how graphical primitives are physically joined is shown. Finally, the logical structure of musical scores is defined with a grammar.

1.2.1 Elements of scores: Musical Notation.

The musical notation (see [5],[13]) in scores consists of the following elements (see Fig.1.12):

- Staff: Where musical symbols are written down. Five equidistant, horizontal lines form a staff.
- Attributive symbols at the beginning: Clef, Time and Key signature.
- Bar lines, that separate every bar unit or measure.
- Rests(pauses) and Notes. Notes are composed of head notes, beams, stems, flags and accidentals.
- Slurs: Curves that join musical symbols.
- Dynamic Markings indicate how loud or soft the music should be played.
- Tempo Markings indicate the speed of the rhythm of a composition.

Some scores include text, so an important task is to determine which objects are text (lyrics), and which are musical symbols. In addition, some words correspond to dynamic markings, so context information should help to distinguish them.

Some terminology used in music is the following:

- Staff line sections: The covered sections of a staff line are those sections where other music symbols intersect the staff line; the remaining sections of the staff line are bare.
- Staff space: The distance between the staff lines within a single staff. The staff space provides a normalized unit of measurement for expressing distances.
- System: A set of staves that are played in parallel. In printed music these staves are connected with braces, and bar lines may be drawn through from one staff to the next. A page of an orchestra score may contain only a single system.
- Staff nucleus: The area of a staff that contains the staff lines and the musical symbols. In order to avoid missing symbols on ledger lines, the staff nucleus can be defined to extend vertically one or two staff spaces above the top staff lines and one or two staff spaces below the bottom staff line.
- Voice: A musical line. A voice may correspond to a single instrument; a piano has two voices (one for the right hand and another for the left one). Some times, several voices may be printed together on one staff: in a orchestra score, the flute 1 and flute 2 voices are printed on the same staff (with opposite stem direction).

- Monophonic: Music consisting of a single voice, where this voice contains no chords.
- Polyphonic: Music consisting of several voices, including chords.

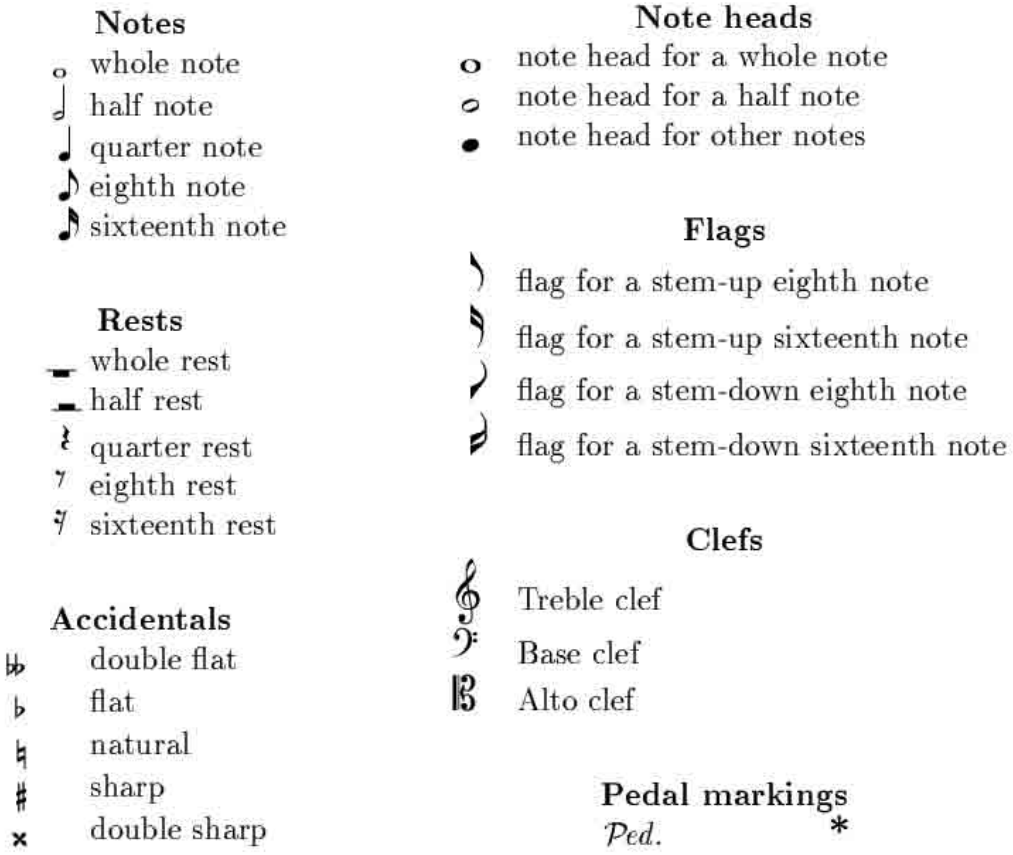


Figure 1.12: Common elements of Music Notation.

1.2.2 Structure of Graphical Primitives: spatial structure.

Graph grammars (grammars which operate on a graph) can solve picture processing problems, and can be applied to determine the meaning of complex diagrams, where the interaction among physically distant symbols is semantically important (see [14]). Graph grammars can describe two-dimensional structures and are flexible to express arbitrary relations between pattern primitives. As musical scores are complex 2-D diagrams, the interaction among physically distant symbols and graphical primitives (such as lines, curves, circles and dots) is semantically important. For our OMR system, we have defined a graph grammar in which graphical primitives form musical symbols:

- Nodes: Will be the primitives, including the kind of primitive (line, curve, circle, dot) and their location (x,y) in the image.
- Arcs: Will be the connections between primitives: joined (left, right, top, down, and its diagonals), closed (left, right, top, down, and its diagonals), or parallel (left, right, top or down).

1.2.3 Logical Structure of musical scores.

Thanks to the fact that music scores follow strict structural rules, context information about musical notation can be effectively used to help in the recognition process. As we know, formal language theory provides useful tools to recognize and solve ambiguities in terms of context-based rules or semantic restrictions using attributes. For that reason, these structural rules can be formalized by grammar rules (describing the musical score structure) and applied in the postprocess stage of the OMR system in order to guide the recognition and validation process.

Informally speaking, a grammar describing a score (see Fig. 1.13) consists of three blocks **G: S** $\rightarrow$ **H[B]E**, where **H** is the heading with the attribute symbols: treble, alto or bass clef, time signature (commonly formed by two numbers that indicate the measure) and key signature (flats, sharps or naturals, which indicate the tonality of the score). All these symbols are very important to provide the meaning of musical symbols. Then, the score is discomposed in bar units **B**, in which notes and rests are written down. The amount of notes and rests in every bar unit depends on the time signature, so it will obviously help to solve ambiguities in the recognition of notes and rests. Finally, there is an ending measure bar (**E**). The **entire grammar** that formalizes the structure of musical scores, and which will be used in our system can be found in the Appendix.

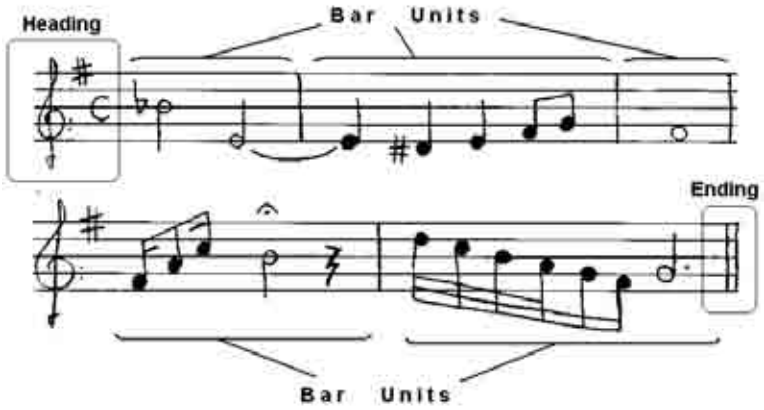


Figure 1.13: Structure of a score.

As Ng comments in [15], global information (such as time and key signatures) influence the layout, grouping and beaming of notes, and introduce probabilistic constraints to the usage of certain note classes in accordance to the tonality of the piece or section of the music:

- Key signature (Tonality): In a musical score, two of the most significant note classes of any key are the tonic (first note) and the dominant (fifth note) and there is a distribution pattern which reflects the harmonic weighting of the note classes in the musical scale. The major utility of knowing the tonality and modality (major or minor) of a musical score is that they will help to verify all isolated accidental signs.
- Time Signature: The timing information of the score establishes the total duration of every measure, so any discrepancies between the estimated duration of a measure and the expected duration of the time signature indicates missing or misclassified items. In addition, a database of commonly found rhythmical patterns for every time signature (e.g. three quavers group in 6/8 time) could solve ambiguous note length and note heads. It must be said that duration of measures can help to recognize time signatures, because many ambiguous time signatures can be resolved using a simple count of the linearly disposed note values between a pair of barlines.

1.3 General Architecture of an OMR system

In this section the stages of an OMR system are exposed: Location and Segmentation, Preprocessing, Feature Extraction, Classification and Postprocessing stages. They are very similar to the ones of an OCR system. Afterwards, levels of the processed information of the system are exposed: image layer, graphical primitives layer, symbol layer and finally, context information layer.

1.3.1 Stages of a OMR system

Our recognition strategy follows a typical OMR architecture (see Fig. 1.14), whose stages are indeed very similar to the ones in OCR systems (see [2],[7]): The input document is scanned by an optical scanner to produce a gray-level or binary image. If the image is in gray-level, it must be binarized using a thresholding method. After that, the system locates the regions on the digitized document in which data is located, and segments the score into isolated symbols. After segmentation, data is subjected to smoothing, elimination of noise, size normalization and other operators to facilitate the extraction of features in the subsequent stage. The identification and classification of symbols is achieved by comparing the extracted features with the statistics of features obtained from the set of samples used in the learning phase (a training set is commonly used). Finally, linguistic, contextual or statistical information can be used to resolve ambiguities of symbols with similar shapes or to correct some sections.

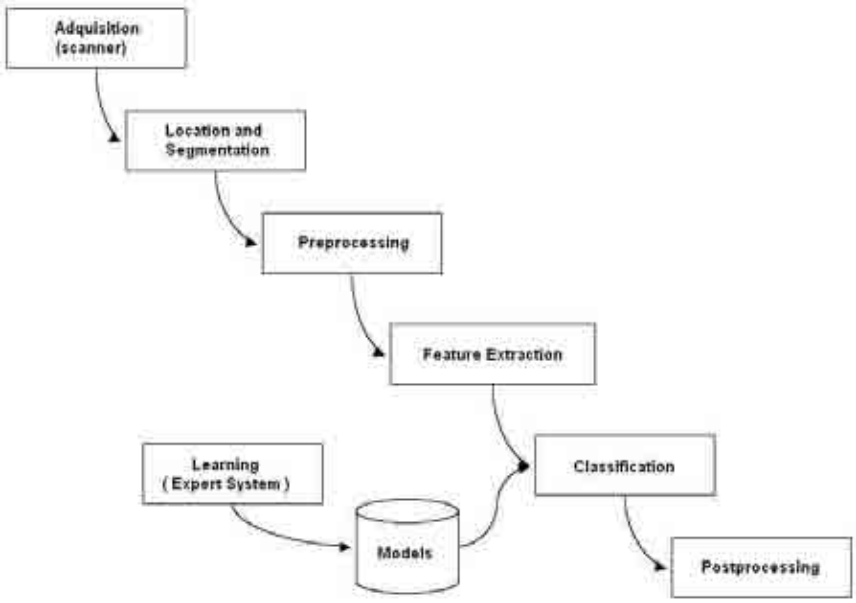


Figure 1.14: OMR Architecture.

Location and Segmentation It determines the regions on the document in which musical information is written down and segment those regions. In OCR systems this task consists in the understanding of the structure of the document and the isolation of text form graphics and images.

Preprocessing Preprocessing is an important phase of pattern recognition process. The main preprocessing techniques available are:

- Binarization and smoothing (which includes filtering, filling and thinning).
- Normalization (see [16]) consists in a slant normalization to reduce the variability of the writing style, stroke width normalization to reduce the writing instrument and size normalization to reduce the variability of the symbols' size.
- Segmentation consists in the distinction of musical data from images and the isolation of musical symbols. The segmentation is easy if symbols do not touch or overlap with one another.
- Line segment approximation is the conversion of a list of pixel coordinates of chain-coded pixels into a set of coordinates, which when joined with straight line segments from a polygonal approximation to the original line. This approximation is performed to reduce the volume of data and is used in connection with recognition techniques that are based on the extraction of features which describe the geometry or topology of the drawing.

Feature extraction Feature extraction is one of the most difficult and important problems of pattern recognition. Many different types of shape features can be extracted and used to recognize characters. The problem is to extract those features which will enable the system to discriminate efficiently one class of characters from the others. The main recognition techniques are: Template matching and correlations, extraction of features from statistical distribution of points, global transformations and series expansions. As Mori exposes ([17]), template matching and structure analysis are converging. The template matching approach has been absorbing structure analysis techniques and now the two approaches seem to be on the verge of fusion.

Classification Phase The classification phase consists of detecting to which class the set of features, extracted from the input symbol, belongs. For the classification, a learning expert system with models are used: a library of feature vectors of every symbol is created during a learning phase. The classification algorithms based on similarity measures calculate the distance between the feature set of the input symbol and every class. The likelihood of the calculation depends on the measures used. The most well-known measure is the Euclidean distance, but Mahalanobis and Minkowski distances are also popular. The character will be assigned to the class of minimum distance. It should be noted that the cost in computation time increases with an increase in the number of library sets used. Also, shape derivation, shape matching and hierarchical feature matching in the form of decision trees are used. In structural

and syntactic Pattern Recognition the classification stage consists in a graph matching or graph parsing process given a relational description of features.

Postprocessing The performance of a recognition system that consists only of a single-character recognition unit is not sufficient. It is necessary to use context information. The application of context makes it possible to detect errors and even to correct them. Some techniques are: diagrams, grammars and dictionaries. In OCR, dictionary methods have proven to be the most effective for error detection and correction: Given a word in which an error may be present, the word is looked up in the dictionary. If the word is present, it does not necessary mean that no error occurred. If there is an error, the error is undetectable without a wider use of context information. If the word is not in the dictionary, then the error has been detected and can be changed to the most similar word in the dictionary. The main disadvantage with the use of a dictionary is that searches and comparisons for error correction are time-consuming and increase with the size of the dictionary. In OMR systems, the use of grammars is the most popular technique in order to validate the scores and solve ambiguities. The grammar formalized in our system has been exposed in section 1.2.3.

1.3.2 System Levels

According to the approach proposed by Kato ([18]), an OMR system has several layers, corresponding to the abstraction levels of the processed information, see Fig. 1.15:

- The image layer is formed by pixels.
- The graphical primitive layer is formed by dots, lines, circles and curves.
- In the symbol layer graphical primitives are combined to form musical symbols, such as notes and rests.
- In the semantic-meaning layer information, the pitch and the beat of every note is obtained, and grammar rules are used to validate it and solve ambiguities.

Feedback among layers is extremely important because each level contains hypothesis of various levels of abstraction, so, if an upper layer rejects a result produced from lower layers (e.g. a certain object is not what it has been determined to be), the system must be able to correct this error and classify the object again.

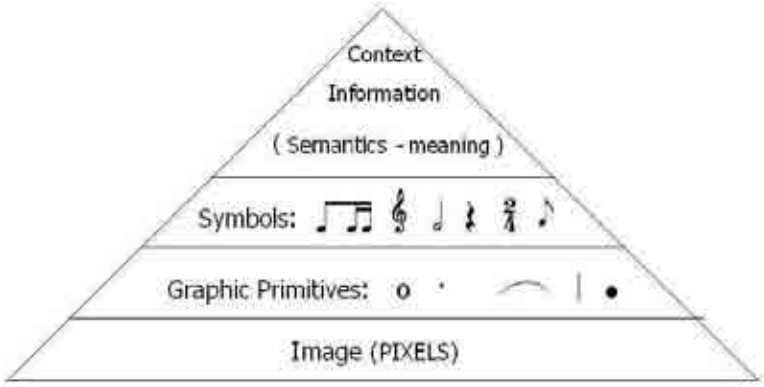


Figure 1.15: (a) Levels of a OMR system.

1.4 Objectives

The main objective of this work is to achieve the early-level stages of the Optical Music Recognition system in old handwritten musical scores: segmentation of staff lines and detection of graphical primitives.

- Segmentation of musical score blocks in documents.
- Detection and extraction of staff lines in modern and old handwritten musical scores (from XVI to XIX centuries).
- Detection of graphical primitives: vertical lines, bar lines, filled head notes and whole head notes.
- Classification of clefs: sol, do and fa.
- Formalize a grammar to validate musical scores.
- Perform a system reliable enough to avoid the need for human verification (using context information).

Concerning the recognition of old documents, the applied objective is the recognition of ancient manuscripts and their conversion to digital libraries, towards the preservation of cultural heritage. In addition, these scores of unknown composers could be edited and published (contributing to the diffusion of artistic and cultural heritage).

1.5 Structure of the dissertation

This dissertation is organized as follows:

In Chapter 1 an introduction to Optical Music Recognition is done: after the framework and applications of OMR, the structure of scores and layers of the system are shown. Afterwards, objectives of this work are exposed.

Chapter 2 presents the state of art in Optical Music Recognition, covering research both in printed and handwritten musical scores.

In Chapter 3 the methodology used in our system is briefly exposed: from general image processing techniques (such as binarization, filtering and morphological operations) to statistical classification techniques and an explanation of Graph Grammars and Graph Matching techniques.

In Chapter 4 our approach to detect primitives in modern and old handwritten scores is shown. First of all, our preliminary work with modern handwritten scores is exposed: the detection of staff lines is performed using Hough Transform and projections, and morphological operations are used for the detection of graphical primitives (vertical lines, filled head notes and whole notes). After that, our work with old scores is shown: for the detection and extraction of staff lines, a contour tracking process is required to cope with deviations in staff. Concerning graphical primitive detection, morphological operations and median filters are used.

In Chapter 5 some illustrative experimental results of modern handwritten scores (first) and old handwritten scores (taken from the Archive of Barcelona) are reported.

In Chapter 6 the concluding remarks and future work are exposed.

Chapter 2

State of the Art in Optical Music Recognition

In this chapter, a review of the research literature concerning Optical Music Recognition, also known as Musical Score Recognition (MSR), is presented. First works in this field were done by Prerau and Pruslin in 1966 and 1970, and interest in OMR has been growing in last decades, appearing several OMR systems and even a keyboard-playing robot in Japan called Wabot-2 described in [19].

As it has been said in the introduction, there is a lot of literature about the recognition of printed scores, whereas few research works have been done in handwritten ones. For that reason, stage of art about OMR will include works done for both printed and handwritten musical scores.

An interesting survey of classical OMR (from 1966 to 1990) can be found in [5], where several methods to segment and recognize symbols are reviewed. It must be said that most work through 1990 has concentrated on locating staves and isolating and recognizing symbols. Nowadays, problems in OMR of printed scores include effective algorithms to interpret the resulting 2-D arrangement of symbols, and precise formalisms for representing the results of interpretation. Referring OMR for handprinted scores, research is in a early stage and more work must be done.

2.1 Overview of techniques used in OMR

In this section, a review of techniques used for different authors for each stage of the OMR system is exposed, but before that, it must be said that the definition of musical symbols varies through authors: In one hand, Prerau ([20]) define music symbols as all four-connected regions that remain after staff lines have been removed (that means, that a beamed note sequence is called a single symbols). He distinguish between characters (which are size invariant and OCR methods can be used), and other malleable symbols (beams, slurs...) which have a parameterized shape (so, OCR techniques will not work). In the other hand, authors such as Mahoney ([21]) consider that music symbols are composed of

pattern primitives such as stems, beams and note heads. He distinguish between the music symbols that describe what is to be played (notes, clefs, key signatures...) and music symbols that describe how it must be played (dynamic and tempo markings). That distinction implies the use of different techniques in the symbol classification stage.

2.1.1 Binarization and Noise Reduction

The first operation in a OMR system is binarizing the gray-scale image into a binary image: Some authors [6] do not apply a binarization method because the scanner performs automatic thresholding to obtain a binary image. Others [20] choose the threshold manually.

Concerning noise reduction, in [6] a horizontal low-pass filter is used to remove short breaks in staff lines and symbols, whereas in [22] a three-by-three mask is used to eliminate isolated black pixels and to fill in isolated white pixels. Recent works ([23]) use adaptive binarization techniques to binarize in a more robust way.

In the vision system for the Wabot-2 robot (see [19]) the image is subdivided and each region separately thresholded to allow for uneven illumination. The image is then rotated as required and normalized to compensate for distortions introduced in scanning.

2.1.2 Detection and extraction of staff lines

Staff lines play a central role in music notation, because they define the vertical coordinate system for pitches, and provide a horizontal direction for the temporal coordinate system. The staff spacing gives a size normalization that is useful both for symbol recognition and interpretation: the size of musical symbols is linearly related to the staff space.

Most OMR systems (except Wabot-2 [19]), after recognizing the staff, remove it from the image in order to isolate musical symbols and facilitate the recognition process.

The approach proposed in [24] eliminates all thin horizontal and vertical lines, including many bare staff-line sections and stems. This results in an image of isolated symbols, such as note heads and beams, which are then recognized using contour tracking methods. This preprocessing step erases or distorts most music symbols other than quarter notes and beamed note groups.

Prerau ([20]) identifies tree ways in which staff lines interfere with symbol recognition:

- The staff lines graphically connect symbols that would normally be disconnected.
- The staff lines camouflage the contour of a symbol.
- The staff lines fill in symbol areas that would normally be blank.

In [20], the process is divided in fragmentation and assemblage. In the fragmentation step, the system scans along the top and bottom edges of staff lines to identify parts of symbols lying between, above and below the staff lines (a new symbol fragment is begun whenever a significant change in slope is encountered). Fragments from a single symbol are separated by the gaps left from crossing staff lines. In

the assemblage step, these symbol fragments are recombined using a simple connection rule: two symbol fragments (separated by a staff line) will be connected if they have horizontal overlap. One disadvantage with this technique is that symbols which merge with staff lines do not always have horizontal overlap, so with this method, would keep disconnected when it should be connected.

Mahoney (see [21]) distinguishes between two types of line removal, removing real lines (bare staff-line sections) and removing ideal lines (complete staff lines). The goal is to remove only those parts of the line that do not overlap other symbols, what is accomplished by removing only those portions of the line satisfying the line's allowed thickness range. For the identification of staff lines, he uses a strategy similar to the symbol identification method: staff-line candidates are constructed (including all thin horizontal lines in the image) and the staff line descriptor (specifying allowable thicknesses, lengths and gap-lengths) is used to classify staff lines. Similarly, the ledger-line descriptor is used to classify ledger lines. With this method, good extraction of staff lines and ledger lines is achieved, although more work is needed for dealing with the more difficult cases of line-region overlap.

Carter and Bainbridge (see [6]) propose a system for segmentation that uses processing based on a Line Adjacency Graph (LAG). Because the detection of places where a thin portion of a symbol tangentially intersects a staff line is difficult, mostly methods create gaps in symbols. Carter proposes a LAG-based analysis that successfully identifies such tangential intersections of symbols with staff lines. In addition, the system locates staff lines despite the image rotation of up to 10 degrees, copes with slight bowing of staff lines and with local variations in staff-line thickness. This method also uses horizontal projections to first locate staff lines (see Fig. 2.1). An extensive explanation of this approach will be exposed in next section.

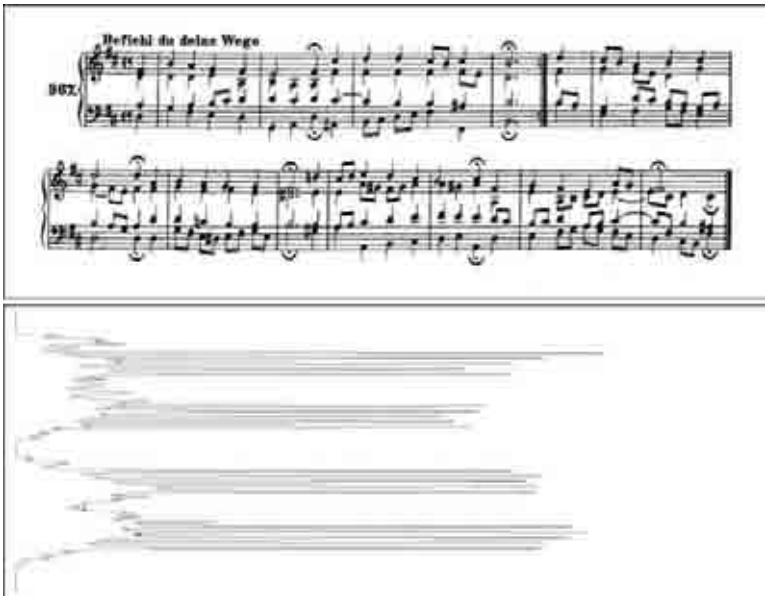


Figure 2.1: Staff detection using projections.

In [18] the detection and extraction of staff lines is performed using histograms, run-lengths and projections. After determining the spacing of staff lines and their location (using run-lengths and histograms), the staff is analyzed (tracking from the left), eliminating short horizontal runs whose width is under a certain threshold. An extensive explanation will be found in next section.

In [22] and [25], projection methods are used to recognize staff lines, which are found in a Y projection. A defined threshold is used to select projections strong enough to be candidate staff lines. These candidates are searched to find groups of five equally-spaced lines. A score after staff removing is shown in Fig. 2.2.

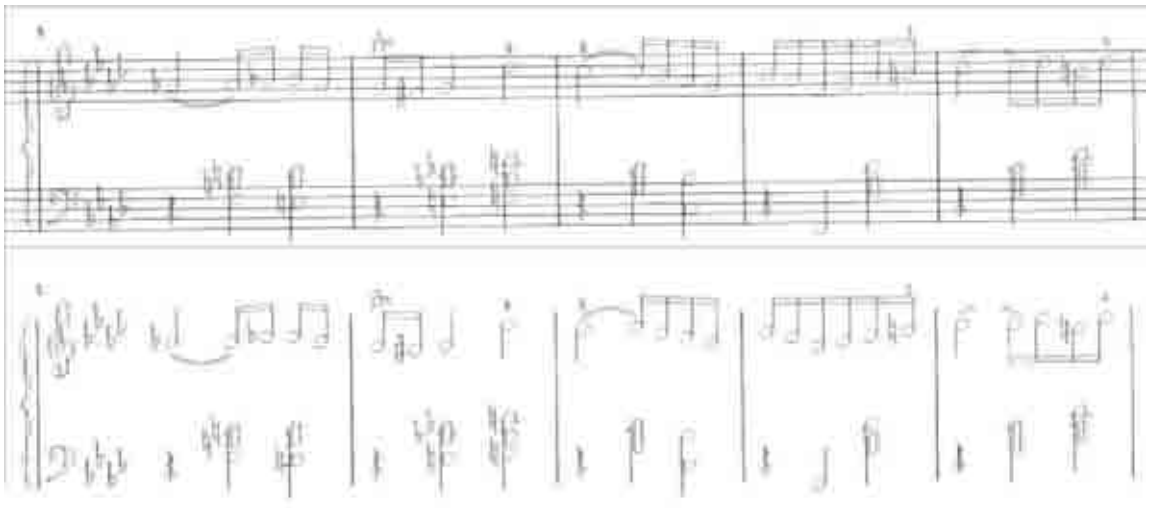


Figure 2.2: Staff removing.

In [26] a method for printed music recognition computationally inexpensive is presented. The staff lines are located by looking for long horizontal runs of black pixels. Then the neighborhood of each staff-line pixel is examined to determine whether a music symbol intersects the staff line at this point. The image is processed one staff at a time, to accommodate the memory limitations on a PC. Staves are located by examination of a single column of pixels near the left end of the system. Large blank sections indicate gaps between staff lines, and are used to divide the image into individual staves. Complete staff separation is no always achievable, because parts of symbols belonging to the staff above or the staff below may be included.

In [27] the method proposed consists in a vertical projection, projection filtering, local minima regions finding (those regions correspond probably to the regions where there is only staff lines), horizontal projection of each local minima regions (to ensure that that those peaks are certainly lines), and linking different peaks between them in order to built up the every staff. The author comments that this technique is robust because even if these lines are bowed, skewed of fragmented they are always found. To delete these staff lines, a straightforward criterion based on thickness is used. The thickness of each line is estimated in according to the width values of the lines peaks. Then if width is smaller than a

threshold (proportional to the estimated thickness) the line points at this place are erased. The problem of split symbols will be carried over in recognition stage.

Leplumey ([28]) presents a method based on a prediction-and-check technique to extract staves, even detecting lines with some curvature, discontinuities and inclination. After determining thickness of staff lines and interlines using histograms and run lengths, some hypotheses on the presence of lines is done grouping compatible observations into lines. Afterwards, those segments are joined to obtain staff lines, thanks to the construction of an interpretation graph. This method process allows little discontinuities thanks to the use of a local predicting function of the staff inclination.

In [29] an OMR system for handwritten scores is described. In such scores, only musical symbols are drawn by hand, because staff lines are printed. His segmentation stage detects staff lines using measures of line angle and thickness. A window is passed over the image to compute a line-angle for every black pixel. The line angle is measured from the center of the window to the furthest black pixel in that window; this furthest black pixel is chosen so that the path from it to the center does not cross any white pixels. To detect staff lines, a large window radius is used. this causes covered staff-line sections to be labelled with a horizontal line-angle despite the interference of the superimposed musical symbols. Once a line angle has been determined, a line-thickness can be measured. These two measurements, combined with adjacency information are used to identify horizontal lines.

Systems that do not remove staff lines The Wabot-2 robot must also be commented (see [19]) because it performs a template matching without removing staff lines: staff lines are detected and used to normalize the image, to determine the score geometry, and also to restrict the search area for music symbols (then, the recognition of musical symbols must learn symbols which include segments of staffs). Staff lines are detected in hardware by a horizontal line filter, tolerating some skew. Where five equally-space lines are found, a staff is deemed to exist. Normalization parameters include staff location, staff inclination, area covered by staff and note-head size. In preparation for further processing, the image of each staff is normalized according to these parameters.

2.1.3 Extraction and Classification of musical symbols

After removing staff lines, the classification of music symbols starts:

Pruslin [24] uses contour tracking to describe connected binary image regions which remain after deleting horizontal and vertical lines. Classification depends both on trace properties as well as on inter-trace measurements (a method for template matching using contour traces is developed).

In [20], relative symbol size is used for an initial classification, because each type of music symbol is significantly different in overall size from almost all other types of music symbols. For that reason, the bounding-box dimensions of each symbols are expressed in staff-space units. The height and width of the bounding box are used to look up a list of possible matches (there is a pre-computed table containing the standard areas of each symbol in a height/width space). Typically there are three to five possible matches for each symbol, so heuristic tests are used to distinguish symbols that overlap in

the height/width space, taking advantage of the syntax, redundancy, position and feature properties of each symbol type. Notice that, this classification os dependant of the publisher, and will not work in handwritten scores.

In [21], pattern primitives (such as note heads, stems beams and flags) are combined to form music symbols (e.g. notes, chords and beamed note sequences). This distinction between pattern primitives and musical symbols simplifies the recognition task, because it is easier to design flexible recognition procedures around simple descriptions. The primitive lines accommodate the variable parameters of the symbols: a parameterized composite symbol such as a beamed note sequence is made up of non-parameterized characters and parameterized lines. It does not use context information for the recognition of primitives, but it is used to infer musical symbols from the relationships between the various kinds of primitives. The pattern primitives used are lines, dots and characters, and classes of primitives are described using ranges of values for parameters. After extracting line primitives, dot primitives are processed and removed. All measures of distance are normalized on staff-line and staff-space thickness. Sample line parameters for describing staff lines, ledger lines, beams, note stems and bar lines are: principal direction (horizontal, vertical), angle, thickness, length, maximum permitted gap. Sample region parameters for describing whole, half and quarter note heads, flags, sharps and dots are: mass, width, height and inclination angle. This process is initially used in an interactive mode to develop object descriptions and to tailor predefined descriptions to new music samples for which the old descriptions do no work well.

In [30] and [6] the recognition system is in an early stage. Objects are classified according to the bounding-box size, and according to the number and organization of their constituent sections. An example of their classification of musical symbols is shown in Fig. 2.3.

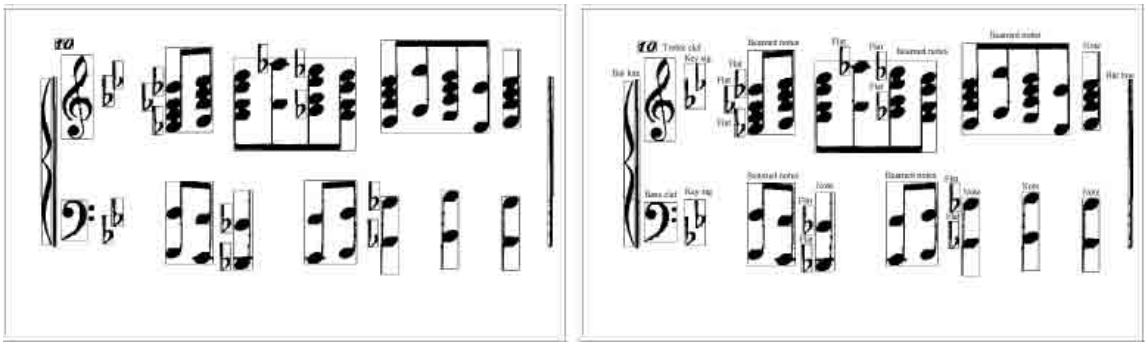


Figure 2.3: Classification of musical symbols.

In the Wabot-2 robot [19], musical symbols are recognized according to a two-level hierarchy: the upper level (in which the recognition of staff lines, note heads and bar lines is done) is implemented in hardware and the lower level in software. The search is performed using hardware-implemented template-matching.

Lee ([22]) uses projection methods to recognize staff lines, bar lines and notes (including notes and

rests). Once an image containing only a staff nucleus is obtained, an X and Y projections are used to find bar lines. Notes are recognized using X and Y projections from a small window around the symbol. Characteristic points in the projections are used for classification; a comparison is made with stored projections for known symbols. The main disadvantage of this method is that is rotation-sensitive.

In [26] an initial classification is obtained from the symbol height and width (as in [20]), and then pixels in few particular rows and columns of the symbol-image are examined (because complete template matching is too computationally expensive). Some preliminary work on chord recognition is also present. An important problem is the noise-sensitivity of the method.

In [18] a sophisticated symbol recognition stage with a top-down architecture is performed. It consists on pattern processing and semantic analysis, which will be described in next section.

In [31] a system that extracts heads and stems in printed piano scores is performed using a neural network: After extracting all regions candidates of stems or heads, a three-layer neural network is used to identify heads; the weights for the network are learned by the back propagation method. In the learning, the network learns the spatial constraints between heads and surroundings rather that the shapes of heads. Afterwards, this networks are used to identify a number of test head candidates. Finally, the stem candidates touching the detected heads are extracted as true stems.

In [32] the recognition system for printed scores is composed of two modules: the low-level vision module uses morphological algorithms for symbols detection; the high-level module context information to validate the results. Because morphological operations can be efficiently implemented in machine vision systems, the recognition task can be performed in near real-time.

Roach ([29]) proposes that knowledge about music notation could be represented in a rule-based system, and this information should be applied starting with the earliest steps of symbol segmentation and recognition. Images are digitalized in low resolution, so it is not clear the effectiveness of the system. The primitives recognized are: circular blobs (for closed note heads), circles (for open note heads), horizontal lines, non-horizontal line segments and arcs (clefs are not recognized). The location and orientation data for each primitive are intended to from the input to a high-level visual expert system. Primitive identification is coded as several passes, using context information in the last pass. Note-head detection is extremely difficult in these handwritten images, and a general-purpose blob detector does no work. Thus, note heads are searched for in constrained locations: first, verticals lines (which might be note stems) are located, then a thickness measure is used to test for wide spots at the ends of each potential stem; if there is a wide spot (whose circularity is under a certain threshold), it is accepted as a note head.

In [33], a probabilistic framework for recognition of printed scores is presented. The system uses an explicit descriptive model of the document class to find the most likely interpretation of a scanned document image, carrying out all stages of the analysis with a single inference engine (which allows for an end-to-end propagation of the uncertainty). The global modeling structure is similar to a stochastic attribute grammar, and local parameters are estimated using Hidden Markov Models (HMM). The HMM parameters are estimated from a training set using the segmental k-means algorithm. It also uses The

Hough Transform and a low-band filter to locate lines and note heads of note groups.

In [27] symbols are isolated by using region growing method and thinning. After the polygonalization of the object, a parameter defined as a minimum distance to the contour is attributed to each segment (the aim is to attribute the maximum distance of its points to the note head segments and the minimum to others), so spurious segments can be eliminated and some configuration segments are transformed. Once the skeleton structure is simplified, the attributed graph is constructed: the graph nodes correspond to the segments and the graph arcs to the links between segments (see Fig. 2.4). After constructing the graph, symbols are classified (using the distance to the contour) in symbols including notes with black heads (there is at least one segment having a distance to the contour exceeding a certain threshold) and the others. Half notes are detected if there is a stem with a little loop in its extremes.

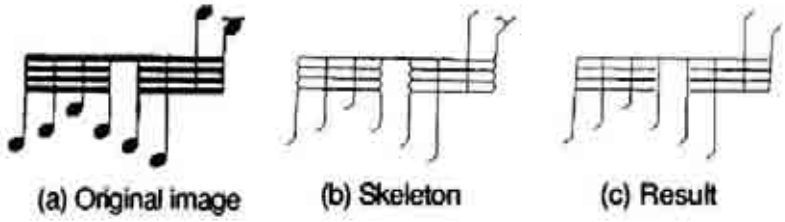


Figure 2.4: Graphical representation of the constructed attributed graph of Randriamahefa.

In [15] the development of a stroke-based segmentation approach using skeletons and mathematical morphology for handwritten scores is exposed. This approach will be described in next section.

In [13], a grammar is formalized to work in the image level to produce an accurate segmentation and thus accurate recognition. Whereas most grammars are usually used at a high level to validate the structured document, Coüasnon's system uses context information (syntax) to control the entire recognition process. Further explanation can be found in next section.

Pinto describes in [34] a system for the recognition of ancient musical scores, where the recognition process is based on a graph structure of classifiers, including the construction of a class hierarchy associated with recognizers that distinguish between clusters of classes based on selected object features. Further information will be found in next section.

2.1.4 Validation

Rules on music notation makes the recognition task easier, because the information of two-dimensional relationships between musical symbols can be captured in a syntactic description of music. For that reason, most authors define grammars describing the organization of music notation in terms of music symbols. In [35], lower level grammars are used to describe the structure of individual music symbols, so they can be used for symbol recognition. High-level grammar is simple, describing a piece as a sequence of staves, where a staff starts with a clef, key signature, and several measures. Five adjacency operators

(above, below, right of, above-right of, above-left of) to relate terminal and nonterminal grammar symbols are defined. The terminal grammar symbols are geometric figures such as white dots, black dots, and oriented lines of various lengths.

In [20] develops algorithmic implementations of syntax rules, used to constraint the possible locations of various symbols. He makes a distinction between notational grammars (to recognize important music relationships between the symbols of the music sample) and higher-level grammars for music (for phrases and larger units of music).

In [19] the robot uses a musical grammar to correct errors such as missing beats or contradictory repeat signs. Constraints applied to three-part organ music are:

- Each of the three parallel voices have the same total note duration.
- A fat double bar appears only at the end of each part.
- A treble or bass clef always appear right at the start of each staff.
- The time and key signatures almost never change within a system, and can be determined by majority rule.
- The number of beats in each ba should match the time signature.

In [32] a high-level reasoning module is developed. The system utilizes prior knowledge of music notation to reason about spatial positions and spatial sequences of recognized symbols. This module is composed of a connected components analysis and a reasoning module (that verifies if every musical symbol accomplishes its own constraints). The high-level module also employs verification procedures to check the veracity of the output of the morphological symbol recognizer.

In [14] a graph grammar for recognizing musical notation is presented. It is a programming style that allows determination of the high-level meaning of music notation given the low-level recognition of the musical symbols. The author comments that it is not clear how best to represent music as a graph, because there are local and non-local interactions between primitives that must be considered in order to obtain the semantics (such as pitch and duration of notes). To address this problem, the input graph to the grammar is constructed as a set of isolated attributed nodes representing the musical symbols. The grammar itself forms the edges representing the significant associations between the primitives that are necessary in determining the meaning. A graph representation of music provides a platform for which the formation of all potentially significant associations between primitives is quite natural; the determination of the truly significant associations occurs only after examination of the context. Although the proposed approach relies on the ability to control the order of application of the productions, there may be some portions of the grammar in which the order need not be specified, so, potential parallelism in the grammar is also made explicit. In this paper, a graph grammar for recognition of pitch, duration and voice chords in music notation is fully presented.

Ng describes in [15] a rule-based system to process the primitives in three levels: local (note by note), intermediate (bar level) and global (global information). Details can be found in next section.

2.2 Key papers in OMR

In this section, the most relevant papers about OMR are presented.

OMR system for printed piano music In [18] a sophisticated recognition system for printed piano music is developed. First, the spacing of staff lines is estimated by scanning 10 evenly-spaced columns in the image. Run-lengths are measured in each of these columns, and histograms of run lengths are calculated. The maximum in the 0-pixel histogram is interpreted as the staff spacing, and the maximum in the 1-pixel histogram is interpreted as the staff-line width. Once the staff size has been established, the staff lines are located. Run-lengths in 10 evenly-spaced columns are used to get accurate local estimates of staff-line spacing and width. Next, small rectangular sections of the staff are analyzed, near the left and right ends of the staff. Short horizontal runs are eliminated; this eliminates most music symbols, but beams and portions of note heads and clefs remain. Next an X projection is used to estimate the locations of the ends of the staff, and a Y projection is used to obtain an accurate height estimate. If the edge of the staff is not found, the window is moved further toward the margin of the page.

In this system, the distance between staff lines is used to restrict the size of symbols, and the width of the staff lines is used to set thresholds for symbol recognition. Staff lines are eliminated from the image before recognition of music symbol begins. The staff lines are tracked from the left, based on initial estimates of their location. A staff line is eliminated wherever the staff width is below a threshold.

Bar lines are found using similar methods, tailored to the recognition of piano music. Rectangular marks are placed on the right-hand staff (top staff), the left-hand staff, and between the staves. Then short vertical runs are eliminated, and hypothesized bar-line locations are extracted from a X projection. The existence of a bar line is established if all three marks hypothesize a bar line at the same X location.

Concerning the symbol recognition stage, a to down approach is used, recognizing one measure of music at a time. Because music symbols differ greatly in size, position and frequency of appearance, is difficult to devise a single method for recognizing all symbols. For that reason, they use a collection of processing modules that communicate by operating on a common working memory, which represents information about the current bar of music at five levels of abstraction: pixel, primitives, music symbols, meaning (pitch and duration of notes) and context information (interpretations). The four processing modules (primitive extraction, symbols synthesis, symbol recognition and semantic analysis) are made up of one or more recognition and verification units. The primitive extraction module contains units for recognizing stems, beams and note heads. Hypothesized primitives are removed from the pixel image. Unacceptable hypotheses are rejected at higher layers, are sent back to lower layers for further processing. Symbol recognition proceeds one measure at a time, and consists on pattern processing (what must cope with overlap between symbols, breaks and unexpected ink spots) and semantic analysis (using context information), required for solving ambiguities of complex notations. In this stage, attributive symbols (clefs, key and time signature) and note symbols are recognized.

In the postprocessing stage, recognition of symbols that span measure boundaries is performed. The

final image interpretation is formed by combining the partial results from each measure. This approach obtains good recognition rates on complicated piano music.

Bainbridge and Carter's system In [6], [30] and [36] a system based on a Line Adjacency Graph (LAG) is exposed. This system is also used to recognize ancient scores (see [37]): scores of madrigals (see Fig. 2.5) notated in *White Mensual Notation*. Symbols are correctly segmented and an early classification stage has been implemented.

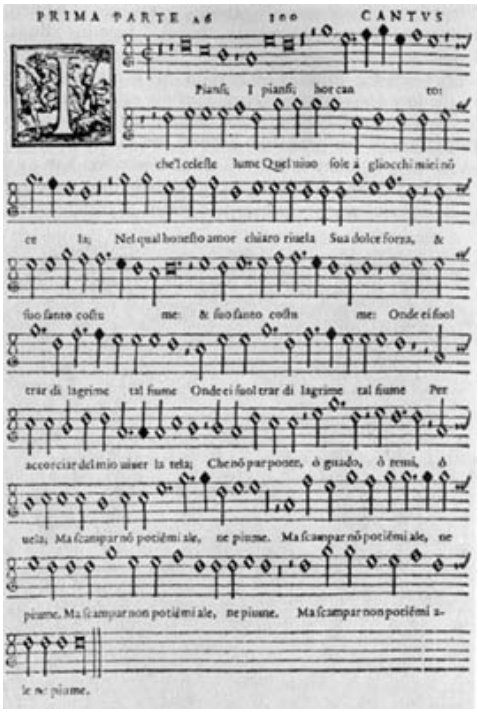


Figure 2.5: A madrigal score of the seventeenth century.

This system successfully identifies tangential intersections of symbols with staff lines, locates staff lines despite the image rotation of up to 10 degrees, copes with slight bowing of staff lines and with local variations in staff-line thickness.

Region information, derived from the LAG, is used to determine whether a symbol has merged with a staff line. The LAG is formed directly from a vertical run-length encoding of a binary image. A transformed LAG is formed by linking together neighboring segments (a segment is defined as an individual vertical run of pixels) to form sections. Sections are formed using a left-to-right scan, in which neighboring vertically-overlapping segments are linked. Junctions occur when a segment in one column overlaps several segments in an adjacent column; sections are terminated at these junctions. In the transformed LAG, each section is represented by a node in a graph, and junctions are represented by edges in the graph. The nodes in the transformed LAG should correspond to structural components

of musical symbols. A rule limiting the rate of change in section thickness helps accomplish this. The rule states that the current section is terminated if its average height differs from the height of the next segment by more than a factor of 2.5. This rule ensures that staff lines and ledger lines are assigned to different sections than are portions of music symbols.

Similarly, a note head is assigned to a different section than the note stem. Section formation is also insensitive to small rotations. The author comments that the use of a LAG is preferable to other common methods of data reduction and feature extraction, such as thinning. The LAG i equally effective describing blobs and lines.

Noise removal on the transformed LAG proceeds by removing isolated or singly connected sections with small area (for example, 5 pixels in a 400dpi image). If removal of these noise sections turns a multi-way junction into a two-way junction, then the two remaining sections are merged provided their heights differ by less than a factor of 2.5.

The transformed LAG is searched for potential staff-line sections (filaments): sections that satisfy criteria related to aspect ratio, connectedness and curvature. Long beams may be included as filaments; these are filtered out using a histogram of filament thickness to determine a threshold for maximum staff-line thickness. Roughly collinear filaments are concatenated together into filament strings, thereby bridging the gaps introduced by superimposed music symbols. The occurrence of five horizontally overlapping and roughly equally-spaced filament strings is recognized to form a staff. After staff lines are identified, the transformed LAG is restructured: further merging of non-staff sections takes place, now that junctions with staff staff-line sections have been specially marked. At this point, musical symbols are effectively isolated from the staff lines. Connected non-staff-line sections are combined to form objects, which correspond to music symbols or to connected components of music symbols.

Concerning the classification stage, after the extraction of staff lines the system performs a description of objects which correspond to music symbols or connected components of music symbols. These segmentation results are interpreted by a recognition system, where the objects resulting from the segmentation are classified according to bounding-box size, and the number and organization of their constituent sections. The author comments that if there are overlapping or superimposed symbols another algorithm will be required.

The Wabot-2 robot The first robot (see Fig. 2.6) that recognizes scores and plays the organ should also be commented. The Wabot-2 robot [19] has a vision system capable of interpreting images taken of sheet music placed on a music stand. The CCD camera is placed on the robot's shoulders; thus, while the robot plays the keyboard, vibrations prevent the CCD camera from being used for score reading. For that reason, the sheet music must be read and interpreted before the robot begins to play. Very fast image interpretation is achieved in the Wabot design (10-15 seconds are needed to interpret one page of music). Special purpose hardware and parallel processing are used to achieve such high speed. If simple scores are used, the recognition rate is nearly 100%. The robot plays three-part organ scores, containing relatively simple notation. There are three staves per system: the top staff is for the right hand, the

middle staff is for the left hand, and the bottom staff is for the feet. The robot's video camera captures images of organ scores that have been placed on a music stand.

Figure 2.6: The WABOT-2 robot.

The following attributes of the imaging must be taken into consideration:

- Distortions occur when the page of music sags on the music stand.
- Some rotation may be present.
- Distance to the page may vary.
- The illumination is uneven.

The image is subdivided and each region separately thresholded to allow for uneven illumination. After staff detection, the image is rotated and normalized to compensate for distortions introduced by scanning.

Musical symbols are recognized according to a two-level hierarchy, where the upper level is implemented in hardware and the lower level in software. Staff lines, note heads and bar lines, which can occur in many places in the image, belong to the upper level. These are searched for using hardware-implemented template-matching (using an AND operation). Eight standard templates for note heads are used, and each template comes in nine different sizes. The correct template size is selected on the basis of the normalization parameters resulting from staff-line detection. The lower level of the hierarchy contains symbols whose possible locations are constrained by the recognition results for the upper level symbols; these symbols are found using sw-implemented localized search. Lower level symbols include rests, stems, flags, repeat signs, staccato and marcato marks, accidentals, prolongation dots, clefs and time signatures.

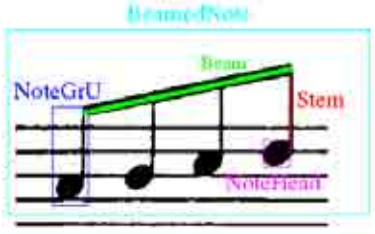
Template matching to detect filled note heads leads to incorrect matches. These are eliminated at a later stage, using knowledge about the syntax of music notation. If this method were applied to more complex notation, the problem of spurious matches might become more serious. As it stands, excellent recognition results are achieved on organ scores containing relatively simple notation.

Couasnon: Polyphonic scores Coüasnon (see [13],[38]) proposes a grammar to formalize the musical knowledge on polyphonic musical scores. The method proposed uses a grammar with operators to define the relative object positions on a score. The grammar can cope with full score, with different voices on a single staff, chords.

Objects are divided in:

- constructs: elements composed of segments (e.g. stems, beams) and noteheads plus a set of construction rules which apply to these segments.
- symbolics: symbols that can be considered as characters, such as clefs, accidentals, dynamics...

The graphical level of the grammar corresponds to the way notes and their attributes are formed and adjusted on the score. An example of the grammar rules defined for a beamed eighth note can be shown in Fig. 2.7



BeamedNote ::= *Beam* (`left_tip` NoteGr :
`closev` [NoteGr | ConsRest]* :
`right_tip` NoteGr)
NoteGr ::= NoteGrU | NoteGrD
NoteGrU ::= *Stem* `closel_downtip` NoteHead
NoteHead ::= Head `closel_same_line` [Accidental]?

Figure 2.7: The grammar rules for a beamed eighth note.

The logical level corresponds to the syntactic way of using notes in music notation. A step is defined as the smallest duration in a column of vertically aligned notes. This notion of step solves some of the problems due to polyphony and full scores, because it can manage the simultaneity of notes on a same staff (polyphony) and on the different staves of one system (full scores). An example of the grammar rules of a bar system is shown in Fig. 2.8

A parser is implemented in λProlog, and a *Delay operator* is defined to modify the way a rule parses the image, so it enables the segmentation of symbols which touch constructs.

The author claims that thanks to the separation between the operating part of the system and the definition of musical rules, the system allows the modification of rules and its adaption (defining a new grammar) to another kind of structured document.

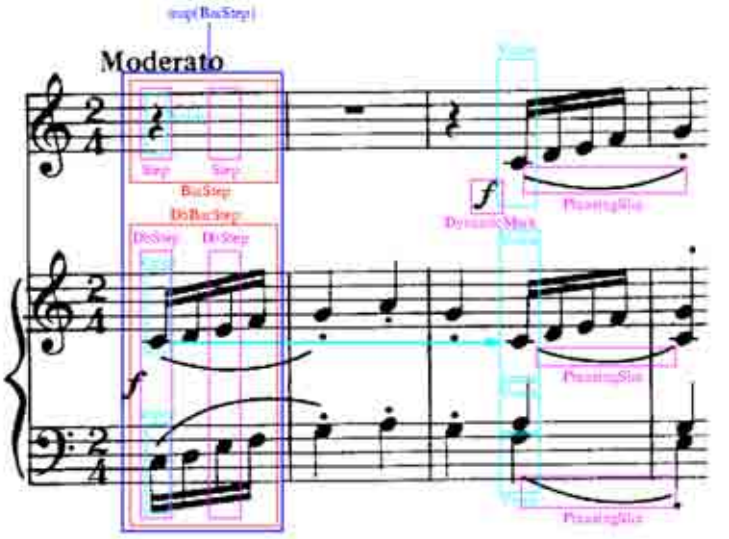


Figure 2.8: The grammar rules for a bar system.

Kia Ng: Handwritten scores Kia Ng exposes in [15] a prototype for printed scores, followed by a prototype for handwritten ones, discussing the limitations of the first one for handwritten scores processing.

In the first one, after binarizing and correcting the skew of the image, stave line location is obtained using horizontal projections. Afterwards, a line tracing algorithm with a local vertical projection window is used to detect the average stave line thickness, and electively mark the pixels that belong to each staff line, and thereby removing the staff lines. After that, a sub-segmentation algorithm disassemble musical symbols into graphical primitives (e.g. note-heads, lines, curves, isolated rests and accidental signs), and the classification stage begins: the first iteration of the classification module recognizes isolated primitive musical symbols (dots and rests) and symbols located at certain positions (clef and time signature). The recognition of other primitives is performed by interplay between the classification and the sub-segmentation modules (symbols not recognized are subdivided depending on its orientation). The classification module uses the aspect ratio of a bounding box, based on a pre-sampled training set, using a k-nearest-neighbor classifier. A screen of the application developed is shown in Fig. 2.9.

Concerning the prototype for handwritten scores, the skeleton of the binary image is obtained in

Figure 2.9: A screenshot of the software developed for the recognition of printed scores.

order to transform musical symbols into a set of interconnected curved lines. Then, junction (which join segments) and termination points are extracted from the skeleton representations.

In the staff detection phase, all horizontal line segments are parsed to determine if they belong to part of a stave line using basic notational syntax and an estimated staff line height, which has been obtained from the histogram of the number of pixels eroded during each skeletonisation iteration.

The sub-segmentation module does not detect joints that are smooth and form continuous curve segments, hence an additional process using a combination of edges, curvature and variations in relative thickness and stroke direction is required to perform further sub-segmentation and segregate the writings into lower-level graphical primitives (lines, curves and ellipses). Afterwards, primitives are classified using a KNN classifier (with appropriate training dataset) and additional feature vectors to make use of information extracted during the skeletonisation process and the junction point distributions. Each terminal point is parsed to search for any other nearby terminal points which are collinear with the current segment or following a polynomial extrapolation from the terminal points of the current segment. The author comments that a tracing routine using a database of isolated handwritten musical symbols would improve the classification stage.

After the classification phase, these sub-segmented primitives are regrouped (applying basic syntactic rules) to form musical symbols. Contextual ambiguities are resolved using relative positions of primitives in the staff, and between primitives. The reconstruction module offers an intermediate stage where extensive heuristic, musical syntax and conventions could be introduced to enhance or confirm the primitive recognition and re-groupings. In fact, there is a rule-based system to process primitives in

three levels:

- Local: At a note by note level. For example, a dot to the right of a note-head implies a dotted note.
- Intermediate: At a bar level, with a database of frequently found rhythmical patterns and beam groupings.
- Global: within an estimated sections with similar tonality (including clef and key information) and timing information to clarify uncertainties.

The author comments that a graphical-user-interface with built-in basic musical syntax and some contextual intelligence is under development to assist the transcription and output. Further work will include formal approaches, such as the usage of graph-grammars, and global information (such as time and key signatures) will help to solve ambiguities.

ROMA: Ancient Music Optical Recognition In [34], a OMR method to recognize ancient musical scores (see Fig. 2.10) is described. This system copes with specific notation of ancient documents, and is developed under the Portuguese project ROMA (Ancient Music Optical Recognition).

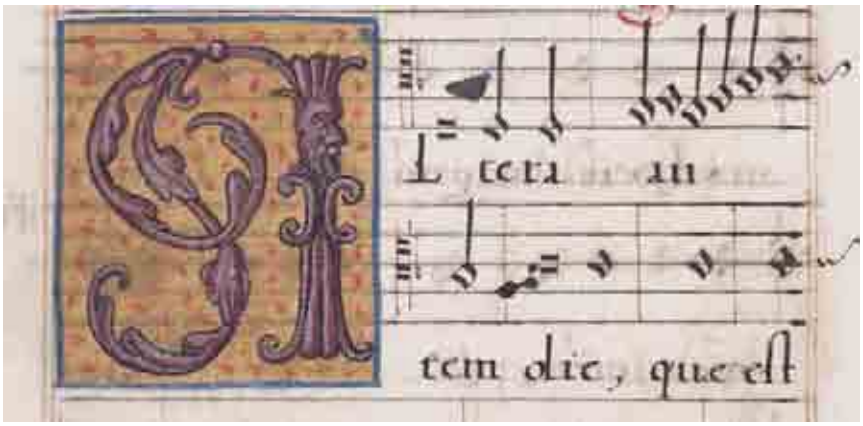


Figure 2.10: Ancient musical score.

After the preprocessing stage, the segmentation module divides the music sheet in staff lines, bars and musical symbols. The staff lines are identified using horizontal projections and small rotations of the image. Then, segments of line whose thickness is not bigger than a certain threshold are removed. Bar lines are located using vertical projections, and objects are segmented using morphological operations and connectivity analysis.

The recognition process is based on a graph structure of classifiers, divided into two steps: feature extraction and classification. The method includes the construction of a class hierarchy associated with recognizers that distinguish between clusters of classes based on selected object features. Then, a method for the search of optimal graph hierarchy (manual and automated) and for the classification algorithms

themselves is proposed. Finally, the reconstruction stage is needed to relate the recognized symbols with each other and with its staff lines and bars, creating the final description of the music.

The author exposes that future work includes the complete automation of recognition graph building and the generalization of the developed techniques to other musical notations.

Identification of a writer's hand in handwritten musical scores In [23] a system for the identification of a writer's hand in musical scores is described. Although the work is in a early stage, it must be commented due to the interesting research field of digital music notation matching. The author claims that an important problem of the current registration of old historical music scores lays in the identification of corresponding writer. This identification is based on the automated analysis of notation graphic features. The process of automated identification requires a definite level of handwriting content understanding. To extract a concrete feature of music objects means at first to recognize these music objects. Only after the recognition of separate note symbols from the whole note graphics, for instance, it is possible to describe them using characteristic feature sets.

The system of significant features of handwriting required for the identification process can not be fully completed neither can all existing variances be included into a system (the more handwritings will be analyzed, the more will the system be increased and adapted). Every note element will be represented by its tree structure, which shows an ideal appearance of the object. The special stochastic rule-based method will be developed to handle an uncertain recognition results. The goal of the recognition problem is to relate the structures found in the image with the underlying object feature models. Once object features are given in the form of structural descriptions, the matching algorithm must solve the following three problems simultaneously: determine which image primitives belong to the same object feature, determine the identity of the structure and assign the correct object features to each image primitive.

Because of the early development of the system (in fact, no results are shown in the paper), the author concludes that further work must be done in the classification stage, performing graph matching techniques with the use of heuristic approaches.

2.3 Conclusions

In this section, main OMR systems have been described. Table 2.1 shows main techniques used in staff detection and removal, whereas table 2.2 shows the main techniques used in classification of musical symbols. The validation phase is performed basically using rules, grammars and graph grammars.

Finally, results of main systems exposed in last subsection should be commented:

The system defined in [24] only recognizes quarter notes, beamed note groups and chords, whereas Prerau's approach [20] recognizes a more complete set of symbols (clefs, accidentals, half quarter and eight notes) with good recognition rates. Andronico [35] describes a system that recognizes clefs, key signatures, notes, rests and accidentals in simple monophonic music. In [21] a system with human interaction recognizes simple polyphonic music, whereas Clarke's system [26] recognizes single line melodies

Staff Detection: Author	Techniques
Prerau	Contour Tracking
Mahoney	Construction of candidates, Staff line descriptors
Carter and Bainbridge	Projections, LAG
Kato, Lee, Dan, Clarke, Randriamahefa	Histograms, Runlengths, Projections
Leplumey	Runlengths, Reconstruction using a Graph
Roach	Slide-window, Orientation of line segments

Table 2.1: Main Techniques used in Staff Detection

Classification: Author	Techniques
Prerau	Bounding-box, Matching
Mahoney	Features of primitives, Descriptors
Carter and Bainbridge	Bounding-box, LAG
Kato	Pattern processing, syntax analysis
Lee	Projections
Clarke	Bounding-box, pixel analysis
Vuilleumier	Hidden Markov Models
Randriamahefa	Polygonalization, Attributed Graph
Ng	Skeletons and Mathematical Morphology Operators
Coüasnon	Grammars

Table 2.2: Main Techniques used in Classification of musical symbols

with a 90% accuracy.

The Wabot-2 can perform fast, accurate recognition of simple three-part organ scores, recognizing notes, clefs, accidentals, time signatures, bar lines, beams, rests, staccato and marcato marks, but it does not recognize words, slurs, ties, expression marks, ornaments and tempo indications.

In [22] the system exposed recognizes staff lines, bar lines, notes, chords and rests.

Carter [6] has developed a system that segments under difficult imaging conditions, without an excess of ad hoc rules. It recognizes solo instrument parts, solo instrument with piano accompaniment and orchestra score with good tolerance to noise, limited rotation, broken print and distortion.

System exposed in [18] handles complex music notation (including two voices per staff with chords and shared note heads, slurs and pedal markings) with high performance rates.

OMR system for handwritten recognition exposed in [29] shows acceptable performance results. Due to the low resolution of digitized images, it is difficult to estimate how this method would compare to others when applied to higher-resolution input.

The grammar formalized by Coüasnon in [13] can recognize notes, rests, chords, accents, clefs, key and time signature, phrasing slurs, dynamic markings. Abbreviations, ornaments and lyrics are not included. The classification system for the symbols and the segmentation and merging of connected components is under development, and no performance results are shown.

The prototype for printed scores described in [15] recognizes 12 different sub-symbols with a 95% reliability. The output is expMIDI, which is compatible with the standard MIDI file format, and capable of storing expressive symbols such as accents and phrase markings. No recognition rates are shown in the recognition of handwritten scores.

System proposed in [34] obtains high performance results (ancient scores are recognized with a 97% of accuracy). The output of the system is a normalized music sheet with the original staff printed using straight staff lines and normalized symbols.

Chapter 3

Methodology Fundamentals

In this chapter, document analysis techniques for pattern recognition used in our OMR system are briefly exposed: binarization is used to separate foreground from background in images. After binarization, some noise removal techniques (such as filtering and morphological operations) are required. Segmentation can be done using contour detection, run length smearing, projections and connected component labelling. For the classification stage, some statistical classification techniques are reviewed (moments and zoning). Finally, the concepts of Graph Grammars and Graph Matching are exposed.

3.1 General Image Transformation Techniques

There are some general image techniques exposed in [39] and [40] to process a document image: geometrical transformations (translation, rotation and scale) are commonly used to correct distortions due to image acquisition (i.e. to deskew an image). Binarization separates foreground from background. Also, different sets of filters have been proposed, either in the spatial or frequential domain to improve this foreground-background separation, feature enhancement and noise reduction. Morphological operations are commonly used to identify objects or boundaries within an image. Finally, thinning and contour detection are used for reducing the image components into their essential information. Let us further describe such techniques.

3.1.1 Binarization Methods

Binarization is an operation used for separating figure (foreground) from background in graylevel images. The aim in binarization is to mark pixels that belong to true foreground regions with a value, and background regions with a different value. Depending on the threshold used for the classification in foreground or background pixels, one can find global or local-adaptive binarization methods. While global methods set up a unique threshold value for the whole image pixels, local adaptive methods decide the threshold for each pixel in terms of its context. A full review of main binarization methods can be

found in [41].

A Global binarization method: OTSU If the document is in good quality and the background is uniform, the separation can be performed using a global binarization method, such as OTSU (see [42]). This method uses a single threshold value for the entire image. Pixels with a gray level darker than the threshold value are labelled black (foreground), otherwise are white (background). An histogram of the gray levels of the image is performed, and after that, the value of the threshold will be the value that divides the histogram for its valley (see Figure 3.1).

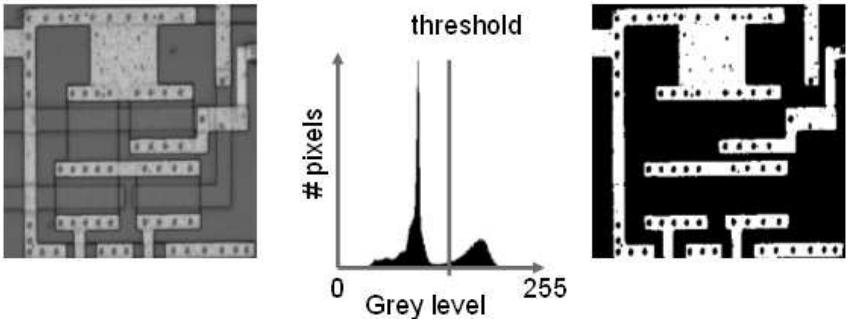


Figure 3.1: Otsu binarization method

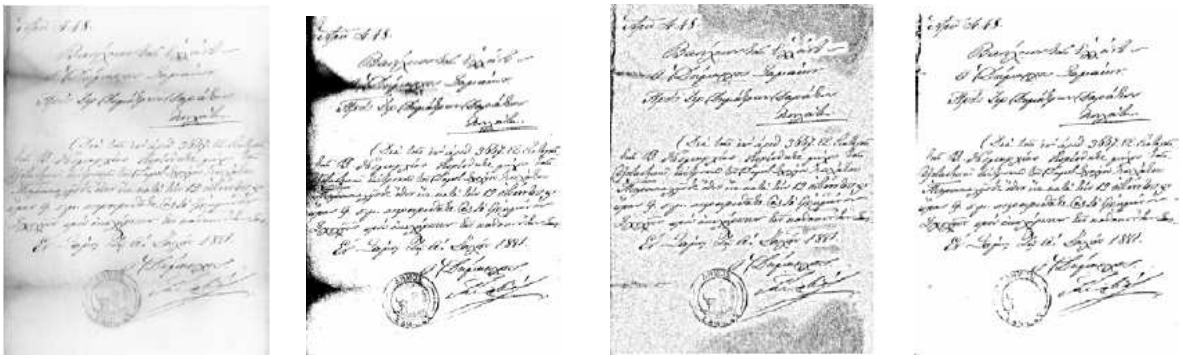
A Local adaptive binarization method: Niblack In old documents, global binarization techniques do not work because paper degradation causes non-uniformities within foreground and background. Documents are in poor condition and background has not an uniform greylevel. For that reason, local adaptive binarization techniques are required, such as Niblack (see [43]), where different threshold values are used in the image: depending on the area, this threshold varies.

The idea of Niblack's method is to vary the threshold over the image, based on the local mean and local standard deviation. The threshold at pixel(x,y) is calculated as follows:

$$T(x,y) = m(x,y) + k \cdot s(x,y) \tag{3.1}$$

where m(x,y) is the sample mean and s(x,y) is the standard deviation, in a local neighborhood of (x,y). The size of the neighborhood should be small enough to preserve local details, but at the same time large enough to suppress noise. The value of k is used to adjust how much of the total print object boundary is taken as a part of the given object.

Gatos describes in [12] an adaptive binarization technique for low quality historical documents. The proposed scheme consists of five distinct steps: a preprocessing procedure using a low-pass Wiener filter, a rough estimation of foreground regions using Niblack's approach, a background surface calculation by interpolating neighboring background intensities, a thresholding by combining the calculated background surface with the original image and finally a post-processing step in order to improve the quality of text



(a) (b) (c) (d)

Figure 3.2: Binarization method's: (a) Original image, (b) Otsu's method, (c) Niblack's method, (d) Gato's method.

regions and preserve stroke connectivity. The author comments that this method works with great success (see Fig. 3.2) even in cases of historical manuscripts with poor quality, shadows, nonuniform illumination, low contrast, large signal-dependent noise, smear and strain.

3.1.2 Filters to denoise a document image

Digital images are prone to a variety of types of noise. In addition, the binarization techniques can introduce noise. The most popular technique to reduce noise in images is filtering, a neighborhood operation consisting in the transformation of an input image I_i(k,l) into an output image I_o(k,l). Every output value is usually a function of input values in a local neighborhood around position (k,l). A pixel's neighborhood is some set of pixels, defined by their locations relative to that pixel.

Filtering is also a technique for modifying or enhancing an image (i.e. for emphasizing certain features or remove other features), and can be classified in Linear Filters and Rank-Order Filters.

Linear Filtering Linear filtering consists in filtering using a linear function, in other words, the value of an output pixel is a linear combination of the values of the pixels in the input pixel's neighborhood.

Linear filtering of an image is accomplished through convolution or correlation:

- Convolution: In convolution, the value of an output pixel is computed as a weighted sum of neighboring pixels. The matrix of weights is called the convolution kernel, also known as the filter. Steps followed are: Rotate the convolution kernel 180 degrees about its center element. Slide the center element of the convolution kernel so that lies on top of the (i,j) element of the image. Multiply each weight in the rotated convolution kernel by the pixel of the image underneath. Finally, sum up these individual products.

- Correlation: The operation called correlation is closely related to convolution. In correlation, the value of an output pixel is also computed as a weighted sum of neighboring pixels. The difference

is that the matrix of weights, in this case called the correlation kernel, is not rotated during the computation.

Linear filters, such as Averaging or Gaussian filters, are useful to smooth and remove certain types of noise:

- Mean Filter (Average):

  The idea of mean filtering is the replacement of each pixel value in an image with the mean ("average") value of its neighbors, including itself. This has the effect of eliminating pixel values which are unrepresentative of their surroundings. Mean filtering is usually thought of as a convolution filter, and it is useful for removing grain noise from images, because each pixel gets set to the average of the pixels in its neighborhood, so local variations caused by grain are reduced.

- Gaussian Filter:

  The Gaussian filter uses a statistical weighting function as the filter kernel. Pixels closer to the center pixel are given more weight than those at the extremities.

  The Gaussian 1-D function is defined as:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}} \tag{3.2}$$

  The Gaussian 2-D function is defined as:

$$p(x, y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3.3}$$

  where sigma is the standard deviation, sigma squared is the variance, and the distribution has a mean of zero (it is centered).

  The gaussian is very popular because the 2-D gaussian filter can be implemented by convolving two one dimensional gaussian filters in X and Y with the image rather than one 2-D gaussian. This results in significantly reduced computation. Since the image is stored as a collection of discrete pixels we need to produce a discrete approximation to the Gaussian function before we can perform the convolution. In theory, the Gaussian distribution is non-zero everywhere, which would require an infinitely large convolution mask, but in practice it is effectively zero more than about three standard deviations from the mean, and so we can truncate the mask at this point.

The both linear filters have side effects in that they includes the value of the noise in the calculation of the result non-linear filters try to reject the noise from the calculation to preserve features important for perception.

Rank-order filtering: Median Filtering One of the most popular rank-order filters (whose functions are non-linear) is Median filtering, because it can eliminate noise without blurring the input image.

Median filtering is a specific case of order-statistic filtering, a nonlinear operation often used in image processing to smooth and reduce "salt and pepper" noise. Median filtering is more effective than convolution when the goal is to simultaneously reduce noise and preserve edges. In median filtering, the value of an output pixel is determined by the median of the neighborhood pixels: The pixels are rank ordered in terms of their grey levels and the middle value of the distribution is selected.

As we can see, objects which are half the size of the median filter or less will be rejected and larger objects will remain intact. Thus the median filter can reject binary noise while preserving edges.

The median is much less sensitive than the mean to extreme values (called outliers). Median filtering is therefore better able to remove these outliers without reducing the sharpness of the image. Unluckily, the Median filter is slower than convolution because of the requirement for sorting, and it does not perform as well on Gaussian noise.

3.1.3 Morphological Operations

Mathematical morphology is a broad set of image processing operations that process images based on shapes. Morphological operations apply a structuring element to an input image, creating an output image of the same size. In a morphological operation, the value of each pixel in the output image is based on a comparison of the corresponding pixel in the input image with its neighbors. By choosing the size and shape of the neighborhood, one can construct a morphological operation that is sensitive to specific shapes in the input image.

While point and neighborhood operations are generally designed to alter the look or appearance of an image for visual considerations, morphological operations are used to understand the structure or form of an image, such as the identification of objects or boundaries within an image.

Although most morphological operations focus on binary images (zero as black or background, and 1 as white or foreground), some also can be applied to grayscale images. The most basic functions for many morphological operators are erosion, dilation and hit-or-miss. Others are special cases of these primary operations or are combinations of them. For a binary image we will define:

- Hit of Miss: the input image is scanned and the neighborhood of each pixel is compared with the structuring element. If there is a perfect match, a predefined value is assigned to the output image at that pixel, otherwise the output is set to its input value. This operation is useful only for simple tasks.
- Dilation: the dilated image of an object I with respect to a structuring element S, is the set of all reference points for which I and S have at least one common point. The output image is an "expansion" of the input image.

$$Dilation = I \oplus \breve{S} = \{\vec{x} - \vec{y} | \vec{x} \in I \wedge \vec{y} \in B\} \tag{3.4}$$

- Erosion: an eroded image of an object I with respect to a structuring element S, is the set of all

reference points for which S is completely contained in I. The output image is a shrinking of the original image.

$$Erosion = I \ominus \breve{S} = \{\vec{x} | \forall \vec{y} \in B : \vec{x} - \vec{y} \in I\} \tag{3.5}$$

- Opening: is defined as an erosion, followed by a dilation. Used for removing small elements and separate those objects joint by thin necks.

$$Opening(I, S) = I \ominus \breve{S} \oplus S \tag{3.6}$$

- Closing: is defined as a dilation, followed by an erosion. Used for fill small holes and join very closed objects.

$$Closing(I, S) = I \oplus \breve{S} \ominus S \tag{3.7}$$

where I is the image, and S is the structuring element.

3.1.4 Thinning

In certain applications where the line thickness is of secondary importance, the reduction of the image components into their essential information (performing a thinning or skeletonization process of the image) can simplify many operations in structural analysis. The main idea of thinning is to repeatedly delete object boundary pixels so as to reduce the line width to one pixel. This must be done without locally disconnecting the object (breaking into parts) nor deleting line end points.

The thinning requirements are formally stated as follows:

- Connected image regions must thin to connected line structures. It is very important that the results of thinning must maintain connectivity. This requirement guarantees the number of thinned connected-line structures to be equal to the number of connected regions in the original image.
- The thinned result should be minimally eight-connected. The resulting lines should always contain the minimal number of pixels that maintain eight-connectedness (a pixel is considered eight-connected to another pixel if the second pixel is one of the eight closed neighbors to it).
- Approximate end line locations should be maintained. The locations of end lines should be maintained. Since thinning can be achieved by iteratively removing the outer boundary pixels, it is important not to also iteratively remove the last pixels of a line. This removal would shorten the line and not preserve its location.
- The thinning results should approximate the medial lines. The resultant thin lines should best approximate the medial lines of the original regions.

- Extraneous spurs (short branches) caused by thinning should be minimized. Although is obvious that noise should be minimized, it is often difficult to determine whether it is noise or not. For that reason, it is best to perform thinning first and the, in a separate process, remove any spurs whose length is less than a specified minimum.

In the thinning algorithm, pixels are deleted if the following conditions are satisfied:

$$P_1 \text{ is deleted if } \begin{cases} 2 \leq NZ(P_1) \\ and \\ NT(P_1) = 1 \\ and \\ ((P_2 \cdot P_4 \cdot P_8 = 0) \text{ or } (NT(P_2 \neq 1)) \\ and \\ ((P_2 \cdot P_4 \cdot P_6 = 0) \text{ or } (NT(P_4 \neq 1)) \end{cases} \tag{3.8}$$

where:
NT = number of 0 to 1 transitions in the ordered sequence P2,P3,...,P9
NZ(P1) = number of nonzero neighbours of P1.
and the neighours of P1 can be seen in Fig. 3.3.

P_3	P_2	P_9
P_4	P_1	P_8
P_5	P_6	P_7

Figure 3.3: Neighbors of the pixel P1.

This process is repeated until there are no more changes in the image. For an example, see Fig. 3.4.

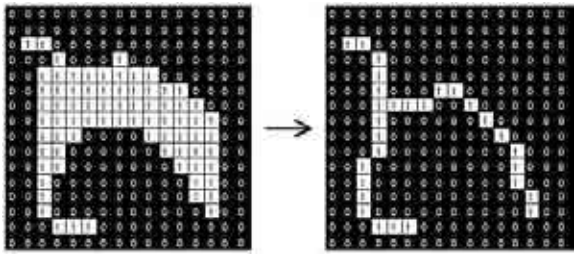


Figure 3.4: Thinning

In [44] an approach for the skeletonization of engineering drawings to achieve the data reduction goal is presented. The basic idea consists in a contour vectorization process applied to the contour image of a drawing, followed by a contour skeletonization process by matching the contour vector pairs to obtain

the skeleton of the drawing.

3.1.5 Contour detection

Contour detection can be thought of as the reciprocal operation of thinning. Whereas thinning yields the inside skeletons, contour detection yields the outside boundaries (contours or edges). Since a single contour envelopes a single region, contour detection can be used for region detection. Contours are comprised of boundary foreground pixels that border background pixels. These contours can be found easily by examining each pixel within a 3x3 window; if the center pixel is a foreground pixel and at least one of its neighborhood pixels is a background pixel, then the center pixel is a contour pixel.

The contour image can be used in a number of ways: the number of contours gives the number of regions, the centroid of the boundary pixels gives a measure of the region location, the length of a contour indicates the enclosed region size, the length and enclosed area can be used to give a measure of how elongated or "fat" the region is, curvature and corner features can be determined from the contour to determine the region shape.

Roberts The Roberts Cross operator performs a simple, quick to compute, 2-D spatial gradient measurement on an image. It thus highlights regions of high spatial frequency which often correspond to edges. In its most common usage, the input to the operator is a grayscale image, as is the output. Pixel values at each point in the output represent the estimated absolute magnitude of the spatial gradient of the input image at that point. The operator consists of a pair of 2x2 convolution kernels. One kernel is simply the other rotated by 90 degrees.

$$G_x = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \qquad G_y = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \tag{3.9}$$

These kernels are designed to respond maximally to edges running at 45ř to the pixel grid, one kernel for each of the two perpendicular orientations. The kernels can be applied separately to the input image, to produce separate measurements of the gradient component in each orientation (call these Gx and Gy). These can then be combined together to find the absolute magnitude of the gradient at each point and the orientation of that gradient. The gradient magnitude is given by:

$$|G| = \sqrt{Gx^2 + Gy^2} \tag{3.10}$$

and the angle of orientation of the edge giving rise to the spatial gradient (relative to the pixel grid orientation) is given by:

$$\theta = \arctan(Gy/Gx) - 3\pi/4 \tag{3.11}$$

In this case, orientation 0 is taken to mean that the direction of maximum contrast from black to white runs from left to right on the image, and other angles are measured clockwise from this.

Sobel The Sobel operator performs a 2-D spatial gradient measurement on an image and so emphasizes regions of high spatial frequency that correspond to edges. Typically it is used to find the approximate absolute gradient magnitude at each point in an input grayscale image. The operator consists of a pair of 3x3 convolution kernels. One kernel is simply the other rotated by 90 degrees. This is very similar to the Roberts Cross operator.

$$G_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \qquad G_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} \tag{3.12}$$

As in Roberts's, these kernels are designed to respond maximally to edges running vertically and horizontally relative to the pixel grid, and can be applied separately to the input image (to produce separate measurements of the gradient component in each orientation) or combined together to find the absolute magnitude of the gradient at each point and the orientation of that gradient. The gradient magnitude and the angle of orientation of the edge are the same that in Roberts's.

Canny The Canny operator was designed to be an optimal edge detector (according to particular criteria, there are other detectors around that also claim to be optimal with respect to slightly different criteria). It takes as input a gray scale image, and produces as output an image showing the positions of tracked intensity discontinuities.

The Canny operator works in a multi-stage process. First of all the image is smoothed by Gaussian convolution. Then a simple 2-D first derivative operator (somewhat like the Roberts Cross) is applied to the smoothed image to highlight regions of the image with high first spatial derivatives. Edges give rise to ridges in the gradient magnitude image. The algorithm then tracks along the top of these ridges and sets to zero all pixels that are not actually on the ridge top so as to give a thin line in the output, a process known as non-maximal suppression. The tracking process exhibits hysteresis controlled by two thresholds: T1 and T2, with T1 $>$ T2. Tracking can only begin at a point on a ridge higher than T1. Tracking then continues in both directions out from that point until the height of the ridge falls below T2. This hysteresis helps to ensure that noisy edges are not broken up into multiple edge fragments.

3.2 Image Segmentation Techniques

Segmentation is the process of dividing an image into regions, each susceptible to containing a simple object or a group of objects of the same type. In order to segment elements of an image, several techniques are commonly used: projection histograms, connected component labelling, Run Length Smearing (also very useful for filling gaps in the image), the Hough Transform. It must be said that morphological operations (described in last section) can also be used in the segmentation stage of a OMR system. Most of these image segmentation techniques are extensively described in [39] and [40].

3.2.1 Projection Histograms

Projection Histograms are commonly used in OCR systems for segmenting characters, words and text lines, or to detect if an input image of a scanned text page is rotated.

In a horizontal projection, y(x_i) is the number of printed pixels in that column, whereas in a vertical projection, x(y_i) is the number of printed pixels in that row. In Figure 3.5, we can see horizontal and vertical projection histograms of number 5.

Figure 3.5: Horizontal and vertical projection histograms

This method is very fast, and accumulative histograms can be very useful in printed characters. It can be invariant to scale using a fixed number of bins on each axis and dividing by the total number of printed pixels in the character image. It is important to remark that whereas the vertical projection x(y) is slant invariant, the horizontal projection is not. In addition, the projection histograms are very sensitive to rotation, variability in writing style, and important information about the character shape seems to be lost.

3.2.2 Connected Component Labelling

Connected Component Labelling assigns to each connected component of the binary image a distinct label. The labels are usually natural numbers starting from one to the total number to connected components in the input image. The algorithm scans the image from left-to-right and top-to-bottom. On the first line containing black pixels, a unique label is assigned to each contiguous run of black pixels. For each black pixel of the next and succeeding lines, the neighboring pixels on the previous line and the pixel to the left are examined. If any of these neighboring pixels has been labelled, the same label is assigned to the current black pixel; otherwise the next unused label is used. This procedure continues to the bottom line of the image. After the scanning process, the labelling is completed by unifying conflicting labels and reassigning unused labels.

3.2.3 Run Length Smearing

It consists in the detection of all runs of a certain value (for example, runs of 0's) of every line (or column) of the image. Then, those runs are converted to a certain value if its length is shorter than a

predefined threshold. Notice that it requires a prior skew correction.

For example, in a binary image, a line with 0's and 1's, the result of applying a Run Length Smearing algorithm with T=3 where 0's are converted to 1's is:

Input line..... 0 0 0 1 1 0 1 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0

Output line.... 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1

As we can see, those runs (segments) of 0's whose length is shorter than 3, are converted to 1's. It is very useful for filling gaps in the image.

The usage of this technique in segmentation, the algorithm is first applied line-by-line and then column-by-column, combined by the logical AND operation. The result could be used as input to the connected component labelling, yielding a set of regions of the image.

3.2.4 Hough Transform

The Hough Transform method is useful when the objective is to find lines or curves (such as circles or ellipses) that fit groups of individual points on the image plane. It is robust against noise and is specially useful for fitting unconnected points (it works although there may be small gaps). In addition, Hough Transform works successfully even when different objects are connected to each other. It is commonly used in the analysis of engineering drawings, where most curves are line segments or circle arcs. It can be used to find skew from text line components. Unluckily, it is costly in terms of computation.

This method involves a transformation from the image coordinate plane to parameter space, in other words, it maps each point in the original (x,y) plane to all points in the (r,theta) Hough plane of possible lines through (x,y) with slope theta and distance from origin r. In the case of searching lines, the method works as follows:

The equation of a line can be expressed as

$$r = x \cdot \cos\theta + y \cdot \sin\theta \tag{3.13}$$

where r is the distance from the origin of the (x,y) space to the line, and theta is the angle of the normal to the line. This produces a sinusoidal curve in the (theta,r) parameter space for each point (x,y).

Each (x,y) location of a foreground pixel in the image plane is mapped to the locations in the transform plane for all possible straight lines through the point (for all possible values or r ant theta). When multiple points are collinear, their transformations will intersect at the same point on the transform plane. Therefore, the (r,theta) locations having the greatest accumulation (maximum) of mapped points indicate lines with those parameters.

Figure 3.6 shows a line and its Hough Transform plane, where an accumulation maximum shows that there is a line, oriented 20 degrees, in the original image.

In [45] one can find an example of the usage of the Hough Transform and other techniques for the automatic baseline extraction in check bank images.

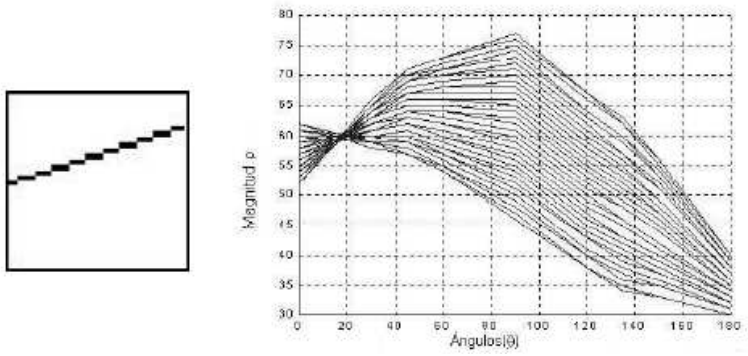


Figure 3.6: A line and its Hough Transform plane

3.3 Statistical Classification

A subject of study in pattern recognition is feature extraction, because it helps in the classification of characters and symbols into classes. Feature extraction is divided two approaches: statistical and structural. In this section, we will expose some descriptors used in statistical classification: geometrical features, moments, Zernike moments and Zoning. These techniques can be found in [46].

3.3.1 Geometrical Features

Simple geometrical features can be very effective in many applications. For instance, the sizes in the x- and y-directions of a connected component may be sufficient to distinguish characters from graphical parts. Geometrical features are:

- Sizes in x and y direction, and their aspect ratio.
- Perimeter
- Area
- Maximum and minimum distances from the boundary to the center of mass
- Number of holes
- Euler number: number of connected components minus number of holes.
- Compactness:

$$\frac{Perimeter^2}{4 \cdot \pi \cdot Area} \tag{3.14}$$

- Signatures: horizontal and vertical projections of black pixels.

In document image analysis, these features are commonly used for the pre-classification of objects into broad categories, such as characters and graphics.

3.3.2 Geometrical Moments

Objects can also be described by their moments, defined by

$$M_{p,q} = \int \int_D i(x,y) \cdot x^p \cdot y^q dxdy \tag{3.15}$$

where p and q are the non-negative integer degrees of the moment, i(x,y) is the gray-level image of the object, and D is the spatial extend of the object. If the image is binary, and takes on the value 1 for black and 0 for white pixels, then:

$$\begin{cases} M_{0,0} = Area \\ M_{1,0} = \text{x-coordinates of the center mass.} \\ M_{0,1} = \text{y-coordinates of the center mass.} \end{cases} \tag{3.16}$$

For large values of p and q, he moments become very sensitive to noise and in practice, only small values of p and q are used.

3.3.3 Zernike Moments

Because geometrical moments are not orthogonal, polar coordinates are used to manage orthogonality and invariance to rotations. Zernike moments are defined over a set of complex polynomials which forms a complete orthogonal set over the unit disk:

$$x^2 + y^2 \leq 1 \tag{3.17}$$

Polynomials of Zernike are denoted by:

$$ZP = \{V_{nm}(x,y) | x^2 + y^2 \leq 1\} \tag{3.18}$$

The form of the Zernike polynomial basis of order n an repetition m is:

$$V_{nm}(x,y) = R_{nm}(x,y) \exp(jm \arctan(y/x)) \tag{3.19}$$

with the condition

$$n \in N^+, m \in N | (n - |m|) even, and |m| \leq n \tag{3.20}$$

The radial polynomial is defined as

$$R_{nm}(x,y) = \sum_{s=0}^{(n-|m|)/2} (-1)^s \frac{(n-s)!}{s!(\frac{n+|m|}{2} - s)! \cdot (\frac{n-|m|}{2} - s)!} \cdot (x^2 + y^2)^{(n-2s)/2} \tag{3.21}$$

where:

$$\begin{cases} r = \sqrt{x^2 + y^2}, \text{length of the vector from the origin to the pixel (x,y)} \\ \theta = \arctan(y/x), \text{the angle between that x-axis and that vector} \end{cases} \tag{3.22}$$

and

$$R_{nm}(r) = \begin{cases} \text{the polar coordinate representation of } R_{nm}(x,y), \\ \text{a polynomial of degree n in r containing terms in } r^n, r^{n-2}, ..., r^{|m|}. \\ \text{contains no power of r less than |m|} \end{cases} \tag{3.23}$$

Then, an image can be expanded using the Zernike polynomial:

$$f(x,y) = \sum_n \sum_m A_{nm} \cdot V_{nm}(r,\theta) \tag{3.24}$$

where the Zernike moments of order n with repetition m is a complex number given by:

$$A_{nm} = \frac{n+1}{\pi} \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f(x,y)[V_{nm}(x,y)]^* \tag{3.25}$$

with * denoting the complex conjugated, and

$$x^2 + y^2 \leq 1 \tag{3.26}$$

The orthogonality property of Zernike polynomials over the unit disk allows the reconstruction of the original image to an approximate image from a finite number of Zernike moment orders. The existence of such reconstruction is important even when it is not intended to be used explicitly in the recognition process. It provides a measure of the representation power of the utilized features, and suggests non-arbitrary matching in the invariant feature space. The reconstructed image can be computed from:

$$f(x,y) = \sum_{n=0}^{n_{max}} \sum_m A_{nm} \cdot V_{nm}(x,y) \tag{3.27}$$

with

$$m \in N | (n - |m|) \text{ even}, and |m| \leq n \tag{3.28}$$

Figure 3.7 shows an image reconstruction of an object using Zernike moments. Low order moments encode information about the global shape of an object. The higher the order moments, the finer the details captured by the reconstruction, such as edges and corners. Although higher order moments are more sensitive to noise (due to the increasingly oscillatory nature of the Zernike polynomials with increasing order n) their use is generally desirable in object recognition as they capture the fine details of the image, and may lead to better discrimination among the different classes of objects. For those

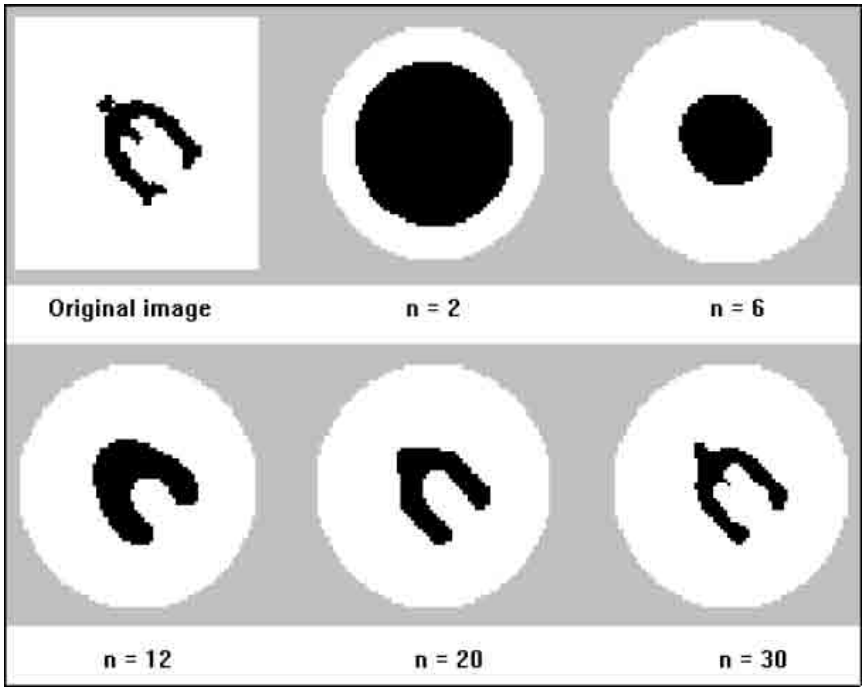


Figure 3.7: Image reconstruction using Zernike moments

reasons, there must be an optimal maximum number of moments which best characterize an image in any given application. Notice that while one might need a large number of moments to achieve an acceptable image reconstruction, a lower number of moments may be enough for the task of object recognition.

3.3.4 Point distribution descriptors: Zoning

Zoning can be used to extract data from specific areas on an image. The method (see [47]) computes the percentage of black pixels in each zone: an m x n grid is superimposed on the character image, and for each of the nxm zones, the average gray level is computed, giving a feature vector of length nxm.

This method is fast and very useful in the recognition of printed characters. Figure 3.8 shows a number 5 and its zoning-matrix.

Radtke exposes in [48] the use of multi objective evolutionary algorithms (MOEAs) applied to the zoning technique for handwritten recognition. A MOEA is a search method based on Darwin's evolutionary theory applied to a population of possible solutions. The algorithm proposed selects suitable zoning strategies (without the requirement of domain expert feedback during the process) using a MOEA, based in genetic algorithms.

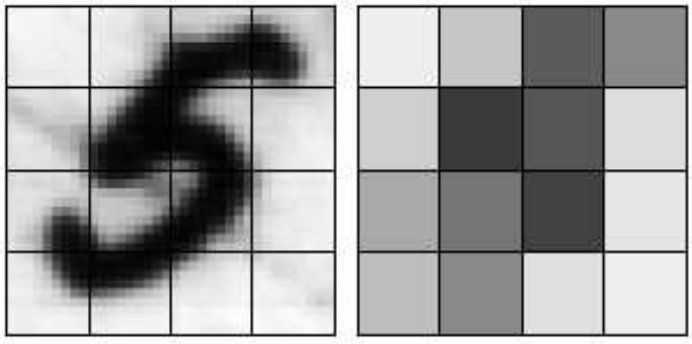


Figure 3.8: Zoning

3.4 Graph Grammars and Graph Matching

3.4.1 Graph Grammars

Graph grammars provide a useful formalism for describing structural manipulations of multi-dimensional data. A full review of graph rewriting can be found in [49]. In the OMR field, a problem to solve is how represent context information associated to music notation. Several authors, such as Fahmy and Baunmann (see [14], [50] and [51]), propose the usage of attributed graph grammars to capture the notational conventions of music in an intellectually manageable way.

In a graph, nodes represent objects (primitives) and edges represent relationships among them (such as *is right of, is below*). Auxiliary information can be expressed by adding attributes to nodes or edges. Grammatical manipulation of graphs permits convenient processing of multi-dimensional pattern classes, nevertheless, picture processing applications of graph grammars remain surprisingly rare. This is due to a variety of factors, including the difficulty of constructing large grammars that are intellectually manageable, and the absence of methods for processing noisy and uncertain data.

These problems could be remedied by the development of readable graph-grammar notations, the provision of graph-grammar construction and debugging tools, and the emergence of techniques for constructing robust graph-grammars.

A graph grammar is specified by a start graph and a set of production rules. The role of the production is to replace one subgraph by another. This process depends on a specification of the desired embedding: edges to/from the old subgraph must be transformed into edges to/from the new subgraph. Figure 3.9 shows an example of an embedding production rule applied to a note symbol with two stems: The embedding specifies that an edge incident on node 1 of the first graph is transformed into two edges, incident on nodes 1' and 4' of the second graph. An edge incident on node 2 is transformed into an edge incident on node 2'; similarly a node-3 edge is transformed into a node-3' edge. This production duplicates doubly-stemmed noteheads to allow subsequent processing as two independent notes.

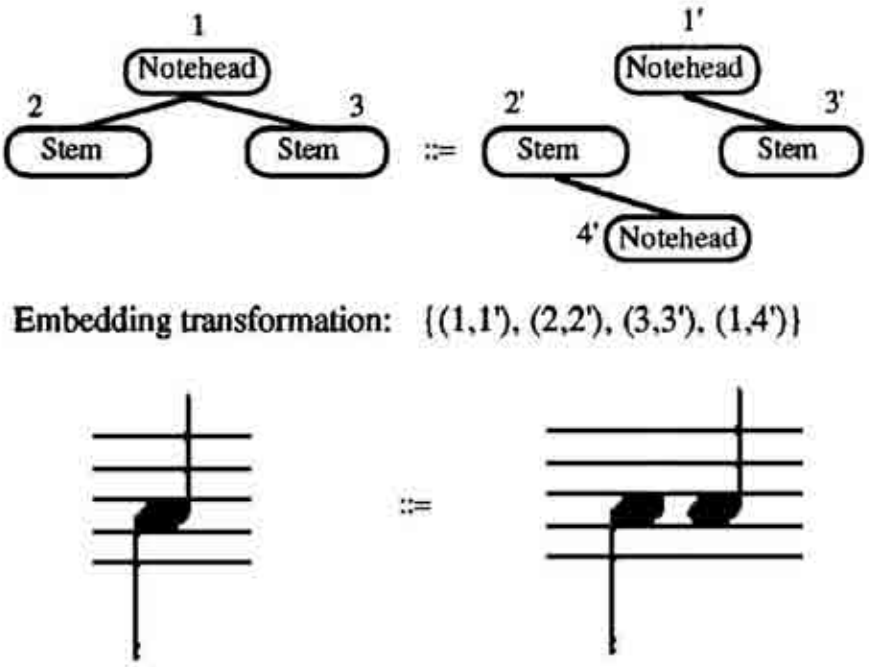


Figure 3.9: An example of embedding rule applied to a note symbol with two stems

3.4.2 Graph Matching

If graphs are used to represent objects of a particular domain, then graph matching will be a suitable strategy for classification, detection or comparison of such structures. In general, graphs may be matched by comparing vertices and edges according to their distribution to a relational distance metric.

Following a mathematical formulation, a matching between two graphs is performed by a graph isomorphism, in other words, a bijective mapping that associates the nodes of the first graph to the nodes of the second graph. The mapping must also preserve the structure so that any pair of adjacent nodes in the first graph are also adjacent in the second one under the mapping. For attributed graphs, it is also required that both node and edge attributes be maintained by the mapping.

Thus, the similarity between attributes must be considered when defining the relational distance metric and, generally, it is application-dependent. If one of the graphs involved in the matching is larger than the other, in terms of the number of nodes, then the matching is performed by a subgraph isomorphism. For example, a subgraph isomorphism from G1 to G2 means finding a subgraph G3 of G2 such that G1 and G3 are isomorphic.

Due to the computational complexity of graph matching, some tractable graph-matching algorithms are developed, using heuristics to cut down the computational effort to a manageable size. In [52] and [53] some of these algorithms are reviewed: optimal algorithms (such as backtrack tree search, forward checking and discrete relaxation), error-tolerant algorithms (branch and bound, random graphs),

approximate algorithms (using probabilistic relaxation), algebraic algorithms (using weight graphs) and indexed search.

Chapter 4

Optical Music Recognition: Our approach

In this chapter, our approach to the recognition of handwritten scores is presented. First, our preliminary work with modern handwritten scores is described. Secondly, limitations of this system when working with old handwritten ones are commented, and new techniques are used to cope with the difficulties in the recognition of old documents.

4.1 Modern Handwritten Scores

Initially, we have been working with modern handwritten musical scores, where paper is in good condition, there is a standard of musical notation and most of staff lines are printed. Here, the early stages (see Figure 4.1) of the OMR system proposed consists in the following: preprocessing (the input image is binarized and deskewed), staff removal and detection of graphical primitives (such as vertical lines and head notes).

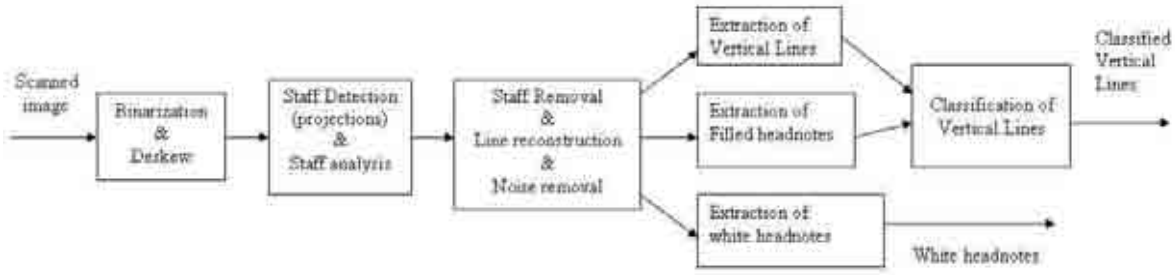


Figure 4.1: First stages of the OMR system for modern handwritten scores.

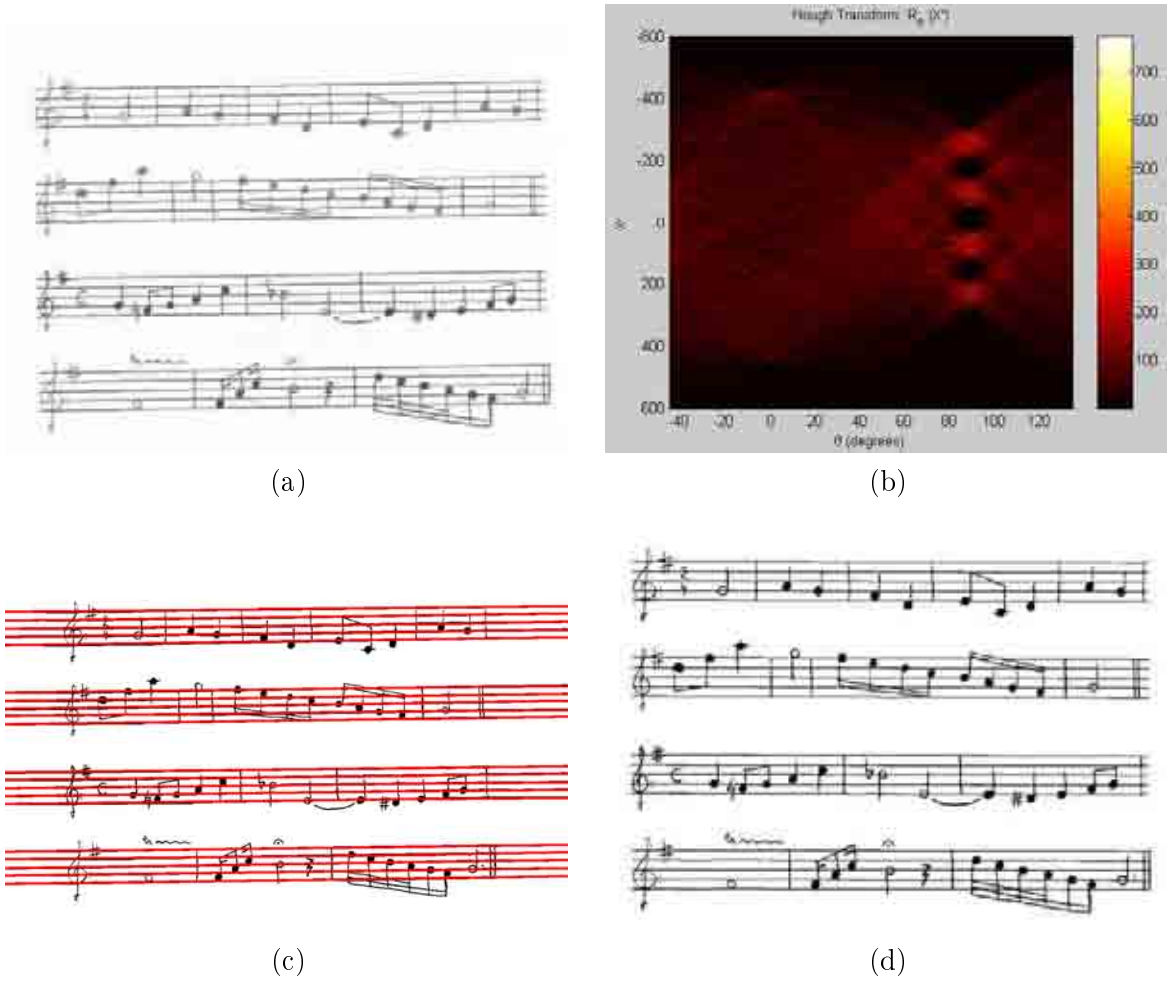


Figure 4.2: (a) Original Image (b) Hough Transform space: yellow points show lines (c) Detected lines (d) Deskewed image

4.1.1 Preprocessing

Binarization First of all, the gray-level scanned image (at a minimum resolution of 300 dpi) must be binarized to separate foreground from background and make the recognition process easier. Thanks to the fact that scores are modern and the music sheet is in good condition (paper sheet is relatively new), the input gray-level image can be effectively binarized with a global binarization method, such as OTSU's method (see Chapter 3 for details): It obtains only one threshold and uses this value to separate all pixels in the image into two categories (pixels whose grey-level is under the threshold and pixels whose value is over this threshold). After that, the image is negated so that white pixels (value=1) will contain information (foreground) and black pixels (value=0) will be the background. It must be said that in this report, **figures are drawn in a non-negated way** in order to make their visualization easier.

Deskewing The second step consists in deskewing the image, so that staff lines would be horizontal and make their recognition easier. The Hough Transform method can be used to detect lines, so if this technique is applied to the image, several dots (corresponding to staff lines) will show the orientation of the staff, and consequently, the orientation of the music sheet. If the orientation of these lines is different from 90 degrees (which corresponds to horizontal lines in the Hough Transform space, see equation 4.1), the rotation angle is calculated and the image is rotated.

$$r = x \cdot \cos\theta + y \cdot \sin\theta \tag{4.1}$$

Figure 4.2(a) shows a skewed image, and how in the Hough Transform space (Fig. 4.2(b)) the staff lines and their orientation are obtained. In Fig. 4.2(c) the staff lines are drawn in red color, applying the equation 4.2 to convert these points in the Hough Transform into the cartesian space. The deskewed image is shown in Fig. 4.2(d).

$$y = \frac{r - x \cdot \cos\theta}{\sin\theta} \tag{4.2}$$

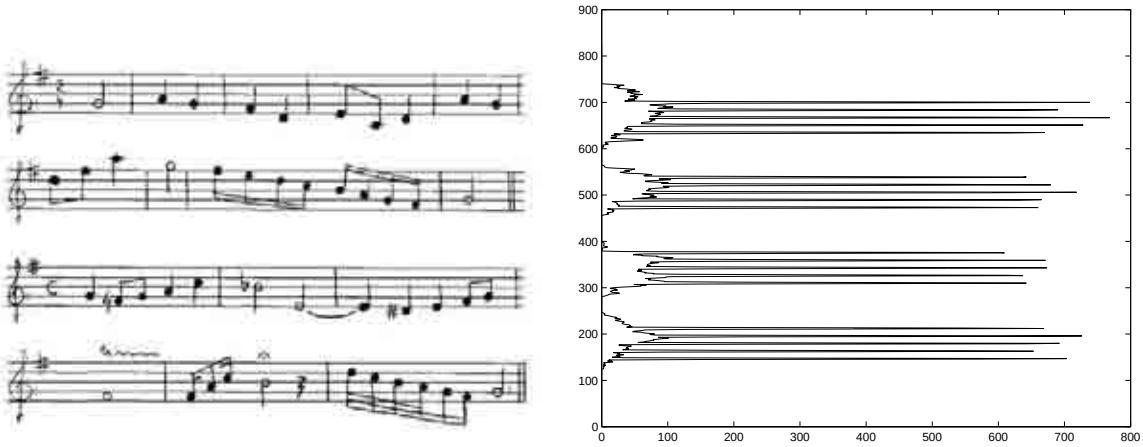
4.1.2 Staff detection and removal

As it has been exposed in Chapter 3, staff lines play a central role in music notation because they define the vertical coordinate system for pitches, provide horizontal direction for the temporal coordinate system and give a size normalization useful for symbol recognition (size of musical symbols is linearly related to the staff space). Unluckily, staff causes distortions in musical symbols (connecting objects that should be isolated), making difficult the recognition process. For that reason, staff removal must be performed in order to isolate musical symbols.

Once the image is deskewed, horizontal projections can effectively be used to detect staff lines: every row whose value is over a certain threshold (depending of the size of the musical score) is likely to contain a staff line (see Fig. 4.3(b)).

In the staff analysis, hypothetical rows detected are grouped (a staff is composed by 5 staff lines), the width of staff lines is set as the average of widths of every line, and the distance between staff lines is set as the average of the distances between every line in every 5-grouped lines. Knowing these parameters, a run-length smearing process deletes staff lines (Figure 4.4 shows the score after the staff removal process).

It must be said that this process can provoke broken objects: when a musical symbol crosses a staff line, the removal of these line can erode also a part of this musical symbol. For that reason, the staff removal process must try to keep unbroken symbols: a pixel of the staff line will be deleted if pixels connected to it in the previous row are zero (see Fig. 4.5(a)). In addition, a module that joins segments of lines is implemented to reconstruct hypothetical broken lines: a slide-window follows the staff line analyzing its area. Wherever the are of the slide-window is over a certain threshold (it means that there is a symbol), the Hough Transform is performed locally in it. The maximum in the Hought Transform



(a) (b)

Figure 4.3: (a) Deskewed Image; (b) Horizontal Projections.

space is found and the line is reconstructed. This technique can join some segments of broken lines, but it also paints pixels that should not be painted (see Fig. 4.5(b)). In order to deal with this problem, pixels painted in the reconstruction module only will change to 1 if in the original image they were 1:

$$\text{Output Image} = \text{ Original Image } \cap \text{ Reconstructed Image} \tag{4.3}$$

Finally, morphological closing and run length smearing are performed to reduce noise.

4.1.3 Detection of Graphical Primitives

In the primitive detection stage, vertical lines and head notes are the first graphical primitives to recognize (they form notes, which are the most important musical symbols). After that, the remaining image can be processed to obtain other graphic primitives.

Detection of Vertical Lines

Detection of vertical lines in printed scores can be easily implemented using projections (because lines are perfectly verticals), whereas their recognition in handwritten ones requires the use of other techniques, because lines are rarely perfectly vertical. First, the input image (without staff lines) is prepared for the recognition of vertical lines:

- Making verticals thicker: A vertical Run Length Smearing process is used to fill gaps (whose length is lower than the width of staff lines) of background color.
- Removing horizontal lines and hypothetical headnotes: In order to remove vertical-short segments (whose lenght is lower than the distance between staff lines), a vertical Run Length Smearing

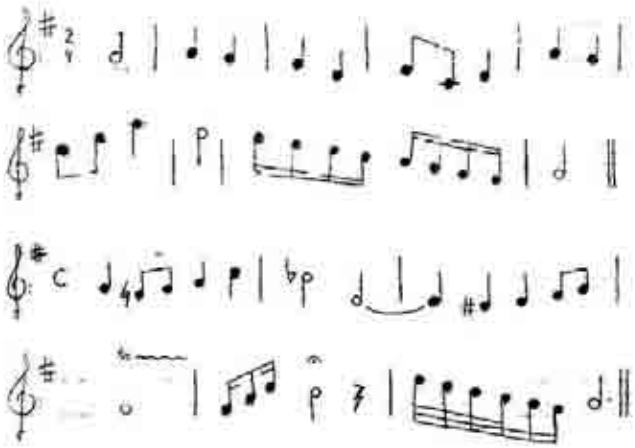
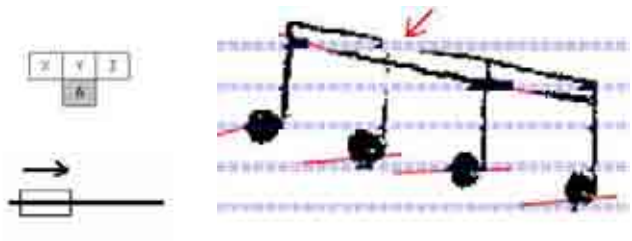


Figure 4.4: Image without staff lines.



(a) (b)

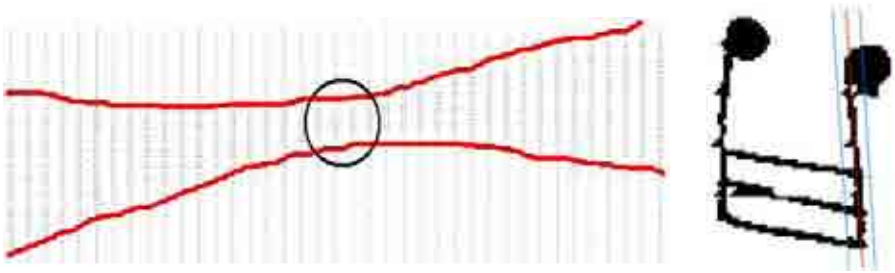
Figure 4.5: (a) In the slide-window, a pixel A with value of 1 will be removed if pixels X,Y,Z are 0; (b) The reconstruction of hypothetical lines: Staff lines are drawn in blue color, reconstructed segments in red color, the arrow shows that some lines with gaps are not reconstructed

process is used for filling gaps in foreground color.

The prepared image will be the input for the following method:

1. Apply Hough Transform method, allowing a skew of 20ž (from -20 degrees to +20 degrees).
2. In the Hough Transform space, find values under a certain threshold and change their values to zero. Then, find shapes similar to a *bow tie* (see Fig. 4.6(a)) and detect their center of mass ($\overline{x}$,$\overline{y}$), which corresponds to the line found:

$$GeometricalMoments = m_{p,q} = \sum_{x=1}^{N}\sum_{y=1}^{M} f(x,y)\cdot x^p \cdot y^q \tag{4.4}$$



(a) (b)

Figure 4.6: (a) In the Hough Transform space, if (values < threshold) are changed to zero, then shapes similar to a bow tie show a line. The center of mass shows the line found; (b) Pixels between blue lines are analyzed to know which ones belong to the red line

$$\overline{x} = \frac{m_{1,0}}{m_{0,0}} = \frac{\sum_{x=1}^{N}\sum_{y=1}^{M} f(x,y)\cdot x}{area} \tag{4.5}$$

$$\overline{y} = \frac{m_{0,1}}{m_{0,0}} = \frac{\sum_{x=1}^{N}\sum_{y=1}^{M} f(x,y)\cdot y}{area} \tag{4.6}$$

3. Draw the line in the cartesian space. Then, a region parallel to the line drawn (see Fig. 4.6(b)) is analyzed in order to decide which painted pixels belong to this line L. Two algorithms have been tested:

   - Measure the distance of the line L to the line that forms every pair of pixels: Every pair of pixels ((x_1,y_1) and (x_2,y_2)) inside the region is used to form a new line N. Then, the equation in polar coordinate system (see eq. 4.1 and eq. 4.7) of this line and its value in the Hough Transform space is obtained. If the distance between both lines is under a certain threshold, then both points belong to the line L.

$$\begin{cases} \theta = \arctan(\frac{x_1-x_2}{y_1-y_2}), & \text{if } y_1 \neq y_2; \\ \theta = 0, & \text{if } y_1 = y_2; \end{cases} \tag{4.7}$$

   - Measure the distance of every pixel to the line L: The distance of every point of the region to the line is obtained and only those points near the line are marked as pixels belonging to the line L:

$$\text{if } |r - x\cdot\cos\theta - y\cdot\sin\theta| < threshold \Rightarrow (x,y) \in L \tag{4.8}$$

   Both options get similar results, so the first option (distance of the line L to the line that forms two points) has been discarded due to its high computational cost.

Figure 4.7 shows vertical lines detected in the musical score.

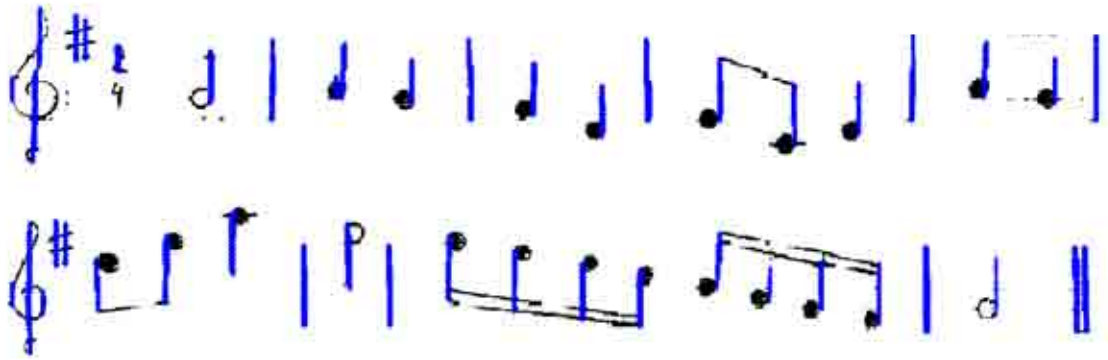


Figure 4.7: An example of the detection of vertical lines: verticals in blue color.

Detection of Filled Headnotes

For the detection of filled headnotes, a first attempt was the use of an edge detection algorithm to convert those filled headnotes in circumferences, so the detection of filled headnotes could be changed into the detection of circles. Two options have been contemplated for detecting circles:

- **Hough Transform** for detecting circles: As we know, The Hough Transform algorithm can be used for detecting lines and curves, including circles.
- **Fast Radial Symmetry**: This method (described in [54]) uses local radial symmetry to extract points of interest within the scene using gradients.

Unluckily, both options fail in the detection of such circles, because headnotes are not bigger enough for being effectively detected.

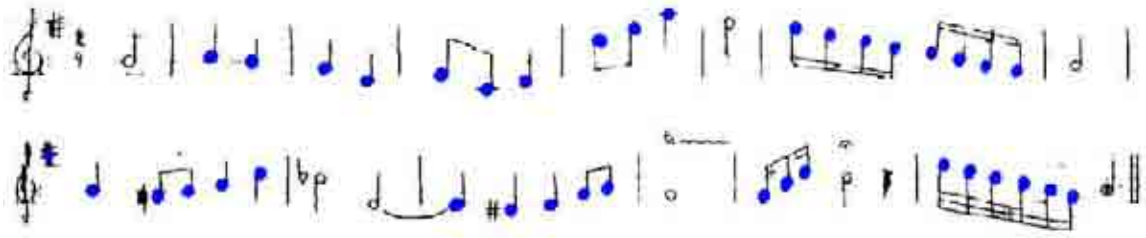


Figure 4.8: An example of the detection of headnotes: hypothetical headnotes in blue color.

For that reason, context information is used to help in their recognition: a filled headnote has always a beam. Thus, a headnote is a filled circle that has a beam in their extreme.

The strategy followed consists in obtaining hypothetical filled circles (see Fig. 4.8) and then, look if they have beams in their extremes. Hypothetical filled circles are obtained performing a morphological opening with a disk (*radius* = *w/3*, where w is the distance between staff lines). Due to the fact that this operation finds all objects bigger than the structuring element (including real headnotes and others), a module must verify if the shape of hypothetical filled circles is similar to the shape of a circle, accomplishing these rules:

- Circularity: a filled circle must have its circularity parameter near to 1 (a perfect circle has circularity = 1):

$$Circularity = \frac{perimeter^2}{4\pi \cdot area} \simeq 1 \tag{4.9}$$

- Area: The area of the filled circle (radius = r) must be between certain thresholds (a perfect headnote should has an area of $\pi \cdot w^2$, where w is the width of staff line):

$$\pi \cdot (0'5 \cdot w)^2 \leq \pi \cdot r^2 \leq \pi \cdot (1'5 \cdot w)^2 \tag{4.10}$$

- Compactness (no concavities): A filled circle has no concavities, so, the area of the convex covering of the figure, must be similar to the area of the filled circle:

$$Compactness = \frac{Area_{\text{ Convex Covering}}}{Area_{\text{ Filled Circle}}} \simeq 1 \tag{4.11}$$

  An easier way to verify compactness is performing a morphological closing and compare with the area of the object:

$$Compactness = \frac{Area_{\text{ Closing}}}{Area_{\text{ Filled Circle}}} \simeq 1 \tag{4.12}$$

- One connected component: The filled circle must not be broken. If it has several parts, then, circularity and compactness are useless.

Classification of Vertical Lines

Once we have detected vertical lines and filled head notes, lines must be classified, see Fig. 4.9, in beams (which have one or more head notes), bar lines (longer than beams, without head notes) and others (e.g. lines that are part of another kind of symbols, such as flats, sharps or naturals). Bar lines are the most important vertical lines, because they divide scores in bar units. Once we have isolated every bar unit, we can process them in an independent way, looking for musical symbols using grammar rules.

The input for the classification stage of vertical lines, is the matrix containing all vertical lines detected and the image with hypothetical headnotes. Then, every vertical line V is classified in:

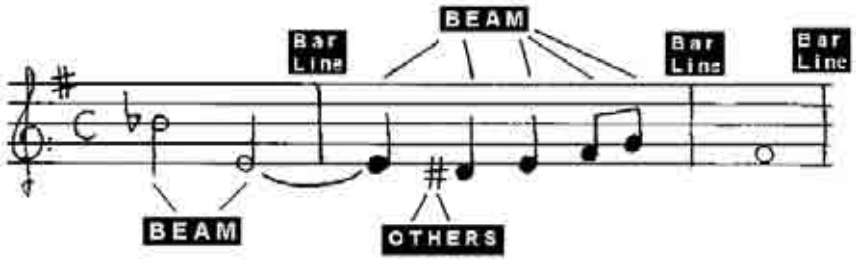


Figure 4.9: Verticals in scores: Beams have a headnote in the top-right or in the bottom-left. Bar lines cover the staff

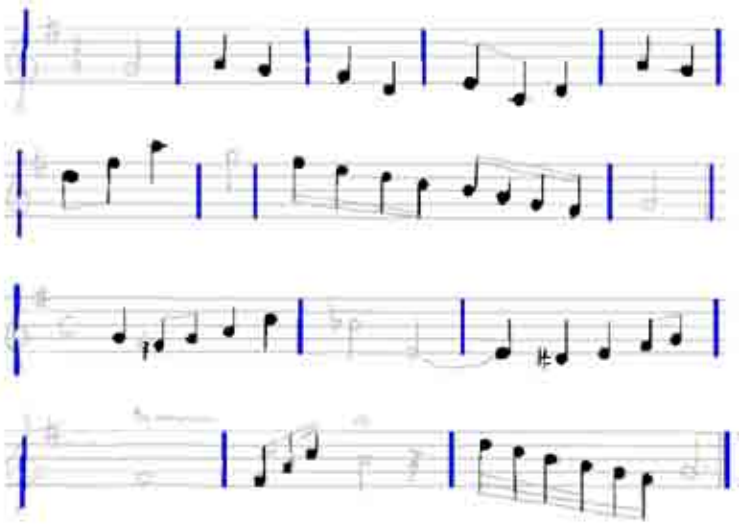
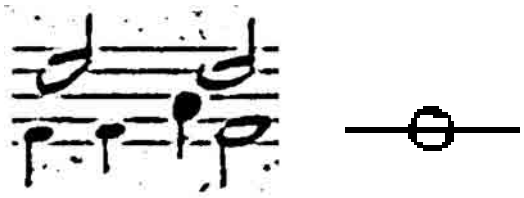


Figure 4.10: Classification of Vertical Lines: Notes with filled headnotes in black color, bar lines in blue color

1. **Beam**: If V has a headnote in its extremes (in the top-right or in the bottom-left) and the length of V is over the double of the distance between staff lines, then V is a beam. The hypothesis here is that the length of beam is usually three of four times the distance between staff lines.
2. **Bar line**: If V is not a beam, and it covers the staff (its length is over four times the width of staff lines), then V is a bar line.
3. **Others**: If V is not a beam nor a bar line, then V is a part of a musical symbol.

In Fig. 4.10 we can see the classification of vertical lines and the detection of notes with fill headnotes: head notes and vertical lines are in black color, and bar lines are drawn in blue color. The remaining score is in grey color.



(a) (b)

Figure 4.11: (a) Half notes: The circumference of two half notes in the top are incomplete because they have gaps; (b) A circle crossing a staff line that has not been completely removed

Detection of white headnotes

Extraction of whole and half notes is reduced to find circumferences in the score, but apart from the fact that circles are very little, the module must cope with:

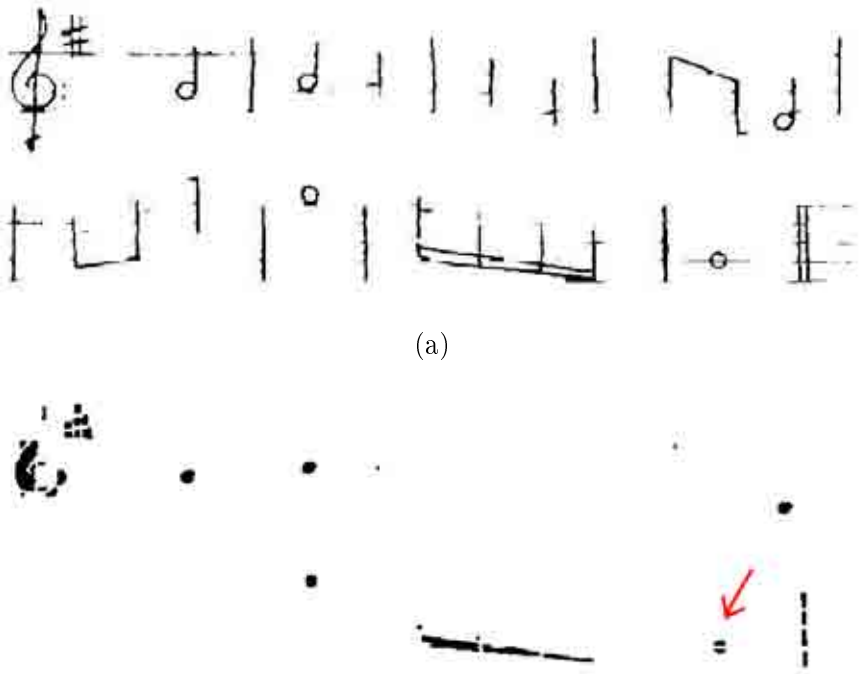
- Gaps: Circles are handwritten, so the circumference frequently is incomplete or has gaps in its contour (see Fig. 4.11(a)).
- Two semicircles: When a circle is drawn over a staff line (crossing it), the staff removal process sometimes does not remove completely the line inside the circumference (see Fig. 4.11(b)). Thus, the detection of circles becomes a problem of detecting two semicircles.

Due to the difficulties commented, a possible solution is to fill these circumferences and recognize filled circles using the module that recognizes filled headnotes:

- Prepare the input image: The image I without filled headnotes (see Fig. 4.12(a)) is prepared to obtain hypothetical white headnotes performing a morphological closing with a structuring element circular, and then removing foreground pixels of the input image I to delete verticals and contour of symbols (see Fig. 4.12(b)).
- Recognize filled circles: As it has been commented in the recognition of filled headnotes, parameters of circularity, area and compactness will be the key for determining which objects are filled circles.

Fig. 4.13 shows detected white headnotes (in blue color). As we can see, not only there are a lot of false positives, but also there is one half note missing (this half note is formed by two semicircles). For that reason, context information will be required here:

- Half note: A half note has a beam with a white headnote. So, if it has a beam in its extremes, then it will be a half note.
- Whole note: A whole note does not have a beam, so the only way to verify if it is a whole note is to count the number of beats in the measure. For example, in a score with the time signature of 4/4 (it means that every measure will have notes whose sum is 4 times), whether there is only one note between two bar lines, then this note should be a whole note (because a whole note has a duration of 4 times).



(a)

(b)

Figure 4.12: (a) Image without filled headnotes ; (b) Input image of the module: Hypothetical white headnotes are filled. The red arrow shows that a circle has been converted into two semicircles

Due to the need of a grammar for the recognition of whole notes, this part will be treated in future stages of the recognition process.

4.2 Old Handwritten Scores

As it has been said in Chapter 1, a growing interest in the Document Analysis area is the recognition of ancient manuscripts and their conversion to digital libraries. Nowadays, our work is focused on the recognition of old handwritten scores (XVIII-XX centuries) so that these scores of unknown composers could be edited and published (contributing to the diffusion and preservation of artistic and cultural heritage).

Working with old scores makes the recognition task more difficult due to:

- Paper degradation: most musical scores are in poor condition, so there is an important problem of low level processing, because it must cope with transparencies, spots, stains and low contrast (examples shown in Fig. 4.14).

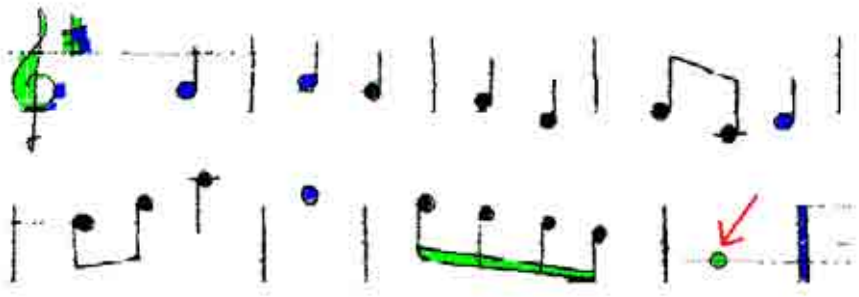
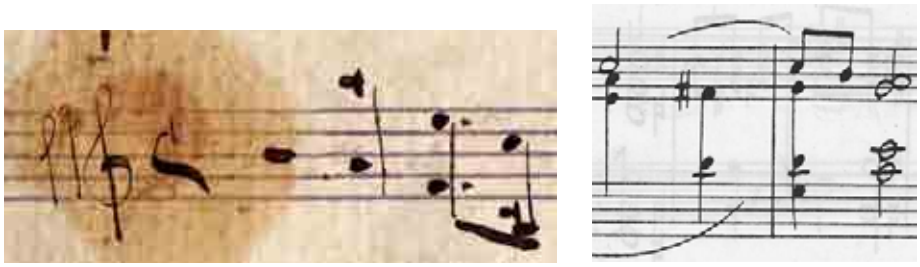


Figure 4.13: Detection of white headnotes: candidates in green color, detected white headnotes in blue color. The red arrow shows that a white headnote (formed by two semicircles) has not been detected.

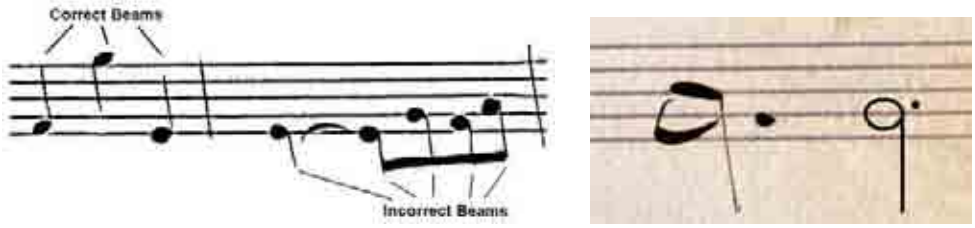


(a) (b)

Figure 4.14: (a) Image with stains. ; (b) Image with transparencies

- The lack of a standard notation in last centuries: there is not a standard in the shape of musical symbols, neither in the graphical relation between primitives. An example of this fact is shown in Fig. 4.15(a), where several notes are drawn with the headnote in the top-left side of the beam, instead of being in the top-right side. The writer style is also important: in Fig. 4.15(b) both notes are the same, but they look different. For those reasons, the creation of a graph grammar becomes more difficult.
- Staff lines are often handwritten (lines with gaps and rarely straight), so, their recognition and removal is hard to achieve.
- Broken and touching symbols as well as high density of symbols make the recognition of musical symbols more difficult.

For those reasons, the preprocessing and segmentation phases described for the recognition of modern scores, must be modified and adapted to this kind of scores: A good solution to paper degradation is the use of local binarization techniques, filtering and morphological operations, whereas staff detection can be performed using a contour tracking process. In order to cope with the lack of standard notation, an expert system will be required to learn every new way of writing, and artificial intelligence based



(a) (b)

Figure 4.15: (a) No Standard in musical notation: some beams have their headnote in the incorrect side; (b) writer style: both notes are the same, but they look different.

techniques will take advantage of higher level musical information.

In the following sections, the method proposed to detect and extract staff lines and graphical primitives is exposed. Figure 4.16 shows the steps followed in the early stages of the system.

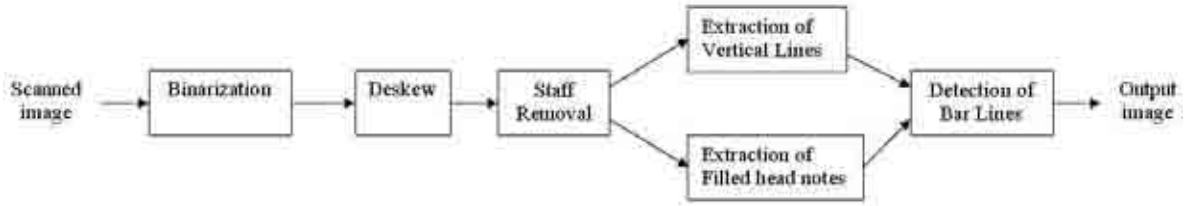
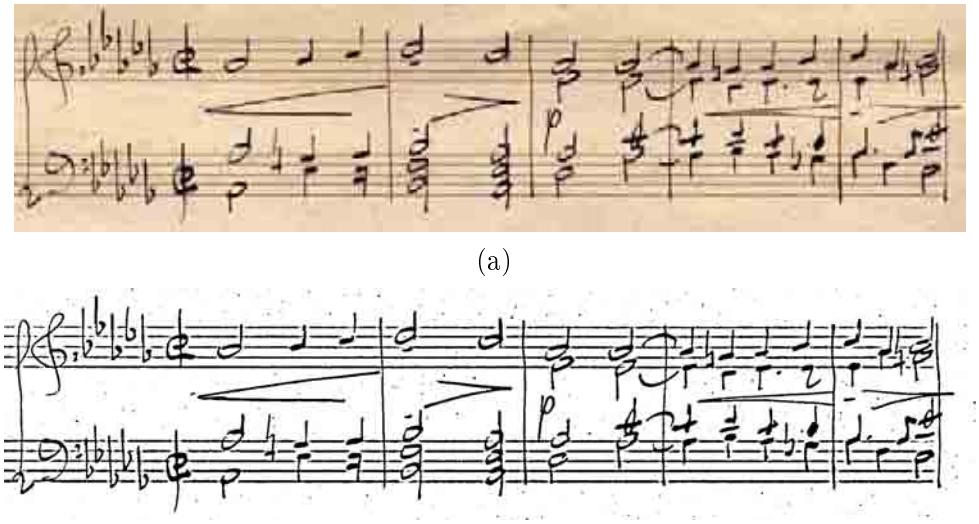


Figure 4.16: Preprocessing Stages of the system.

4.2.1 Preprocessing

As in the recognition of modern scores, the gray-level scanned image (at a minimum resolution of 300 dpi) must be binarized to separate foreground from background, but with such these old scores, global binarization techniques do not work because of degradation of the scores. Thus, adaptive binarization techniques are required, such as Niblack binarization method (see Chapter 3 for details): the threshold used to classify pixels in black or white varies over the image, based on the local mean and local standard deviation of the neighborhood of every pixel (the size of the neighborhood rectangle has been experimentally set to 21x21 pixels). Then, filtering and morphological operations are used to reduce noise.

After binarizing the image (see Fig. 4.17), the image must be deskewed in order to make the recognition of staff lines easier. As in the OMR for modern scores, the the Hough Transform method is used to detect lines, obtaining the orientation of the music sheet, and rotating the image if necessary.



(a) (b)

Figure 4.17: (a) Original Image; (b) Binarized image using Niblack's method

4.2.2 Staff detection and removal

The detection of staff lines is more difficult due to distortions in staff (lines present often gaps in between), and contrary to modern scores, staff lines are rarely perfectly horizontal. This is caused by the degradation of old paper, the warping effect and the inherent distortion of handwritten strokes (staff lines are often drawn by hand, see Fig. 4.18). For those reasons, a more sophisticated process is followed (see Fig. 4.19): After analyzing the histogram with horizontal projections of the image, detecting staff lines, a rough approximation of every staff line is performed using skeletons and median filters. Afterwards, a contour tracking algorithm is performed to follow every staff line and remove segments that do not belong to a musical symbol.

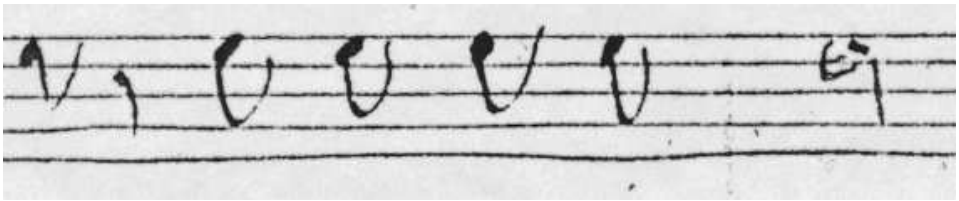


Figure 4.18: Staff written by hand.

Detection of five grouped lines

Due to the fact that there are deviations in the staff, the detection of staff lines can not be done using horizontal projections (see Fig. 4.20(a)), because sometimes, local maximums do not correspond

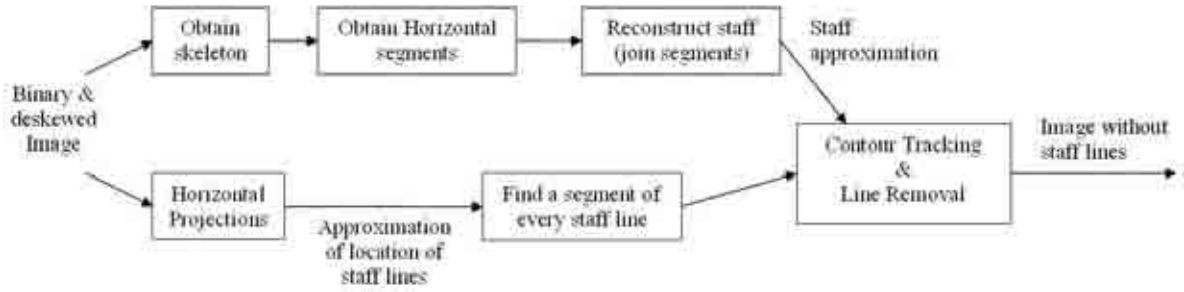


Figure 4.19: Stages of the extraction of staff lines.

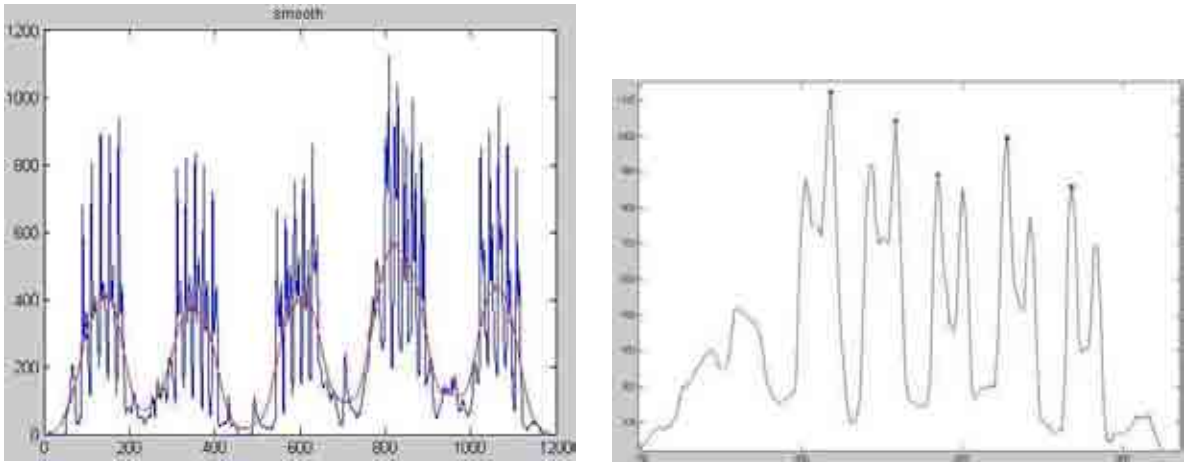
to staff lines (e.g. two local maximums correspond to one staff line, see Fig. 4.20(b)).

Thus, the solution proposed consists in the following steps:

1. Perform a horizontal projection (obtaining an histogram) of the entire score.
2. Smooth the histogram until there is only one oscillation (hill), with only one maximum, for every staff (see the red line in Fig. 4.20(a)).
3. For every oscillation, determine which ones correspond to staffs (a staff has five peaks, corresponding to the five staff lines):
   - If the maximum of a hill is too low, then it is not a staff.
   - Smooth the segment of the histogram (SH) corresponding to this staff until there are only five hills, corresponding to the five staff lines.
   - If there are not five hills, then, it is not a staff.
   - If there are five hills but the distance between them is not constant, then it is not a staff (a staff has five equidistant staff lines).
4. For every staff detected:
   - Obtain five maximums and six minimums in the smoothing histogram (SH) corresponding to this staff.
   - Get the maximum M of the histogram between every two minimums of the smoothing image. Also, this maximum M must be near every peak of the hill. Every maximum M correspond to a staff line (see the red dots in Fig. 4.20(b)).

Determining pixels belonging to staff lines

Once we have a rough approximation of the location of every staff line, pixels belonging to every staff line must be determined. Some failed attempts to solve this problem were:



(a) (b)

Figure 4.20: (a) Histogram of the Horizontal Projection of the musical score: the red line corresponds to the smoothing process of the histogram; (b)A segment of the histogram: There are several local maximums corresponding to a staff line, and the red dot corresponds to the staff line.

- Evaluate a window containing the staff line, and mark those pixels that are near the staff line. But this method does not work when the staff has deviations in its lines.
- Use the **structural tensor** to detect orientations of lines, and then, follow those segments that belong to the same orientation. This method (described in [55]) performs smoothing and partial derivatives to obtain vectors and orientation of lines. Unluckily, these method does not work when the staff has high density of musical symbols, because in such cases, orientation of horizontal segments results to be very similar to the orientation of symbols.
- Calculate gradients of the image in order to obtain horizontal segments. Then, obtain a **polynomial function** for the curve that goes through those segments (notice that the function is a curve because it must allow deviations). Then, join those segments closer to that curve. This option has been discarded for two reasons: it requires a polynomial of high degree (minimum 12ž), so the computational cost is too much expensive; and it is extremely difficult to decide which horizontal segments must belong to the same function (distance between staff lines is small, so when distortions are present, segments of different staff lines are very closed).

The method proposed consists in the use of horizontal runs as seeds to detect a real segment of every staff line. Afterwards, a contour tracking process is performed in both directions following the best fit path according to a given direction. In order to avoid deviations (wrong paths) in the contour tracking process, a coarse staff approximation needs to be consulted.

Reconstruction of the hypothetical staff lines

The steps applied to obtain an image with horizontal segments (which will be candidates to form staff lines) are:

1. Obtain the skeleton of the image.
2. Use a median filter with a horizontal mask.
3. Go to step 2 until last two images are similar.

Thanks to the usage of median filters with a horizontal mask, most symbols are deleted from the skeleton of the image, and only staff lines and those horizontally-shaped symbols will remain. Median is less sensitive than mean in front of outliers (extreme values) in the image. Notice that the fact of working with binary images, simplifies the computation of the filter, because the possible values of the pixel are 0 or 1. For that reason, to compute the output value it is only necessary to count the number of 0's and 1's in the neighborhood and choose the greater value:

$$Output(i,j) = \begin{cases} 0, & \text{if number of 0's is greater than number of 1's;} \\ 1, & \text{otherwise;} \end{cases} \tag{4.13}$$

The size of this horizontal mask is constant (experimentally, the best dimensions in pixels are: 1 width $\times$ 9 height), because in the skeletonized image, each line is one pixel-width, so the width of lines in the original image is irrelevant. The process applies median filters deleting iteratively segments that are not horizontal until stability (last two images are similar).

Once the image with horizontal segments is obtained, these segments must be used to reconstruct staff lines. The method chosen for discarding or joining segments basically looks the orientation, distance and area of segments:

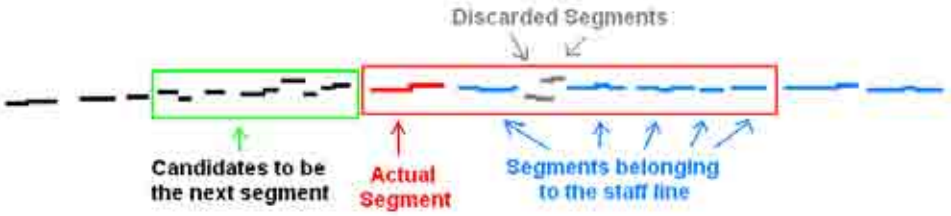


Figure 4.21: Reconstruction of staff lines: Blue segments are chosen to be part of the staff line; the red segment is the actual segment, and the next segment will be chosen between segments inside the green square. Segments inside the blue square will be used to calculate the mean of the orientation.

1. Remove little segments with a morphological opening with a structuring element circular.
2. Obtain connected components of the image.

3. For every staff line:

   (a) Chose the initial segment as the larger one.

   (b) Calculate the slope and the orientation (in degrees) of the segment: Obtain the equation of the line that fits the segment, the vector director, and finally, its orientation:

$$y = m \cdot x + n \Rightarrow \alpha = \arctan(m); \tag{4.14}$$

   (c) Reconstruct the staff line, joining segments that are in the left side of the segment. Repeat until the staff line is reconstructed:

      i. Obtain the statistical mean of the orientations of segments inside a window which contains the segments belonging to the staff line: In order to make comparisons between orientations of segments, the orientation of the next segment chosen must be compared to the orientation of the last segments which belong to the line (see Fig. 4.21). Thus, the orientation of a single segment will not be so important in order to make comparisons. The mean α orientation of n segments can be calculated as follows:

$$\overline{\alpha} = [\frac{\sum_{i=1}^{n} \cos(2 \cdot \alpha_i)}{n}, \frac{\sum_{i=1}^{n} \sin(2 \cdot \alpha_i)}{n}]; \tag{4.15}$$

      ii. Choose the next segment belonging to the staff line:

- For every candidate segment in the search window, calculate the area, distance to the actual segment A, position of their extremes, orientation of the candidate segment, and orientation J of the line that joins every candidate with the actual segment.
- Calculate the distance d between orientations of candidates segments and actual segment. The range of the angle of orientations is between [-180,180], so the distance d will be:

$$d = min\{abs(\alpha_1 - \alpha_2), 180 - abs(\alpha_1 - \alpha_2)\} \tag{4.16}$$

- Return a segment R that complies:

$$\begin{cases} distance(R \rightarrow A) < threshold; & and \\ area(R) > threshold; & and \\ orientation(R) \simeq orientation(A); & and \\ orientation(J) \simeq orientation(A); \end{cases} \tag{4.17}$$

  where A is the actual segment, R is the candidate segment,and J is the line that joins R with A.

  If no segment complies these rules, then return $\emptyset$.

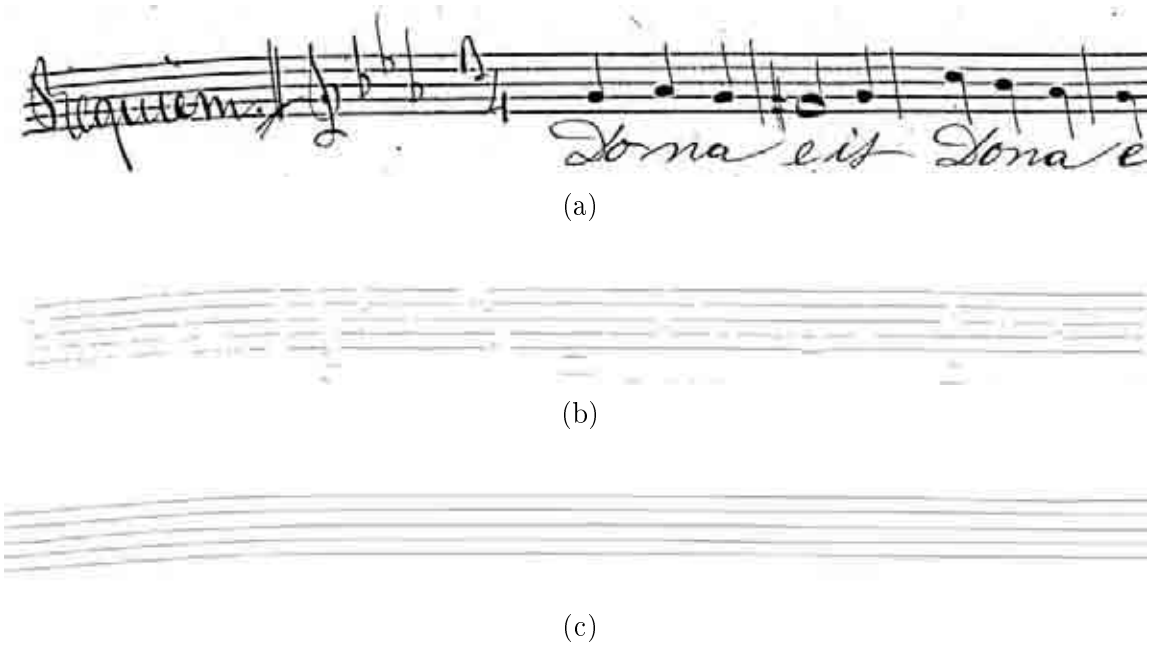


Figure 4.22: (a) Original Image; (b)Horizontal segments of the score; (c) Reconstruction of the hypothetical staff lines.

iii. If there is not a segment inside the window, then, paint the window with a line according to the mean α orientation.

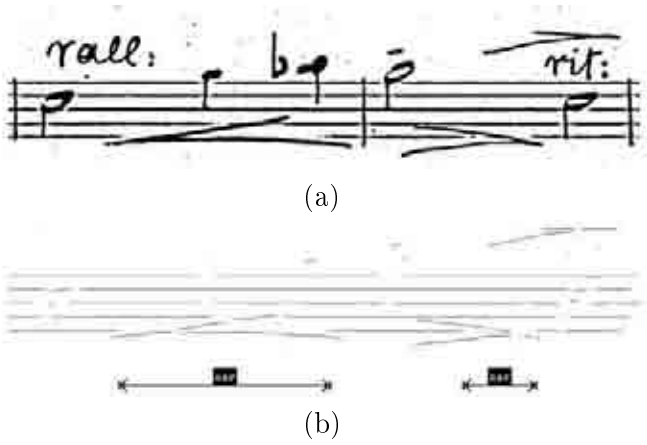
iv. If there is a segment chosen inside the window, then mark those segment as belonging to the staff line and paint the line that joins the actual segment with the chosen one.

(d) Reconstruct the staff line, joining segments that are in the right side of the segment: The method is identical to the one described in (c).

(e) Delete those segments discarded that are near the staff line.

Fig. 4.22(a) shows the original score suffering from a warping effect and Fig. 4.22(b) shows horizontal segments obtained using skeletons and median filters. The reconstruction of staff lines joining segments is shown in Fig. 4.22(c).

If there are big gaps in staff lines in presence of horizonal symbols this method could fail and follow a segment of this symbol instead of a segment of the staff line. Figure 4.23(c) shows a big gap with a crescendo marking and Fig. 4.23(d) shows its reconstruction. An initial solution to this problem consists in increasing the size of the slide-window, but it could not work in scores with large deviations in staff lines.



(a)

(b)

Figure 4.23: (a) Original Image (b) Line segments of staff lines with gaps and horizontal symbols

Contour Tracking

After the obtention of the reconstructed staff lines, the contour tracking process can be performed following the best fit path according to a given direction. The aim is to remove pixels belonging to staff once their location is roughly determined.

For every staff line:

1. Take a window that includes the staff line and obtain the width of this staff line: perform a Run Length Smearing vertical and catch the longer segment. The width of that staff line W will be the statistical mode of the width of this segment.

2. Perform a Run Length Smearing vertical with a segment of length = W, and detect a segment SG longer and closer to the horizontal line detected in the histogram of horizontal projections.

3. Take the segment SG and perform the contour tracking towards the left direction. Repeat until the beginning of the image:

   (a) Take a little column in the left side of the segment, and detect positions of the pixels which belong to the contour in that column:

   - If there is no pixel in the little column but there are pixels in a section near it, then determine if there is a change of line or not (depending on the distance and orientation).
   - Chose the connected component in the column with bigger area and closer to the actual positions of the segment. Then, calculate its extremes. If those positions are too far from the positions of the actual segment, or they are too far from the hypothetical reconstructed staff line, then reject those component.

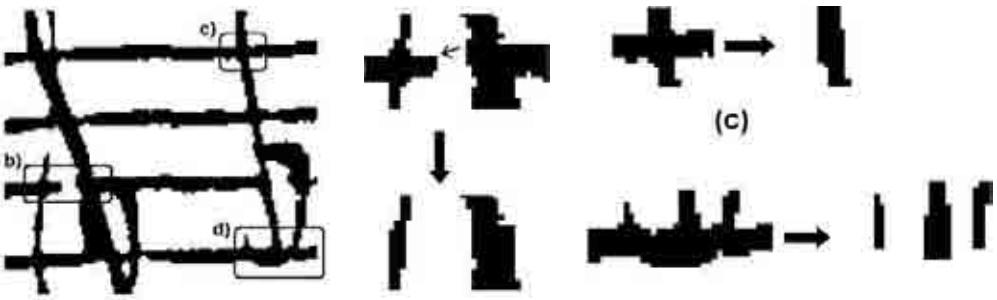
   (b) If points are returned, mark them as belonging to the staff line. If no points are returned, then mark next points, depending on the hypothetical reconstructed staff line.

4. Take the segment *SG* and perform the contour tracking towards the right direction until the end of the image: The method is identical to the one described in (c).

As it has been described, if there is no presence of staff line (a gap), the contour tracking process will be able to continue according to the location of the reconstructed staff line.

Staff Removal

Concerning line removal, we must decide which line segments can be deleted from the image, because if we delete staff lines in a carelessly way, most symbols will become broken. For that reason, only those segments of lines whose width is under a certain threshold (experimentally set to 1'2 * *width of staff lines*) will be removed. As it has been commented, width of staff lines has been calculated in the contour tracking process, using the statistical mode of line-segments. Figure 4.24 shows some examples of line removal: Figure 4.24(a) is the original image, where in Fig. 4.24(b) we can see how in presence of a gap, the process can detect next segment of staff line to continue; in Fig. 4.24(c) a symbol crossing the line will keep unbroken, because the width of the segment is over the threshold.

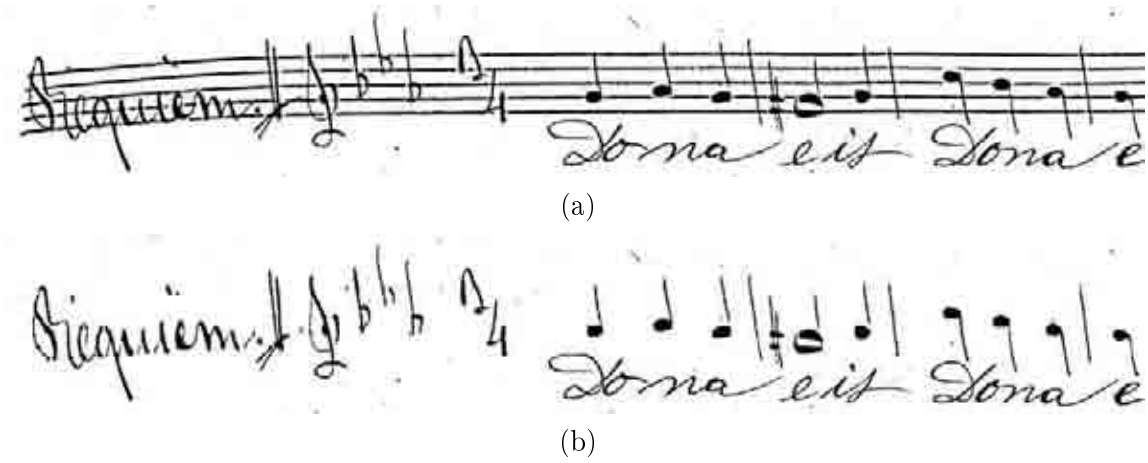


(a) (b) (d)

Figure 4.24: Examples of Line Removal in Contour Tracking process. a) Original Image, b) Gap in line, c) Symbol crosses the staff line, d) Symbol is tangent to staff line: Symbol becomes broken

In this level of recognition, it is almost impossible to avoid the deletion of segments of symbols that overwrite part of a staff line (they are tangent to staff line, see Fig. 4.24(d) and whose width is under this threshold, because context information is not still available.

Fig. 4.25 shows an example of the results of the staff removal module, which can cope with deviations in staff lines.



(a)

(b)

Figure 4.25: (a) Original Image; (b) Image without staff lines.

4.2.3 Graphical Primitive Detection

After removing the staff from the image and calculating the width of staff lines and distance between them, vertical lines and head notes are the first graphical primitives to recognize. The input image for the primitive detection module is the image without staff in which some morphological operations and run length smearing techniques are applied to reduce noise.

Recognition of vertical lines

The main idea for the detection of vertical lines is the usage of median filters with a vertical structuring element, so only symbols with vertical shape will remain (see Fig. 4.26):

1. Perform a thinning process of the image, so the width of strokes will not be determinant.
2. Obtain vertical segments performing the median filter operation with a vertical element (whose length is experimentally set to 0'4 * *distance between staff lines*).
3. Filtrate results: Perform a morphological closing and a run length smearing process. Then, discard vertical segments with little area.

Contrary to extraction of staff lines, here the size of the structuring element depends on the distance between staff lines. We have also tested Hough Transform to detect vertical lines (as we do in modern scores), but results using median filters are better and the algorithm is faster.

Recognition of filled head notes

Working with printed scores makes this process easier, because all head notes have similar shape. A morphological opening operation (with a circular structuring element), and choosing the ones with ade-

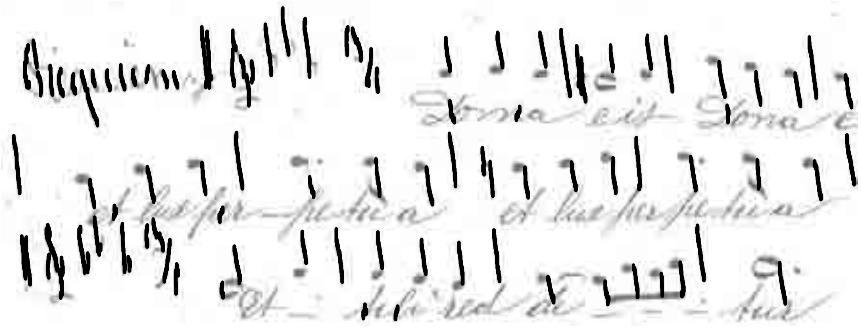


Figure 4.26: Vertical lines detected are in black color.

quate circularity and area, does not work with handwritten scores, because there is too much variability in ways of writing to perform a process that detects exactly all head notes.

The method proposed performs a morphological opening with elliptical structuring element (whose size depends on the distance between staff lines), oriented 30 degrees. After that, elements with large area are discarded. This approach gets all filled head notes and false positives (see Fig. 4.27), but it is better to discard false positives in next stages than forgetting real head notes.

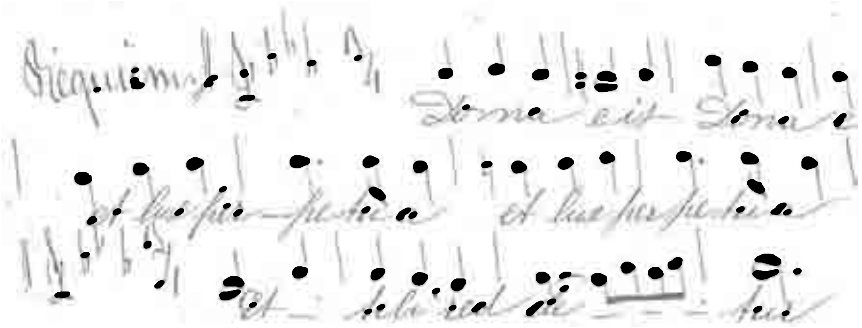


Figure 4.27: Filled head notes detected in black color.

Because of the lack of a standard notation in old scores, some modern rules of musical notation are not applied in old scores: For example, in modern musical notation if a head note is allocated under the third staff line, it has a beam on the right side; otherwise, it has a beam on the left (see Fig. 4.28). In some old scores, notes have the beam on the right side whatever their allocation on the staff. In addition, several musical notes has the headnote in the top-left side of the beam, instead of being in the top-right side.

For those reasons, the classification of notes (filled head notes with beams) will be performed in higher-level stages, using grammar rules and the knowledge of time signature.

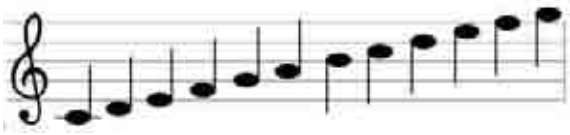


Figure 4.28: Positions of beams in the notes of a musical scale.

Recognition of bar lines

As it has been exposed in the recognition of modern musical scores, the detection of bar lines is very important because they divide the score in measures.

A first approximation of bar lines is performed assuming these two hypotheses: bar lines cover all staff and there are no head notes in their extremes. The input of the module are vertical lines and filled headnotes detected in last stages. Then, for every vertical line, mark it as a bar line if it satisfy these constraints:

- The vertical line is longer than 4 * distance between staff lines.
- There is no headnote in its extremes.
- If covers the staff at those position: Due to distortions in staff, in order to know if a bar line covers the staff, it is necessary to look the real staff location in that section to decide (see Fig. 4.29).

Figure 4.29: Bar lines in black color.

Figure 4.30 shows results of the bar line detection module. As it can be seen, there are some false positives, which will be detected in higher layers.

Once every bar line is obtained, the musical score can be divided into measures (see Fig. 4.31), and every one can be sent to the classification module of musical symbols, where grammar rules will be used.

4.2.4 Classification of clefs

As it has been said in last section, once every measure of the score is obtained, it is processed independently in order to recognize and classify all musical symbols. The heading of every score is formed of the clef (treble, alto or bass clef), time signature (commonly formed by two numbers that indicate the measure) and key signature (flats, sharps or naturals, which indicate the tonality of the

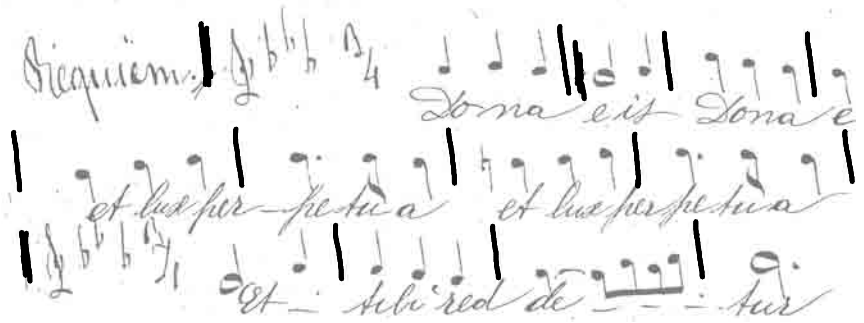


Figure 4.30: Bar lines in black color.

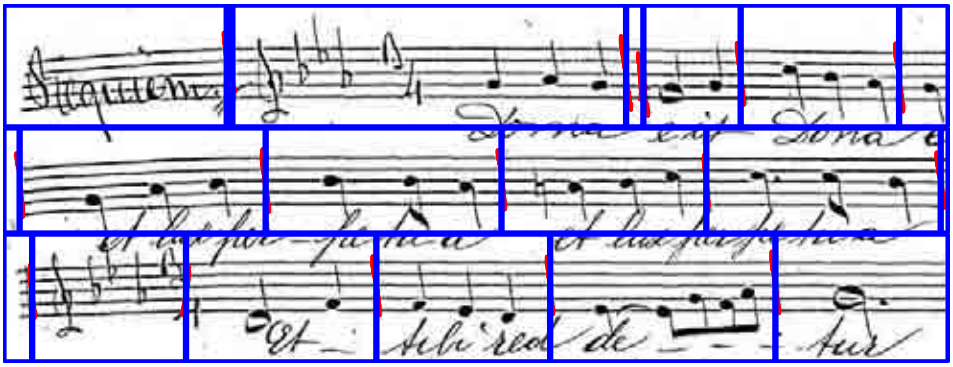


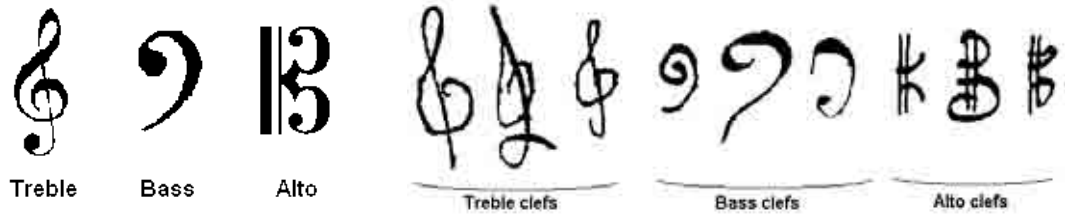
Figure 4.31: Measures of the musical score in blue color.

score). Because the clef of the score labels every note with his pitch (every note will be labelled as a Do, Re, Mi...), should be one of the first elements to recognize.

In old handwritten scores, styles in writing are very important, so there is a lot of variations in clefs (see printed clefs in Fig. 4.32(a) and see variations in handwritten clefs in Fig. 4.32(b)).

Different approaches are proposed in the literature to recognize shapes in image documents, including OCR techniques (see [2], [34], [7], [46]). Among these techniques, an initial attempt to classify clefs of the score (into treble, alto and bass clef) using horizontal and vertical projections have failed due to the enormous variations in handwritten clefs. The solution proposed consists in the usage of *Zernike moments* and *Zoning*. Zernike moments has been tested because it is a typical feature descriptor which maintains properties of the shape, being invariant in front of deformations (as happens in handwritten documents). Zoning has been also tested because it is easy and has low computational cost.

Zernike moments The main idea of our approach to classify clefs is the usage of Zernike moments (the computation of Zernike moments is fully described in Chapter 3) to construct the feature vector of a model for every class; then compare this vector with the feature vector obtained from the clef to classify and choose the class whose distance is minimum. The more moments used, the most accurate



(a) (b)

Figure 4.32: (a) Printed Clefs; (b) Handwritten clefs: there are important variations in style notation.

will be the reconstruction of the clef (see Fig. 4.33). Experimentally, 12 moments are used (less moments are not enough to classify efficiently, but the computation of more moments implies high computational cost) and 8 model classes. The steps followed by the module proposed are:

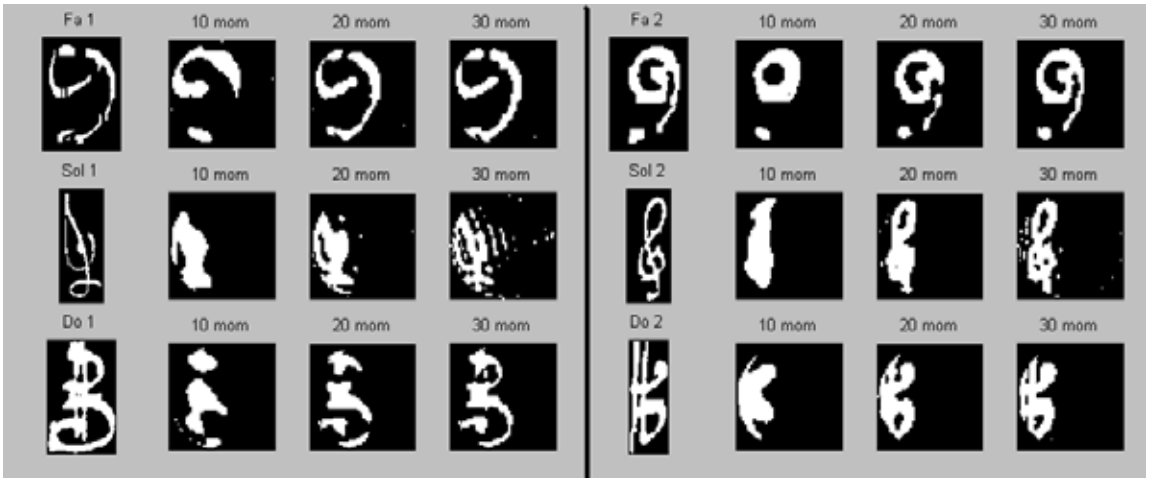


Figure 4.33: Clefs and its reconstruction using Zernike moments.

1. Normalize the image of every model of the class.
2. Compute the Zernike moments (or order 12) for every model of the class.
3. Get the feature vector of every model using the moments just obtained.
4. Normalize, compute the Zernike moments (real and imaginary) and the feature vector of the new clef C to be classified.
5. Compute the euclidean distance between the clef C and the model of every class.
6. The clef C will be labelled as the clef whose class has minimum distance to the clef C:

$$class = min\{distance(FeatureVector(clef), FeatureVector(models)\}; \quad (4.18)$$

Figure 4.34: Models used in the classification.

The rate classification of treble and alto clefs is quite good (see Chapter 5), but low rates in the classification of bass clefs must be improved combining results with *Zoning*.

Zoning As it has been said, Zoning has been also implemented because of its low computational cost, and the fact that it is able to codify shapes based in statistical distribution of points in a compact and easy way.

Thanks to the fact that in bass clefs, the top of the clef has the bigger area, so the zoning algorithm will be useful to improve the recognition of bass clefs. Thus, it will be used for a initial classification of bass clefs. Then, clefs not classified will be sent to the Zernike module to further classification.

For using the fact that a bass clef has the bigger area in the top, the image must be divided in several rows but only one column. The method consists in the following steps:

1. Normalize the image of the clef, resizing it.
2. Divide the image in sections (3 rows and 1 column, see red lines dividing every clef in Fig. 4.35).
3. Fill the zoning vector (3x1) with the normalized area of every row i:

$$mZoning(i) = area(\text{section } i)/area(image); \quad (4.19)$$

4. If the first row (mZoning(1)) is the biggest area of the vector(the white square in Fig. 4.35), then the clef is a bass clef.

Further work will be focused in the combination of Zoning and Zernike moments to improve the performance rates.

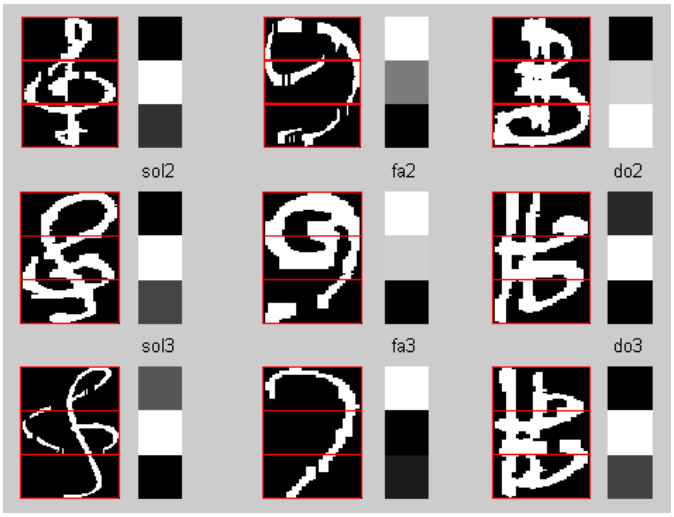


Figure 4.35: The application of Zoning technique to clefs using 3 files and 1 column to divide the images.

4.2.5 A syntactic approach for the modelization of the score structure

As it has been exposed in Chapter 1, formal language theory provides useful tools to recognize and solve ambiguities in terms of context-based rules or semantic restrictions using attributes. Grammars are usually used to describe the score structure. Therefore, parsers guide the recognition and validation process. Informally speaking, a grammar describing a score consists of three blocks **G: S** $\rightarrow$ **H[B]E**, where **H** is the heading with the attribute symbols: treble, alto or bass clef, time signature (two numbers), and key signature (flats, sharps or naturals). All these symbols are very important to provide the meaning of musical symbols. Then, the score is discomposed in bar units **B**, in which notes and rests are written down. The amount of notes and rests in every bar unit depends on the time signature, so it will obviously help to solve ambiguities in the recognition of notes and rests. Finally, there is an ending measure bar (**E**).

The grammar formalized to help in the recognition and classification stage is fully described in the Appendix of this dissertation.

Chapter 5

Results

In this chapter, results of the optical music recognition system are shown: first, results in the recognition of staff, notes and white headnotes in modern handwritten musical scores are exposed; secondly, results of the recognition of staff and graphical primitives in old handwritten musical scores are described.

5.1 Modern Handwritten scores: Results

Results on the segmentation and primitive extraction of modern handwritten scores is presented. The OMR system is tested with a set of modern images extracted from [56] and other scores written for performing tests.

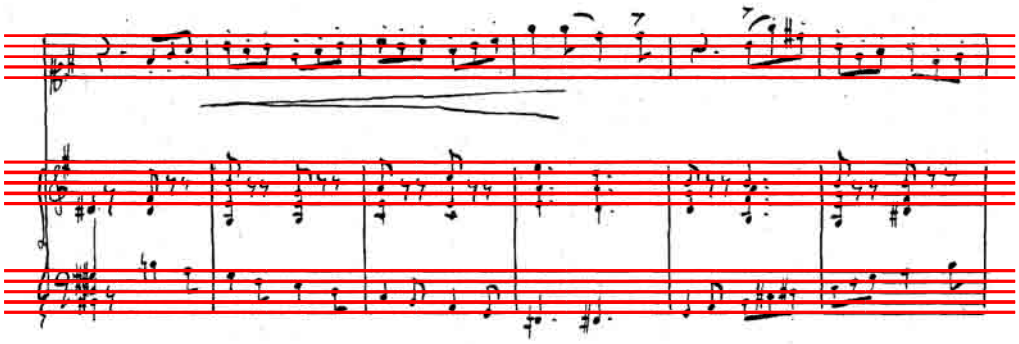
5.1.1 Staff removal

The staff removal process detects the staff lines using Hough Transform and Horizontal Projections (see Fig. 5.1(a)). Because staff lines are perfectly horizontal, they are always detected (100% detection).

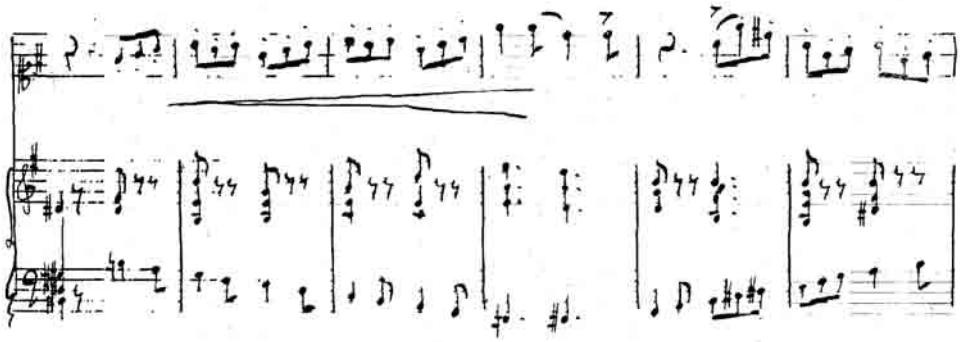
The removal process only removes those rows which value in the histogram is over a threshold. For that reason, if width of the staff line is not constant, some parts of the staff line are not perfectly removed (see Fig. 5.1(b))

5.1.2 Detection of graphical primitives

Several modern scores have been tested. Apart form the examples shown in chapter 4, another example of the graphical primitive detection is shown in Fig. 5.1. As it can be seen, most notes (filled headnotes plus a beam) are correctly detected, whereas the detection of white headnotes (in green color) has a lot of false positives and sometimes a false negatives.



(a)



(b)

Figure 5.1: Modern handwritten score: (a) Original Image with detected Staff lines in red color; (b) Image without staff lines.

In table 5.1 we can see tests performed with modern scores: detection of verticals, notes (filled headnotes with beams), and white headnotes. For the first two, recognition rates are shown, whereas for white headnotes, the percentage of false positives is a more significant parameter due to the large number of false positives. As it has been discussed, the use of context information will improve the performance rates (implemented in high-level layers).

5.2 Old Handwritten scores: Results

We have tested our method with a set of images of several composers, which have been obtained through the archive of Seminar of Barcelona. Results are divided in two subsections: results from staff removal and results from graphical primitive detection.

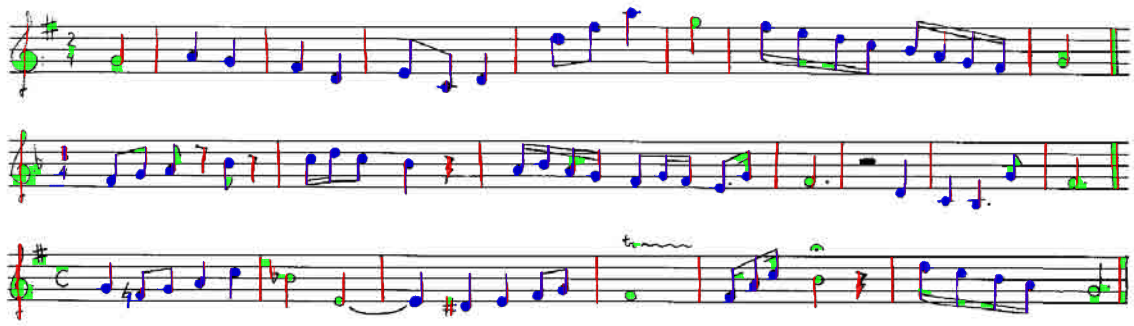


Figure 5.2: Graphical Primitive detection in modern handwritten score: notes (beam plus filled headnote) are drawn in blue color; verticals (not beams) in red color; white headnotes in green color;

Nž staffs	Verticals: Detected/Existing, (%)	Notes: Detected/Existing, (%)	White Headnotes: Detected/Existing, (%FP)
3	112 / 118 , 94%	60 / 63 , 95%	70 / 10 , 85%
4	76 / 80 , 95%	39 / 39 , 100%	43 / 8 , 81%
2	42 / 46 , 91%	23 / 29 , 79%	31 / 1 , 96%
3	141 / 150 , 94%	126 / 139 , 90%	38 / 0 , 100%
2	128 / 141 , 90%	93 / 107 , 87%	25 / 3 , 88%

Table 5.1: Results in the detection of graphical primitives in Modern Scores: Notes and Verticals (with their recognition rates); White headnotes and its percentage of False Positives(FP)

5.2.1 Description of the ground truth

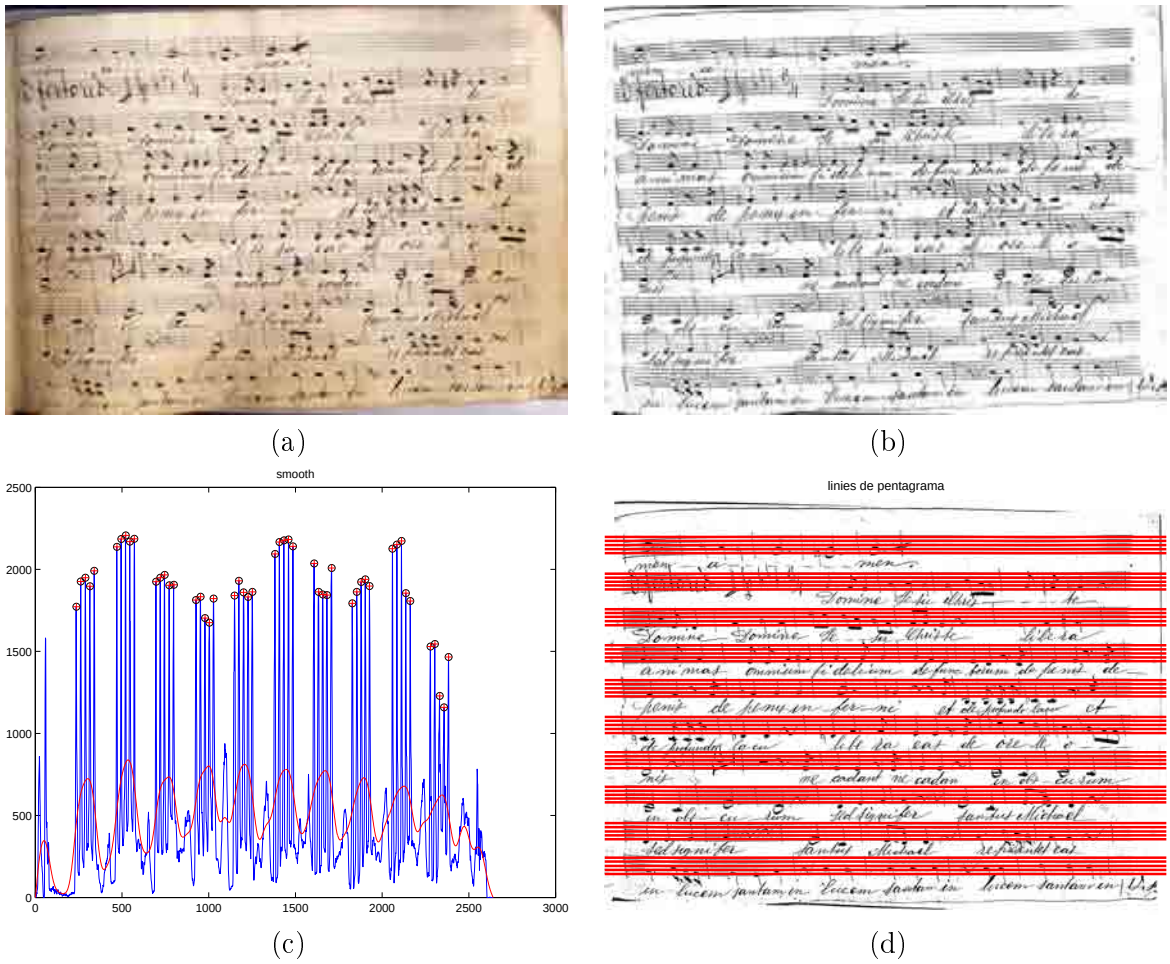
Experimental framework has been obtained from images of scores scanned from the archive of Seminar of Barcelona, where hundreds of old handwritten scores have never been edited or published. For performing tests, 19 images of old scores (from the XIX century) of three different authors have been scanned at a resolution of 300dpi. In these images one can see the lack of standard in musical notation in old scores.

The execution of the system proposed has been performed using those scores, and the evaluation of the results in detection of staff and graphical primitives are discussed in next sections.

5.2.2 Staff removal

The detection of staff lines is performed using Hough Transform, Horizontal projections and an analysis of the maximums in the histogram, trying to find five local maximums equidistant and located between local minimum (see Fig. 5.3).

Sometimes, when the author has used a staff to write down lyrics of the musical score, some strokes



(a) (b) (c) (d)

Figure 5.3: (a) Original Image (b) Binarized Image (c) Histogram with horizontal projections (d) Detected staff lines

cause too much distortion in staff lines. Then, the staff detection module can fail in the search of five maximums in the histogram and could not be able to detect a staff with text (see Fig. 5.4).

Concerning staff lines reconstruction, in table 5.2 we can see that most staff lines are perfectly reconstructed, but not everyone because in some cases a horizontal symbol is drawn over a staff line and causes the staff reconstruction to follow wrongly this symbol. An example of staff reconstruction can be seen in Fig. 5.5, where the system correctly detects those pixels belonging to staff lines although there are distortions and oscillations in the staff lines.

It must be said that whether the score is in very bad condition (and some sections of the staff can be missing), it could not reconstruct the whole staff correctly (see Fig. 5.6). Then, staff will not completely removed.

The staff deletion process (see Fig. 5.7) depends on the perfectly reconstruction of hypothetical staff

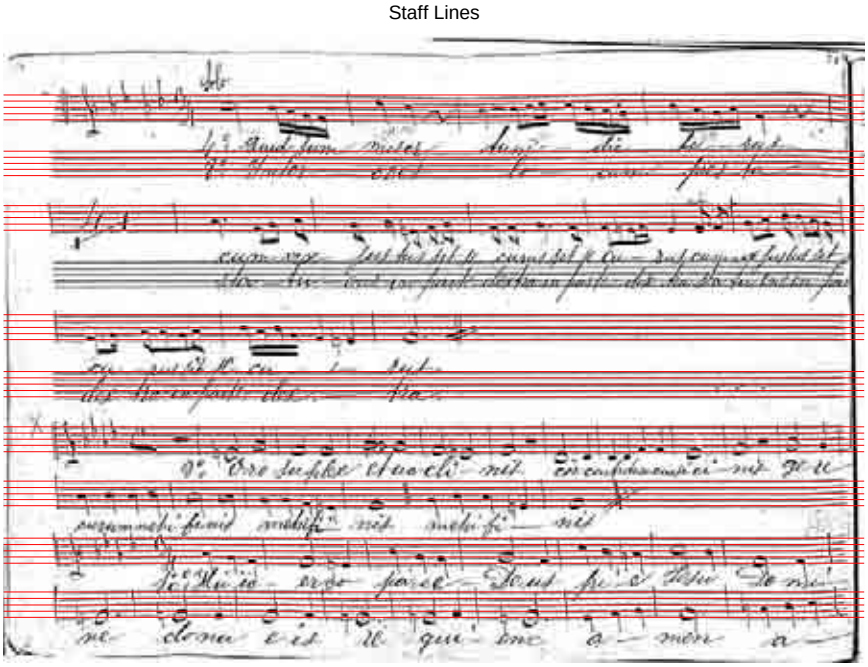


Figure 5.4: Detected Staff lines: There is one staff missing

lines, and then, in the number of symbols that are tangent to the staff line. In table 5.2 it can be seen how performance rates of staff removal depend on the staff reconstruction.

5.2.3 Detection of graphical primitives

In chapter 4 we have shown some results from a section of the Requiem Mass of the composer Aleix. In Figure 5.8 results from a section of a score of several composers are shown: Staff is removed, verticals are shown in green color, bar lines in blue color, and filled headnotes in red color.

In table 5.3 we can see that head notes, vertical and bar lines detected and the percentage of false positives (which will be detected in high-level layers). As it can be seen, most verticals are correctly recognized using filters, and some false positives are due to the verticals in lyrics (text). The detection of bar lines is also good, and false positives can be detected easily when two bar lines are very closed. Performance in detection of filled head notes decreases when strokes are very thick, so in such cases, other objects (e.g. half notes and rests) are also detected as filled head notes. Although there are many false positives, it is better to discard them in next stages than having false negatives (filled head notes in thin strokes that are not detected).

Page	Number of staffs	Perfectly Reconstructed / Total, (%)	Perfectly Removed / Total, (%)
1	10	49 / 50 , 98%	48 / 50 , 96%
2	10	50 / 50 , 100%	50 / 50 , 100%
3	10	45 / 50 , 90%	45 / 50 , 90%
4	10	49 / 50 , 98%	48 / 50 , 90%
5	12	54 / 60 , 90%	53 / 60 , 88%
6	14	70 / 70 , 100%	70 / 70 , 100%
7	14	69 / 70 , 98%	69 / 70 , 98%

Table 5.2: Staff removal results: When lines are not perfectly reconstructed, it is impossible to reach rates of 100% in staff removal

Page	Nž staffs	Verticals: Correct / Detected, (%FP)	Bar lines, (%FP)	Head notes, (%FP)
1	10	236 / 352 , 33%	71 / 80 , 11%	99 / 462 , 78%
2	10	177 / 237 , 25%	54 / 57 , 5%	96 / 465 , 79%
3	7	225 / 269 , 16%	40 / 43 , 7%	135 / 382 , 64%
4	7	218 / 284 , 23%	48 / 49 , 2%	128 / 365 , 65%
5	6	227 / 271 , 16%	38 / 41 , 7%	110 / 390 , 71%
6	6	180 / 254 , 29%	37 / 48 , 23%	122 /435 , 72%

Table 5.3: Results in the recognition of graphical primitives: 100% of Head notes, Vertical and Bar lines detected. FP= % of False Positives

5.2.4 Classification of clefs

Some tests have been done using different models for every clef. Firstly, the use of 4 models for 3 clefs will not produce good rates (almost half the number of treble and bass clefs are misclassified), due to the enormous variation in writing styles. For that reason, 4 models have been added to the system, producing better results (see table 5.4). Finally, the addition of zoning improve the classification of bass clefs, and the final module obtains classification rates of 86%.

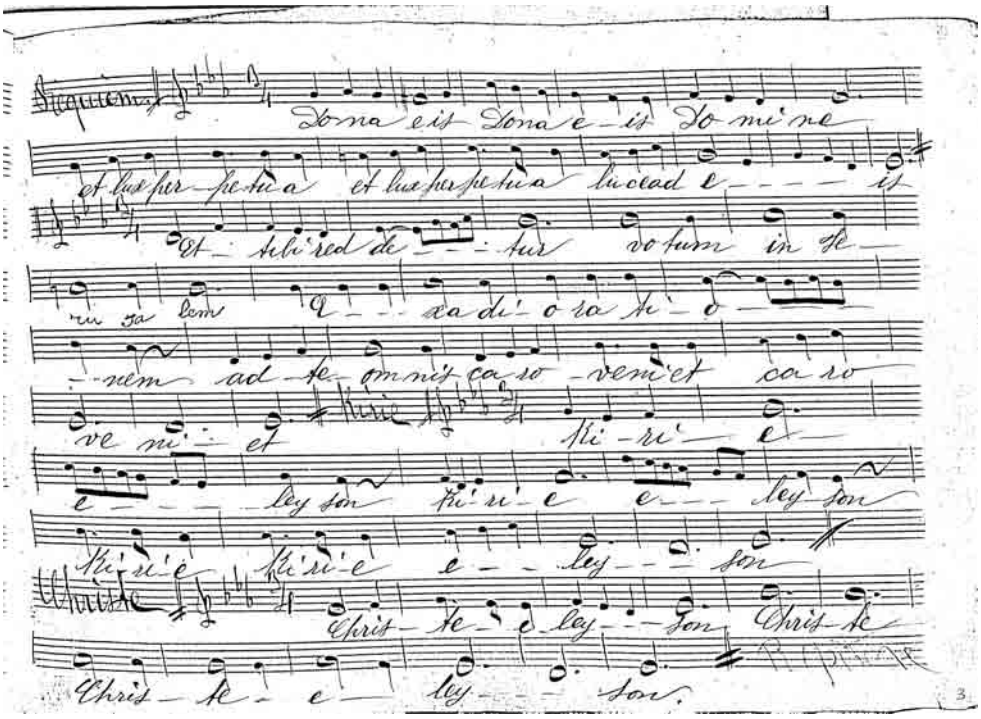
Number of models	Treble clefs Detected/Existing, (%)	Alto clefs, (%) D/E (%)	Bass clefs, (%) D/E (%)	Total D/E (%)
4	12 / 26, 46 %	3 / 14 , 21%	10 / 11 , 91%	25/51, 49%
8	22 / 26, 85 %	7 / 14 , 21%	10 / 11 , 91%	39/51, 76%
8 & zoning	22 / 26, 85 %	12 / 14 , 86%	10 / 11 , 91%	44/51, 86%

Table 5.4: Results in the classification of clefs: Classification of Treble, Alto and Bass Clefs and its performance rates. More number of models normally help to reach higher performance rates.

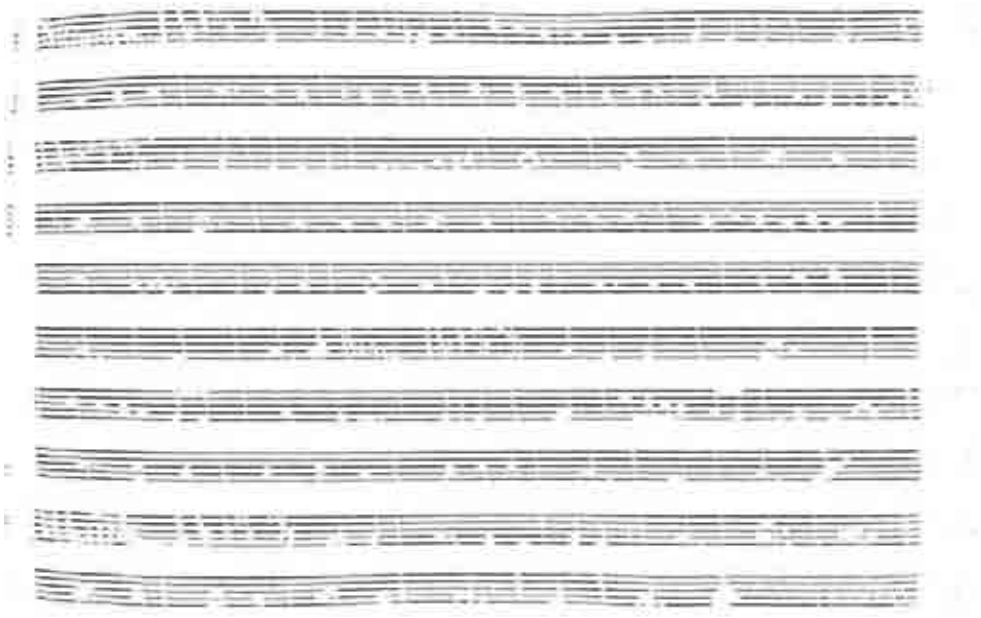
5.3 Conclusions

As it can be seen, in modern handwritten musical scores, good performance rates are reached in the detection of staffs, lines and filled headnotes, whereas the detection of white headnotes requires the use of grammars and will be further treated (in higher layers).

In the recognition approach for old handwritten scores, although it is in a preliminary stage, we have obtained high performance rates. False positives in the recognition process are due to the enormous variation in handwritten notation and the lack of a standard notation commented. Thus, the use of context information will be used into the recognition module for reducing the number of false positives and misclassified elements.

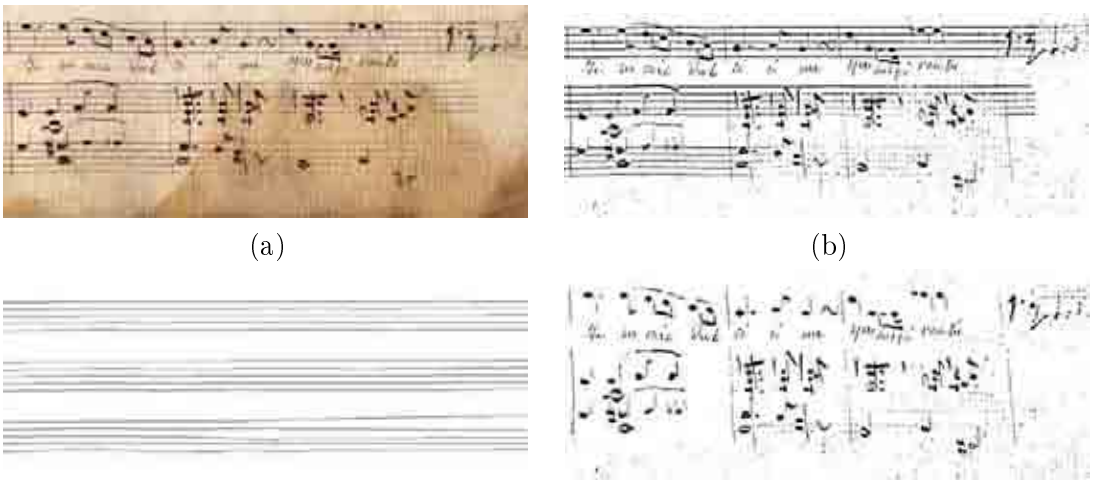


(a)



(b)

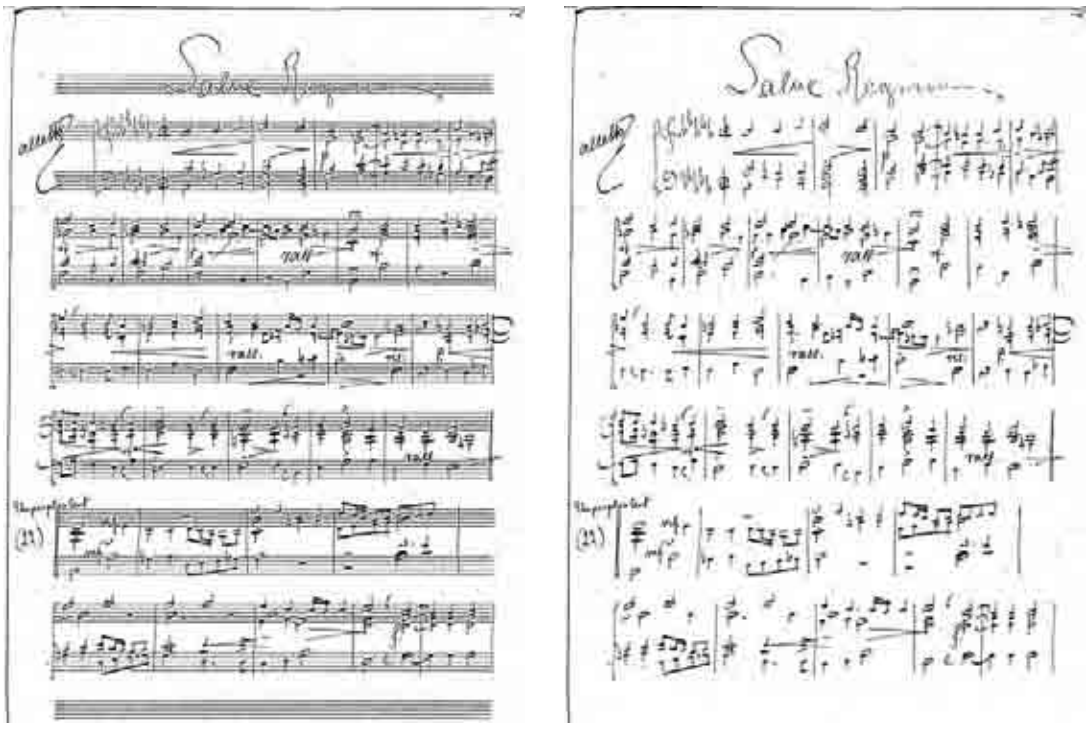
Figure 5.5: Staff Removal: (a) Original Image (b) Pixels belonging to staff lines which will be removed from the image



(a) (b)

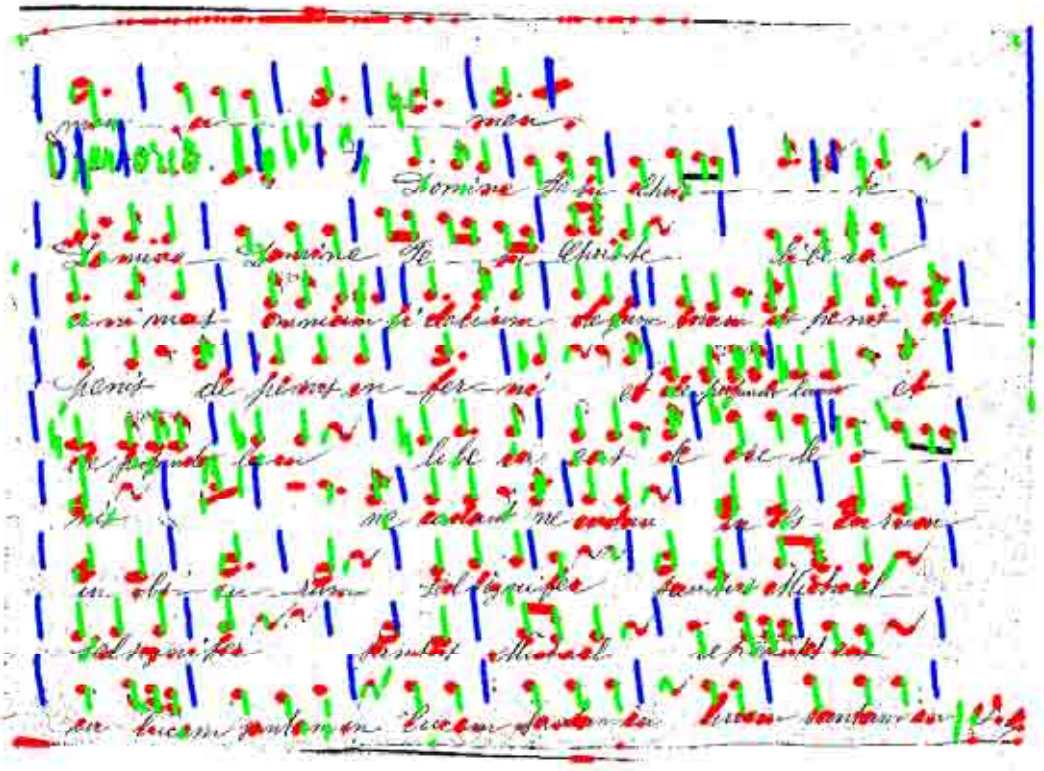
(c) (d)

Figure 5.6: Staff reconstruction: (a) Original Image; (b) Binarized image; (c) Staff reconstruction: The final part is not correctly reconstructed; (d) Staff Removal: The end of section is not completely removed

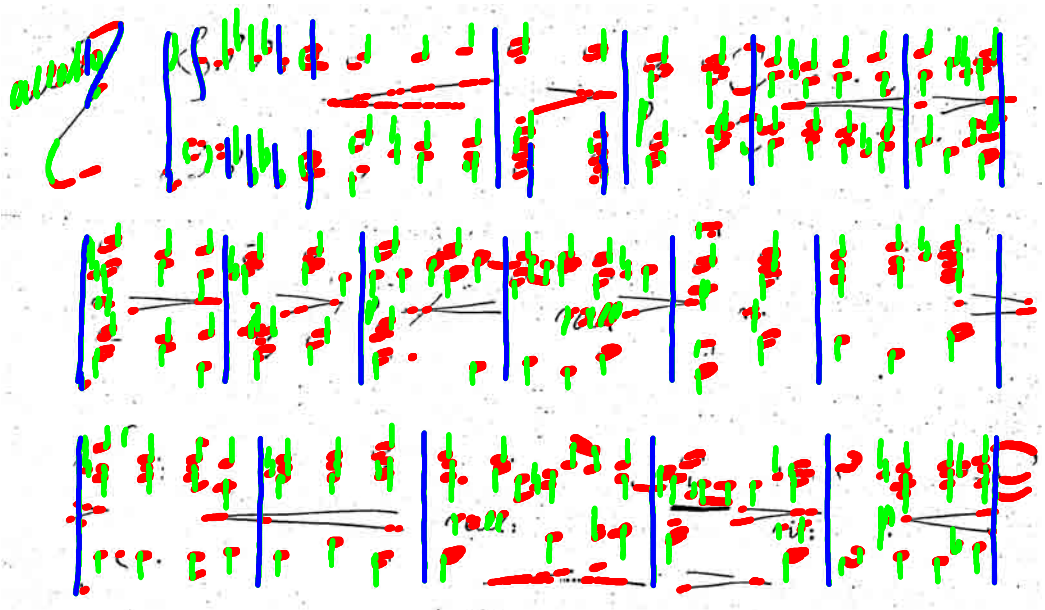


(a) (b)

Figure 5.7: Staff Removal: (a) Original Image (b) Image without staff lines



(a)



(b)

Figure 5.8: Graphical primitive detection: (a) a section of the Requiem Mass of the composer Aleix; (b) a section of "Salve Regina" of the composer Aichinger. Verticals in green color, bar lines in blue color, filled headnotes in red color.

Chapter 6

Conclusions and Future Work

In this chapter, conclusions of the Optical Music Recognition system developed is presented. After that, future work to improve the system is briefly presented.

6.1 Conclusions

In this work an approach to segment primitive elements in handwritten musical scores has been presented. Preliminary work has been performed in the recognition of modern handwritten scores: staff detection has been performed using Hough Transform, and projections; filled headnotes are detected using morphological operators and parameters of circularity, area and compactness; vertical lines have been detected using Hough Transform and classified in beams, bar lines and others. The detection of white headnotes is performed using morphological operations to fill circles, and then, applying a method similar to the detection of filled headnotes.

Good performance rates are reached in the detection of staffs, lines and filled headnotes, whereas the detection of white headnotes requires the use of grammars and will be treated in higher layers.

The system proposed has been adapted to the difficulties of the recognition of old handwritten scores:

- Old documents: working with old documents means dealing with distortions and degraded paper, so adaptive binarization techniques, morphological operations and filters must be used to preprocess the image.
- Handwritten documents: the writer style and the lack of a standard in musical notation requires the use of context information (formalized using grammars) and an expert system to cope with variations in notations.

The strategy of old handwritten scores consists of the following steps: first, score line detection and removal has been performed using Hough Transform and projections to obtain the rough approximation of the location of staff lines; then thinning and filters are used to obtain horizontal segments, and the

reconstruction of staff lines is performed joining segments according to their area and orientation. Finally a line tracking algorithm marks every pixel belonging to the staff line.

Second, the detection of vertical lines uses median filters and run length smearing; third, circular primitives corresponding to filled head notes have been extracted using morphological operators, and bar lines have been detected from the reconstruction of staff lines and head notes. Finally, classification of clefs is performed using *zoning* and *Zernike moments*.

Our work with old handwritten scores is in a preliminary stage, but we have obtained high performance rates in this primitive segmentation stage. False positives in the recognition process are due to the enormous variation in handwritten notation and the lack of a standard notation commented. Thus, the use of context information is the key to improve the recognition module and reduce the number of false positives.

6.2 Future Work

Further work will be focused on improving the reconstruction of staff lines and obtaining other graphic primitives and combining them to classify musical symbols. An important task will be the segmentation of text from musical symbols, because sometimes they are touching (see blue items in Fig. 6.1).

In front of ambiguities in the recognition process, the system could be **semi-assisted**. So, if there is any symbol difficult to recognize, the system could prompt for help to the user.

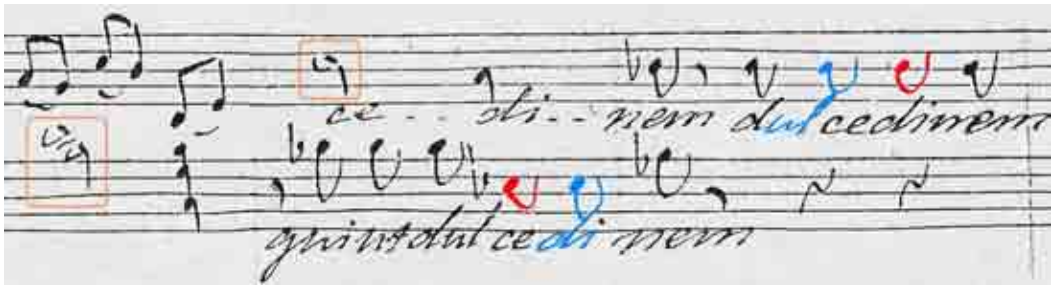


Figure 6.1: Old Score with staff lines written by hand. There are difficulties in the recognition of graphical primitives: some eighth notes (filled headnote + beam + flag) whose beams are not lines in red color; text connected to notes in blue color; white headnotes (with the circle not closed) inside an orange rectangle.

6.2.1 Staff Removal

In old handwritten scores, an important task is the detection and removal of staff lines, dealing with distortions and oscillations in lines. The contour tracking process will success wherever the hypothetical staff lines are perfectly reconstructed. Due to the fact that the reconstruction of horizontal segments

into staff lines can fail if there are big gaps or distortions in the staff, this method should be improved in order to reach higher performance rates. Possible solutions could be:

- Looking the five parallel staff lines at the same time when reconstructing. Thus, if there are deviations or ambiguities, the system can look which path the other four lines follow and then choose a path trying to keep five lines equidistant. This solution will improve the reconstruction module when working with scores with distortions and warping effect. Contrary, this constraint must be relaxed with staff lines written by hand, because sometimes, five lines are not equidistant enough (see Fig. 6.1).
- Constructing a graph with horizontal segments as nodes. Then, every reconstructed staff line will be the output of an algorithm which follows the best fit path (with backtracking). An example of contour tracking as a best fit path is described in [57].

6.2.2 Recognition of Attribute symbols

The recognition of attribute symbols at the beginning of the score (key, clef and compass signature) will require an expert system to learn every way of writing. As it has been said in the dissertation, *Zernike Moments* and *Zoning* are used to classify clefs; and they could also work in the detection of key and compass signature.

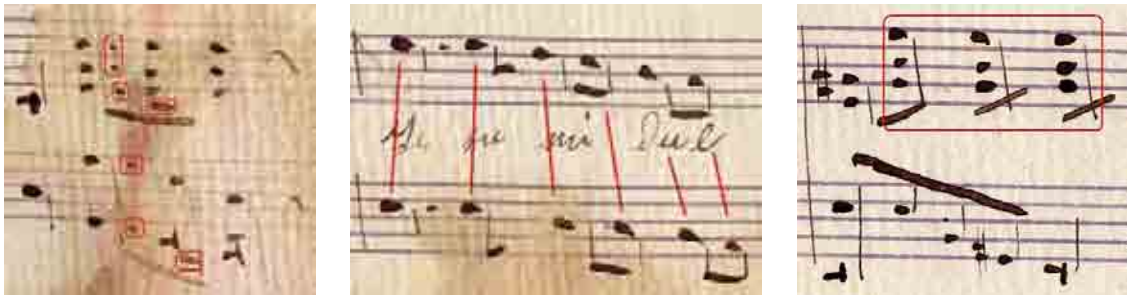
In addition, a learning module with clustering could learn every writing style: every similar unknown item could be grouped in the same class; then, when an item (belonging to this class) is recognized (using context information), automatically, all elements in this class are also labelled as the same musical symbol. For example, in Fig. 6.1, when one blue item is recognized as an eighth note, then other blue elements will also be labelled as eighth notes.

6.2.3 Recognition of Text and Lyrics

In order to cope with the extraction of text (lyrics), several possibilities could be tested:

- Look the shape of bounding box of connected components: The size of bounding box for text is different from the size of musical symbols.
- Orientations of strokes in text are changing constantly, so the Structural Tensor could be used to find sections with a lot of changes in orientation in their strokes.
- Fractal Dimension: A graphic with the area and number of dilations (iterations) will show that lines have a function different from the function of text.

Another subject is determining which text corresponds to musical notation (e.g. dynamics such as *allegro, adagio, forte, mf, coda, ritardando...*). Thanks to the fact that there are a finite set of words in musical notation (less than 200 words), a dictionary could be used to distinguish musical words from lyrics.



(a) (b) (c)

Figure 6.2: Solving ambiguities: (a) Filled black circles inside a red rectangle are not filled headnotes; (b) Each note in the first staff has a corresponding note in the second staff: notes in the second staff are the ones in the first one, but transposed two tones in a descendent way; (c) Three chords inside the red rectangle are the same; Notice that filled headnotes are not physically joined to their correspondent beam.

6.2.4 Grammars for the Graphical Primitive Detection stage

The detection of white headnotes (whole and half notes) is more difficult than filled headnotes, because handwritten circles are often broken or incomplete (see white headnotes in Fig. 6.1), so morphological operations cause too many false positives. In addition, different styles in the writing make the symbol recognition extremely difficult (e.g. a beam is a vertical line, but some beams have a semicircular shape, see red beams in Fig. 6.1). For those reasons, the grammar described in the Appendix will be used to detect white headnotes and other graphical primitives. It could also used to discard false positives in elements that look like others: in Fig. 6.2(a) we can see that some filled black circles inside a red rectangle are not really filled headnotes (they are duration dots and others).

Finally, the use of stochastic grammars with harmonic musical rules will also be very useful in front of ambiguities: In scores with several voices, the recognition can be made in parallel. For example, in Fig. 6.2(b) we can see that each note of the first voice (in the first staff) has its corresponding note in the second voice (second staff), where notes in the second staff are the ones in the first one but transposed two tones in a descendent way. Thus, if a note is not clear, the system can notice that two voices are played together and be treated in parallel. Another example is shown in Fig. 6.2(c) , where three chords inside the red rectangle are the same, so if some notes are not very clear, the other chords can be consulted.

Notice that these rules should work with classical scores, but in modern scores (XX century) harmonic musical rules are more free, so dissonances are allowed: e.g. in piano scores, if the staff for the right hand has a *Do* note, then it is possible that in the staff for the left hand, a note that seems a *Si* or *Do*, will probably be a *Do* note. Contrary, in XX and XXI century, a *Si* and a *Do* note can be played together.

Appendix A: Grammar formalized

As it has been discussed in the dissertation, context information has been formalized using a grammar to help in the recognition and classification tasks ([ ] means optional, * means repeat zero or more times, + means repeat one or more times):

- <Score = <Heading with time signature> <Section> [<Final> <Heading> <Section>]* <Conclusive Ending>.
- <Section> = <Measure> [<Bar line> <Measure>]*.
- <Heading with time signature> = <clef> [<initial key signature>] <time signature>.
- <Heading> = <clef> [<key signature>] [<time signature>].
- <Final> = <double bar line> | <beggining repeat bar line>| <ending repeat bar line>.
- <Conclusive Ending> = <double bar line> | <ending repeat bar line>.
- <Clef> = <Treble> | <Alto> | <Bass>.
- <Initial Key signature> = [♭]* | [♯]* .
- <Key signature> = [♭]* [♮]* | [♯]* [♮]* | [♮]* [[♭]* | [♯]*].
- <Time Signature> = 2/4 | 3/4 | 4/4 | C | 2/2 | 3/8 | 6/8 | 9/8 | 12/8 ...
- <Measure> = [<Note> | <Rest>]$^{+}$.
- <Note> = <White headnote> | <Headnote with a beam> | <beamed notes>.
- <Rest> = <whole rest> | <half rest> | <quarter rest> | <eighth rest> | <sixteenth rest>.
- <beamed notes> = [<headnote with a beam> <joining bar>]$^{+}$.
- <Headnote with a beam> = <beam> <headnote> | <headnote> <beam>.
- <headnote> = [accidental] <circle> <duration dot>.
- <beam> = <vertical line> [<flag>]*.
- <circle> = <white circle> | <filled circle>.

- <accidental> = <♭> | <♮> | <♯> | <𝄫> | <×>.
- <accidental> = <♭> | <♮> | <♯> | <𝄫> | <×>.
- <duration dot> = [<dot>]$^{+}$.

Appendix B: Publications

- "Staff and graphical primivite segmentation in old handwritten music scores". Alicia Fornés, Josep Lladós, Gemma Sánchez. Congrés Català d'Intel.ligència Artificial (CCIA), October 2005, Alghero, Italy.
- "Primitive segmentation in old handwritten music scores". Alicia Fornés, Josep Lladós, Gemma Sánchez. International Workshop on Graphics Recognition (GREC), August 2005, Hong Kong, China.

Bibliography

[1] V. Govindan and A. Shivaprasad, "Character recognition, a reiew," in *Proceedings in Pattern Recognition*, vol. 23, no. 7, 1990, pp. 671–683.

[2] S. Impedovo, L. Ottaviano, and S. Occhinegro, "Optical character recognition, a survey," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, no. 1, pp. 1–24, 1994.

[3] S. N. Srihari, S. W. Lam, J. J. Hull, R. K. Srihari, and V. Govindaraju, "Intelligent data retrieval from raster images of documents," in *Proceedings of Digital Libraries*, 1994, pp. 34–40.

[4] J. Lladós, E. Valveny, G. Sánchez, and E. Martí, "Symbol recognition: current advances and perspectives," in *Selected Papers from the Fourth International Workshop on Graphics Recognition Algorithms and Applications, GREC 2001*, Japan, September 2001, pp. 104–127.

[5] D. Blostein and H. S. Baird, *Structured Document Image Analysis.* Springer Verlag, 1992, ch. A critical survey of music image analysis, pp. 405–434.

[6] N. P. Carter and R. A. Bacon, *Structured Document Image Analysis.* Sringer-Verlag, 1991, ch. Automatic Recognition of Printed Music, pp. 169–203.

[7] R. Plamondon and S. N. Srihari, "On-line and off-line handwriting recognition: A comprehensive survey," in *Proceedings in IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22(1), 2000, pp. 63–84.

[8] S. Impedovo, *Fundamentals in Handwriting Recognition.*

[9] A. L. Spitz, "Tilting at windmills: adventures in attempting to reconstruct don quixote," in *Proceedings of 6th international workshop of Document Image Analysis, DAS 2004*, Italy, 2004.

[10] B. Gatos, K. Ntzios, I. Pratikakis, S. Petridis, T. Konidaris, and S. J. Perantonis, "A segmentation-free recognition technique to assist old greek handwritten manuscript ocr," in *Proceedings of 6th international workshop of Document Image Analysis, DAS 2004*, Italy, 2004, pp. 63–74.

[11] M. S. Kim, K. T. Cho, H. K. Kwag, and J. H. Kim, "Segmentation of handwritten characters for digitalizing korean historical documents," in *Proceedings of 6th international workshop of Document Image Analysis, DAS 2004*, Italy, 2004.

[12] B. Gatos, I. Pratikakis, and S. Perantonis, "An adaptive binarization technique for low quality historical documents," in *Proceedings of 6th international workshop of Document Image Analysis, DAS 2004*, Italy, 2004, pp. 102–113.

[13] B. Coüasnon and B. Rétif, "Using a grammar for a reliable full score recognition system." [Online]. Available: citeseer.nj.nec.com/33332.html

[14] H. Fahmy and D. Blostein, *Machine Vision and Applications*. Springer Verlag, 1993, ch. A graph grammar programming style for recognition of music notation, pp. 83–99.

[15] K. Ng, "Music manuscript tracing," in *International Workshop on Graphics Recognition Algorithms and Applications, GREC 2001*, Japan, September 2001.

[16] N.Bartneck, T.Bayer, J.Franke, E.Mandler, M.Oberländer, and J. Schürmann, "Document analysis, from pixel to contents," in *IEEE Proceedings: Special Issue on OCR and Document Analysis*, 1992, pp. 1101–1119.

[17] S. Mori, C. Y. Suen, and K. Yamamoto, "Historical review of ocr research and development," in *Proceedings of the IEEE*, vol. 80(7), July 1992, pp. 1029–1058.

[18] H. Kato and S. Inokuchi, *Structured Document Image Analysis*. Springer-Verlag, 1991, ch. A recognition system for printed piano music using musical knowledge and constraints, pp. 435–455.

[19] T. Matsushima, S.Ohteru, and S. Hashimoto, "An integrated music information processing system: Psb-er," in *In proceedings of 1989 International Computer Music Conference*, Columbus, Ohio, November 1989, pp. 191–198.

[20] D. Prerau, "Computer pattern recognition of standard engraved music notation, phd thesis," 1970.

[21] J. Mahoney, "Automatic analysis of musical score images, b.s. thesis," 1982.

[22] M. W. Lee and J. S. Choi, "The recogniton of printed music score and performance using computer vision system," *Journal of the Korea Institute of Electronic Engineers*, vol. 22, no. 5, pp. 429–435, setember 1985.

[23] N. Luth, "Automatic identification of music notations," in *Proceedings of the Second International Conference on WEB Delivering of Music, WEDELMUSIC 2002*, 2002.

[24] D. Pruslin, "Automatic recognition of sheet music, phd thesis," 1966.

[25] L. S. Dan, "Automatic optical music recognition, final year project report," 1998.

[26] A. Clarke, B. Brown, and M.P.Thorne, "Inexpensive optical character recognition of music notation: a new alternative for publishers," in *Proceedings, Computers in Music Research Conference*, Bailrigg, Lancaster, UK, April 1988.

[27] R. Randriamahefa, J. Cocquerez, C. Fluhr, F. Pépin, and S. Philipp, "Printed music recognition," in *In Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, Japan, 1993, pp. 898–901.

[28] I. Leplumey, J. Camillerapp, and G. Lorette, "A robust detector for music staves," in *Proceedings of the International Conference on Document Analysis and Recognition*, Japan, 1993.

[29] J. Roach and J. Tatem, "Using domain knowledge in low-level visual processing to interpret handwritten music: an experiment," in *In Proceedings of Pattern Recongition*, vol. 21(1), 1988, pp. 33–44.

[30] D. Bainbridge and N. Carter, "Automatic reading of music notation," *Handbook of Character recognition and document image analysis*, vol. 24, no. 8, pp. 583–603, 1997.

[31] H. Miyao and Y. Nakano, "Head and stem extraction from printed music scores using a neural network approach," in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, 1995, pp. 1074–1079.

[32] B. R. Modayur, V. Ramesh, R. M. Haralick, and L. G. Shapiro, "Muser: A prototype musical score recognition system using mathematical morphology," *Machine Vision and Applications*, vol. 6, no. 2-3, pp. 140–150, 1993.

[33] M. V. Stückelberg and D. S. Doermann, "On musical score recognition using probabilistic reasoning," in *In Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 1999, pp. 115–118.

[34] J. Pinto, P. Vieira, and J. Sosa, "A new graph-like classification method applied to ancient handwritten musical symbols," *International Journal of Document Analysis and Recognition, IJDAR*, vol. 6, no. 1, pp. 10–22, 2003.

[35] A.Andronico and A.Ciampa, "On automatic pattern recognition and adquisition of printed music," in *Proceedings of the International Computer Music Conference, ICDAR'95*, Venice, Italy, August 1982, pp. 245–278.

[36] D. Bainbridge and T. Bell, "An extensible optical music recognition system," in *Proceedings of the Nineteenth Australasian Computer Science Conference*, Melbourne, 1996, pp. 308–317.

[37] N. P. Carter, "Segmentation and preliminary recognition of madrigals notated in white mensural notation," *Machine Vision and Applications*, vol. 5, no. 3, pp. 223–230, 1995.

[38] B. Couasnon and J. Camillerapp, "A way to separate knowledge from program in structured document analysis: Application to optical music recognition," *Third International Conference on Document Analysis and Recognition*, August 1995.

[39] H. Bunke and P. Wang, *Handbook of character recognition and document image analysis.*

[40] L. O'Gorman and R. Kasturi, *Document Image Analysis.*

[41] Øivind Due Trier, "Goal-directed evaluation of binarization methods," in *Proceedings of IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17(12), December 1995.

[42] N. Otsu, "A threshold selection method from gray-level histograms," in *Proceedings in IEEE Trans. Systems, Man and Cybernetics*, vol. 9(1), 1979, pp. 62–66.

[43] W. Niblack, *An introduction to digital image processing.* Prentice Hall, 1986.

[44] C. Han and K. Fan, "Skeleton generation of engineering drawings via contour matching," in *Proceedings in Pattern Recognition*, vol. 27, no. 2, 1994, pp. 261–275.

[45] K. Liu, C. Y.Suen, M.Cheriet, J. N.Said, C. Nadal, and Y. Y.Tang, "Automatic extraction of baselines and data from check images," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 11, no. 4, pp. 675–697, 1997.

[46] Øivind Due Trier, A. K.Jains, and T. Taxt, "Feature extraction methods for character recognition - a survey," in *Proceedings of Pattern Recognition*, vol. 29(4), December 1996, pp. 641–662.

[47] M. Bokser, "Omnidocument technologies," in *Proceedings of the IEEE*, vol. 80, July 1992, pp. 1066–1078.

[48] P. V. W. Radtke, L. E. S. de Oliveira, R. Sabourin, and T. Wong, "Intelligent zoning design using multi-objective evolutionary algorithms," in *Proceedings in the 7th International Conference on Document Analysis and Recognition, ICDAR 2003*, vol. 2, 2003, pp. 824–828.

[49] D. Blostein, H. Fahmy, and A. Grbavec, "Practical use of graph rewriting," *Technical Report*, no. 95-373, pp. 1080–1083, August 1995.

[50] H. Fahmy and D. Blostein, "A survey of graph grammars: Theory and applications," in *Proceedings of 11th International Conference on Pattern Recognition*, vol. 2, Netherlands, September 1992, pp. 294–298, pattern Recognition Methodology and Systems.

[51] S.Baunmann, "A simplified attributed graph grammar for high-level music recognition," *Third International Conference on Document Analysis and Recognition (ICDAR'95)*, vol. 2, pp. 1080–1083, August 1995.

[52] H. Bunke and B. Messmer, "Recent advances in graph matching," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11, pp. 169–203, 1997.

[53] J. Lladós, E. Martí, and J. J. Villanueva, "Symbol recognition by error-tolerant subgraph matching between region adjacency graphs," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23(10), October 2001, pp. 1137–1143.

[54] G. Loy and A. Zelinsky, "Fast radial symmetry for detecting points of interest," in *IEEE Transactions on pattern analysis and machine intelligence*, vol. 25(8), August 2003.

[55] J. Garding and T. Lindeberg, "Direct computation of shape cues using scale-adapted spatial derivative operators," *International Journal of Computer Vision*, vol. 17, no. 2, pp. 163–191, February 1996.

[56] A. Abreu, P. Serra, and J. Zamacois, *Solfeo*.

[57] D. H. Ballard and C. M. Brown, *Computer Vision*. Prentice Hall, 1982, ch. 4.