FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Departamento de Engenharia Electrotécnica e de Computadores

Quality Assessment of Manufactured Ceramic Work Using Digital Signal Processing

Vasco Daniel Carvalho Ferreira dos Santos

Licenciado em Engenharia Electrotécnica e de Computadores
pela
Faculdade de Engenharia da Universidade do Porto

Dissertação submetida para satisfação parcial dos
requisitos do grau de mestre
em
Engenharia Electrotécnica e de Computadores
(Área de especialização em Sistemas Digitais e Informática Industrial)

Dissertação realizada sob a supervisão do
Professor Doutor Aníbal João de Sousa Ferreira,
do Departamento de Engenharia Electrotécnica e de Computadores
da Faculdade de Engenharia da Universidade do Porto

Porto, Novembro de 2004

Abstract

It has been a long time since computational *Pattern Recognition* techniques have begun being successfully implemented in the industry for the efficient automation of several kinds of tasks, namely the ones related to the quality control of manufactured objects. Still, there are some industry sectors that lack effective robust solutions for this kind of task. Examples of such industries are the industry of ceramic objects and the industry of metal casting.

This dissertation addresses the problem of the automatic quality assessment applied to manufactured ceramic pieces, more precisely to clay roof-tiles, since it still consists of a manual process and, in most cases, is only applied to a small fraction of the whole production output. The characterization technique presented herein is based of computational algorithms for *Digital Signal Processing* applied to audio and image data which are directly extracted from the pieces subjected to the proposed process. The analysis results obtained by these algorithms are presented to a classification stage that is based of *Pattern Recognition* techniques, for the effective quality assessment of the pieces.

A laboratory prototype and a set of computational applications have been developed in the context of this research, where the real time constraint is one of the main key issues.

Resumo

As técnicas computacionais de *Reconhecimento de Padrões* têm vindo desde há bastante tempo a ser implementadas com sucesso no domínio industrial para a automatização eficiente de diversos tipos de tarefas, nomeadamente as relacionadas com o controlo de qualidade de peças produzidas. No entanto, existem ainda alguns sectores industriais que carecem de soluções robustas para este tipo de tarefas. Exemplos disso são a indústria de peças cerâmicas e a indústria de fundição de peças metálicas.

A presente dissertação aborda o problema do controlo automático de qualidade em peças cerâmicas, nomeadamente em telhas de barro, já que este é ainda efectuado manualmente e, na maior parte dos casos, aplicado em apenas parte da produção. A técnica de caracterização apresentada baseia-se em algoritmos computacionais de *Processamento Digital de Sinal* aplicado a dados áudio e imagem, directamente extraídos a partir das peças sujeitas ao processo proposto. Os resultados de análise obtidos através destes algoritmos são apresentados a uma infraestrutura de classificação baseada em técnicas de *Reconhecimento de Padrões* para a aferição da qualidade das peças analisadas.

Para ajudar à realização deste trabalho foi desenvolvida uma plataforma laboratorial e um conjunto de aplicações computacionais, onde o funcionamento em tempo real se apresenta como um dos desafios principais.

Résumé

Les techniques de *Reconnaissance de Patrons* sont, depuis très longtemps, introduites avec succès dans le domaine industriel pour l'efficiente automatisation de divers types de tâches, particulièrement celles des contrôles de qualité des pièces produites. Cependant, il existe encore quelques secteurs industriels qui ont besoin de solutions robustes pour ce genre de tâches. Exemples de cela sont l'industrie de pièces céramiques et l'industrie de fonderie de pièces métalliques.

La présente dissertation aborde le problème du contrôle automatique de la qualité des pièces céramiques, particulièrement des tuiles, puisque celui-ci est encore fait manuellement et, dans la plupart des cas, est seulement appliqué dans une petite fraction de la production. La technique de la caractérisation présentée se fonde en algorithmes de *Traitement Numérique du Signal* appliqué à des données audio et d'images, directement extraits a partir des pièces soumises au procès proposé. Les résultats de l'analyse obtenus à travers ces algorithmes sont présentés à une infrastructure de classification basé en techniques de *Reconnaissance de Patrons* pour l'évaluation de la qualités des pièces analysées.

Pour aider à la réalisation de ce travail, une plateforme de laboratoire et un ensemble d'applications d'ordinateur ont été développés, où le fonctionnement en temps réel se présente comme un des principaux défis.

Preface

Ceramic industries have a long tradition within the Iberic Peninsula and, although they have benefited from significant technological improvements of the production process, quality control procedures have basically been the same for many centuries. As there are a number of possible defects that are impossible to detect by simple visual inspection (internal micro-fissures and clay imperfections), the quality control process presents many challenges. Traditionally, the control of quality consists of a manual procedure, conducted by human experts, who apply non-destructive mechanical stimuli, i.e., a stroke on the ceramic pieces, using a metallic object. The structural quality of the pieces is directly assessed through the resulting audio impulse response heard by the expert. This is a physically and psychologically agressive task that frequently leads to classification mistakes, bringing additional costs to the production flow. As a result, the production in large quantities makes the manual assessment to the whole output not a feasible process, which in turn causes this quality control procedure to only be applied to a few selected samples, except for the higher-end product families. Therefore, the need to develop reliable automatic solutions for this problem is evident. Such a solution would provide the manufacturers with the ability to assess the quality of the whole production, thus increasing the quality standards of the sector.

This dissertation addresses this problem by replicating the work of the human expert, in an automatic way. The presented solution is based on *Digital Audio Processing* techniques for analyzing the captured response of the pieces when submitted to a mechanical stimulus. A *Pattern Recognition* algorithm is applied on the resulting analysis for producing the final *conforming/non-conforming* binary classification. A *Digital Image Processing* system that recognizes the kind of ceramic object in the production line is also included in order to make the system a more robust and flexible one. This visual object recognition module provides the sound analysis module with the ability to behave in a different way, i.e., to adapt to different types of ceramic objects without the need to manually reconfigure it, allowing the continuous flow of the pieces without any interruptions, and allowing the appearance of different types of ceramic objects in the same production line.

This dissertation has inherited a previous and important work on the analysis of the captured audio signals from previously classified roof-tile samples and on the development of an empirical, yet powerful *Pattern Recognition* technique. A laboratory prototype had also partially been developed in this previous context to assist the research.

In this phase of the project, the operation in real-time has revealed to be one of the main constraints (and challenges) in the development flow, somewhat conditioning the types of algorithms and analyses implemented, as well as the kinds of platforms used.

Acknowledgements

First of all, I have to thank my supervisor, Prof. Aníbal Ferreira, for his dedication in supporting my work on this project, for the carefull revision of this text, and for having invited me to join the Audio Group at INESC Porto. These three years of such a stimulating research environment will always be present (and helpful) in my professional career as an engineer.

I would also like to thank INESC Porto, in particular the Multimedia and Telecommunations Unit to which I have belonged during these three years, for the possibility of writing this dissertation and for the chance of being able to show my work to the rest of the world.

I also owe a lot of gratitude to Chipidea Microelectrónica, S. A., the company which I currently work at, for giving the opportunity to work in the areas that I like most, and to all my colleagues for such an overwhelming working experience.

Thanks to my closest friends who fortunately have been accompanying and encouraging me since my earliest childhood. Yes, you know who you are...

Many thanks to my family, in particular to my mother and father, for their never ending support. This dissertation is dedicated to them.

I keep thinking but I cannot find the right words to thank you Mariela. It just seems that too much is not enough. Still, I just want to let you know that it is your smile that gives me the strengths to move big mountains like this one. But then again, you already know it, don't you?...

Reader notes

This document was written using the LaTeX typesetting system. Normal text is written using the Roman font. Computer code and the identification of some hardware devices are written with the `Typewriter` font. In the beginning of this document, a list of acronyms is presented, in order not to overload the text with excessive definitions. Bibliographical references are listed at the end of the document, and are referred to with the initials of the author followed by the year of publication, bounded with rectangular brackets.

Contents

List of Figures

List of Tables

List of Acronyms

ADC	Analog-to-Digital Convertion
AGC	Automatic Gain Control
ANSI	American National Standards Institute
CCD	Charge Coupled Device
DAC	Digital-to-Analog Convertion
DAT	Digital Audio Tape
DSP	Digital Signal Processing
FFT	Fast Fourier Transform
GUI	Graphical User Interface
IDE	Integrated Desktop Environment
JTAG	Joint Test Action Group
LPC	Linear Predictive Coding
ODBC	Open DataBase Connectivity
PCA	Principal Component Analysis
PCI	Peripheral Component Interconnect bus
PR	Pattern Recognition
PSD	Power Spectral Density
SNR	Signal-to-Noise Ratio
SPL	Sound Pressure Level
SQL	Structured Query Language
TCP	Transmission Control Protocol
UART	Universal Asynchronous Receiver/Transmitter

Chapter 1

Introduction

Semantic inspection of media by machines is one of the most challenging areas of engineering, since a number of decades ago. Technologies for the automatic interpretation of either audio events, still images or video, form without any doubt, one of the most intense research areas of the recent years. Moreover, with the fast growth of computer technology that we have been observing, impelled by the advances in electronic chip fabrication, the search for real-time solutions is emerging in a way that never had before. This is evident in several industry sectors, where these current advances in available technology are leading to faster and more efficient automatizations of many types of manufacturing processes. One of the production tasks that have been benefiting from these automatizations is the quality control. Quality control of manufactured materials is a form of semantically interpreting their characteristics and definitely constitutes a crucial task in industrial competitiveness.

This dissertation presents a solution for the automatic assessment of the structural quality of manufactured ceramic work, based on visual object recognition and digital sound processing. The method works by first detecting the type of ceramic piece that is present in the production line, and then by analyzing their acoustic impulse responses, which reflect possible non-conformities present in their structure. The real-time constraint applied to the development flow serves to show that this is more than a theoretical study. The main pursued objective is to apply this technique in the industry.

There are already several patented techniques for the quality assessment of ceramic objects, some of them being quite old. For example, [1] presents a crack detection system that is based on heating the ceramic pieces. In this heating process, acoustic emissions from the piece are detected by a acoustic-electric sensor that is coupled to the ceramic product by means of a waveguide. By applying this thermal stress, an internal crack is forced to grow, and an acoustic emission is released. The time that it takes to heat the piece and to detect the cracks makes this system of low applicability in the cases where the flow of the ceramic pieces in the production line is fast. Moreover, as the acoustic emissions caused by thermal stress are of very low amplitude, this system is likely to fail in a noise-hostile environment.

Another example is [2], where only surface cracks are evaluated. This system uses a solution of silver nitrate, applied to the surface of the pieces, and that penetrates into the defects. The object is then dried out so that the silver nitrate remains in the defects. The analysis of the defects is carried out by means of an X-ray radiograph where any defects and micro-cracks will appear. This system lacks the important ability to detect micro fissures

that are beneath the surface of the pieces, and apparently is also unable to deal with rapid production flows.

The system described in [3] uses a piezoelectric transducer beneath a supporting base for the analyzed pieces. The pieces are then submitted to a load. If a crack is present in the piece, a stress wave is emitted, captured by the transducer and converted to electric pulses. Because of the properties of ceramic, these pulses are of low amplitude, short duration, and fast rise time. This requires the use of special (and expensive) electronic circuitry in order to detect these pulses and to distinguish them from the background noise.

Other systems like the one in [4] use ultra-sonic excitation in which the acquisition and analysis of the produced echoes enables the identification of the properties of the tested materials, namely in the quality assessment of ceramic and metallic pieces. However, this method relies in the use of complex ultra-sonic generation and acquisition devices which, in most times, are not compatible with the conditions found in the ceramic factories.

Generally, the only quality control process applied in the ceramic industry consists of a manual process. The human operator takes the pieces by hand and uses a metallic object to apply a non-destructive stroke to them. The acoustic response heard by the operator suffices to judge upon the structural quality of the pieces. This operation is carried out at the end of the production flow. In fact, if a ceramic object has a good structural integrity, the acoustic vibrations along its body will propagate freely from end to end. In the case of an object containing cracks or micro fissures, these vibrations will constantly be attenuated upon passing through the defects. These different vibration modes are directly evidenced in the emitted acoustic responses. This manual procedure is used with almost all kinds of ceramic products.

Since this is a costly process in terms of hand labour, assessing the quality of the whole production output is only justified with higher grade products. Furthermore, as it is a monotonous and tedious task for the operators and that requires much of their concentration, frequent flaws arise in the discrimination criteria.

This work describes a method for the automatic detection of structural flaws applied to the particular case of clay roof-tiles, using the same principles described above. The idea is to develop a robust prototype system using only low cost devices, in contrary to other techniques. By using efficient processing platforms, the system is able to assess the quality of the manufactured materials in real-time, without affecting their continuous flow on the production line. Thus, this automatic procedure would allow the manufacturers to apply a quality control procedure to their whole production, thereby increasing the quality standards of the ceramic industry sector, as well as customer satisfaction. The feasibility of the method has previously been validated using red-bricks [5].

Chapter 2

Prototype System Apparatus

In this chapter, a brief overview of the prototype system that aided this research is presented. It comprises the following 6 modules:

- An aluminum structure that holds the ceramic pieces to analyze, simulating a real production line;
- A striking system composed of a metallic arm and a pneumatic actuator;
- A sound acquisition system composed of two microphones and an audio mixing console;
- A DSP board;
- An image acquisition system that comprises a digital acquisition camera and a frame grabber card;
- A PC running a Windows operating system.

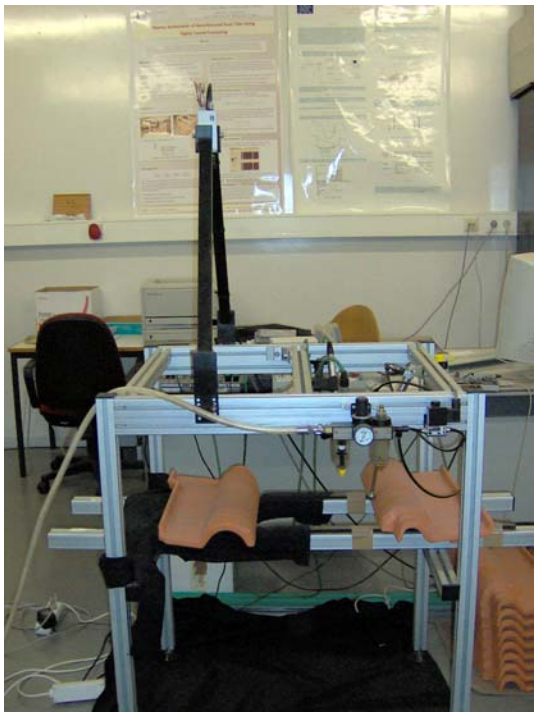
In the next sections of this chapter, a more detailed description of each prototype module is presented. Note that some of its parts have been inherited from a previous phase of the project [6, 7].

2.1 Overall system aspect

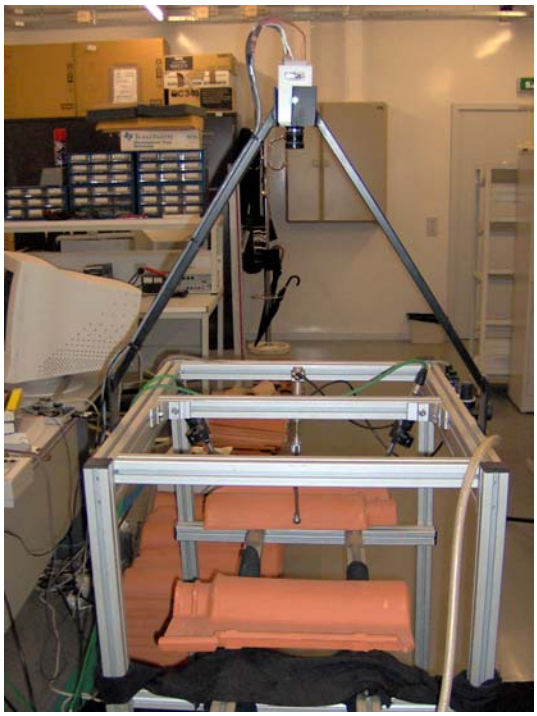
Figure 2.1 presents a set of 4 photographs of the prototype, showing the position of the image acquisition camera, the two microphones, and the metallic hammer that physically interacts with the ceramic tiles. The position of the hammer in the pictures show the hit point of the tiles. This spot has been selected following the informations given by operators that apply the manual quality control procedure. Several other hit points have been tested in the laboratory [6], but this one has shown to be adequate for any kind of fracture. The black cloth on the floor is for easing the segmentation process in the image processing algorithm (see Chapter 4 for details).

2.2 The striking system

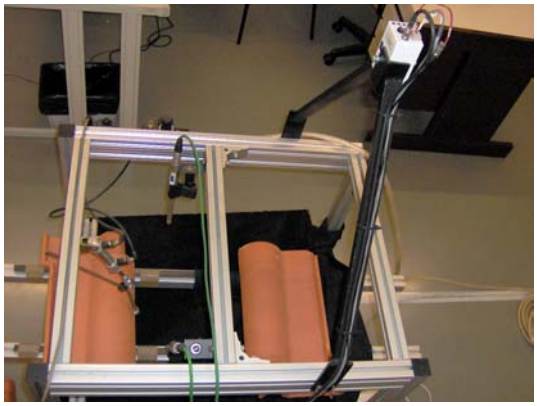
The striking system comprises an air compressor, an electro-pneumatic valve, an electronic triggering system connected to the parallel port of the PC, and a metallic arm (hammer)



(a) Side view.



(b) Front view.



(c) Top view.

(d) Detail of microphone positioning and the hammer.

Figure 2.1: Photographs of the metallic structure that holds the image acquisition camera, the two microphones, and the striking hammer.

that hits the ceramic pieces.

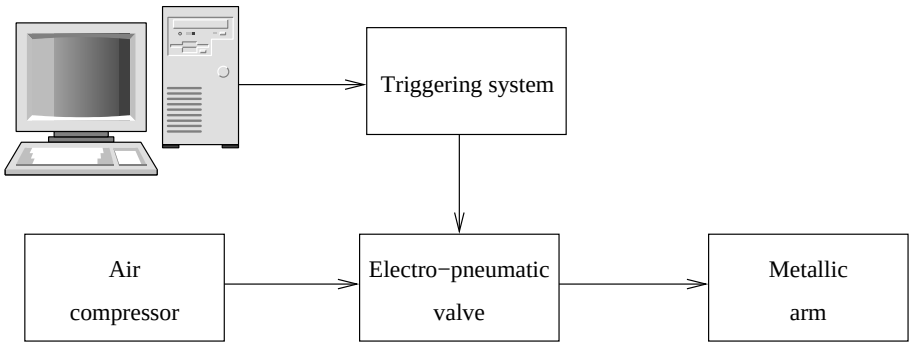


Figure 2.2: An illustration of the connections between the PC and the striking hammer.

Figure 2.2 shows an illustration of the connections between the PC and the hammer. The electro-pneumatic valve is opened for pulling the hammer up and closed for striking the piece (hammer goes down). The opening and closing orders come from the parallel port of the PC, and are software-triggered. In a real production line, these striking orders would need to be automatically given upon the detection of a piece with the aid of a presence detection circuit. Figure 2.3 presents an illustration of the constitution of the metallic arm

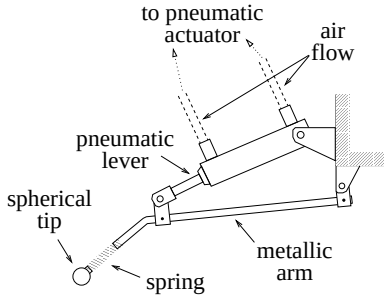


Figure 2.3: An illustration of the striking hammer and its constituting parts.

that physically interacts with the pieces after the pneumatic command is issued. The spring attached between the metallic arm and the spherical tip serves to eliminate rebounds when striking the piece [6].

2.3 The sound acquisition system

The sound acquisition module uses two `Behringer ECM8000` measurement microphones and a `Behringer Eurorack MX1602` audio mixing console for capturing the impulse responses of the roof-tiles. The microphones are omni-radial and active (phantom powered at 48 Volts), and the mixing console is also used for leveling the signal amplitudes. The use of a conventional AGC system could be used instead of the mixing console for maintaining the microphone levels within a certain amplitude range, but this system would also change the signal in a way that the feature extraction process described in Chapter 3 could no longer

be applied in an effective way, i.e., the algorithm would have to cope with the changes that the AGC introduces in the signal, which would not be an easy task.

There are two main reasons for using more than one microphone for capturing the acoustic impulse response of the roof-tiles. On one hand, the acoustic energy emitted by the piece varies along with its body, and thus it becomes reasonable to place more than one microphone in carefully chosen spots. On the other hand, acoustic reflections and other perturbations related to sound propagation can influence the sound analysis process. By considering more than one capture spot, this problem is somewhat attenuated. In Figure 2.1d one can see the position of the microphones relatively to the tile. These positions have been carefully chosen so as to capture the most important characteristics of the acoustic signals coming from the body of the tiles [6].

2.4 The DSP board

The DSP board is used for analyzing the captured acoustic impulse responses and for feature extraction. The `Analog Devices ADSP-21061 SHARC EZ-KIT Lite` board was chosen for this task. These are some of the specifications of this board:

- 32-bit IEEE floating point processing with the `ADSP-21061` processor, operating at a 40 MHz clock frequency;
- Non-intrusive code debugging with the JTAG probe connection;
- Includes the 16-bit `Analog Devices AD1847 Soundport` $\Sigma\Delta$ stereo audio ADC/DAC interface featuring a dynamic range above 76 dB and sampling frequencies up to 48 kHz;
- Includes an `RS-232` compliant serial port connector through the `PC16550` UART chip.

Programming and debugging DSP algorithms with this kit is made easy through the `Analog Devices VisualDSP++` IDE. The algorithm for feature extraction and their transmission to the PC for the final classification stage was programmed using the C language.

2.5 The image acquisition system

The image acquisition system is composed of a color CCD acquisition camera (`JAI CV-S3300`) and a PCI digital frame grabber (`Imagenation PXC200AL`). The frame grabber card is software-programmed for capturing RGB color images in real-time, with 384 (h) $\times$ 288 (v) pixels, and at a rate of 50 frames per second.

The position of the camera is set so as to capture image planes that are parallel to the horizontal plane of the roof-tiles, as shown in Figure 2.1.

2.6 Overview of the developed computer software

All developed computer software was programmed using the C++ language and the `Borland C++ Builder` IDE. This IDE makes the development of GUIs a rapid and easy task. The following 4 graphical applications were developed (see Chapter 5 for further details):

- A training set storage application for the image analysis module;
- A training set storage application for the sound analysis module;
- A results demonstration application for the image analysis module;
- A results demonstration application for the sound analysis module.

All developed applications use a database that stores all required data. The training set storage applications store the data pertaining to the training set samples in this database. The results demonstration applications retrieve these stored data and use it for classification purposes. The database system used is the `MySQL` open source database management system (more information at `http://www.mysql.org`). Section 5.1 presents more detailed information on the structure and implementation of the used database.

The choice of an external database management system is advantageous in many aspects. Some of these aspects are:

Increased speed. Advanced database management systems like `MySQL` implement highly efficient algorithms for data storage and retrieval, as well as optimized internal search mechanisms.

Data integrity. Data integrity is guaranteed through the use of commonly implemented database transaction mechanism like atomic operations [8].

Flexibility. This kind of relational database systems permit flexible ways of creating new database tables, storing, retrieving and deleting data through the use of structured queries. Also, the task of re-allocating the database server in a remote machine is made easier.

All applications use an ODBC `MySQL` driver layer for accessing the database, that must be previously installed for the applications to work correctly[1]. The applications then send queries and receive data through this driver, always using the ANSI SQL-92 language [9].

The two results demonstration applications need to communicate, so as the results of the image analysis module (the recognized type of ceramic piece) be sent to the sound analysis module. The latter module will then load the appropriate training set for a more accurate classification. This communication is accomplished with the use of Internet socket

[1] Actually, not only the driver must be present, but also the database server service must be running for the correct operation of the applications.

programming [10]. This communication mode was chosen for its flexibility if, for instance, the two applications need to run on different machines. This topic is further detailed in Chapter 5.

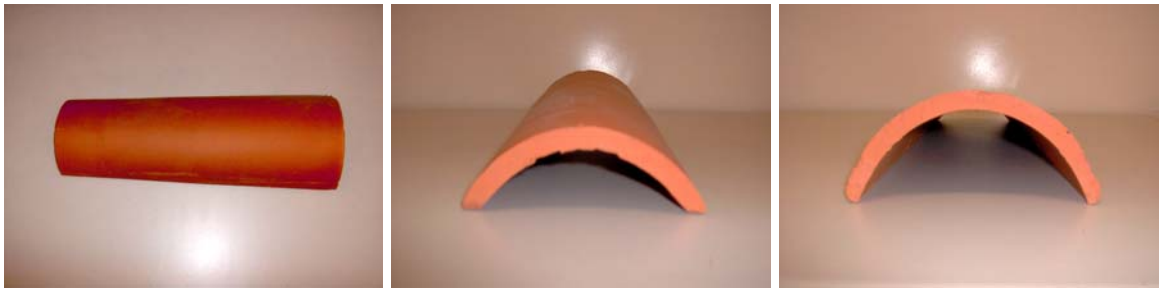
All applications run on a 350 MHz Pentium II PC, supported by a Windows XP operating system.

2.7 Roof-tile samples present in the laboratory

Several roof-tiles are present in the laboratory for the experiments. The set consists of 3 different types of tiles which are of the most commonly used. Figure 2.4 presents some photographs of each of these types.

(a) Spanish or "S"-type.

(b) French or Marseille.



(c) Barrel.

Figure 2.4: Photographs of the types of tiles present in the laboratory.

Unfortunately, only the spanish roof-tiles have enough pre-classified samples to be used in the experiments. The other two types are used only for designing and testing the performance

of the image analysis module.

For the spanish type, 34 pre-classified samples (17 of class *Conforming* and 17 of class *Non-conforming*) are used in the experiments for the sound analysis module. These samples come from the same manufacturer and have been manually classified by human experts.

Chapter 3

The Sound Analysis Module

This chapter presents the description of the part of the system that takes care of capturing, analyzing and classifying the acoustic responses coming from the roof-tiles. As mentioned in Section 2.7, only the roof-tiles of the spanish kind are considered in the experiments. Most part of this module has been inherited from the work described in [6], namely the feature extraction process and the classifying scheme used. However, further developments have been performed involving the communication between the DSP board and the PC, as well as deeper analyses on the discriminating power provided by the extracted features.

3.1 Triggering, detecting and capturing impulse responses

The process of triggering strokes on the ceramic pieces is started by the results demonstration application that runs on the PC. This command is then transmitted to the pneumatic system through the parallel port of the PC.

Both the detection and capture of the acoustic signals tasks are performed by the DSP platform and can be summarized in the following way:

1. Each 128-samples signal interval (≈2.9 msec.) passes through a fifth order inverse linear prediction filter whose coefficients are listed in table 3.1. These coefficients have been experimentally obtained using high quality audio signals in a previous perceptual audio coding research context [11]. The resulting frequency response is shown in Figure 3.1.

2. The energy of the signal resulting from the previous filtering stage is computed. A fixed threshold is then applied to determine if the energy of the signal interval is enough to be considered as containing the initial part of an acoustic impulse response that is the result of the application of a stroke on the piece.

3. After the transient detection, exactly 4864 audio samples (including the 128-samples interval used in the previous stages) are acquired, which represents approximately 95 msec. of audio content. All acquired samples remain in memory for the next algorithm steps.

The above procedure is applied to both audio channels.

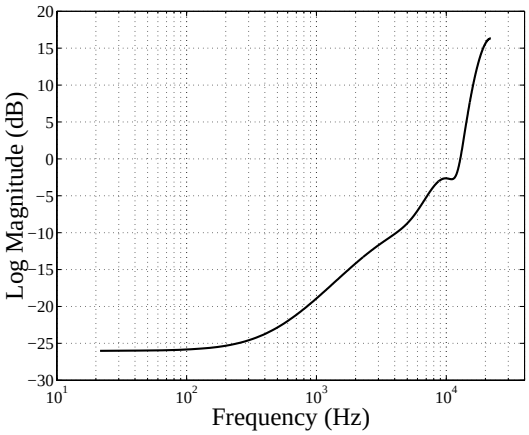


Figure 3.1: Frequency response of the inverse linear prediction used in the transient detection stage. Its impulse response is given by $h(n) = \sum_{k=0}^{5} \alpha_k x(n-k)$.

Coefficient	α_0	α_1	α_2	α_3	α_4	α_5
Value	1.00	−1.74	1.57	−1.27	0.73	−0.24

Table 3.1: Coefficients of the linear prediction filter that is used for stroke detection.

3.2 Feature extraction

Figures 3.2 and 3.3 show the time, frequency and spectogram representations for two acquired acoustic responses, coming from tiles of classes C and NC, respectively. Clearly, it is seen that direct analysis of only time or frequency information of the captured signal does not permit a reliable evaluation of the structural quality of the pieces under test [6]. However, in the spectograms, one can see that tiles from both classes exhibit different behaviours. For example, it can be seen that the spectral content pertaining to the tiles of class C varies in a much more structured way than with the NC tiles, whose spectograms show a more chaotic behaviour. The visible "bumps" on some of the spectograms are due to slight rebounds of the spherical tip of the hammer in the surface of the tiles. These bumps are not a problem if the analyses applied to the signals are restricted to only their first, say, 100 msec.

Based on the information of the spectograms, a mixed time-frequency scheme is used. The base philosophy is to monitor the evolution of certain characteristics of the spectral content of the signal over time.

The following sequence of operations is performed to the audio samples acquired on both channels:

1. The audio content is segmented in an analysis filter bank composed of 16 Fast Fourier Transformations (FFTs), overlapping with each other by 75%. The hanning window function is used for segment separation.

2. The 512-point linear spectral content is mapped to 4 non-linear frequency bands whose limits are shown in Table 3.2.

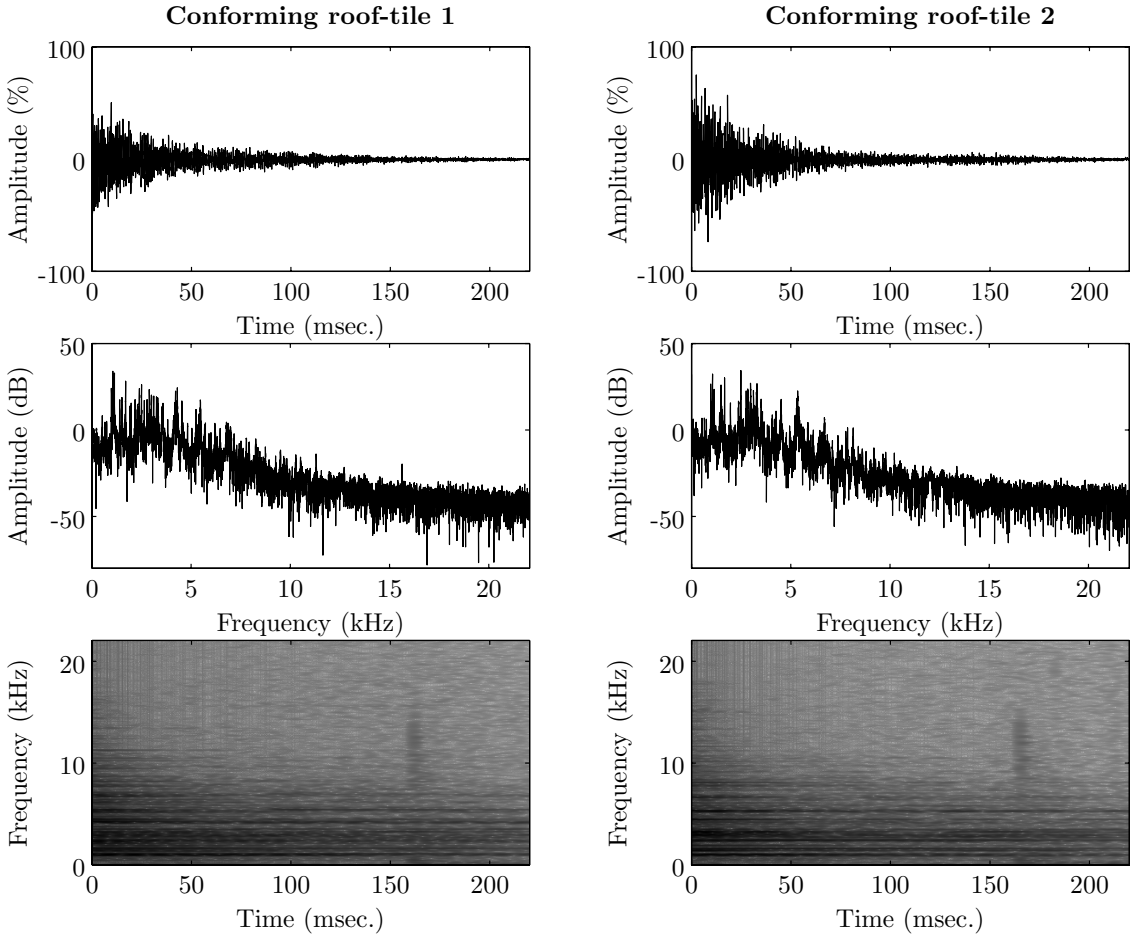


Figure 3.2: Time, frequency and spectrogram representations for the acoustic responses of two tiles of the *conforming* class.

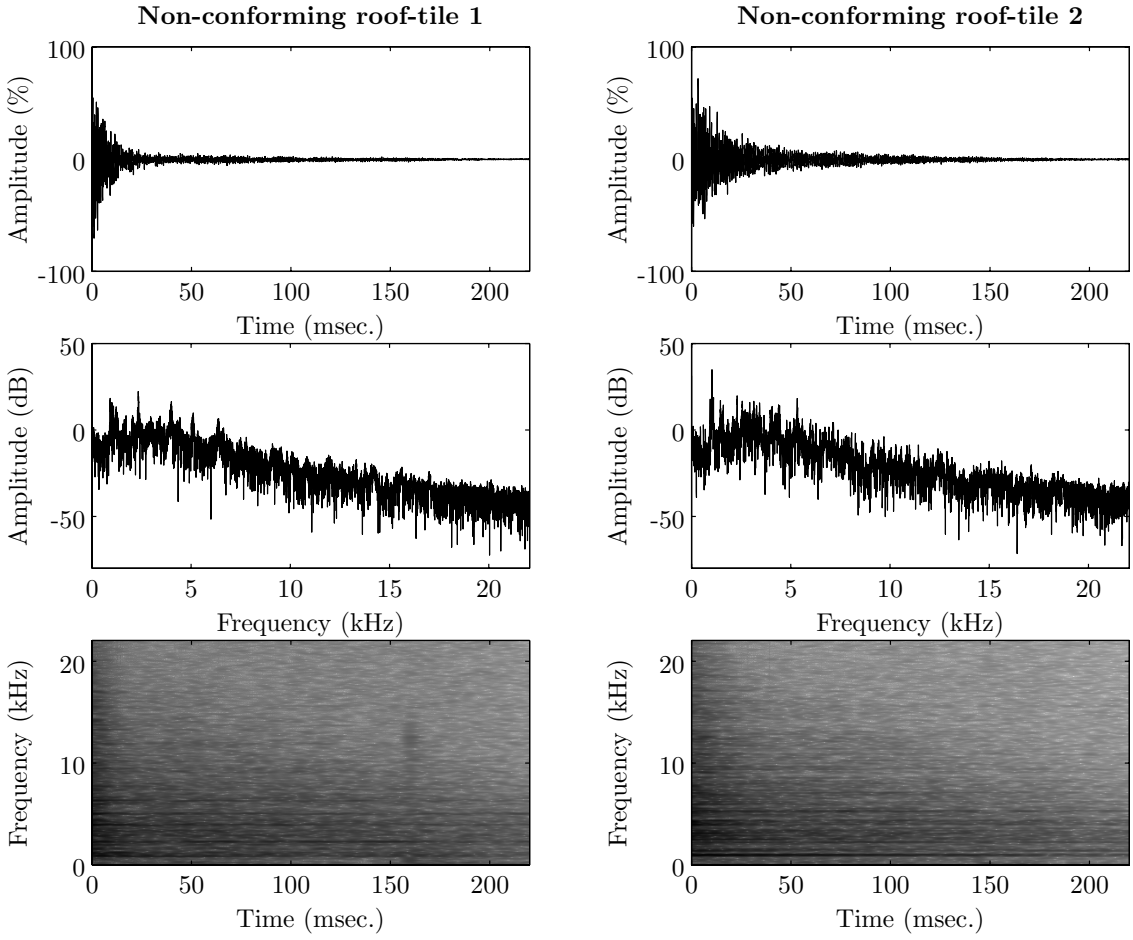


Figure 3.3: Time, frequency and spectrogram representations for the acoustic responses of two tiles of the *non-conforming* class.

Band	1	2	3	4
Frequency range (bins)	5–51	52–105	106–174	175–348
Frequency range (Hz)	258–2261	2261–4566	4566–7558	7558–15031

Table 3.2: Frequency partition scheme used for feature extraction. Note that the frequency indexes in bins go from 0 to 511.

3. The value of the energy contained in each band, for each of the 16 time segments, is computed. The expression for obtaining these values is

$$E_b(s) = 10\log_{10}\left\{\frac{1}{N_b}\sum_{k=\text{start}_b}^{\text{start}_b+N_b-1}[X_s(k)]^2\right\}, \tag{3.1}$$

where $b = 1, 2, 3, 4$ represents the frequency band index, start_b and N_b are the starting frequency bin and the number of bins contained in band b, respectively, $s = 1, 2, \ldots, 16$ is the time segment index, and $X_s(k)$ represents the FFT value at the k-th frequency bin, for time segment s.

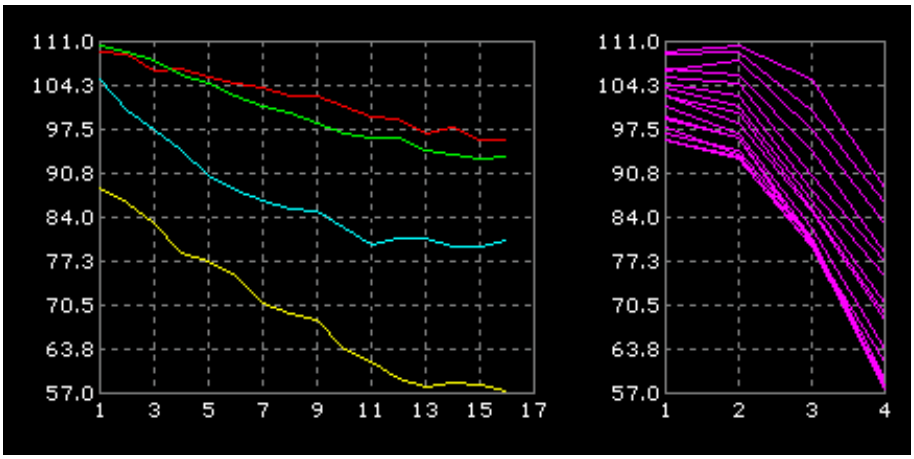
4. Six features ($\mathcal{F}_1$ to $\mathcal{F}_6$) are extracted which reflect the behaviour of the evolution of the energy values for each frequency band through time. These 6 features are categorized in terms of their relation to the energy evolution analysis parameters [6, 7] as follows:

   - *Persistence* – ($\mathcal{F}_1$ to $\mathcal{F}_4$)
   - *Stability* – ($\mathcal{F}_5$)
   - *Dispersion* – ($\mathcal{F}_6$)

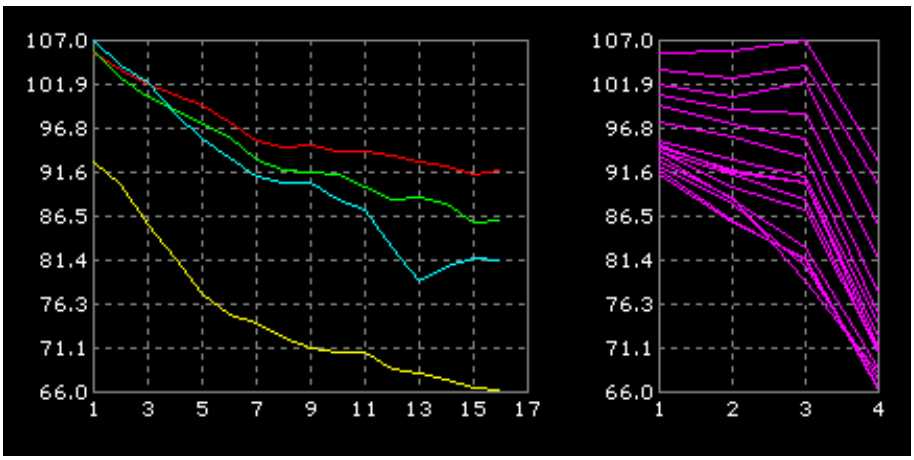
Figure 3.4 shows plots of the typical energy evolutions for the conforming and non-conforming classes, depicted in two ways. On the left side, the energy evolutions are plotted against the time segment indexes, for each of the 4 frequency bands. The first band is plotted in red color, the second in green, the third in cyan, and the fourth in yellow. On the right side, energy is plotted for each of the time segments, against the frequency bands. As can be seen on the plots, the two classes exhibit distinct energy evolution profiles. For instance, the energy evolves in a more "well-behaved" or regular fashion for the tiles of the conforming class than for the non-conforming ones.

Persistence (energy decay per band) - $\mathcal{F}_1$ to $\mathcal{F}_4$

Comparing the typical acoustic impulse responses of both C and NC kinds of pieces, one can see that the former kind has a longer, thus more persistent, response. If this comparison is made within each of the 4 frequency bands, the difference is even more evident [6]. The first



(a) Conforming class.



(b) Non-conforming class.

Figure 3.4: Plots of typical energy evolution profiles for classes C and NC.

4 features, one for each band, are extracted as

$$\mathcal{F}_b = \sum_{s=1}^{16} \left| \max[E_b(\cdot)] - E_b(s) \right|, \qquad b = 1, 2, 3, 4. \tag{3.2}$$

Stability (energy decay regularity) - $\mathcal{F}_5$

By analyzing the energy decay within each frequency band, it is seen that the logarithmic value of the energy evolves in an approximately linear way. However, for the NC class, this evolution is more unstable [6]. The fifth feature is obtained through the standard deviation of the energy differences between consecutive time segments of the captured signal, and is computed as

$$\mathcal{F}_5 = \frac{1}{4} \sum_{b=1}^{4} \sqrt{\frac{1}{15} \sum_{s=1}^{15} \left\{ \Delta_E^b(s) - \text{avg} \left[\Delta_E^b(\cdot) \right] \right\}^2}, \tag{3.3}$$

where

$$\Delta_E^j(i) = E_j(i) - E_j(i+1) \tag{3.4}$$

is the energy difference between the segments i and $i+1$, given the frequency band j.

Dispersion (spectral evolution profile) - $\mathcal{F}_6$

From a subjective analysis, it can be seen [6] that there are differences in the timbres of the two kinds of pieces, although they are not structured in an evident way. From here, another relevant feature can be extracted that evaluates the spectral balance of the response of the piece under analysis.

From the existing 4 frequency bands, the 3 most consistent ones are selected, which have been found to correspond to the 3 lower-index bands. From these bands, an average value of their energy for each time segment s, is computed:

$$A(s) = \frac{1}{3} \sum_{b=1}^{3} E_b(s), \qquad s = 1, 2, \ldots, 16. \tag{3.5}$$

To obtain $\mathcal{F}_6$, the Euclidean distances between all $E_b(s)$ and $A(s)$ are used, as can be seen in the expression

$$\mathcal{F}_6 = \frac{1}{16} \sum_{s=1}^{16} \sqrt{\sum_{b=1}^{4} \left[E_b(s) - A(s) \right]^2}. \tag{3.6}$$

The value of $\mathcal{F}_6$ from Equation 3.6 can be seen as a measure of the syntony or spectral openness through time.

3.2.1 Class separation

Figure 3.5 shows the class distribution obtained extracting the features described in Section 3.2 for a set of 17 C and 17 NC roof-tiles which have been previously classified by human experts. Because of the somewhat different results obtained within different strokes on the same piece [6], 10 strokes have been applied on each analyzed roof-tile, slightly varying the hit point on the pieces, which totals to 340 training samples. Assuming Gaussian

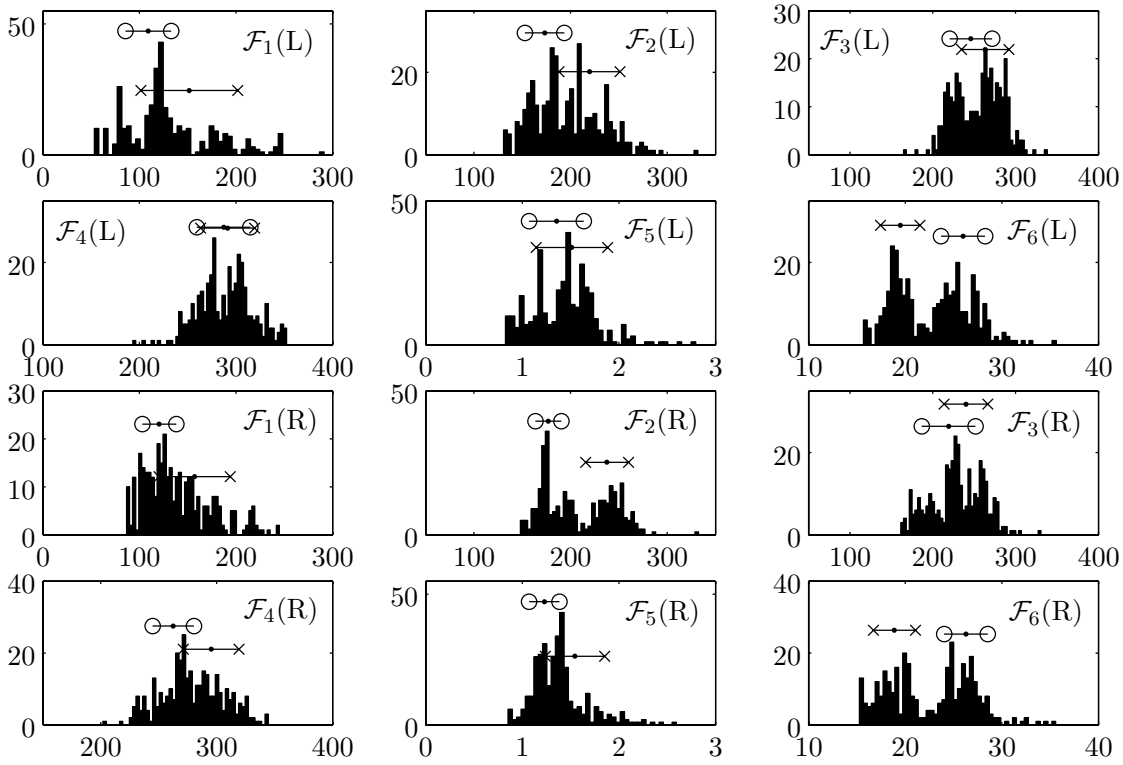


Figure 3.5: Histograms for the distributions of the two classes for each of the extracted features, showing the corresponding mean values ('**.**'), and the standard deviations ('$\circ$' and '$\times$' for classes C and NC, respectively).

functions for modeling the distributions of the C and NC classes by using the expression

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \tag{3.7}$$

where μ represents the mean or average value of each distribution and σ its standard deviation, the plots depicted in Figure 3.6 can be made that better show the separation of the classes with each extracted feature. The plots show that there are some features that provide excellent class separation, like $\mathcal{F}_2$ and $\mathcal{F}_6$, while others provide very low visible discrimination, as with the case of $\mathcal{F}_3$. Furthermore, it can be seen that the Gaussian model is a fairly good approximation to the distributions.

The results shown in the figures shows the advantages of using the two microphones for

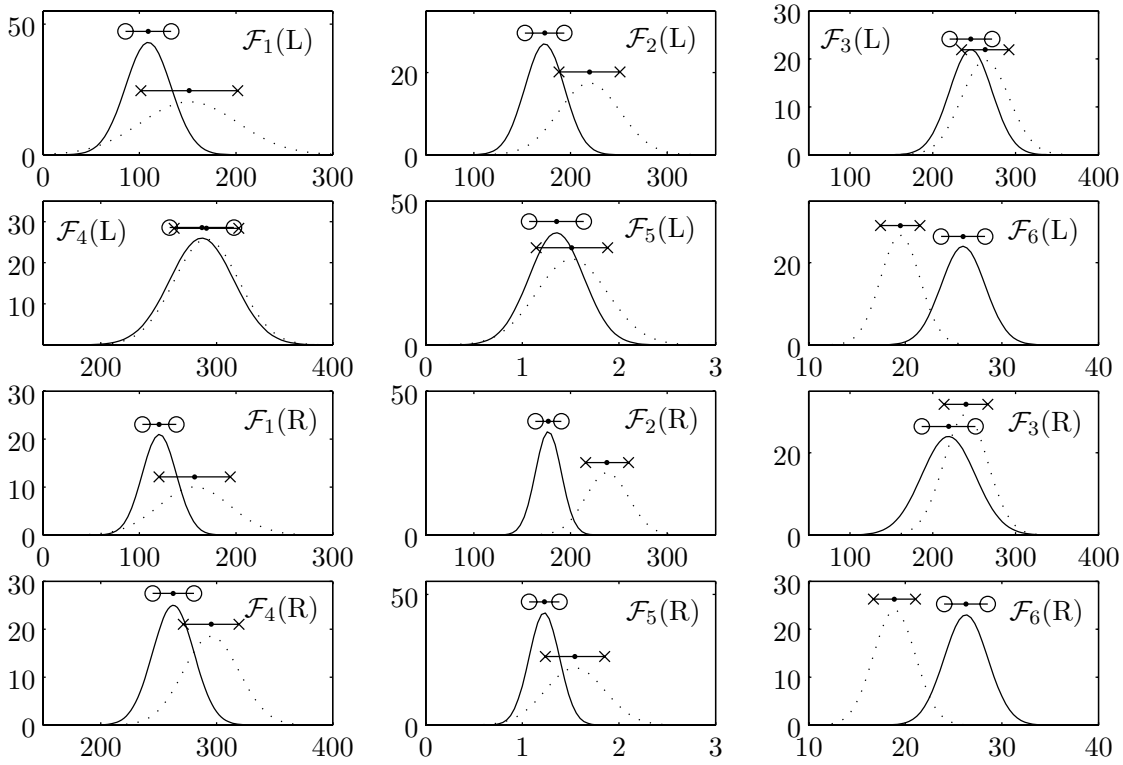


Figure 3.6: Gaussian model representations for the histograms in Figure 3.5, denoting the discriminating power given by each feature when separating classes C (solid line) and NC (dashed line).

capturing the acoustic signals. It seems that there are some acoustical properties in the signals that, when not captured by one of the microphones, are captured by the other [6]. This is evident for feature $\mathcal{F}_4$, where the left-channel distributions are much more overlapped than the right-channel ones.

A carefull observation of the plots of Figure 3.6 shows that for the class of non-conforming pieces, the data points generally appear in more sparse areas than for the conforming class. This is a somewhat expected result, since there is a high number of possible structural defects in the NC pieces that lead to different acoustic impulse response characteristics.

Another common way of analyzing the class separation power given by the features is through their mean and correlation matrices. The correlation matrix is a square matrix in which the tendency of each pair of features (x_i, x_j) varying in the same direction is measured. The covariance between features x_i and x_j is estimated as

$$c_{ij} = \frac{1}{N-1} \sum_{k=1}^{N} (x_i(k) - \mu_i)(x_j(k) - \mu_j), \tag{3.8}$$

where N is the number of patterns and μ_a is the mean value of feature a.

The covariance is related to the Pearson correlation matrix, whose coefficients can be obtained using Equation 3.9.

$$r_{ij} = \frac{1}{\sigma_i^2 \sigma_j^2 (N-1)} \sum_{k=1}^{N} (x_i(k) - \mu_i)(x_j(k) - \mu_j) = \frac{c_{ij}}{\sigma_i^2 \sigma_j^2}. \tag{3.9}$$

The resulting correlation matrix can be interpreted as a normalized version of the covariance.

Tables 3.3 to 3.8 show the mean, covariance and correlation matrices for classes C and NC, using the same data that produced the results of Figures 3.5 and 3.6.

$\mathcal{F}_1$ (L)	$\mathcal{F}_2$ (L)	$\mathcal{F}_3$ (L)	$\mathcal{F}_4$ (L)	$\mathcal{F}_5$ (L)	$\mathcal{F}_6$ (L)	$\mathcal{F}_1$ (R)	$\mathcal{F}_2$ (R)	$\mathcal{F}_3$ (R)	$\mathcal{F}_4$ (R)	$\mathcal{F}_5$ (R)	$\mathcal{F}_6$ (R)
109.17	173.19	245.91	287.16	1.35	25.99	120.62	177.04	219.06	262.55	1.23	26.29

Table 3.3: Mean values of the features for the *conforming* class.

From the tables, one can see that the features exhibit different mean values for the two classes. These different values, associated with tight covariances, permit a reliable class separation. Also, the covariance matrices show that the covariance values of the NC class are higher than the ones of class C. This somewhat correlates with the above graphical analysis.

The correlation matrices present high values for some pairs of features, mainly when comparing equivalent features extracted for the two audio channels. However, it is seen that these correlations generally present much lower values for class C than for class NC. In a way, this proves that capturing the same acoustic response from more than one spot helps to discriminate conforming from non-conforming pieces.

	$\mathcal{F}_1$ (L)	$\mathcal{F}_2$ (L)	$\mathcal{F}_3$ (L)	$\mathcal{F}_4$ (L)	$\mathcal{F}_5$ (L)	$\mathcal{F}_6$ (L)	$\mathcal{F}_1$ (R)	$\mathcal{F}_2$ (R)	$\mathcal{F}_3$ (R)	$\mathcal{F}_4$ (R)	$\mathcal{F}_5$ (R)	$\mathcal{F}_6$ (R)
$\mathcal{F}_1$ (L)	558.28	-140.29	88.60	172.59	5.06	-25.43	243.68	-47.58	-182.25	147.69	1.64	-16.31
$\mathcal{F}_2$ (L)	-140.29	418.89	99.21	-29.22	-2.31	10.73	-146.18	75.35	413.12	72.69	-0.13	1.61
$\mathcal{F}_3$ (L)	88.60	99.21	657.67	87.11	-0.73	-11.35	117.05	1.30	-148.08	59.05	0.22	-8.51
$\mathcal{F}_4$ (L)	172.59	-29.22	87.11	766.99	3.73	-14.03	126.85	116.26	-177.59	52.23	-0.54	-5.89
$\mathcal{F}_5$ (L)	5.06	-2.31	-0.73	3.73	0.08	-0.34	3.13	-0.24	-2.04	1.70	0.02	-0.16
$\mathcal{F}_6$ (L)	-25.43	10.73	-11.35	-14.03	-0.34	5.17	-19.82	-6.71	7.91	-8.82	-0.15	4.56
$\mathcal{F}_1$ (R)	243.68	-146.18	117.05	126.85	3.13	-19.82	311.44	-47.90	-141.73	89.90	-0.21	-7.95
$\mathcal{F}_2$ (R)	-47.58	75.35	1.30	116.26	-0.24	-6.71	-47.90	178.79	9.76	3.19	-0.11	-10.33
$\mathcal{F}_3$ (R)	-182.25	413.12	-148.08	-177.59	-2.04	7.91	-141.73	9.76	1048.55	92.01	-0.16	-4.96
$\mathcal{F}_4$ (R)	147.69	72.69	59.05	52.23	1.70	-8.82	89.90	3.19	92.01	318.51	0.65	-5.11
$\mathcal{F}_5$ (R)	1.64	-0.13	0.22	-0.54	0.02	-0.15	-0.21	-0.11	-0.16	0.65	0.02	-0.13
$\mathcal{F}_6$ (R)	-16.31	1.61	-8.51	-5.89	-0.16	4.56	-7.95	-10.33	-4.96	-5.11	-0.13	5.14

Table 3.4: Covariance matrix for the *conforming* class.

	$\mathcal{F}_1$ (L)	$\mathcal{F}_2$ (L)	$\mathcal{F}_3$ (L)	$\mathcal{F}_4$ (L)	$\mathcal{F}_5$ (L)	$\mathcal{F}_6$ (L)	$\mathcal{F}_1$ (R)	$\mathcal{F}_2$ (R)	$\mathcal{F}_3$ (R)	$\mathcal{F}_4$ (R)	$\mathcal{F}_5$ (R)	$\mathcal{F}_6$ (R)
$\mathcal{F}_1$ (L)	1.00	-0.29	0.15	0.26	0.76	-0.47	0.58	-0.15	-0.24	0.35	0.44	-0.30
$\mathcal{F}_2$ (L)	-0.29	1.00	0.19	-0.05	-0.40	0.23	-0.40	0.28	0.62	0.20	-0.04	0.03
$\mathcal{F}_3$ (L)	0.15	0.19	1.00	0.12	-0.10	-0.19	0.26	0.00	-0.18	0.13	0.05	-0.15
$\mathcal{F}_4$ (L)	0.26	-0.05	0.12	1.00	0.48	-0.22	0.26	0.31	-0.20	0.11	-0.12	-0.09
$\mathcal{F}_5$ (L)	0.76	-0.40	-0.10	0.48	1.00	-0.52	0.63	-0.06	-0.22	0.34	0.38	-0.25
$\mathcal{F}_6$ (L)	-0.47	0.23	-0.19	-0.22	-0.52	1.00	-0.49	-0.22	0.11	-0.22	-0.41	0.88
$\mathcal{F}_1$ (R)	0.58	-0.40	0.26	0.26	0.63	-0.49	1.00	-0.20	-0.25	0.29	-0.08	-0.20
$\mathcal{F}_2$ (R)	-0.15	0.28	0.00	0.31	-0.06	-0.22	-0.20	1.00	0.02	0.01	-0.05	-0.34
$\mathcal{F}_3$ (R)	-0.24	0.62	-0.18	-0.20	-0.22	0.11	-0.25	0.02	1.00	0.16	-0.03	-0.07
$\mathcal{F}_4$ (R)	0.35	0.20	0.13	0.11	0.34	-0.22	0.29	0.01	0.16	1.00	0.23	-0.13
$\mathcal{F}_5$ (R)	0.44	-0.04	0.05	-0.12	0.38	-0.41	-0.08	-0.05	-0.03	0.23	1.00	-0.38
$\mathcal{F}_6$ (R)	-0.30	0.03	-0.15	-0.09	-0.25	0.88	-0.20	-0.34	-0.07	-0.13	-0.38	1.00

Table 3.5: Correlation matrix for the *conforming* class.

$\mathcal{F}_1$ (L)	$\mathcal{F}_2$ (L)	$\mathcal{F}_3$ (L)	$\mathcal{F}_4$ (L)	$\mathcal{F}_5$ (L)	$\mathcal{F}_6$ (L)	$\mathcal{F}_1$ (R)	$\mathcal{F}_2$ (R)	$\mathcal{F}_3$ (R)	$\mathcal{F}_4$ (R)	$\mathcal{F}_5$ (R)	$\mathcal{F}_6$ (R)
151.73	219.63	263.47	291.26	1.51	19.49	157.11	237.84	239.95	295.29	1.55	18.89

Table 3.6: Mean values of the features for the *non-conforming* class.

	$\mathcal{F}_1$ (L)	$\mathcal{F}_2$ (L)	$\mathcal{F}_3$ (L)	$\mathcal{F}_4$ (L)	$\mathcal{F}_5$ (L)	$\mathcal{F}_6$ (L)	$\mathcal{F}_1$ (R)	$\mathcal{F}_2$ (R)	$\mathcal{F}_3$ (R)	$\mathcal{F}_4$ (R)	$\mathcal{F}_5$ (R)	$\mathcal{F}_6$ (R)
$\mathcal{F}_1$ (L)	2513.35	110.38	169.28	295.96	10.68	-65.40	1475.29	-94.45	-75.06	-244.47	8.57	-39.84
$\mathcal{F}_2$ (L)	110.38	999.26	25.59	298.75	6.17	10.03	113.76	235.42	385.61	250.34	4.32	14.17
$\mathcal{F}_3$ (L)	169.28	25.59	815.90	93.97	-0.15	-4.01	222.19	397.19	91.77	140.77	-0.50	-18.14
$\mathcal{F}_4$ (L)	295.96	298.75	93.97	778.15	-0.28	6.47	177.58	-1.48	462.66	303.59	-0.60	22.98
$\mathcal{F}_5$ (L)	10.68	6.17	-0.15	-0.28	0.14	-0.21	6.15	0.28	0.70	-0.02	0.10	-0.11
$\mathcal{F}_6$ (L)	-65.40	10.03	-4.01	6.47	-0.21	4.18	-30.46	-2.20	22.16	24.08	-0.17	3.50
$\mathcal{F}_1$ (R)	1475.29	113.76	222.19	177.58	6.15	-30.46	1363.64	-125.76	-150.50	25.33	4.78	-24.34
$\mathcal{F}_2$ (R)	-94.45	235.42	397.19	-1.48	0.28	-2.20	-125.76	491.92	162.17	115.59	1.26	-8.74
$\mathcal{F}_3$ (R)	-75.06	385.61	91.77	462.66	0.70	22.16	-150.50	162.17	697.62	279.15	0.56	31.34
$\mathcal{F}_4$ (R)	-244.47	250.34	140.77	303.59	-0.02	24.08	25.33	115.59	279.15	579.08	1.06	22.91
$\mathcal{F}_5$ (R)	8.57	4.32	-0.50	-0.60	0.10	-0.17	4.78	1.26	0.56	1.06	0.09	-0.10
$\mathcal{F}_6$ (R)	-39.84	14.17	-18.14	22.98	-0.11	3.50	-24.34	-8.74	31.34	22.91	-0.10	4.71

Table 3.7: Covariance matrix for the *non-conforming* class.

	$\mathcal{F}_1$ (L)	$\mathcal{F}_2$ (L)	$\mathcal{F}_3$ (L)	$\mathcal{F}_4$ (L)	$\mathcal{F}_5$ (L)	$\mathcal{F}_6$ (L)	$\mathcal{F}_1$ (R)	$\mathcal{F}_2$ (R)	$\mathcal{F}_3$ (R)	$\mathcal{F}_4$ (R)	$\mathcal{F}_5$ (R)	$\mathcal{F}_6$ (R)
$\mathcal{F}_1$ (L)	1.00	0.07	0.12	0.21	0.58	-0.64	0.80	-0.08	-0.06	-0.20	0.56	-0.37
$\mathcal{F}_2$ (L)	0.07	1.00	0.03	0.34	0.53	0.16	0.10	0.34	0.46	0.33	0.45	0.21
$\mathcal{F}_3$ (L)	0.12	0.03	1.00	0.12	-0.01	-0.07	0.21	0.63	0.12	0.20	-0.06	-0.29
$\mathcal{F}_4$ (L)	0.21	0.34	0.12	1.00	-0.03	0.11	0.17	-0.00	0.63	0.45	-0.07	0.38
$\mathcal{F}_5$ (L)	0.58	0.53	-0.01	-0.03	1.00	-0.27	0.45	0.03	0.07	-0.00	0.88	-0.14
$\mathcal{F}_6$ (L)	-0.64	0.16	-0.07	0.11	-0.27	1.00	-0.40	-0.05	0.41	0.49	-0.28	0.79
$\mathcal{F}_1$ (R)	0.80	0.10	0.21	0.17	0.45	-0.40	1.00	-0.15	-0.15	0.03	0.42	-0.30
$\mathcal{F}_2$ (R)	-0.08	0.34	0.63	-0.00	0.03	-0.05	-0.15	1.00	0.28	0.22	0.19	-0.18
$\mathcal{F}_3$ (R)	-0.06	0.46	0.12	0.63	0.07	0.41	-0.15	0.28	1.00	0.44	0.07	0.55
$\mathcal{F}_4$ (R)	-0.20	0.33	0.20	0.45	-0.00	0.49	0.03	0.22	0.44	1.00	0.14	0.44
$\mathcal{F}_5$ (R)	0.56	0.45	-0.06	-0.07	0.88	-0.28	0.42	0.19	0.07	0.14	1.00	-0.16
$\mathcal{F}_6$ (R)	-0.37	0.21	-0.29	0.38	-0.14	0.79	-0.30	-0.18	0.55	0.44	-0.16	1.00

Table 3.8: Correlation matrix for the *non-conforming* class.

3.3 Communication with the PC

After the DSP platform has finished computing all the 12 features of the captured acoustic impulse response (6 features for each of the two channels), it has to transmit them to the PC for the classification stage.

On the PC side, an application is running that continuously monitors the serial port. On the DSP side, all features are stacked into an internal buffer as they are being computed. After completing its calculations, the DSP platform forms a data package with additional control information and a two-way serial communication with the PC is started which obeys to a simple data link protocol.

The developed data link protocol uses some 8-bit `ASCII` control characters that are non-printable. Some of these characters are presented in Table 3.9 (a complete list containing all `ASCII` characters can be found at `http://www.dynamoo.com/technical/ascii.htm`).

Character	Value (hex.)	Description
`SYN`	`0x016`	“Synchronous idle”
`ENQ`	`0x005`	“Enquiry”
`SOH`	`0x001`	“Start of heading”
`STX`	`0x002`	“Start of text”
`ETX`	`0x003`	“End of text”
`DLE`	`0x010`	“Data link escape”
`ACK`	`0x006`	“Acknowledge”
`NAK`	`0x015`	“Negative acknowledge”

Table 3.9: List of non-printable `ASCII` characters used in the data link protocol.

3.3.1 Data link protocol – control sequences

DSP platform online verification

This message is continuously sent by the PC to the DSP platform, and is used to verify if the connection between them is working. The format of this message is as follows:

SYN	SYN	ENQ

After receiving this message, the DSP platform responds with:

SYN	SYN	ACK

The PC uses a timeout of 100 msec. after which, if no valid response is received, the DSP connection is given as "not ok".

Data reception confirmation

After the DSP has sent a data package containing the results of the feature extraction process for one of the two audio channels, it waits for a response coming from the PC, confirming that the data package has been correctly received. This is what the PC sends after determining that a valid data package has been received:

SYN	SYN	DLE	'channel'

The byte that refers to the audio channel can be one of the 'L' or 'R' characters, depending on the audio channel which the data package refers to. If the data package sent by the DSP has been malformed or corrupted, the folowing message is returned:

SYN	SYN	NAK

After receiving this message, the DSP platform will repeat the process of sending the same data package that has been corrupted until a positive confirmation is returned by the PC.

3.3.2 Data link protocol – data package

The data package is sent by the DSP and has the following format:

SYN	SYN	SOH	'nbH'	'nbL'	'channel'	'HBCC'	STX	'DATA'	ETX	'BCC'

In this message, the SOH character signals the beginning of the message header. This header is composed of the 'nbH' and 'nbL' characters which represent the most and the least significant bytes of the number of bytes contained in the 'DATA' field, respectively. The 'HBCC' byte serves as a protection of the header data and is obtained by a bitwise XOR operation between all header bytes. This header protection is essential so that the PC can exactly know the number of bytes contained in the remaining part of the package. The 'DATA' field is delimited by the STX and ETX characters. The last byte to be transmitted is

the ‘BCC’ character and is formed through a bitwise XOR operation between all data bytes (including the STX and ETX characters) to serve as a protection of the integrity of the bytes pertaining to the most important part of the communication, which are the results of feature computation stacked into the ‘DATA’ field.

3.4 Classification

The task of assessing the structural quality of the roof-tiles using the features extracted by the DSP platform is performed by the application that runs on the PC. The method uses an empirical PR algorithm that was developed in a former phase of this project [6, 7].

3.4.1 The dimensionality problem

It is well known that the performance of a classifier is in great part determined by the class discrimination power given by the features that are used. Naively, one may think that continuously adding more dimensions to the feature space will make the classifier more and more robust. However, for the class distribution functions within the feature space to remain well characterized, it is required that the number of samples needed in the training set increase exponentially with the feature space dimension [12, 13]. This is usually known as the “curse of dimensionality”. In fact, it is often observed in practice that the added features actually increase the probability of misclassification if the number of training samples is small relative to the number of features, i.e., if one is using a low *dimensionality ratio*[1]. This paradoxical behaviour is often termed as the peaking phenomenon [12].

To see how this dimensionality problem affects cluster distributions across the feature space, one can start by imagining a one-dimensional Gaussian distribution. From the common Statistics tables one can find that about 68.3% of the distributed values are found within a one standard deviation boundary around the mean. If, instead of the one-dimensional representation, a two-dimensional distribution is considered, then only about 46.6% of the data will lie within a σ-radius circle around the mean. For an M-dimensional representation, one has $(0.683)^M \times 100\%$ points within a σ-radius hypersphere [14]. This means that, for the 12-dimensional problem of this dissertation, only about 1% of the data for each class is in the neighbourhood of its mean value.

Principal component analysis

Principal component analysis (PCA) is a commonly used method for reducing the dimensionality of PR problems. It is based on applying an orthonormal transformation to the covariance matrix $\mathbf{C}$ so as the set of supposedly correlated features be transformed into a

[1] The dimensionality ratio of a given PR problem is given by dividing the number of available pattern samples by the dimension of the feature space [14].

set of new, uncorrelated features. Assuming an M-dimensional feature space, the process is started by obtaining the M eigenvalues of $\mathbf{C}$ through the equation

$$\det(\mathbf{C} - \lambda\mathbf{I}) = 0, \tag{3.10}$$

where $\mathbf{I}$ represents the identity matrix. Solving the above equation results in M solutions λ called the eigenvalues of $\mathbf{C}$. The eigenvectors $\mathbf{z}$ of $\mathbf{C}$ can now be computed by solving the homogeneous system of equations:

$$(\mathbf{C} - \lambda\mathbf{I})\mathbf{z} = 0, \tag{3.11}$$

Grouping the M eigenvectors of $\mathbf{C}$ as

$$\mathbf{Z} = [\mathbf{z}_1\ \mathbf{z}_2\ \ldots\ \mathbf{z}_M], \tag{3.12}$$

results in an orthonormal matrix [14]. A linear transformation can then be applied to the initial feature vectors $\mathbf{x}$ using the transpose of matrix $\mathbf{Z}$ [2]:

$$\mathbf{u} = \mathbf{Z}^T\mathbf{x}. \tag{3.13}$$

This orthonormal transformation is also known as the Karhunen-Loève transformation. The new covariance matrix will be diagonal (uncorrelated features). Furthermore, it can be seen that the values lying on its diagonal correspond to the eigenvalues of the initial matrix $\mathbf{C}$ [14], i.e., the eigenvalues of $\mathbf{C}$ are the new variances after the transformation.

A *principal component* is an eigenvector that corresponds to an eigenvalue representing significant variance of the whole data. Dimensionality reduction can be performed by retaining only the most significant eigenvectors in matrix $\mathbf{Z}$ and then applying the orthonormal transformation.

The eigenvalues of the covariance matrices for the conforming and non-conforming classes are presented in Tables 3.10 and 3.11, respectively. The cumulative contributions to the vari-

Component	1	2	3	4	5	6	7	8	9	10	11	12
Eigenvalue	3.78	2.00	1.44	1.33	1.13	0.86	0.58	0.35	0.23	0.17	0.08	0.04
% of total variance	31.53	16.68	12.00	11.05	9.45	7.18	4.86	2.93	1.91	1.40	0.66	0.33
Cumulative %	31.53	48.21	60.22	71.27	80.72	87.90	92.77	95.70	97.60	99.01	99.67	100.00

Table 3.10: Sorted eigenvalues of the covariance matrix for class C.

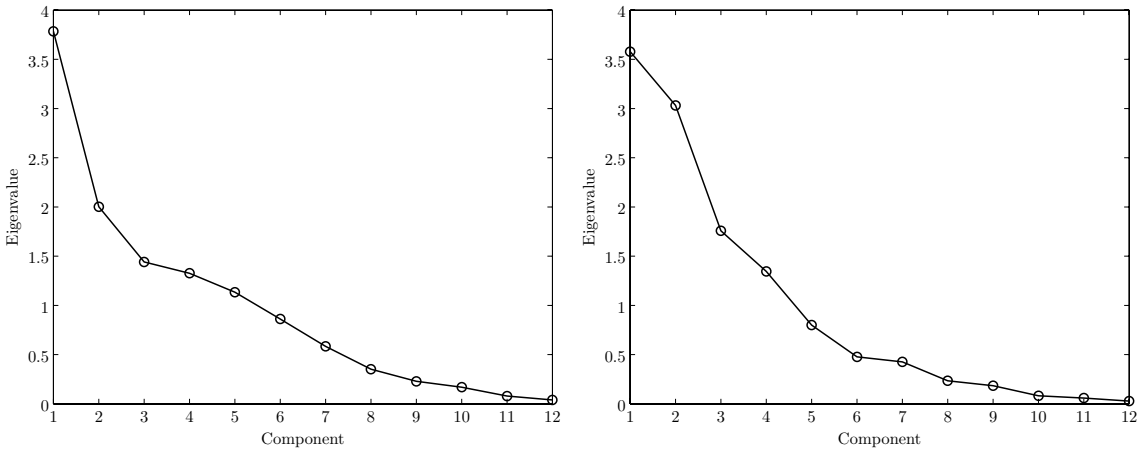
ance of the 12 components suggest that it would suffice to retain only the first 8 eigenvectors (obtained as linear transformations of the original features) in order to obtain at least 95% of the total variance for both classes. It is also worthy to observe the plots of the eigenvalues

[2] Further details about this transformation and its effects can be seen in [14, 15, 16].

Component	1	2	3	4	5	6	7	8	9	10	11	12
Eigenvalue	3.58	3.03	1.76	1.34	0.80	0.48	0.43	0.23	0.18	0.08	0.06	0.03
% of total variance	29.82	25.26	14.65	11.20	6.66	3.98	3.54	1.95	1.52	0.68	0.49	0.23
Cumulative %	29.82	55.08	69.73	80.93	87.60	91.57	95.12	97.08	98.60	99.28	99.77	100.00

Table 3.11: Sorted eigenvalues of the covariance matrix for class NC.

as shown in Figure 3.7. These plots are often termed *scree tests*. Based on the visible sparse



(a) Class C. (b) Class NC.

Figure 3.7: Plots of the eigenvalues for classes C and NC.

distribution of the variance with the number of components (especially for class C), it seems that the best solution is to discard only components 11 and 12. However, such a reduction is not very significant. Furthermore, by discarding components with very low contribution to the total variance, but that play important roles in pattern discrimination, one may compromise the classification performance. Moreover, there is no obvious semantic relation between the principal components and the original features, i.e., the obtained eigenvectors and their associated eigenvalues are not directly related to the set of initial features. Such semantic meaning would help arriving a better classification solution. Hence, instead of using the orthonormal transformation for dimensionality reduction, weighting factors are assigned to each of the 12 available features that measure their class discriminating capabilities. It is the responsability of the final decision function to use them and to automatically weight the set of features that will probably provide the best classification solution.

Assigning weighting factors to features

Taking the example in Figure 3.8, where two Gaussian distributions are shown that intersect each other, one can arrive a factor that measures the class separation for a given feature. The weighting factors are obtained through the computation of the intersection point of the

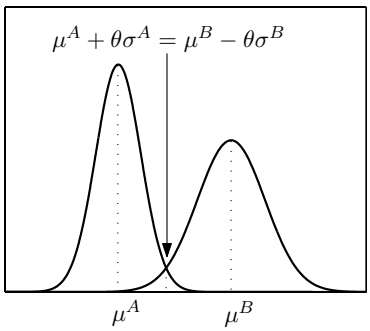


Figure 3.8: Obtaining the intersection point of two Gaussian probability density functions.

two class models, for each of the features. This intersection point (as shown in the figure) is given by

$$\theta_i = \frac{|\mu_i^B - \mu_i^A|}{\sigma_i^B + \sigma_i^A}, \tag{3.14}$$

where σ_i^α represents the standard deviation value of the distribution of class α, for feature i. Normalization is accomplished by using the expression

$$W_i = \frac{\theta_i}{\sum_{m=1}^{M} \theta_m}. \tag{3.15}$$

These weighting factors measure the class separation obtained in the training set, by each feature. Therefore, a feature having a low weighting factor will have a lower decision power, compared to the higher-weighted features. This provides the classification system with a higher robustness degree, since certain features that are not highly correlated to the structural quality of a certain type of roof-tiles (hence assigned to low weighting factors) may have an important role when assessing the quality of other kinds of pieces (assuming the same feature set). Furthermore, this strategy also helps to attenuate the dimensionality problem discussed above.

3.4.2 The Gaudio classifier

The *Gaudio* classifier is a parametric, supervised, statistical PR technique, which means that it uses statistical parameters from samples in a training set whose classes have been determined *a priori*. Furthermore, it assumes that the probability density function that best models the statistical distribution of each class within the feature space follows a Gaussian distribution [6, 7].

For the simple case of a two-class problem and given an M-dimensional feature space, the output response of the classifier C_x to the input vector $x = [x_1, x_2, \ldots, x_M]$ will be

$$C_x = \sum_{i=1}^{M} W_i \cdot \max\left[0, \min\left[1, \frac{x_i - \mu_i^A}{\mu_i^B - \mu_i^A}\right]\right], \tag{3.16}$$

where W_i is the weighting factor assigned to feature i, and μ_i^α represents the mean value of the distribution of class α for feature i. The output of the classifier is limited in the interval $[0, 1]$, which requires that $\sum_{i=1}^{M} W_i = 1$. This is guaranteed by the normalization procedure in Equation 3.15. The final classification decision is taken through the use of the following rule:

$$x \in \begin{cases} \text{class } A, \text{ if } C_x \leq 0.5 \\ \text{class } B, \text{ if } C_x > 0.5 \end{cases} . \tag{3.17}$$

Decision border

Figure 3.9 shows an example of a decision border obtained using the *Gaudio* classifying strategy, for the simple case of a two-feature problem. The data points in the figure were randomly generated using Gaussian functions. To be noticed is the non-linear decision

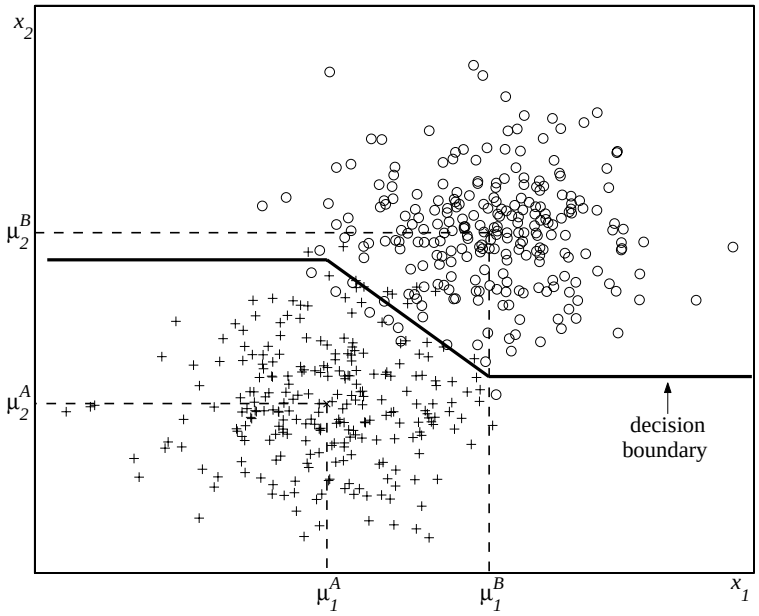


Figure 3.9: Example decision border obtained with the *Gaudio* classifier, for a simple two-feature problem.

function which, in this case, seems to provide a good approximation to the best possible solution.

3.4.3 Classification performance

The performance evaluation of this system has already been evaluated in [6] and compared to the implementation of other classifying schemes. However, the experiments with the *Gaudio* classifier were repeated in order to validate the new implementation. Table 3.12 summarizes the previously obtained results, where a performance comparison is shown between the *Fisher Linear Discriminant* function (FLD), the *k-Nearest Neighbours* classifier (kNN), the *Linear Vector Quantization* (LVQ) neural network approach, and the *Gaudio* classifier. Performance

is also compared for sets of 4, 6 and 12 features that are automatically selected by means of a *Forward Sequential Search* algorithm [17]. The *Forward Sequential Search* algorithm uses the

Technique	FLD			kNN ($k=1$)			kNN ($k=3$)			*Gaudio*		
Number of features	4	6	12	4	6	12	4	6	12	4	6	12
Performance (error %)	2.1%	1.8%	2.6%	1.5%	2.1%	1.8%	2.1%	2.1%	2.4%	1.8%	1.2%	1.5%
Technique	LVQ (2×4 cells)			LVQ (3×3 cells)			LVQ (4×5 cells)			LVQ (5×5 cells)		
Number of features	4	6	12	4	6	12	4	6	12	4	6	12
Performance (error %)	4.1%	2.4%	3.8%	3.8%	2.6%	4.4%	3.5%	2.4%	4.7%	4.7%	4.4%	4.7%

Table 3.12: Some results obtained for each of the experimented classifying techniques.

classifier itself to choose the set of features that heuristically provide the best classification performance[3]. The algorithm works as follows:

1. Two subsets are created that include the *selected* and *candidate* features, respectively. The selected feature set starts empty, while the candidate set includes all available features.

2. Features from the candidate set are evaluated one by one, and the one providing the best classification performance is moved to the set of selected features.

3. The previous step is performed until N features are found in the selected set.

The resulting set of selected features is a sub-optimal set [14]. Performance is measured in terms of an error percentage, computed using a cross-validation strategy. This validation strategy consists in partitioning the available set of 340 patterns into 34 subsets containing 10 arbitrarily selected patterns. The classifier parameters are extracted, at each iteration, from 33 of the available sets, and the remaining set is used for testing the performance of the classifier. The test set is rotated at the end of each iteration. When the 34 iterations are complete, the final performance measure is given by the average of the performances obtained at each of the iterations. This validation strategy provides a good estimate of the performance for a large number of samples as patterns are used independently for design and test. Also, it helps to further attenuate the problem of the reduced dimensionality ratio of this problem.

[3] A different classifier can also be used to select the feature set. If the same classifier is used both for feature selection and for the classification itself, the method follows a *wrapper* approach, otherwise the *filter* approach is used [18].

The results in Table 3.12 show that the *Gaudio* classifier outperforms the remaining tested algorithms, consistently showing error percentages below 2%. Also, it is seen that the best results are generally obtained when only 6 features are used[4]. However, as discussed in Sections 3.2.1 and 3.4.1, the use of all 12 extracted features permit the classification of pieces of different types, i.e., with different acoustic impulse response characteristics. Moreover, the general adaption of the sound acquisition system to different scenarios is made more robust by using the whole feature set. Also, the performance difference between the cases of 6 and 12 features is not very significant for the *Gaudio* classifier.

As mentioned above, the experiments were repeated for the *Gaudio* classifying scheme, in order to validate the new implementation and to arrive at new conclusions about the overall behaviour of this system. The same cross-validation strategy was used, again using the same set dimensions as above, but this time the test sets were forced to have exactly 5 patterns of each class. This somewhat avoids over-estimating performance in the cases when, coincidently, the classifier has been designed with a training set that better models the behaviour of the class with the larger pattern count in the test set. The results are shown in Table 3.13. Clearly, the performance results presented herein are in the proximity of the

Total number of test sets	34
Number of test sets with errors	2
Min/max error percentages	20%/40%
Average error percentage	**1.76%**

Table 3.13: Table of the results obtained with the *Gaudio* classifier.

ones obtained in the previous evaluation, and thus the new implementation is successfully validated. Also, the table shows that, from the 34 test sets used for evaluation, only 2 of them have presented classification errors. The total number of misclassifications is 6, and all of them correspond to non-conforming pieces classified as conforming. Obviously, it results in lower production costs if a conforming piece is classified as being non-conforming and thus be rejected. However, the higher dispersion of the non-conforming class along the feature space brings additional difficulties to the classifying process. The analysis of how the dispersion of non-conforming cluster could be made more tight (by re-analyzing the feature extraction process or the sound acquisition scheme, for example) is a subject that is beyond the scope of this thesis, and thus is pointed out as a topic for further developments.

3.5 Influence of noise

The performance of the system was also tested in a simulated industrial environment which, for this kind of industry, is rather hostile for an application based on sound analysis. Several

[4]Interestingly, it was shown [6] that the 6 selected features do not strictly correspond to the ones extracted from a single audio channel. Again, this proves that it is advantageous to compute the same features from both audio channels.

minutes of the noise present in the production unit were recorded and digitally stored on a DAT tape. The classifying system was tested in the laboratory, with the help of a DAT player. A speaker was directed to the system prototype and the sound volume set to 80 dB SPL, which is a typical value of the sound intensity in the factory [6]. The *Gaudio* classifier was evaluated using all the 12 available features. The training set comprises all the 340 samples taken from the 34 available tiles in the laboratory. All 34 pieces were then submitted for classification, one by one, with the noise source constantly held activated. At this acoustical noise intensity, the system showed no classification errors, as shown in Table 3.14.

The number of stroke undetections represents the number of times that the audio transient detector was unable to identify that a stroke has been applied to the piece. The same

Number of tested tiles	34
Number of classification errors	0
Number of stroke undetections	0

Table 3.14: Results obtained with the *Gaudio* classifier in the presence of industrial noise at 80 dB SPL.

process was repeated for 90 dB SPL, and the results are presented in Table 3.15. This time,

Number of tested tiles	34
Number of classification errors	2
Number of stroke undetections	6

Table 3.15: Results obtained with the *Gaudio* classifier in the presence of industrial noise at 90 dB SPL.

the process failed in classifying 2 of the 34 pieces, and missed the transient detection for 6 times. The 2 classification errors were observed only for tiles of class NC. The conforming class only presented 2 of the 6 stroke undetections. It is worth to compare the intensity of noise with the peak intensity of the strokes. Table 3.16 presents the minimum, maximum and mean values of the acoustic intensities produced by C and NC tiles when submitted to a mechanical stimulus. It can easily be seen that the noise intensities at 90 dB SPL are very

Intensity	Class C	Class NC
Minimum	101.0 dB SPL	87.4 dB SPL
Average	101.9 dB SPL	95.7 dB SPL
Maximum	102.7 dB SPL	101.1 dB SPL

Table 3.16: Minimum, average and maximum values of the sound intensity produced by strokes on C and NC tiles.

close to the intensities produced by the strokes. For instance, at 90 dB SPL, the captured impulse response of a NC roof-tile presents an SNR of about 5.7 dB SPL in average, which is a very low value. Nevertheless, the noise vulnerability of this system is considered to

be low, which in turn shows that the system is capable of operating correctly in real-world conditions.

There are a number of measures that can be carried out to further reduce the noise influence on the classification system. One of those measures is to protect the acoustic environment near the sound acquisition spot. Low cost materials like cork oak, for example, can be used for efficiently absorbing external acoustic noise. Another measure can be the use of directional microphones instead of the omni-radial ones that were used in this dissertation. Directional microphones can concentrate their capturing direction to the zone of the tiles, highly attenuating the acoustic emissions coming from external directions. However, directional microphones having good frequency response characteristics and sensibility are very expensive, which would imply a considerable increase on the cost of the prototype.

Chapter 4

The Image Analysis Module

Roof-tiles of different types (different geometric and texture characteristics) produce different sounds when submitted to a mechanical stimulus. Consequently, the extracted features from the audio signals (see Chapter 3) will also differ.

The development of this analysis module intends to make the classification process become more robust by accepting more than one type of piece (roof-tiles in this case) in the same production line. Its function is to visually recognize the type of roof-tile that is present so that the sound analysis module may use specific classification parameters for each kind of tile in the training set database.

The solution proposed by this dissertation uses a real-time digital image acquisition system composed of a color acquisition camera and a frame grabber card (see Section 2.5), and a digital image processing algorithm implemented on the PC. The 3 types of tiles in Figure 2.4 are used in this development stage.

4.1 Algorithm description

The image processing algorithm developed divides itself into 4 different stages: pre-processing, segmentation, feature extraction, and pattern matching (recognition). Because of the real-time operation constraint, each step of the image processing algorithm must be designed so as to be simple and effective, while not requiring excessive computational requirements.

4.1.1 Pre-processing

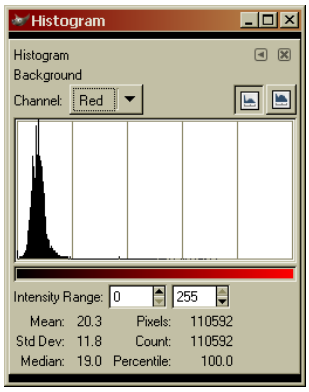
This processing stage intends to enhance the acquired color image, filter it for noise removal, and prepare it for the segmentation process.

Figure 4.1 shows an image of the background of the roof-tile scene that was acquired by the acquisition camera. The individual histograms for components R, G and B are also shown in the figure. Notice that there are some possibly disturbing objects in the figure (mainly on the left side) not covered by the black cloth on the floor, causing the histograms to exhibit high values above the zero intensity value.

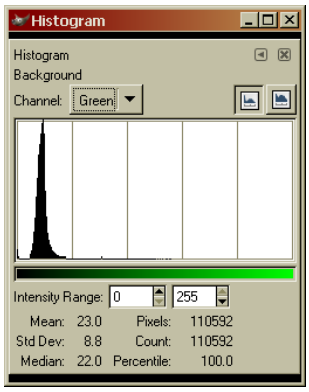
Inserting a spanish roof-tile in the scene transforms the RGB histogram levels as shown in Figure 4.2. Obviously, the presence of the roof-tile increases the amount of red components in the image. Compared to the red histogram in Figure 4.1b, one can see that an isolated section has appeared on the new histogram. This is due to the fact that the color tonality of

the roof-tile is mainly reddish. Also, one can see that a nearly isolated section has appeared on the histogram of the green component, also because of the presence of the tile. The blue component has almost disappeared from its histogram, which indicates that the tiles have a blue color component that is almost zero.

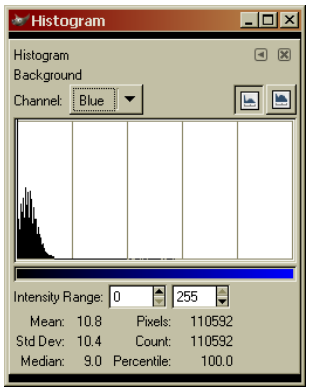
(a) Background image.



(b) Red component histogram.



(c) Green component histogram.



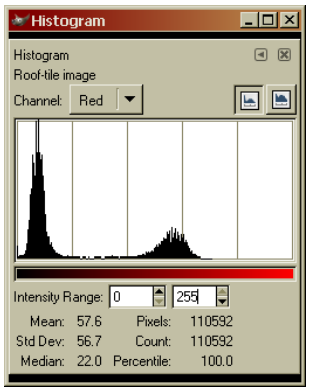
(d) Blue component histogram.

Figure 4.1: Typical background image obtained by the acquisition camera, showing the individual RGB component histograms.

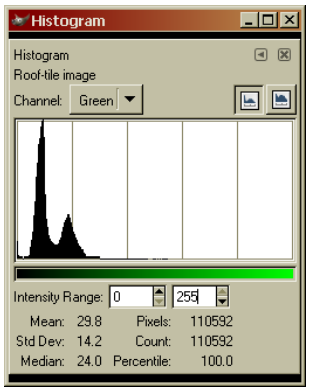
RGB to gray transformation

After comparing the RGB histograms of Figures 4.1 and 4.2, a transformation of the RGB color image to a gray (or intensity-level) image can now be performed. This step is important because, for the subsequent algorithmic steps, each pixel will have only one dimension (its intensity level) instead of the three dimensional RGB components, which will make them more simple and thus perform faster.

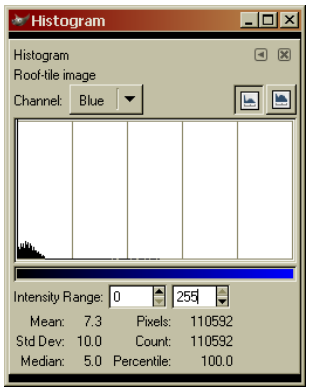
(a) Roof-tile image.



(b) Red component histogram.

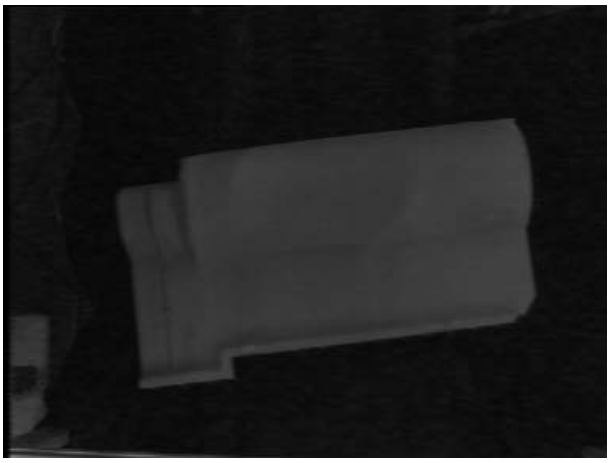


(c) Green component histogram.

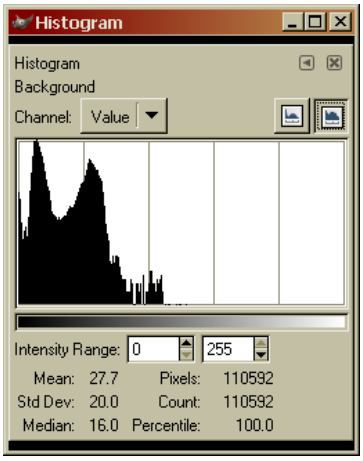


(d) Blue component histogram.

Figure 4.2: Typical roof-tile image obtained by the acquisition camera, showing the individual RGB component histograms.



(a) Resulting image.



(b) Logarithmic level histogram.

Figure 4.3: Image resulting from the direct color to gray transformation (a) and the corresponding logarithmic intensity histogram (b).

Figure 4.3 shows the result of the direct RGB to gray transformation using the expression

$$P(x,y) = \frac{R(x,y) + G(x,y) + B(x,y)}{3}, \tag{4.1}$$

where $P(x,y)$ represents the new value of intensity of pixel with coordinates (x,y). The coordinate axes directions are assumed throughout this dissertation as shown in Figure 4.4. The logarithmic intensity histogram was chosen so as to better evaluate the discrimination

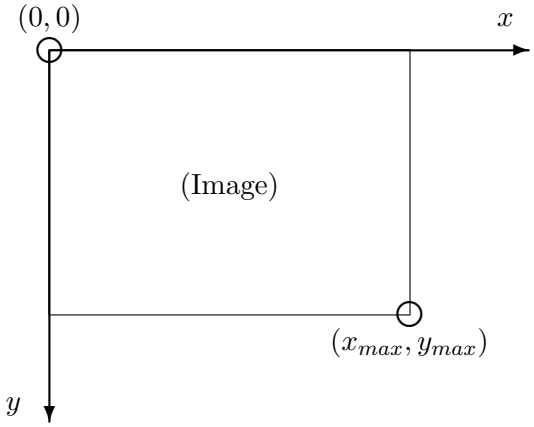


Figure 4.4: Coordinate axes directions.

between the two main components (segments) of the scene: the root-tile and the background (plus other disturbing objects). Clearly, there are two main peaks in the histogram, denoting the separation between the tile and the rest of the scene. However, by examining the red component histograms of Figures 4.1 and 4.2, the separation between the tile and the rest of the scene is more evident. Transforming the RGB color image into a gray image by simply using the expression

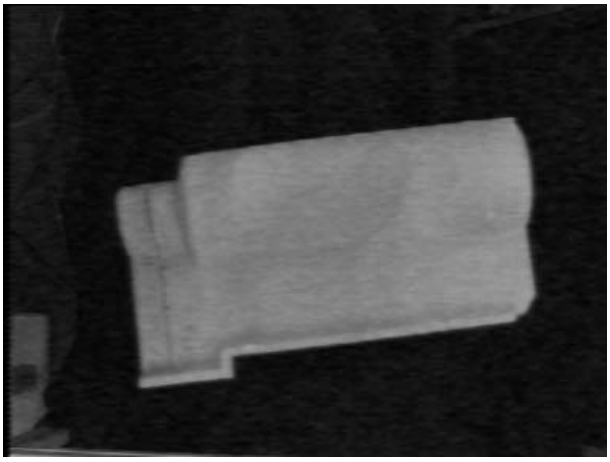
$$P(x,y) = R(x,y), \tag{4.2}$$

leads to the results shown in Figure 4.5. The separation of the two segments of the scene is still evident, perhaps even more pronounced. However, the disturbing objects in the scene are still evident. If, for example, one object crosses the tile boundary in the image plane, and if that object contains enough red component (whitish objects have high values for the three RGB components, for example), the segmentation process may be compromised.

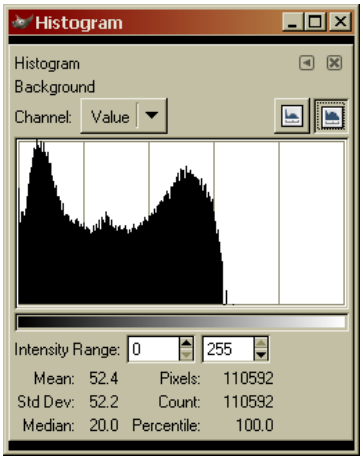
A better solution has been obtained that preserves the separation between the tile and the rest of the scene while attenuating the interference of other objects. The basic principle is to make the green and blue color components be attenuated in favour of the red component. The empirically derived expression is

$$P(x,y) = \frac{2R(x,y) - G(x,y) - B(x,y)}{3}. \tag{4.3}$$

The result obtained with this expression is show in Figure 4.6. This result shows that not

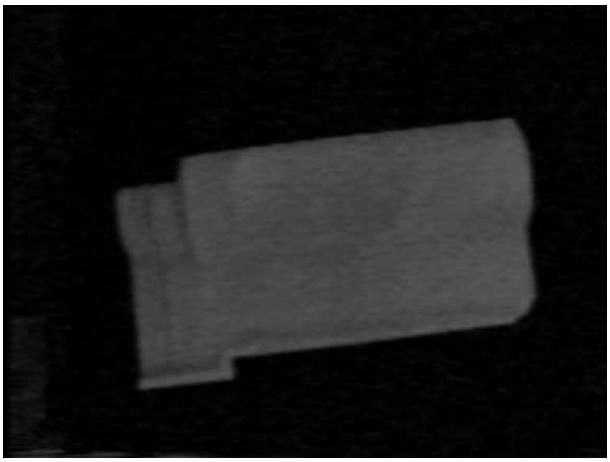


(a) Resulting image.

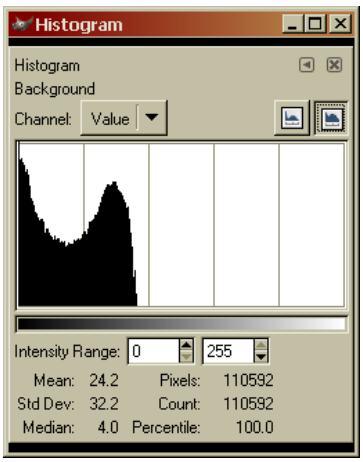


(b) Logarithmic level histogram.

Figure 4.5: Image resulting from the color to gray transformation by direct mapping of the red color component (a) and the corresponding logarithmic intensity histogram (b).



(a) Resulting image.



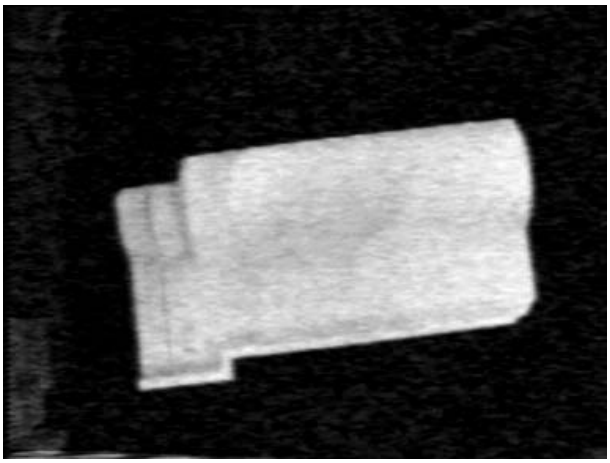
(b) Logarithmic level histogram.

Figure 4.6: Image resulting from the color to gray transformation by using Equation 4.3 (a) and the corresponding logarithmic intensity histogram (b).

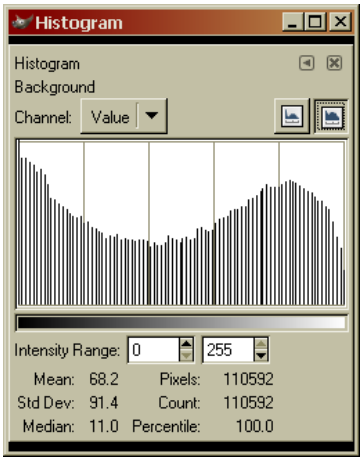
only the disturbing objects have almost disappeared, but also the separation between the tile and the rest of the scene has become more evident in the histogram. By analyzing Equation 4.3, one can see that only reddish disturbing objects can, in fact, compromise the segmentation process if they accidentally cross the roof-tile boundary in the image plane. While not completely eliminating the problem, this solution at least reduces its probability of occurrence.

Histogram stretching

The operation of histogram stretching is often termed as the auto-contrast operation by some commercial image processing computer software. Its aim is to interpolate the histogram so as to distribute its data across all the intensity level range. This processing stage is essential herein because it helps to normalize the histograms of the roof-tile images, without the need of a specially dedicated scene lighting strategy. Figure 4.7 shows the result of the auto-contrast operation performed on the image in Figure 4.6a.



(a) Auto-contrasted image.



(b) Logarithmic level histogram.

Figure 4.7: Image resulting from the auto-contrast operation applied to the image in Figure 4.6a (a) and the corresponding logarithmic intensity histogram (b).

Median filtering

Median filters are a class of nonlinear order-statistics spatial filters that are very popular because, for certain types of random noise, they provide excellent noise reduction capabilities, with considerably less *blurring* than linear smoothing filters of similar size [19].

Spatial filtering works as follows:

- A filter mask is created that traverses all points in an image. This filter mask is also often termed as *filter*, *kernel*, *template* or *window* in common literature.

- At each point $P(x, y)$ of the image, the response of the filter is given by a predefined relationship between all points under the mask.
- For linear spatial filtering, the response is given by a sum of products of the filter coefficients and the corresponding image pixels in the area spanned by the filter mask. The value of the pixel in the center of the mask is replaced by the computed filter response.
- For order-statistics filtering, the response is based on ordering (ranking) the pixels contained in the subimage area encompassed by the filter mask, and then the center pixel is replaced by the ranking result.

Figure 4.8 illustrates the mechanics of this process, showing a 3×3 mask and the subimage area under it.

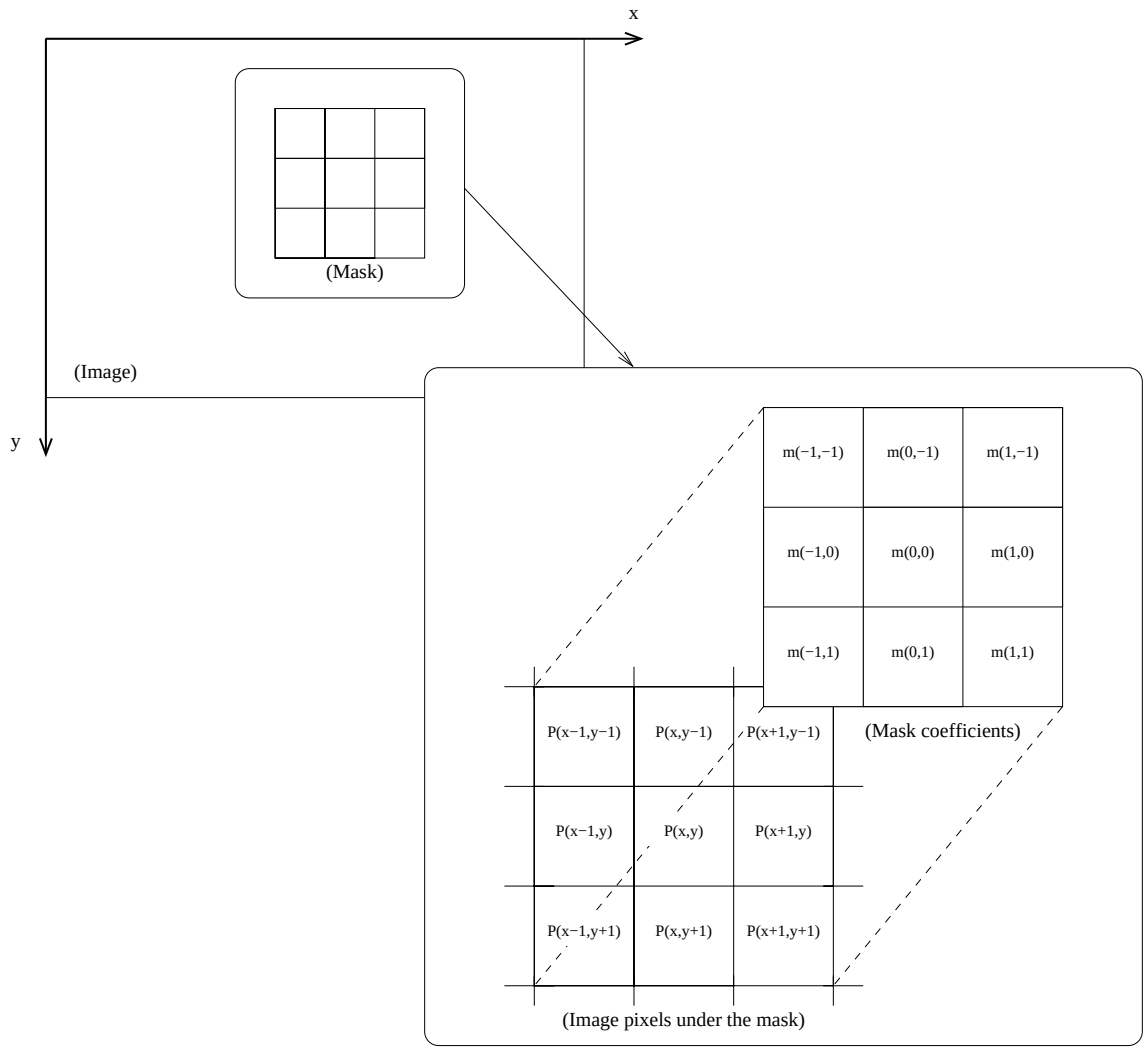


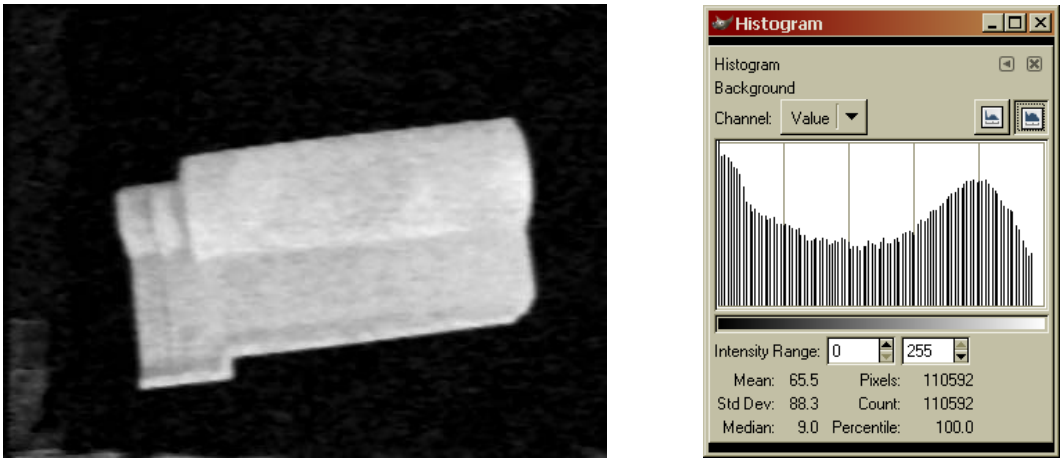
Figure 4.8: Illustration of the mechanics of spatial filtering.

The size of the masks is usually an odd number (3×3, 5×5, and so forth) so that its

center position coincides exactly with a pixel of the underlying image.

The median filter, as its name implies, replaces the central pixel value with the median of the intensity levels in its neighbourhood (the original value of the central pixel is also included in the computation of the median). This operation causes pixels whose intensity levels lie outside the range of the rest of the pixels contained within the mask to be rejected. This is why this class of filters is particularly effective in the presence of impulsive noise (also called *salt and pepper* noise) [19].

Figure 4.9 shows the result of the median filtering operation applied to the image in Figure 4.7a, and its corresponding histogram. The size of the filter mask is 3×3.



(a) Median-filtered image.

(b) Logarithmic level histogram.

Figure 4.9: Result of a 3×3 median filtering operation applied to the image in Figure 4.7a (a) and the corresponding logarithmic intensity histogram (b).

Median filters are computationally intensive, even for small-sized masks. The most immediate way of computing the median of a one-dimensional array is to first sort it (either in ascending or descending order), and then take the value at the middle index of the resulting array. For performing a 3×3 median filter in an image with $384 \times 288 = 110592$ pixels would require 109252 sorting operations (excluding the pixels at the boundary of the image) on 9-element arrays. Even with efficient sorting algorithm implementations, this would require very expensive computational resources.

There are other kinds of median computation algorithms that do not pre-sort the input array. In fact, only the median value is needed, and whether the array is pre-sorted or not is of little or no concern.

Nicolas Devillard [20] wrote an article about the fast implementation of median search algorithms. In this article, a comparison is made between various popular algorithms, both with and without pre-sort. Also, the author presents code for various implementations, written in the C language. Apparently, there is an optimized algorithm for dealing with the cases where the filter mask size is small. The C language code of these routines is presented

in the article for arrays with 3, 5, 7, 9 and 25 elements. Below is a slightly changed version of the code for the 9-element case, which corresponds to the 3×3 median filter mask.

```
typedef Byte pixelvalue;

#define ELEM_SWAP(a,b) { pixelvalue temp = (a); (a) = (b); (b) = temp; }
#define PIX_SORT(a,b) { if ((a) > (b)) ELEM_SWAP((a),(b)); }

pixelvalue opt_med9(pixelvalue * arr)
{
  /* 1st stage */
  PIX_SORT(arr[1], arr[2]); PIX_SORT(arr[4], arr[5]); PIX_SORT(arr[7], arr[8]);
  /* 2nd stage */
  PIX_SORT(arr[0], arr[1]); PIX_SORT(arr[3], arr[4]); PIX_SORT(arr[6], arr[7]);
  /* 3rd stage */
  PIX_SORT(arr[1], arr[2]); PIX_SORT(arr[4], arr[5]); PIX_SORT(arr[7], arr[8]);
  /* 4th stage */
  PIX_SORT(arr[0], arr[3]); PIX_SORT(arr[5], arr[8]); PIX_SORT(arr[4], arr[7]);
  /* 5th stage */
  PIX_SORT(arr[3], arr[6]); PIX_SORT(arr[1], arr[4]); PIX_SORT(arr[2], arr[5]);
  /* 6th stage */
  PIX_SORT(arr[4], arr[7]);
  /* 7th stage */
  PIX_SORT(arr[4], arr[2]);
  /* 8th stage */
  PIX_SORT(arr[6], arr[4]);
  /* 9th stage */
  PIX_SORT(arr[4], arr[2]);

  return(arr[4]);
}
```

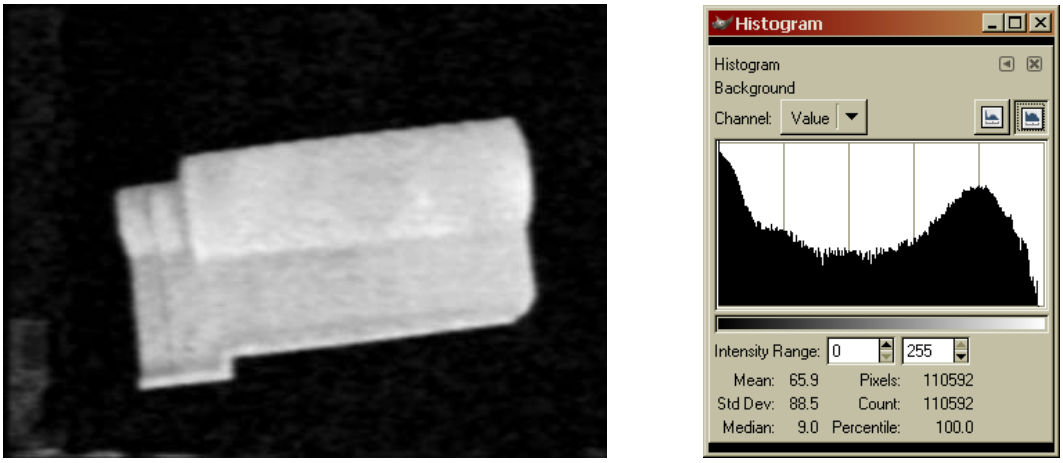
The algorithm shown above does not sort the input array in an ordered way, but guarantees that its median value is available at its the center position, at the end of the 9 stages. As mentioned in [20], in theory, there is no faster way of computing the median value of a 9-element array, without further assumptions on the data contained in the array.

Average filtering

Averaging is linear spatial filtering technique for smoothing images and that helps to attenuate noise. Each pixel of the resulting "averaged" image will be given by the average or mean value of the pixels in its neighbourhood. The mask for computing a 3×3 averaging filter is

$$\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

Figure 4.10 shows the result of an averaging filter using the above mask, applied to the image in Figure 4.9a. As can be seen, the result is a softened (blurred) image. Notice that



(a) Average-filtered image. (b) Logarithmic level histogram.

Figure 4.10: Result of a 3×3 averaging filtering operation applied to the image in Figure 4.9a (a) and the corresponding logarithmic intensity histogram (b).

the histogram data is now distributed across nearly all its intensity level axis. Also, the histogram valley that separates the roof-tile portion from the rest of the scene is now more pronounced, which will favour the segmentation process.

4.1.2 Segmentation

Image binarization

This stage of the image processing algorithm aims the binarization of the pre-processed image. This binarization process will try to separate the tile from the rest of the image components. This is done by applying a threshold T to the histogram data. In this case, all histogram data above this threshold is considered to be as part of the roof-tile body, while the remaining data will be considered as being part of the background. The threshold value T can be a fixed value or it can be obtained by using an algorithm that automatically computes it using the histogram data, for instance.

The thresholding strategy used for this problem uses a heuristic approach found in [19]. After visually inspecting the histograms of the images resulting from the pre-processing stage, the following algorithm can be applied to obtain T automatically.

1. Select an initial estimate for T.

2. Partition the histogram into two parts. The group of pixels with intensity levels above T is labeled G_1, while the pixels with levels below or equal to T are grouped into G_2.

3. Compute the average intensity values μ_1 and μ_2 for the pixels in regions G_1 and G_2, respectively.

4. Compute a new threshold value using

$$T = \frac{\mu_1 + \mu_2}{2}.$$

5. Repeat steps 2 through 4 until the absolute value of the difference in T between consecutive iterations is smaller than a predefined parameter T_0.

This algorithm is of very little computational complexity since all processing is done using the histogram data, i.e., not recurring to the individual pixel data in the image.

After observing various histograms resulting from the pre-processing stage, and because of the histogram stretching operation, the initial value for T was chosen to be 127, i.e., the center position of the histogram. The value chosen for T_0 is zero. There is no problem in using $T_0 = 0$ because the absolute difference between T for successive iterations is performed using only integer operations.

Figure 4.11: Result of the image binarization process applied to the image of Figure 4.10a. The automatically obtained value for threshold T was 68.

The result of this binarization process, applied to the image in Figure 4.10 is shown in Figure 4.11. The image pixels have now only two possible values: 0 or 255 (black and white, respectively). As can be seen, the binarization process successively separated the tile body from the rest of the image. Still, there are some small points in the lower-left corner of the image that need to be removed.

Isolation of the roof-tile

The effective isolation of the roof-tile in the binary image is performed using a region-based segmentation process and some assumptions on the image data. This process is based on a region identification and labeling strategy, and works as follows:

1. All pixels that lie on the image boundary are set to zero.
2. Every pixel on the image (excluding the image boundary pixels) is visited one by one, from the top-left corner to the bottom-right corner, moving on the left-right direction in each row. For each pixel `P` with non-zero value that is found, an analysis is performed on the neighbour pixels `A`, `B`, `C` and `D` that have been already visited:

A	B	C
D	P	

   (a) If all pixels `A`, `B`, `C` and `D` are zero, a new label is assigned to pixel `P`. Otherwise, the first of `A`, `B`, `C` and `D` to be different from zero will have its label copied to `P`.

   (b) If there are more than one different label in pixels `A`, `B`, `C` and `D`, one (or more) entry is added to a list of label equivalences (note that, at the end of this step, a contiguous object in the image may have pixels with different labels).

3. After all pixels on the image have been visited, the list of label equivalences is processed so as to assign a unique label to each group of equivalent labels.
4. A second visit is performed to the pixels on the image, this time to assign them their new labels.
5. The number of pixels pertaining to each label is counted. The tile region is then identified as being the one whose label has the largest pixel count[1]. Regions with less pixel count are eliminated from the image.

Figure 4.12 shows the result of performing this algorithm to the image of Figure 4.11. Clearly, this approach has been successful in eliminating the small, disturbing objects in the original image. Figure 4.13 shows the result of applying a bitwise *and* operation between the gray level image of Figure 4.10a and the image in Figure 4.12, showing that the tile region has effectively and correctly been discriminated.

4.1.3 Feature extraction

After the roof-tile body has been completely separated from the rest of the scene, the feature extraction process begins. This stage aims at selecting which features (characteristics) in

[1]The assumption that the tile body is the largest object present in the binary image makes sense because there is not much space left in the image area for contiguous additional objects to appear.

Figure 4.12: Result of the tile isolation process applied to the image of Figure 4.11.

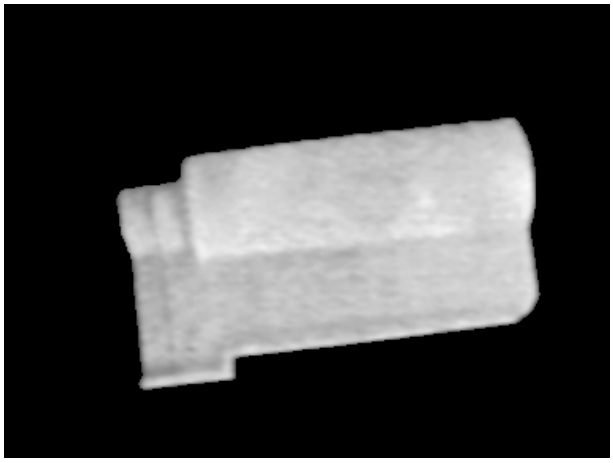


Figure 4.13: Result of a bitwise *and* operation between the images of Figures 4.10a and 4.12.

the roof-tile image are extracted so that the type of tile is recognized. These features must be carefully chosen because the robustness of the recognition stage highly depends on their discrimination power.

Carefully inspecting the roof-tile images of Figures 2.4 as well as the ones obtained in the pre-processing and segmentation algorithm stages, two ways for feasible analysis of the visual discriminating characteristics of the tiles quickly arise: by using their texture (relief) or by using the contour defined by their borders. Clearly, these are characteristics that easily enable instantaneous discrimination between the 3 types of tiles present in the laboratory (at least for the human eyes). Tile rotation invariance is an important requirement in feature extraction because it is seen that, in common roof-tile production lines, tiles do appear in somewhat arbitrary directions on the horizontal plane.

Texture analysis

Texture analysis can be performed using the variations of luminosity emitted by the tiles across their body. Taking these variations in the vertical axis of the images seems to be the most obvious solution. Taking the image resulting from the bitwise *and* operation shown in Figure 4.13, the following operation can be performed:

1. Identify the horizontal rectangular box that best fits the tile region in the image. This can be done by extracting the coordinate values of the pixels that are at the most top, the most bottom, the most left and at the most right part of the tile.

2. Take the pixel values that lie on the vertical center line in the tile box, from the top to the bottom of the rectangle, and form an array in memory with these values.

3. Do the same with the N vertical lines to the left and right of the center line and add the pixels values to the array, at the corresponding left and right columns. The final array is an extracted portion of the image, containing the central vertical part of the tile.

Figure 4.14 shows an illustration of this process, clearly highlighting the 4 pixels that define the best-fit rectangular box and the central vertical section above mentioned.

Figure 4.15 shows a 3D plot of the array obtained by the above described process applied to a spanish roof-tile sample image, using $N = 15$. This plot provides a better visual perception of the texture properties of the image of the roof-tile as captured by the acquisition camera, than the image itself. The same process was repeated for the french (Figure 4.16) and barrel (Figure 4.17) roof-tiles for comparison purposes.

Clearly, it is seen that there are notorious differences in the image textures between the different roof-tiles. However, after examining different vertical texture profiles taken from different images of the same roof-tiles, three weaknesses of this process have been found that compromise the robustness of feature extraction, namely, the heavy dependence on the illumination conditions of the scene, the rotation variance problem, and noise vulnerability.

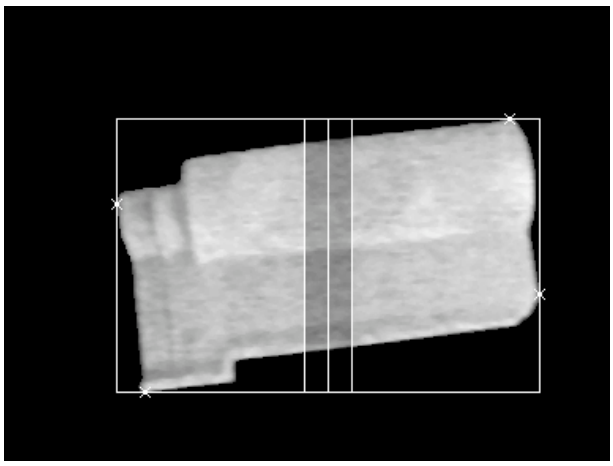


Figure 4.14: Illustration of the process for obtaining the rectangular box that best fits the tile region, highlighting the center vertical line of the rectangle and the region delimited by the N neighbour vertical lines.

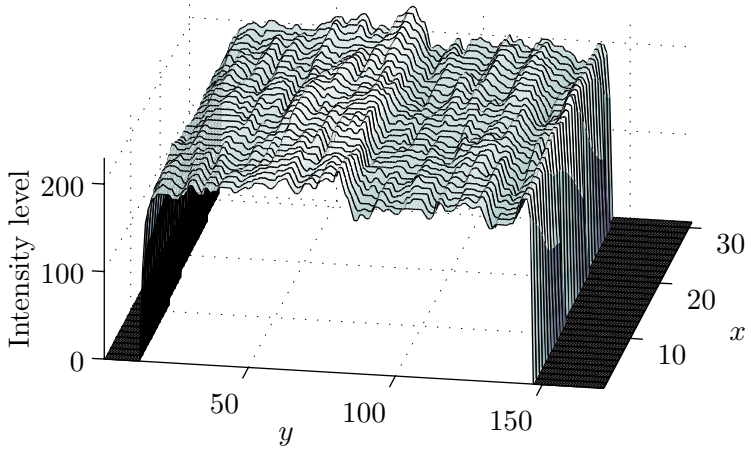
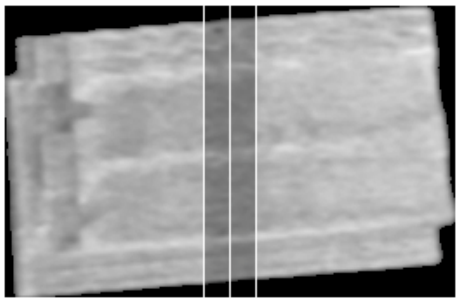
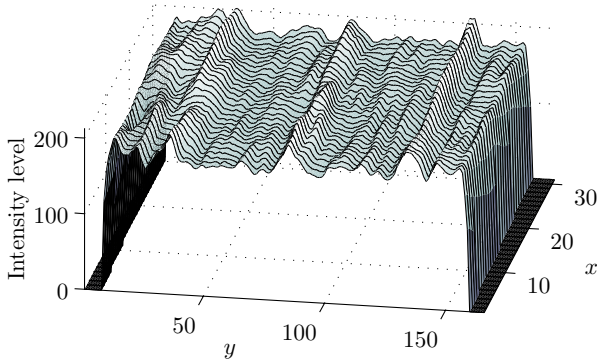


Figure 4.15: Three dimensional plot of the 31 vertical center lines of the image of a spanish roof-tile that better define its texture properties.

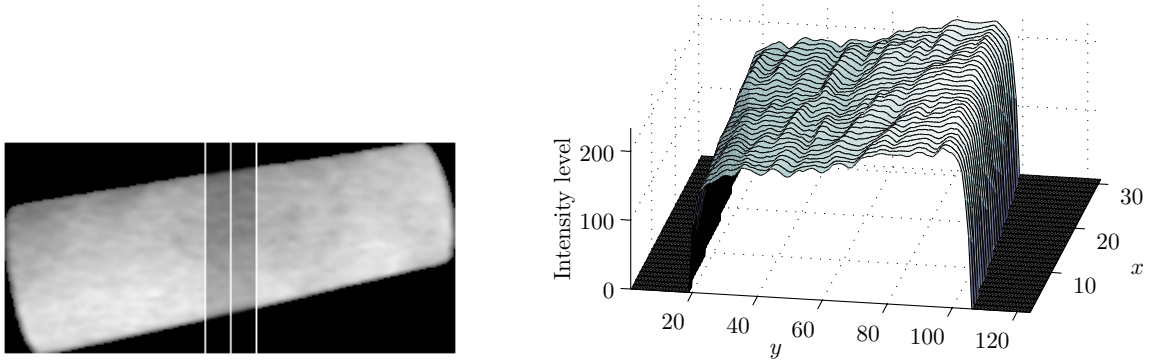


(a) Segmented french roof-tile image.



(b) 3D plot of the 31 vertical center lines of the image.

Figure 4.16: Extraction of the vertical texture characteristics applied to a french tile image.



(a) Segmented barrel roof-tile image.

(b) 3D plot of the 31 vertical center lines of the image.

Figure 4.17: Extraction of the vertical texture characteristics applied to a barrel tile image.

The tile rotation dependence can be solved by rotating the image so that the tiles always appear on an horizontal position. For this to be done, an estimate of the rotation angle of the tile in the image is needed. An easy way to obtain this estimate is to use the pixel coordinates of some points that lie on the top boundary line of the tile. A simple linear regression algorithm [21] would then provide the equation of the line that best fits the pixel locations. The slope of this line would be the final angle estimate.

The problem of the dependence on lighting conditions (shadows and reflexions) could only easily be solved by developing a dedicated scene illumination apparatus, which would imply the cost of re-designing the laboratory prototype.

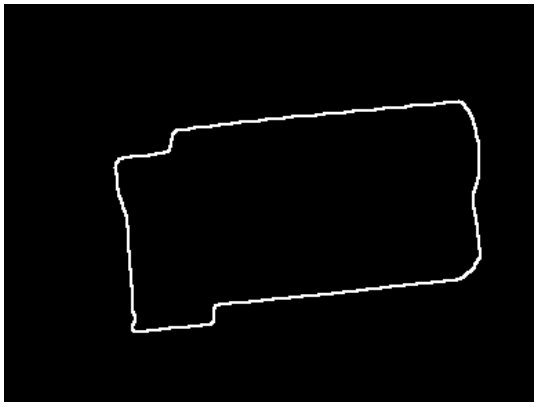
Noise vulnerability can be attenuated by averaging the lines of the extracted vertical texture profile along the x axis, resulting in a single texture profile line. Note that noise can also easily create perturbations in obtaining the estimate for angle of rotation.

Practical experiments have shown that, despite the strategies presented for attenuating these problems, the task of discriminating the spanish from the french roof-tiles is not an easy task. Searching for alternate feature extraction processes seems the most obvious way to follow.

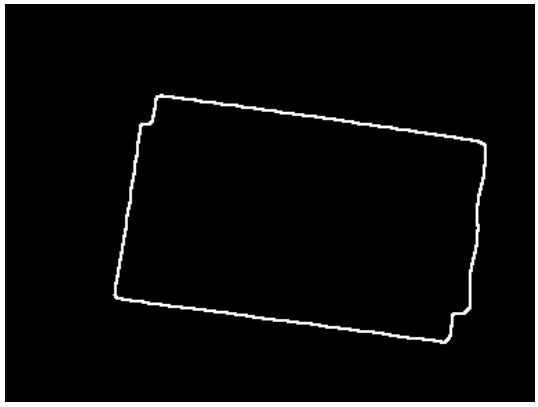
Contour analysis

By carefully observing the images of the different kinds of roof-tiles, one can see obvious differences on their boundary regions (contours). To make them more evident, a contour extraction process was applied to a sample image of each of the 3 tile types. Results are shown in Figure 4.18. These contours have been extracted by a two-step procedure. First, *eroded* versions of the binary images (like the one in Figure 4.12) are obtained. Next, a bitwise *xor* operation is performed between the original binary image and the eroded one. The binary erosion operation works basically like normal image filters:

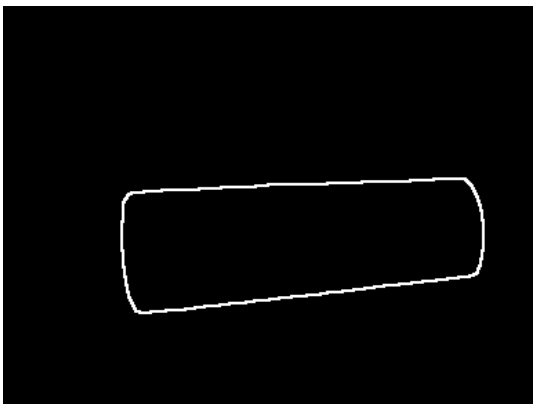
1. The $N \times N$ mask traverses all pixels on the image in the same way as with linear filtering.



(a) Spanish roof-tile contour.



(b) French roof-tile contour.



(c) Barrel roof-tile contour.

Figure 4.18: Results of contour extraction applied to images of 3 different kinds of roof-tiles.

2. If, for a given pixel, any of its neighbours lying below the mask is black, the center pixel value is replaced by zero, otherwise its value is maintained.

A 5×5 mask was used for producing the results of Figure 4.18. Clearly, the three roof-tiles present very different contour properties. However, for discrimination purposes, these contour differences must be quantitatively differentiated.

One of the most popular form of representing the boundary of an object is through the use of chain codes. However, it is seen [19] that small disturbances along the boundary due to noise or imperfect segmentation cause changes in the coding process that may lead into a more complicated contour matching process. Some normalization procedures exist that help chain codes to lead to more robust contour representations. However, these normalizations are exact only if the boundaries themselves are invariant to rotation and scaling, which is not the case of this particular problem.

Another method for representing the tile boundaries was chosen that extracts the *signature* of the contours. A signature is a one dimensional representation of a boundary which can be obtained by plotting the distance r from the centroid of an object to its boundary as a function of angle θ [2]. This process is illustrated in Figure 4.19.

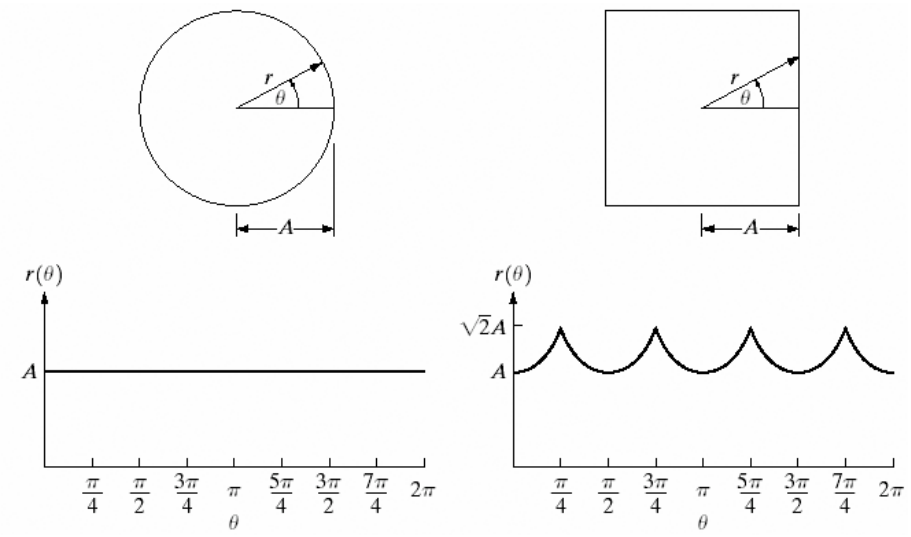


Figure 4.19: Illustration of the extraction of the signature of the contour of an object (taken from `http://www.prenhall.com/gonzalezwoods`).

Using this signature extraction method, the following algorithm is applied for obtaining the features used for pattern matching:

1. The coordinates of the centroid of the roof-tile is computed by finding the average value of the x and y coordinates of all pixels of the extracted boundary.

2. 360 lines are defined (1 degree spacing) from the centroid position to the farthest pixels of the boundary that intersect each of these lines.

[2] Other methods exist for signature extraction. For additional information, please refer to [19, 22].

3. The Euclidean distances between the centroid and each of the intersected pixels are taken and stored in memory.

4. An FFT operation is performed on the array containing those distances, and the logarithmic absolute value of the squared result (logarithmic PSD) is retained.

By using the Fourier coefficients for feature extraction, the problem of rotation invariance is completely and elegantly solved since the discrete Fourier transform treats the input signal as if it was repeated indefinitely through time [23]. Hence, the only information that could indicate tile rotation lies on the phase of the input signal, which is discarded in the subsequent analysis.

Figure 4.20 shows the results of this algorithm applied to the roof-tile boundary images of Figure 4.18. Clearly, it is seen that the spectral component distribution along the frequency axis presents different characteristics for the 3 types of tiles. Moreover, it is intuitive to see that this process has a low vulnerability to noise because such disturbances appearing in the signatures will be translated into high frequency spectral components. Restricting subsequent analyses to the lower frequency part of the spectrum will eliminate the problem.

Consecutive experiments carried out using different tile images have shown very consistent spectral properties within tiles of the same type (mainly in the lower frequency part), which indicates that the recognition problem can be reduced to a template matching problem, without having the need to go into more complex PR techniques like statistical classifiers or neural networks.

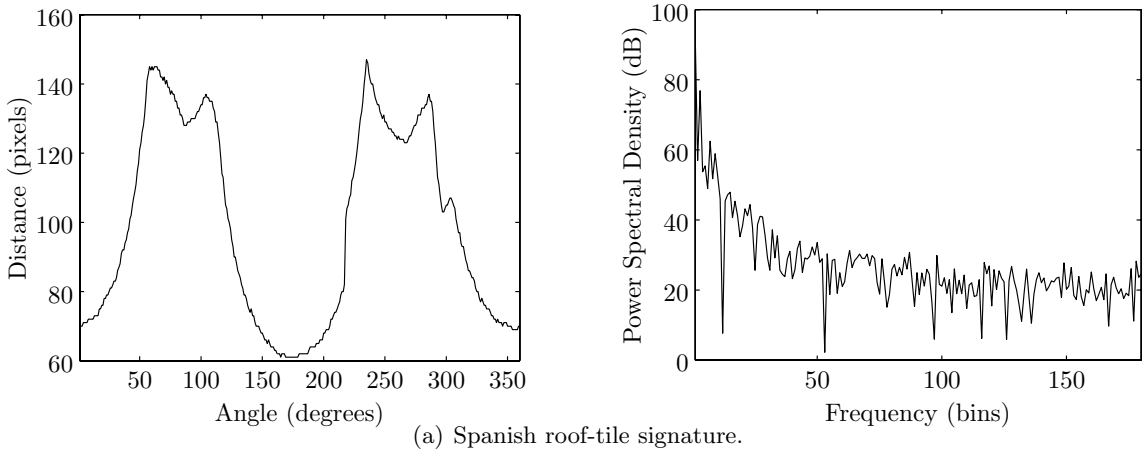
For performing the above mentioned FFT operation, an external C library was used that implements highly optimized FFT algorithms. This library is known as `FFTW` and is made freely available at `http://www.fftw.org`. This library contains routines that automatically determine the fastest way to perform FFT computations for the underlying PC machine [24].

4.1.4 Pattern matching

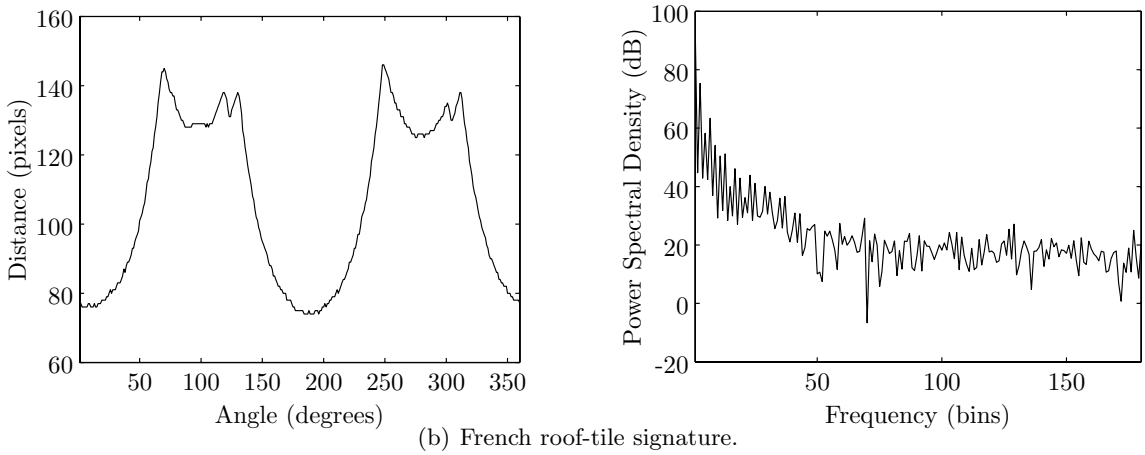
Pattern matching is the final part of the image processing algorithm and that effectively tries to recognize the type of tile present in the captured images. For this, the result of the feature analysis by contour signature extraction is used.

For a given roof-tile type to be recognized among other types, it has to be compared in some way with at least one *template* of each of the available patterns. In this case, a single template for each of the available roof-tiles exists in the database, whose features have been previously extracted using a training set storage application (see Section 2.6 and Chapter 5 for details). Quantitative comparisons between the roof-tile samples and the available templates are accomplished by means of the Pearson correlation coefficient [21] of the first N log PSD values of the contour signatures, between the tile sample under test and each of the stored templates.

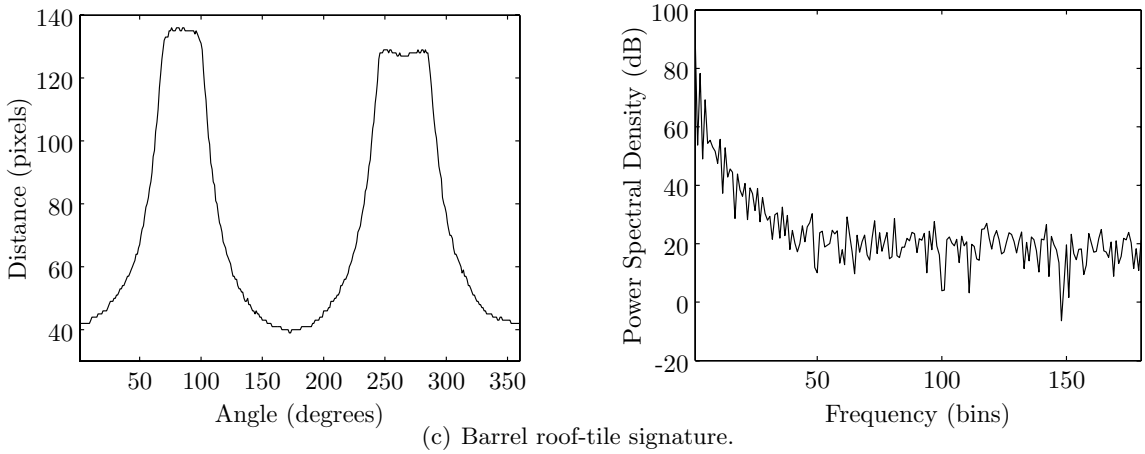
For a given input sequence of pairs of quantities (v_i, w_i), $i = 1, \ldots, N$, the Pearson



(a) Spanish roof-tile signature.



(b) French roof-tile signature.



(c) Barrel roof-tile signature.

Figure 4.20: Results of signature extraction applied to the tile contour images in Figure 4.18.

correlation coefficient r between them is given by

$$r = \frac{\sum_{i=1}^{N}(v_i - \bar{v})(w_i - \bar{w})}{\sqrt{\sum_{i=1}^{N}(v_i - \bar{v})^2}\sqrt{\sum_{i=1}^{N}(w_i - \bar{w})^2}}, \tag{4.4}$$

where $\bar{v}$ and $\bar{w}$ represent the mean values of vectors v and w, respectively. The value of r is limited in the interval $[-1, 1]$. A value of -1 indicates that the two input sequences have complete negative correlation, meaning that if the input vector v was plotted against w (or vice-versa), the data points would lie on a perfectly straight line with negative slope. If r is 1, then the sequences have complete positive correlation, meaning that the data points would lie on a straight line with positive slope. A value of zero indicates that the two sequences are uncorrelated. This shows that the obtained correlation coefficients are invariant to scaling and to offsets. In fact, if v and w were two versions of the same data, with arbitrarily different constant offsets (mean values) and positive scaling factors[3], plotting the two vectors against each other would always result in the data points lying in a straight line with positive slope, i.e., the correlation coefficient would be $r = 1$ for every possible combination.

If the Pearson correlation coefficients between the tile under analysis and the different templates are treated as matching scores, then the output of this recognition stage can simply be given as the type of roof-tile whose template has the highest score result.

4.2 Performance analysis

4.2.1 Recognition performance

A test set was created to evaluate the performance of the pattern matching system and the influence of the number of log PSD coefficients retained from the extracted signatures. The set consists of 10 images from each of the kinds of tiles in the laboratory, taken with arbitrary rotation angles. Some of the captured images show angle rotations of nearly 45 degrees. A new kind of spanish tile was also introduced in the set, adding some contour variations to the spanish tile presented above. Figure 4.21 shows one captured image of this type of tile. In the analysis below, the formerly presented spanish roof-tile will be known as `spanish1` and the newer one as `spanish2`.

A single template image of each one of the tested tile types was added to the database, also with arbitrary tile rotations. The Pearson correlation coefficients obtained for all test images, using only the first 10 log PSD values of the extracted contour signatures are listed in Table 4.1. Figure 4.22 shows the same information, but recurring to a graphical representation for easier comparison purposes. The same results, obtained for the first 20, 30, 40 and 50 log PSD coefficients are presented in Tables 4.2 to 4.5 and in Figures 4.23 to 4.26, respectively.

[3]Note that this is exactly what happens to the contour signatures if the tiles are scaled in the images.

Figure 4.21: Image of the new type of spanish roof-tile introduced in the test set for evaluating the pattern matching system.

	Test tile image									
Template	1	2	3	4	5	6	7	8	9	10
Spanish1	**0.9974**	**0.9971**	**0.9984**	**0.9956**	**0.9985**	**0.9985**	**0.9965**	**0.9961**	**0.9897**	**0.9852**
Spanish2	0.9456	0.9423	0.9535	0.9549	0.9451	0.9452	0.9607	0.9517	0.9528	0.9449
French	0.9153	0.9126	0.9242	0.9203	0.9205	0.9220	0.9230	0.9138	0.9062	0.8995
Barrel	0.8806	0.8831	0.8778	0.8904	0.8865	0.8874	0.8822	0.8775	0.8804	0.8794
Spanish1	0.9172	0.9395	0.9522	0.9490	0.9473	0.8976	0.8999	0.9237	0.9089	0.9399
Spanish2	**0.9891**	**0.9864**	**0.9899**	**0.9914**	**0.9784**	**0.9765**	**0.9778**	**0.9746**	**0.9842**	**0.9936**
French	0.9442	0.9174	0.9777	0.9786	0.9165	0.9405	0.9415	0.9538	0.9488	0.9508
Barrel	0.9228	0.8739	0.9197	0.9305	0.8675	0.9259	0.9269	0.9371	0.9235	0.9264
Spanish1	0.9201	0.9039	0.9013	0.9130	0.9117	0.9070	0.9101	0.9352	0.8761	0.9145
Spanish2	0.9575	0.9740	**0.9743**	0.9618	0.9697	0.9469	0.9728	0.9692	**0.9308**	0.9788
French	**0.9661**	**0.9875**	0.9668	**0.9727**	**0.9784**	**0.9792**	**0.9903**	**0.9872**	0.9248	**0.9909**
Barrel	0.8656	0.8948	0.8912	0.8767	0.8742	0.8970	0.9209	0.9313	0.8782	0.9214
Spanish1	0.8702	0.8685	0.8707	0.8678	0.8739	0.8698	0.8779	0.8752	0.8659	0.8789
Spanish2	0.9221	0.9249	0.9258	0.9349	0.9269	0.9151	0.9356	0.9330	0.9113	0.9368
French	0.9028	0.9064	0.9132	0.9299	0.9120	0.8950	0.9214	0.9119	0.8888	0.9275
Barrel	**0.9993**	**0.9967**	**0.9986**	**0.9874**	**0.9981**	**0.9988**	**0.9960**	**0.9944**	**0.9991**	**0.9949**

Table 4.1: Pearson correlation values for a test set containing 40 roof-tile images, using the first 10 log PSD values of the extracted contour signature. The highlighted fields represent the type of tile presented to the system (left side) and the highest correlation value obtained (right side).

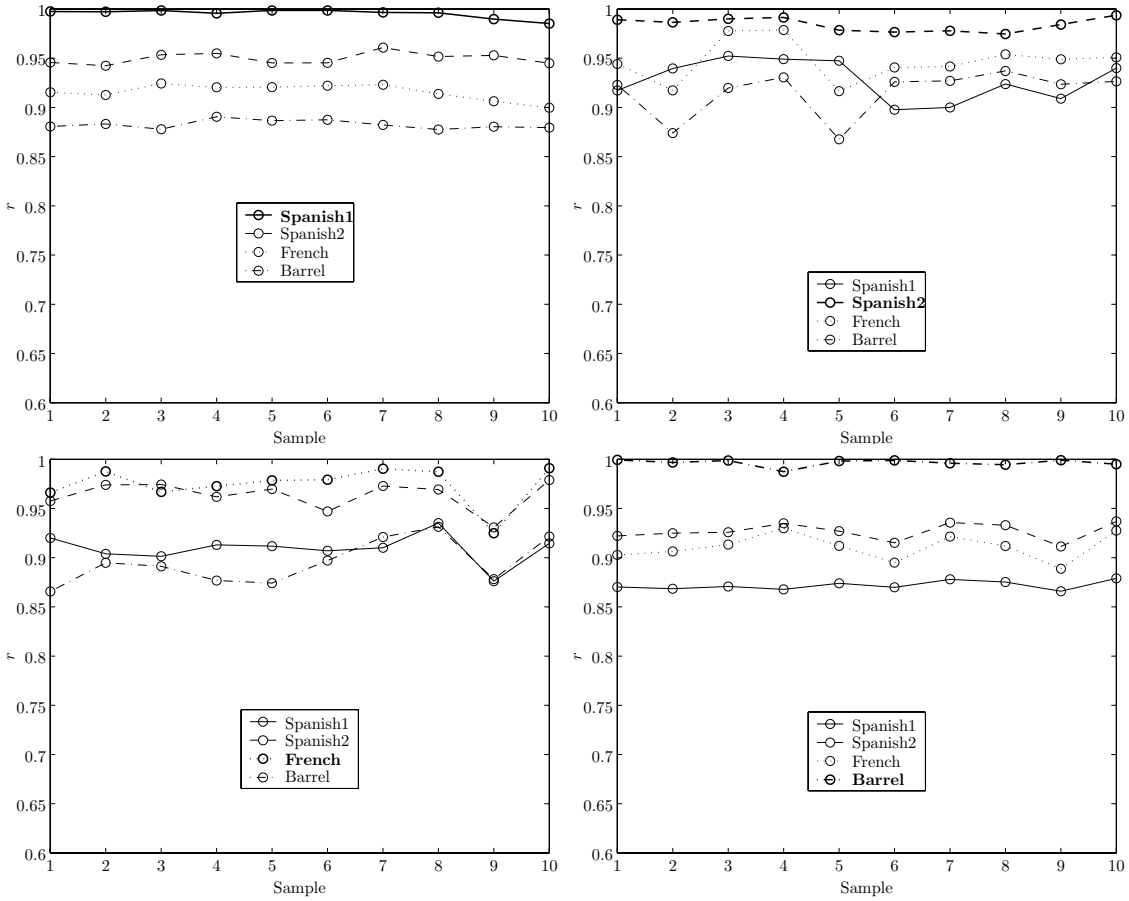


Figure 4.22: Graphical representation of the results shown in Table 4.1.

	Test tile image									
Template	1	2	3	4	5	6	7	8	9	10
Spanish1	**0.9848**	**0.9898**	**0.9285**	**0.9744**	**0.9975**	**0.9928**	**0.9808**	**0.9905**	**0.9845**	**0.9788**
Spanish2	0.8687	0.8571	0.8090	0.8448	0.8596	0.8481	0.9040	0.8851	0.8942	0.8903
French	0.8459	0.8297	0.8019	0.8402	0.8553	0.8512	0.8666	0.8405	0.8472	0.8275
Barrel	0.8728	0.8696	0.8481	0.8695	0.8676	0.8677	0.8819	0.8621	0.8685	0.8634
Spanish1	0.8459	0.8494	0.8625	0.8609	0.8648	0.8451	0.8391	0.8934	0.8288	0.8770
Spanish2	**0.9909**	**0.9863**	**0.9869**	**0.9880**	**0.9802**	**0.9771**	**0.9793**	**0.9691**	**0.9854**	**0.9936**
French	0.8714	0.8633	0.9130	0.9063	0.8625	0.8579	0.8657	0.8660	0.8717	0.8785
Barrel	0.8479	0.8214	0.8574	0.8628	0.8283	0.8454	0.8425	0.8671	0.8375	0.8615
Spanish1	0.7956	0.7873	0.7824	0.7928	0.7801	0.8290	0.8106	0.8641	0.7841	0.8404
Spanish2	0.8823	0.9299	0.9240	0.9108	0.9137	0.8933	0.8804	0.8960	0.8215	0.9206
French	**0.9620**	**0.9667**	**0.9524**	**0.9611**	**0.9607**	**0.9762**	**0.9647**	**0.9723**	**0.9356**	**0.9856**
Barrel	0.7714	0.7909	0.7856	0.7759	0.7701	0.8217	0.8123	0.8514	0.7413	0.8265
Spanish1	0.8687	0.8542	0.8465	0.8461	0.8612	0.8384	0.8629	0.8547	0.8181	0.8603
Spanish2	0.8673	0.8540	0.8290	0.8666	0.8503	0.8313	0.8646	0.8596	0.8118	0.8700
French	0.8422	0.8266	0.8164	0.8586	0.8369	0.8029	0.8515	0.8438	0.7752	0.8676
Barrel	**0.9976**	**0.9968**	**0.9941**	**0.9862**	**0.9966**	**0.9933**	**0.9942**	**0.9932**	**0.9890**	**0.9883**

Table 4.2: Pearson correlation values for a test set containing 40 roof-tile images, using the first 20 log PSD values of the extracted contour signature. The highlighted fields represent the type of tile presented to the system (left side) and the highest correlation value obtained (right side).

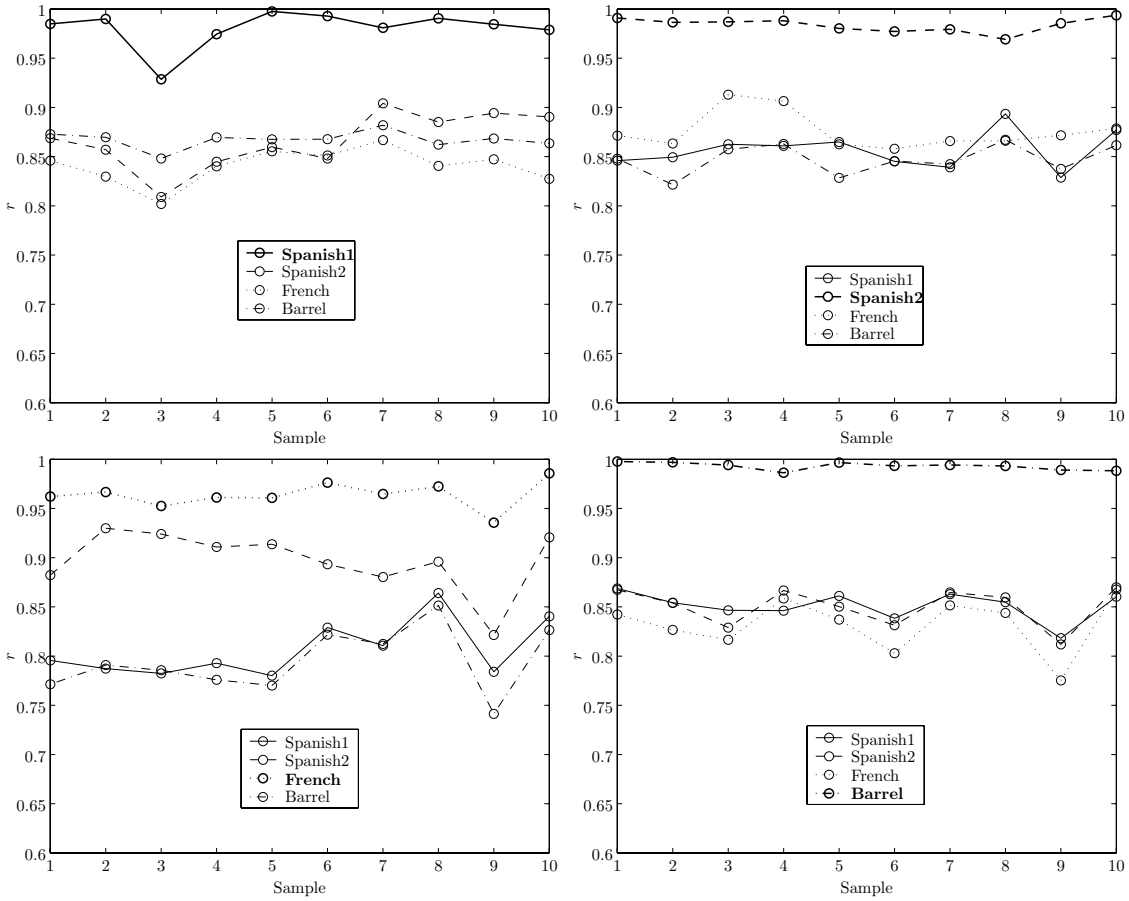


Figure 4.23: Graphical representation of the results shown in Table 4.2.

	Test tile image									
Template	1	2	3	4	5	6	7	8	9	10
Spanish1	**0.9839**	**0.9874**	**0.9288**	**0.9535**	**0.9930**	**0.9902**	**0.9651**	**0.9865**	**0.9834**	**0.9654**
Spanish2	0.8705	0.8554	0.8183	0.8278	0.8615	0.8536	0.8998	0.8687	0.8797	0.8666
French	0.8266	0.8077	0.8078	0.7946	0.8362	0.8405	0.8568	0.8155	0.8221	0.7875
Barrel	0.8593	0.8531	0.8273	0.8395	0.8638	0.8558	0.8716	0.8428	0.8483	0.8369
Spanish1	0.8536	0.8467	0.8407	0.8402	0.8621	0.8383	0.8353	0.8798	0.8273	0.8695
Spanish2	**0.9897**	**0.9861**	**0.9832**	**0.9840**	**0.9824**	**0.9788**	**0.9814**	**0.9673**	**0.9867**	**0.9868**
French	0.8350	0.8324	0.8792	0.8707	0.8282	0.8216	0.8246	0.8168	0.8293	0.8195
Barrel	0.8665	0.8285	0.8481	0.8608	0.8437	0.8591	0.8606	0.8780	0.8554	0.8892
Spanish1	0.7836	0.7621	0.7614	0.7633	0.7416	0.8200	0.8082	0.8608	0.7529	0.8319
Spanish2	0.8287	0.8842	0.8788	0.8635	0.8504	0.8303	0.8334	0.8648	0.7497	0.8654
French	**0.9541**	**0.9568**	**0.9498**	**0.9590**	**0.9410**	**0.9627**	**0.9619**	**0.9624**	**0.9230**	**0.9736**
Barrel	0.7209	0.7437	0.7373	0.7264	0.7048	0.7501	0.7455	0.7968	0.6659	0.7658
Spanish1	0.8563	0.8359	0.8322	0.8532	0.8133	0.8404	0.8560	0.8506	0.7632	0.8776
Spanish2	0.8706	0.8495	0.8460	0.8757	0.8469	0.8565	0.8619	0.8673	0.7859	0.8863
French	0.7763	0.7732	0.7629	0.8036	0.7463	0.7680	0.8128	0.7921	0.7089	0.8363
Barrel	**0.9947**	**0.9899**	**0.9857**	**0.9840**	**0.9800**	**0.9870**	**0.9806**	**0.9861**	**0.9591**	**0.9752**

Table 4.3: Pearson correlation values for a test set containing 40 roof-tile images, using the first 30 log PSD values of the extracted contour signature. The highlighted fields represent the type of tile presented to the system (left side) and the highest correlation value obtained (right side).

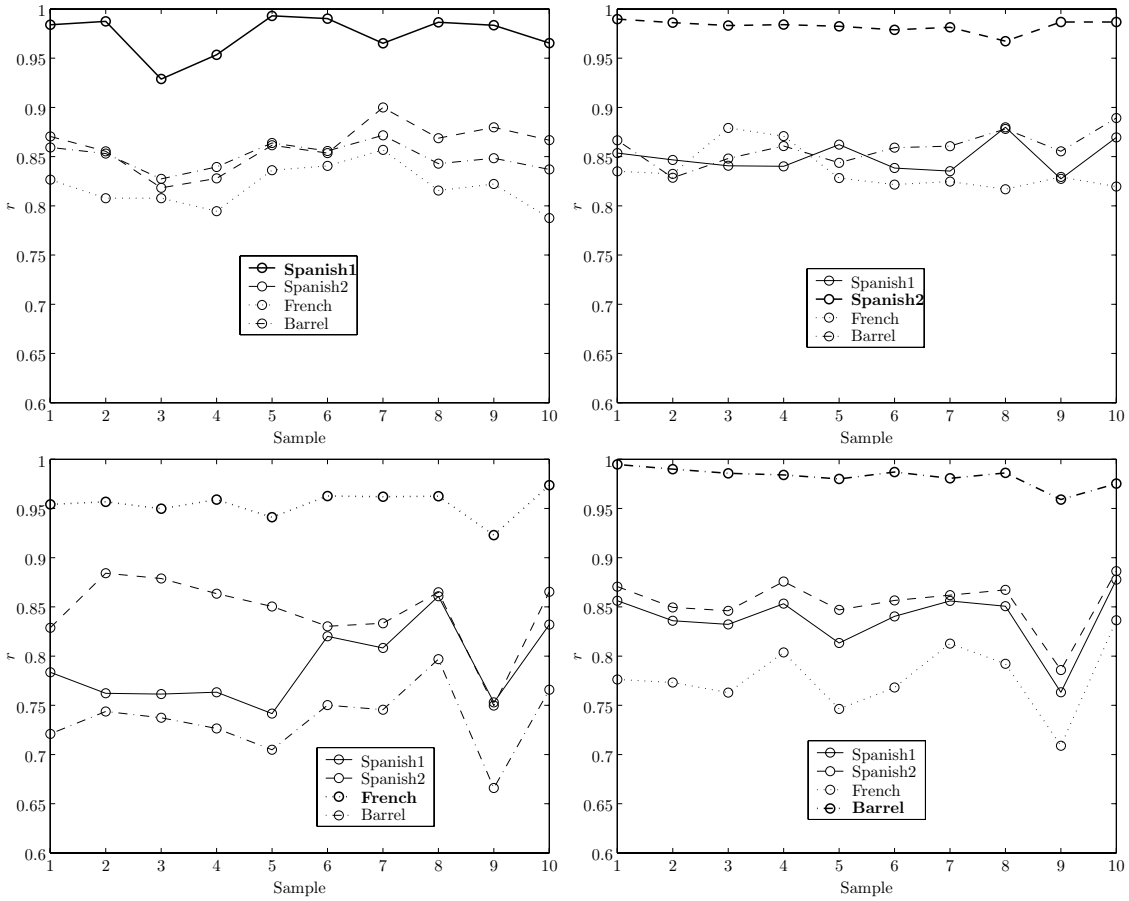


Figure 4.24: Graphical representation of the results shown in Table 4.3.

	Test tile image									
Template	1	2	3	4	5	6	7	8	9	10
Spanish1	**0.9761**	**0.9851**	**0.9195**	**0.9590**	**0.9875**	**0.9884**	**0.9190**	**0.9779**	**0.9576**	**0.9544**
Spanish2	0.8998	0.8994	0.8466	0.8698	0.8967	0.8933	0.8682	0.8872	0.8638	0.8717
French	0.8198	0.8082	0.8163	0.7915	0.8370	0.8480	0.8005	0.8050	0.7959	0.7842
Barrel	0.8842	0.8745	0.8585	0.8668	0.8883	0.8785	0.8734	0.8650	0.8627	0.8666
Spanish1	0.8989	0.8800	0.8649	0.8661	0.8878	0.8601	0.8659	0.9034	0.8590	0.9045
Spanish2	**0.9870**	**0.9848**	**0.9509**	**0.9592**	**0.9760**	**0.9469**	**0.9563**	**0.9549**	**0.9710**	**0.9850**
French	0.8399	0.8384	0.8626	0.8810	0.8383	0.8107	0.8136	0.8299	0.8449	0.8119
Barrel	0.8673	0.8366	0.8320	0.8698	0.8664	0.8442	0.8472	0.8898	0.8638	0.8706
Spanish1	0.7750	0.7817	0.7650	0.7587	0.7504	0.8203	0.8009	0.8644	0.7397	0.8362
Spanish2	0.8153	0.8717	0.8577	0.8324	0.8388	0.8181	0.8146	0.8609	0.7398	0.8498
French	**0.9447**	**0.9525**	**0.9479**	**0.9515**	**0.9325**	**0.8999**	**0.9482**	**0.9540**	**0.8963**	**0.9457**
Barrel	0.7073	0.7545	0.7495	0.7262	0.7053	0.7013	0.7489	0.8081	0.6601	0.7667
Spanish1	0.8590	0.8466	0.8303	0.8327	0.8237	0.8388	0.8567	0.8423	0.7758	0.8768
Spanish2	0.8679	0.8517	0.8281	0.8509	0.8407	0.8473	0.8645	0.8486	0.8031	0.8831
French	0.8052	0.7955	0.7804	0.8038	0.7758	0.7855	0.8374	0.8064	0.7369	0.8422
Barrel	**0.9858**	**0.9882**	**0.9789**	**0.9657**	**0.9785**	**0.9731**	**0.9698**	**0.9697**	**0.9563**	**0.9693**

Table 4.4: Pearson correlation values for a test set containing 40 roof-tile images, using the first 40 log PSD values of the extracted contour signature. The highlighted fields represent the type of tile presented to the system (left side) and the highest correlation value obtained (right side).

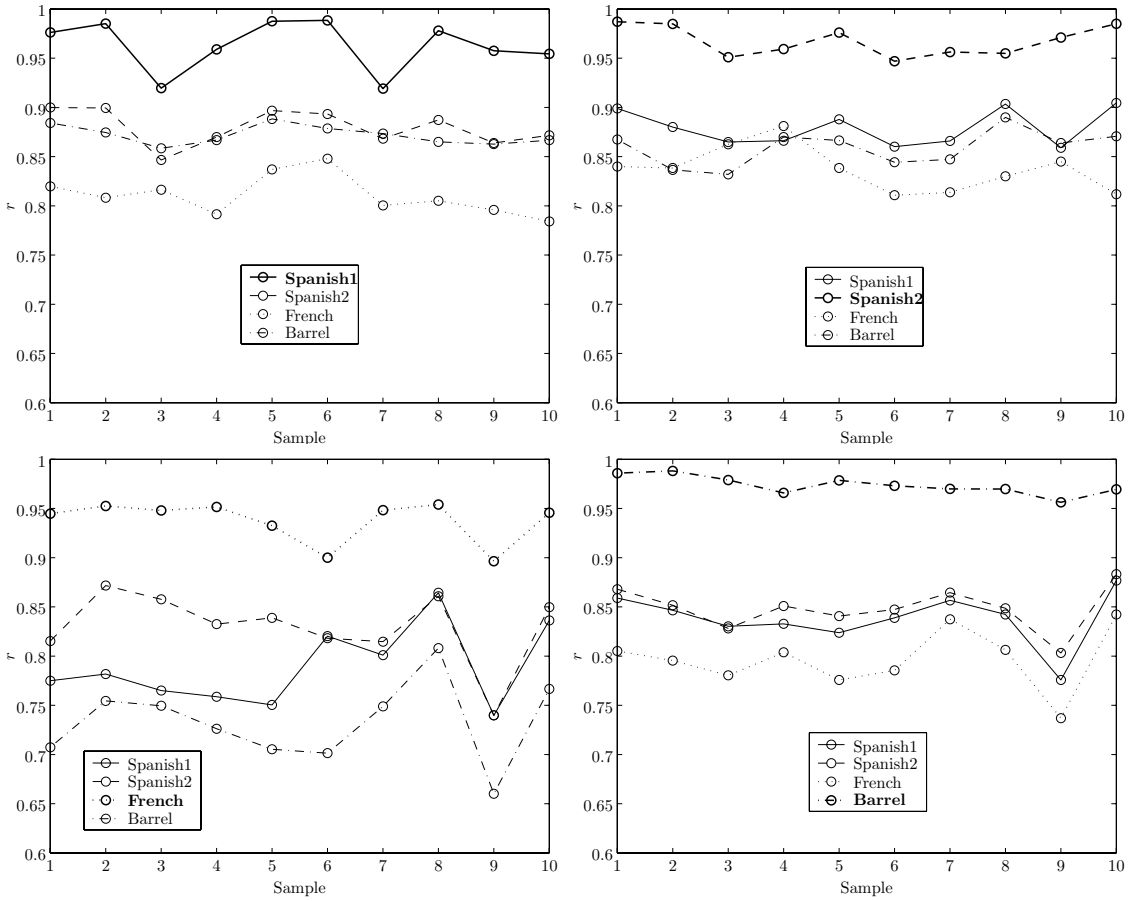


Figure 4.25: Graphical representation of the results shown in Table 4.4.

	Test tile image									
Template	1	2	3	4	5	6	7	8	9	10
Spanish1	**0.9728**	**0.9820**	**0.9203**	**0.9552**	**0.9862**	**0.9869**	**0.9219**	**0.9751**	**0.9595**	**0.9458**
Spanish2	0.8941	0.8946	0.8522	0.8695	0.8892	0.8915	0.8717	0.8830	0.8583	0.8586
French	0.8261	0.8141	0.8209	0.8044	0.8475	0.8577	0.8228	0.8188	0.8116	0.7931
Barrel	0.8789	0.8750	0.8523	0.8526	0.8804	0.8771	0.8667	0.8797	0.8753	0.8789
Spanish1	0.9042	0.8666	0.8650	0.8694	0.8823	0.8639	0.8550	0.8969	0.8651	0.9027
Spanish2	**0.9782**	**0.9762**	**0.9500**	**0.9501**	**0.9597**	**0.9511**	**0.9426**	**0.9482**	**0.9622**	**0.9729**
French	0.8546	0.8408	0.8557	0.8787	0.8566	0.8225	0.8117	0.8482	0.8523	0.8317
Barrel	0.8662	0.8423	0.8400	0.8677	0.8624	0.8573	0.8475	0.8959	0.8594	0.8690
Spanish1	0.7885	0.8047	0.7846	0.7660	0.7736	0.8359	0.8175	0.8624	0.7608	0.8390
Spanish2	0.8017	0.8605	0.8544	0.8280	0.8385	0.8253	0.8199	0.8553	0.7408	0.8556
French	**0.9237**	**0.9470**	**0.9399**	**0.9356**	**0.9269**	**0.9068**	**0.9426**	**0.9514**	**0.8961**	**0.9443**
Barrel	0.7205	0.7817	0.7620	0.7596	0.7229	0.7338	0.7812	0.8246	0.6785	0.7846
Spanish1	0.8420	0.8248	0.8167	0.8409	0.8332	0.8402	0.8639	0.8457	0.7882	0.8804
Spanish2	0.8705	0.8324	0.8420	0.8572	0.8413	0.8510	0.8665	0.8507	0.8268	0.8825
French	0.8077	0.7911	0.7810	0.8115	0.7938	0.8012	0.8601	0.8237	0.7496	0.8513
Barrel	**0.9713**	**0.9700**	**0.9612**	**0.9644**	**0.9715**	**0.9657**	**0.9631**	**0.9592**	**0.9528**	**0.9655**

Table 4.5: Pearson correlation values for a test set containing 40 roof-tile images, using the first 50 log PSD values of the extracted contour signature. The highlighted fields represent the type of tile presented to the system (left side) and the highest correlation value obtained (right side).

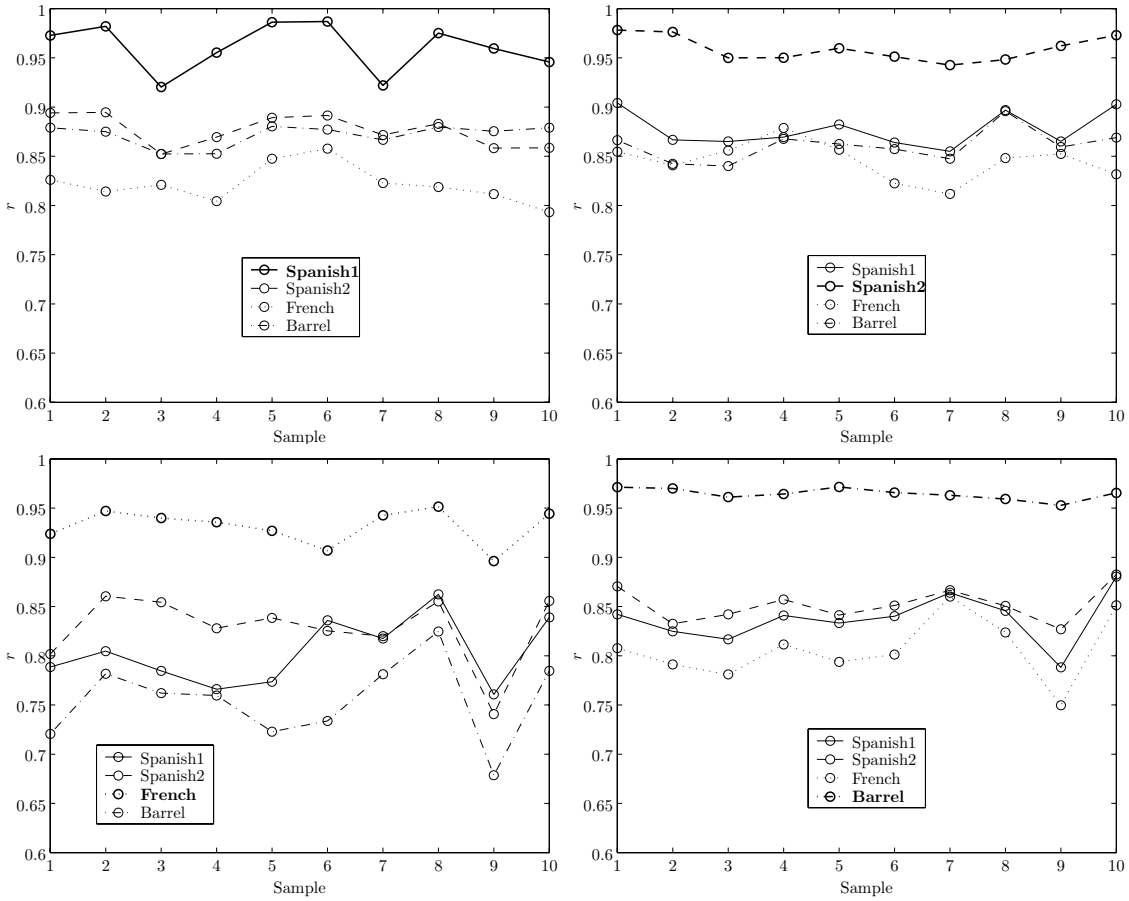


Figure 4.26: Graphical representation of the results shown in Table 4.5.

The various results show that using only the first 10 log PSD coefficients is not a good choice for obtaining good results because of the recognition flaws observed for the french tile type. Also, comparing to the results for 20, 30, 40 and 50 coefficients, the degree of confidence obtained for the 10-coefficient recognizer is very low as the correlation values obtained for the various templates are too close. For higher numbers of log PSD coefficients, the designed pattern matching system shows no major recognition difficulties.

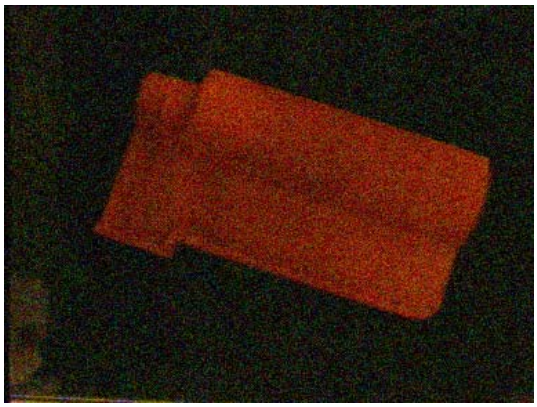
From the 4 types of tiles, it is seen that the easiest one to be discriminated from the remaining ones is the barrel roof-tile. Also, the spanish2 roof-tile has brought an evident confusion to the system when it comes to recognize a french roof-tile. This is because the contours of both kinds of tiles have somewhat similar properties.

Carefully inspecting the plots in Figures 4.22 to 4.26 shows that retaining only the first 30 log PSD coefficients is the most reasonable choice for a robust recognition system. For 40 coefficients and up, the correlation values for the various templates slowly start to get closer to the one corresponding to the expected recognition result, thereby increasing the risk of misclassification.

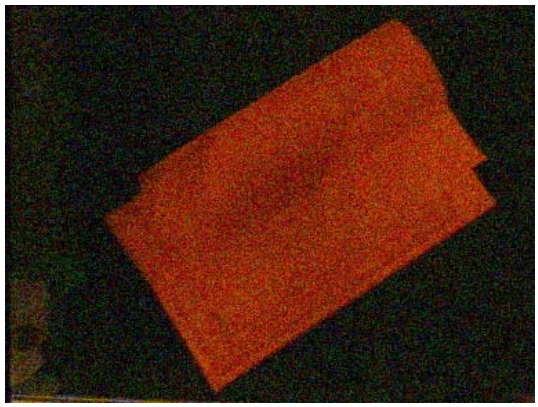
4.2.2 Influence of noise

In this system, noise would appear right on the initial color images captured by the acquisition camera. To simulate the appearance of noise, a sample image from each of the roof-tile types used for recognition performance evaluation was taken and added with coloured Gaussian noise having a standard deviation of 10% of the scale, using a commercial image processing software. Each of the RGB channels gets its own independent noise components. The resulting images are shown in Figure 4.27. Table 4.6 lists the correlation coefficients obtained using the first 30 log PSD values of the extracted contour signatures.

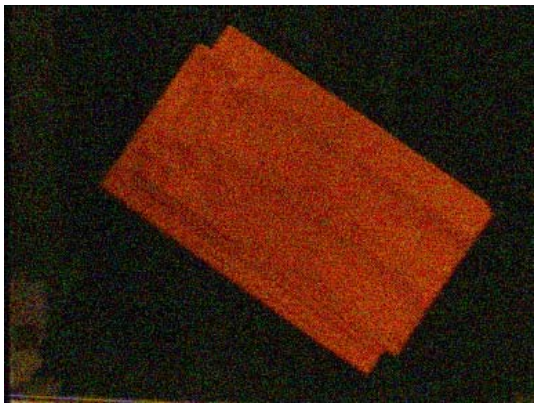
As can be seen in the results, despite the somewhat *rugged* contour extraction caused by the noisy components, the method of pattern matching by correlation of the signature information seems to provide a high degree of robustness to the overall tile recognition system. Note that the noise added to the signatures is removed when only the lower frequency part of their spectral information is retained.



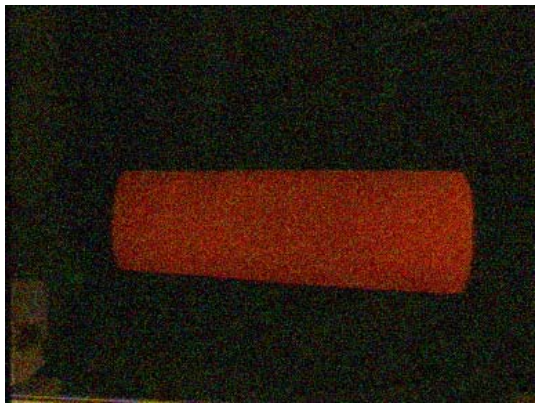
(a) Noisy spanish1 roof-tile image.



(b) Noisy spanish2 roof-tile image.



(c) Noisy french roof-tile image.



(d) Noisy barrel roof-tile image.

Figure 4.27: Images resulting of adding coloured Gaussian noise with 10% standard deviation to roof-tile images.

Template	Correlation coefficient	Template	Correlation coefficient
Spanish1	**0.8486**	Spanish1	0.8236
Spanish2	0.7719	**Spanish2**	**0.9241**
French	0.6762	French	0.8186
Barrel	0.8089	Barrel	0.8380
Spanish1	0.8877	Spanish1	0.8382
Spanish2	0.9062	Spanish2	0.8746
French	**0.9553**	French	0.7904
Barrel	0.8315	**Barrel**	**0.9485**

Table 4.6: Correlation coefficients obtained for the roof-tile images shown in Figure 4.27, using the first 30 log PSD values of the extracted contour signatures.

Chapter 5

Computer Software Implementation

One of the most important parts of this project is related to the development of efficient PC applications for implementing the various algorithms described throughout this document. Without going into extensive programming details, this chapter presents some descriptions of the graphical interfaces built for aiding the work presented in this dissertation.

5.1 Database implementation

All developed PC software applications share a single database implementation that was built using the `MySQL` database engine. This database stores all training samples related to both the sound analysis module and the image analysis module. Since `MySQL` uses only relational database implementations (at least at the time of this development), the SQL language is used for data storage and retrieval.

The physical database implementation consists of 8 different tables. The corresponding relational database diagram is presented in Figure 5.1. A brief description of the role of each of the tables follows next:

Manufacturers This table stores the names of the various tile manufacturers corresponding to the tiles present in the laboratory.

Tiles Each of the records in this table represents each of the types of tiles available for a given manufacturer.

SamplingSessions In this table, generic information related to the data acquired from the analysis of the acoustic properties of the various tile samples are stored.

BadSamples This table stores the data received from the DSP board, related to the roof-tile samples that belong to the *non-conforming* class.

GoodSamples This table stores the data received from the DSP board, related to the roof-tile samples that belong to the *conforming* class.

TileImages Here, the data acquired from the image analysis module is stored. Both the raw image data and all the computed Fourier coefficients are extracted and put into this table.

TrainingSets Data from various sampling sessions of a given kind of piece can be stored in this table, so as to form a training set (including the mean values and standard deviations for each feature of both classes) to be used by the classifying scheme implemented in the sound processing module.

TstCorresp This table plays an important role in the linkage between the sound and image processing modules, as described in the following sections, because it creates a kind of a "bridge" between a given type of tile and a given training set. This allows the selection between different training sets for the same kind of tile.

The use of a dedicated database management system like `MySQL` brings many advantages, some of them were already presented in Section 2.6. This particular system, although being an open-source solution, provides well-known high performance solutions. For an easier interaction with the tables of the database, `MySQL` provides a graphical tool called `MySQL Control Center`, that facilitates some administration tasks like creating and deleting tables, or defining database access privileges[1].

5.2 Graphical user interface applications

For programming the various graphical applications, the `Borland C++ Builder` IDE was used. This environment is very user-friendly, and makes the development of this kind of applications an easier task, as opposed to other available commercial solutions.

All the 4 developed applications share the database described in the previous section. For that, an ODBC driver must be installed that provides a link between the `MySQL` database engine and the operating system. This driver is made available in the `MySQL` website.

5.2.1 Training set storage application for the sound analysis module

This application manages data pertaining to the acquisition of acoustic impulse responses from tile samples, in order to store them in the database for posterior creation of training sets for the sound processing module. In Figure 5.2, a screenshot of this application is presented that shows its initial window. In the left side of the window, a tree-like view is provided, in which the manufacturers, the various kinds of tiles and their corresponding sampling sessions are listed. Within this application, the user has the ability to:

- Add/delete manufacturers, and change their properties (name only);
- Insert/delete types of tiles and assign them to the existing manufacturers;
- Create/delete sampling sessions for the different available tiles and store them in the database;

[1] For more information, please refer to `http://www.mysql.org`.

- View the data pertaining to the already stored sampling sessions.

When a sampling session creation is requested, the window shown in Figure 5.3 is presented. In this window, the user chooses the class to which the tile that is being sampled belongs[2] (upper-left corner of the window). Then, he may choose the number of times to hit the tile and the time interval between successive strokes. This may be useful when determining if the tiles are sensitive to the variations obtained within consecutive strokes. The plots of the energy variation profiles are shown in the right side of the window[3]. Also, the results of the feature extraction process associated with the energy profiles can be seen. The percentage values shown next to the energy decay plots represent the maximum dynamic range used by the captured signals, which is useful for determining the adequate signal levels at the audio mixing console. After the user has hit the 'Save' button, a new sampling session is recorded in the database, and the acquired data is put in tables 'GoodSamples' and 'BadSamples', accordingly. The application allows to mix data from both classes in a single sampling session.

Data from already stored sampling sessions can also be seen with this application. In this case, the window in Figure 5.4 is presented. In this window, the user has the ability to browse through the records of a given sampling session using the 'Previous' and 'Next' buttons, where the previously acquired data is re-processed to the screen.

5.2.2 Training set storage application for the image analysis module

This application was developed for capturing and storing images from tiles for providing the image analysis module with the necessary image templates. For each of the different available kinds of tiles, only one image sample is needed[4]. Figure 5.5 shows a screenshot of its initial window. From this application, the user can insert a sample for a new tile, delete existing samples, replace existing images in the database, and view the already stored images. When the user orders the capture of a new tile image, both the raw image data and the Fourier coefficients, extracted from the signature of the contour of the tiles, are stored in the database. In a future version, this application will probably be implemented as a sub-module of the training set storage application developed for the sound analysis module.

5.2.3 Results demonstration application for the sound analysis module

This application shows the online feature extraction and the classification results, obtained from the tiles that are under test in the prototype. A screenshot of the main window of this application can be seen in Figure 5.6. In the window, the results of a piece classified as being *conforming* ('GOOD') are shown, along with the output of the *Gaudio* classifier function. In the bottom-left corner, the status of the connection with the DSP board is shown.

[2]The class of the tile must be previously determined by human experts.
[3]This data is transmitted from the DSP board.
[4]This has already been discussed in Section 4.1.3.

Several measures have been carried out so that real-time operation is obtained within this application. Besides careful programming, a multi-threaded scheme was implemented. The use of multiple threads for implementing applications having intensive input/output activity is well known to be very advantageous. In this case, 4 threads are running in parallel. One of these threads is responsible for the user interaction with the application, so that the application does not "freeze" when it is waiting for the completion of some kind of operation. Another thread takes care of the interaction between the application and the parallel port of the PC, so as to transmit the striking orders to the pneumatic system. Another thread is solely dedicated to the communication with the DSP board. The remaining of the 4 threads is reserved for the computation of the data received from the analysis performed and transmitted by the DSP.

Obviously, the use of multiple threads where the program sequences of some of them depend on the flow of others requires certain kinds of additional synchronization mechanisms. The type of thread synchronization mechanisms used by this application is based on the triggering of events. For instance, if a given thread needs the results that are provided by another thread, then the former is put waiting for a *flag* that is signaled by the latter thread. Modern operating systems provide highly efficient implementations for this type of synchronization procedures. For example, when a thread is put in a wait state, it consumes 0% of processor resources.

The overall appreciation of this application is that it is capable of real-time operation, being able to process tiles coming in the production line with intervals as low as 200 msec., i.e., about five tiles per second (even running on a deprecated PC machine). This high performance is due not only to the fact of using an efficient multi-threaded design, but also to the fact that a significant part of the processing flow is performed by the DSP board.

5.2.4 Results demonstration application for the image analysis module

This application shows, in real-time, the results of the image analysis process, applied to the tiles that traverse the production line. Figure 5.7 shows a screenshot of the main window of this application, where a recognition result is shown, applied to a spanish roof-tile. The sub-frame on the left side continuously shows the images captured by the acquisition camera, frame by frame, in real-time. This is accomplished using a multi-threaded approach similar to the one described in the previous section. When the user presses the 'Take a shot' button, the frame that is currently displayed in the left side is copied to the frame on the right side. Therefore, the image on the right frame shows a still image. After this, the user may press the 'Process' button, thereby triggering the image analysis procedure. The results of the centroid computation, as well as the extracted contour (see Section 4.1.3) are displayed in the right-side frame, as shown in the figure. Also, the result of the pattern matching procedure is displayed under the left-side frame.

This application is able to process a tile image sample in about 600 milliseconds. This

is a good result, even using the low computational resources that were available for this project. Surely, this time value will descend dramatically if a more recent PC machine is used.

5.2.5 Linking the sound and image processing modules

This part of the software implementation deals with the communication between the two results demonstration applications that were presented in the previous sections. This is necessary so that the image processing module sends its results to the sound processing module. If this is purely a communication between two processes running on the same machine, a simple implementation using operating system mechanisms can be used. Examples of these mechanisms are *pipes* and *FIFO*s [25]. These mechanisms are based on the creation of temporary files, in which data is exchanged between the applications by using standard file reading and writing operations.

If one considers that the two applications may run on separate machines, then a network programming strategy must be used. Internet socket programming is an example of such strategies. For the sake of flexibility, this option was chosen for implementation in this dissertation.

In [10], an informal introduction is given to the programming techniques regarding the use of Internet sockets for communication between different machines connected in a computer network. The type of sockets used for the implementation in this project is connection-oriented. This means that a logic connection is established between the applications, prior to any data exchange. This type of sockets is termed as *stream sockets*, and use the TCP protocol layer for data transmission. This type of communication is assured to be error free (guaranteed by lower-level error detection, correction and/or data re-transmission) and provides data package reception in the same order they were transmitted.

As in many computer network communication implementations, a server and a client must be defined. The client sends requests to the server, and the latter provides the former with the responses. In this case, the client-side is with the image module application, and the server is the sound module application. When the image processing application sends its results to the sound module, a request is being performed. The client does not need to respond to these requests in this case. The only thing the server has to do is analyze the data package that contains a label corresponding to the tile that was recognized, and switch to the training set currently defined for that type of tile. These correspondences are stored in the `TstCorresp` table in the database.

The results demonstration application for the sound analysis module presented above provides the user with the ability to create tile/training set correspondences in an intuitive way. This function may be accessed through the 'Tools/Training set correspondences' menu. Figure 5.8 shows a screenshot of the window that implements this function. In the screenshot, two already defined correspondences are shown between tiles with names "Spanish" and

"Marselha" and training sets "M09" and "B03", respectively. This window enables the creation, deletion and edition of the correspondences, which are immediately reflected in the records of the `TstCorresp` table in the database. For instance, if the image processing module recognizes that "Spanish" roof-tile in present in the production line, then a data package with this tag is sent to the sound module. The server then switches its current training set to the one named "M09".

This communication scheme is of simple implementation, but because it is beyond the scope of this thesis, and because of the lack of available time for its implementation, it has only partially been built, and therefore it is posted herein as a topic for further developments.

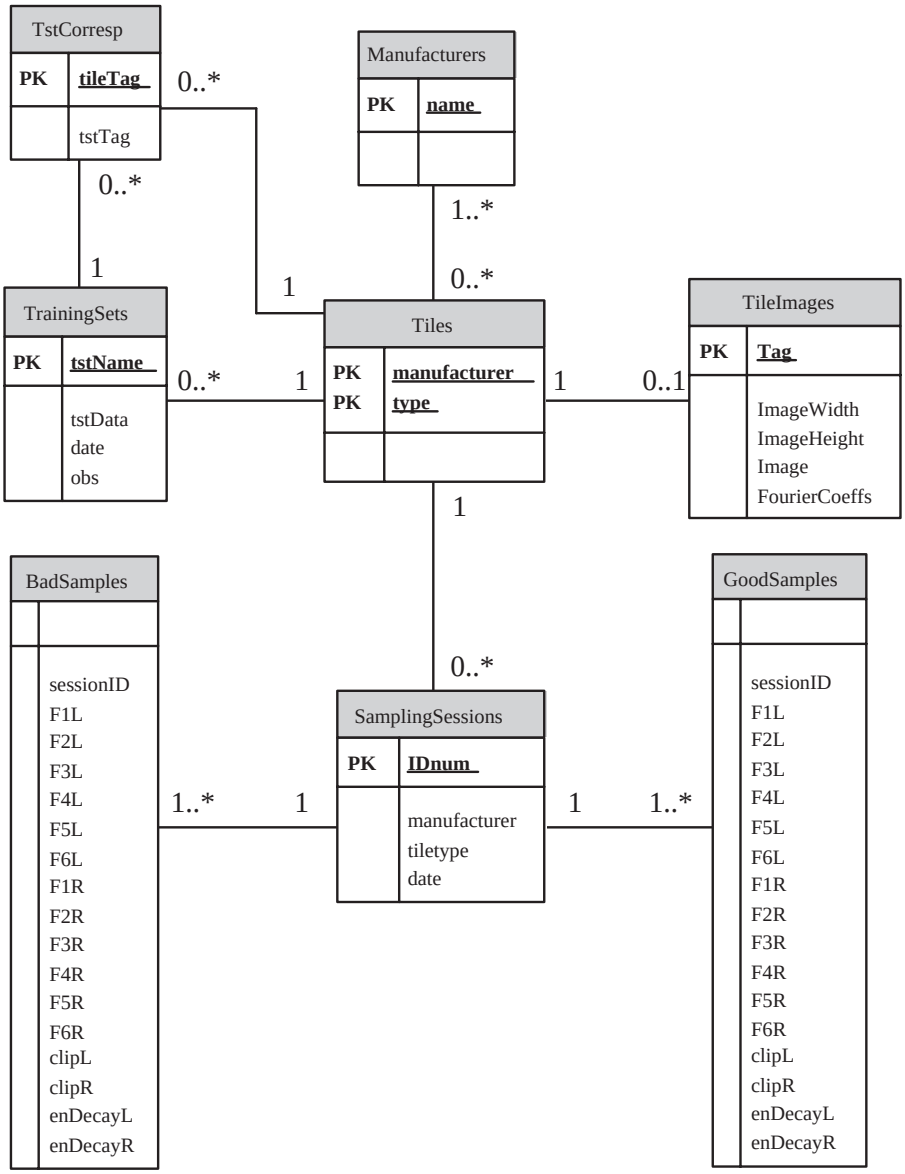


Figure 5.1: Relational diagram describing the implemented database.

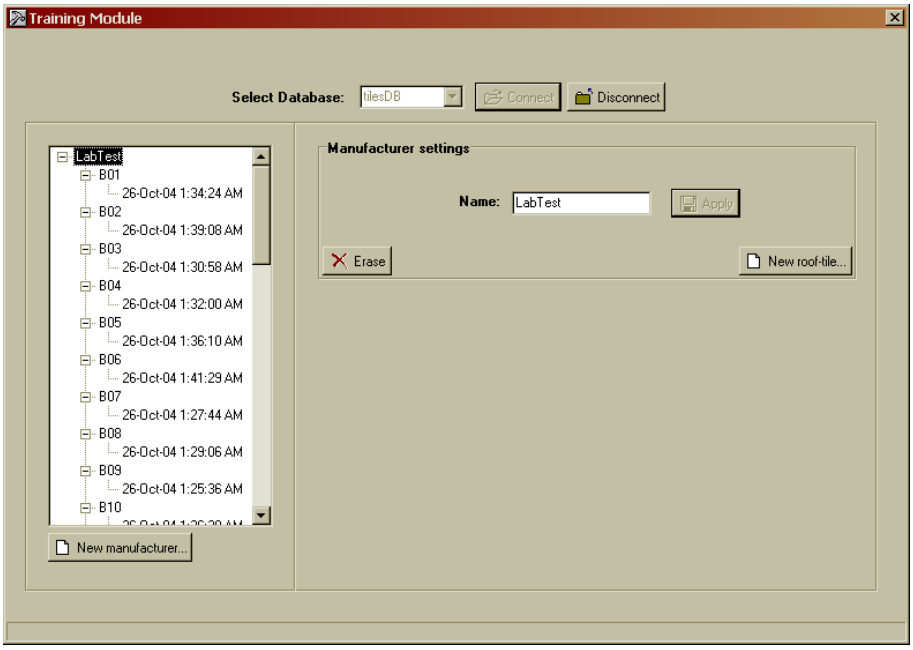


Figure 5.2: Screenshot of the training set application for the sound analysis module, showing its initial window.

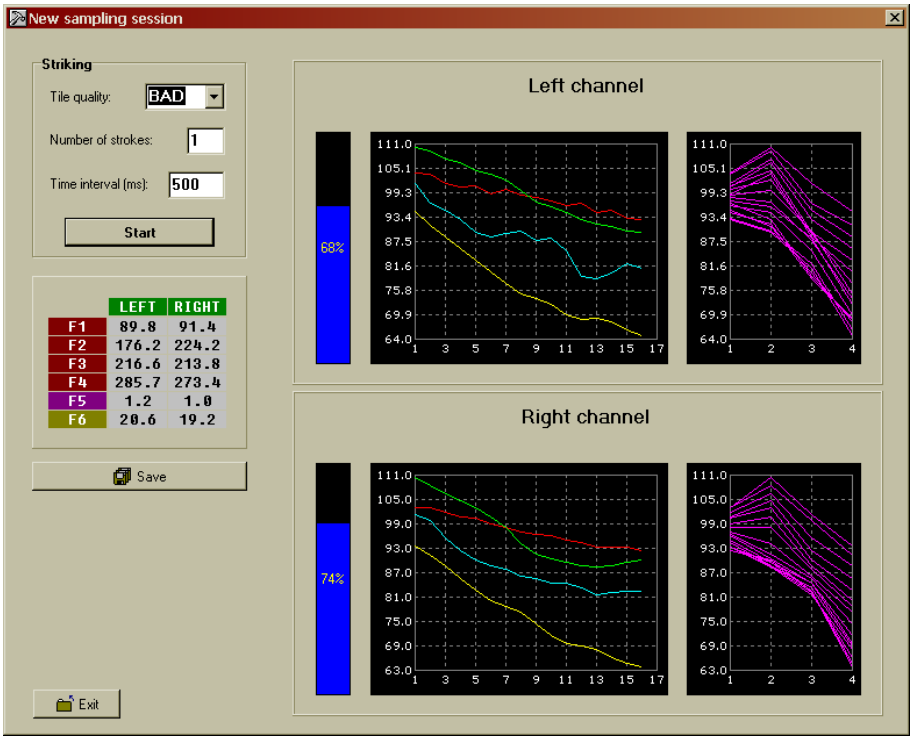


Figure 5.3: Screenshot of the training set application for the sound analysis module, showing the 'New sampling session' window.

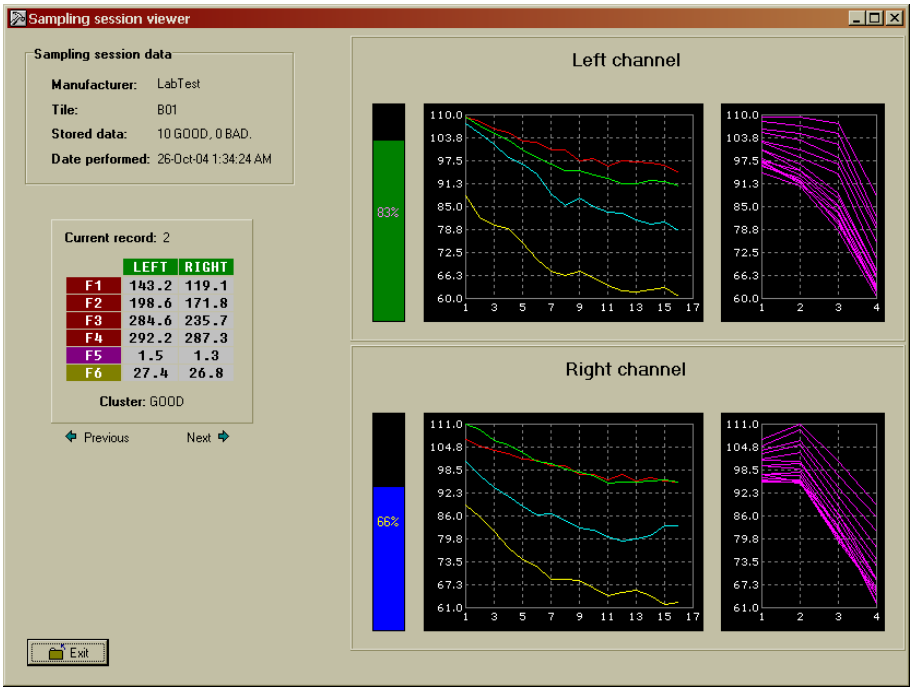


Figure 5.4: Screenshot of the training set application for the sound analysis module, showing the 'Sampling session viewer' window.

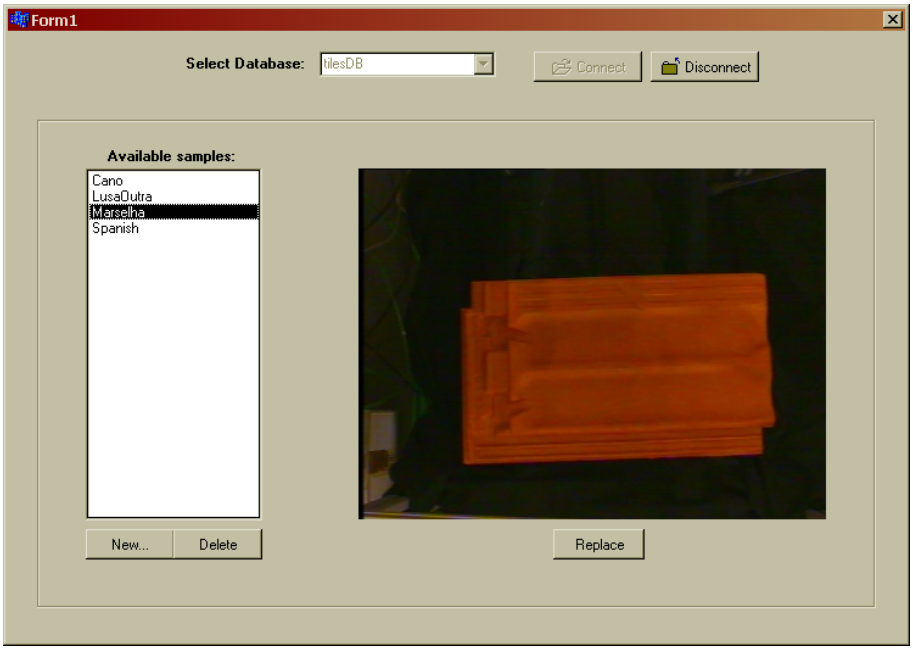


Figure 5.5: Screenshot of the training set application for the image analysis module, showing the initial window.

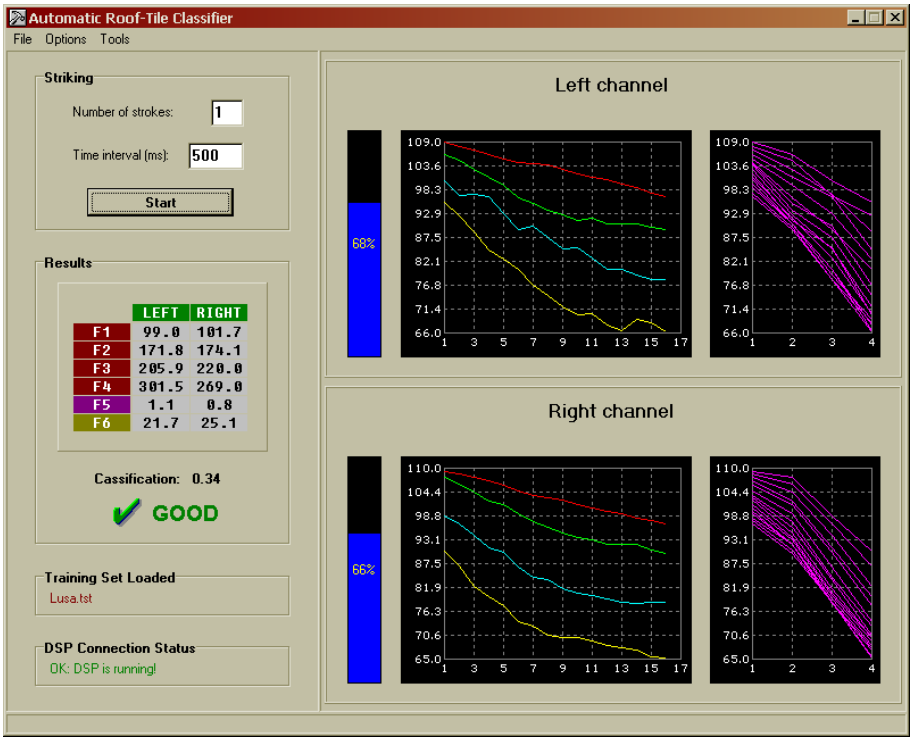


Figure 5.6: Screenshot of the demonstration application for the sound analysis module, showing its main window.

Figure 5.7: Screenshot of the demonstration application for the image analysis module, showing its main window.

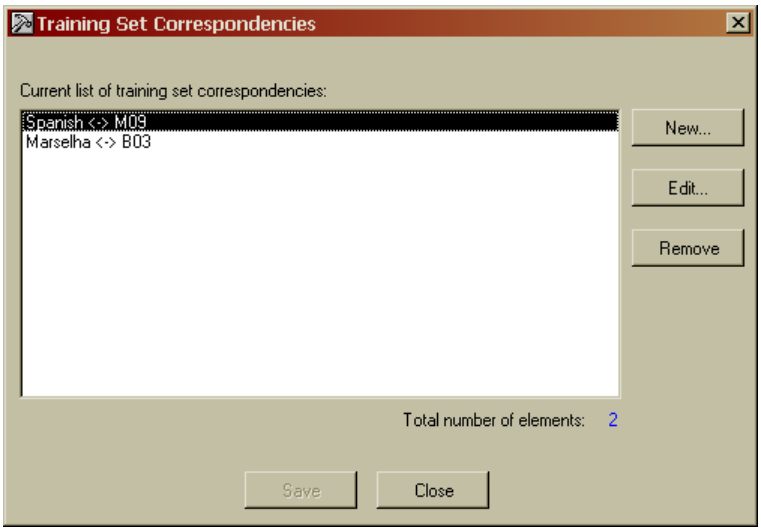


Figure 5.8: Screenshot of the window that enables the definition of tile/training set correspondences.

Chapter 6

Conclusions

This dissertation proposes a method for automatically inspecting the structural quality of ceramic objects which has been tested with clay roof-tiles. The method uses *Digital Signal Processing* techniques applied to audio and image signals, and uses *Pattern Recognition* techniques implemented on a mixed PC/DSP platform, where the real-time constraint is one of the main key issues. A prototype has been built to assist the research, whose mechanical structure was designed to facilitate the insertion on a real production line, installed in a factory.

The system includes an object recognition system that uses artificial visual inspection to determine the type of ceramic object present in the production line. A striking system applies a non-destructive stroke on the piece, initiating the acquisition of the acoustic impulse response therefrom produced. This acquired signal is submitted to an analysis algorithm for extracting relevant features that enable a reliable quality discrimination that is performed on a PC, implementing a fast PR algorithm.

Several computer applications have been developed both for storing training samples in a database system and for the actual implementation of the algorithms for PR and image analysis.

6.1 Strengths and limitations

The results obtained for the sound and image analysis modules are very promising. The PR strategy implemented for sound analysis has shown a 1.76% error percentage, which is a good performance value. The *Gaudio* classifying strategy, developed in the context of this project, by its simplicity and low computational requirements, has proven to constitute a suitable choice for real-time implementation. The analysis on the discriminating power of the 12 extracted features (6 for each audio channel) has shown that some of them actually provide low class separation capabilities. The principal component analysis shown in Section 3.4.1 indicates that low (or no) dimensionality reduction can be performed. However, the results obtained in the previous phase of this project (see Section 3.4.3) show that good results can be obtained by using a reduced set of the 12 features. Nevertheless, by the excellent results obtained, by the low computational effort required to classify a given sample, and by the degree flexibility thereby gained (in terms of adaption to different kinds of pieces and different signal acquisition conditions), the choice of using all 12 features shows to represent a good

compromise. The use of weighting factors assigned to the features, measuring their ability to discriminate between the conforming and non-conforming classes, and their inclusion in the decision function somewhat reduces the problem of the "curse of dimensionality" referred in Section 3.4.1. The advantages of using more than one microphone for capturing the acoustic signals have also been demonstrated. Noise has proven to have low effect on the system, as only with SNR values below 6 dB SPL the classifying scheme starts to fail.

The results obtained for the image analysis module showed a 0%-faulty performance in discriminating 4 different kinds of roof-tiles. The implemented method is also invariant to rotation of the pieces in the captured images, and has shown to have a great degree of immunity to noise. However, because it solely analyzes the contour of the roof-tiles, it will present difficulties if tiles have visible defects caused by severe cracks. Texture analysis through the variations of gray-level intensities along the vertical axis by itself led to no reliable solution. This, in part, was due to the fact of not having a dedicated lighting scheme for the scene. Perhaps a combined recognition strategy using both texture and contour analysis could provide a better degree of robustness to this system.

An overall analysis of the implemented algorithms shows that efficient solutions for addressing the problem of automatically assessing the structural quality of roof-tiles have been presented. Moreover, the developed system seems to be able to be implemented with relative ease in a real production line.

6.2 Topics for further development

The presented solution for automatically inspecting the quality of ceramic work was developed with the aim of actually being used by the ceramic industry. For this integration to be successfully accomplished, a number of additional developments and research activities need to be performed, some of them of difficult replication in laboratory environment, namely:

Socket communication scheme. As described in Chapter 5, the part of linking the image and sound processing modules was only partially built during this dissertation. Therefore, for the sake of completeness, this is one of the future developments that could be firstly carried out, without requiring too much time and effort.

Influence of the movement of the pieces. In order to evaluate the behaviour of the system with the movement of the pieces an evolved version of the prototype needs to be built. Such prototype would have presence detectors, signaling the presence of objects in the line. The supporting carpet beneath the tiles could move in a circular way. This evolved prototype would most certainly raise the attention of industrial manufacturers.

Alternative time/frequency decompositions. Despite the good performance results obtained in the sound analysis module, there is still room for analyzing alternative feature extraction processes, in order to overcome the fact that the data points pertaining to

the non-conforming class are more sparse in the feature space. This fact makes the classifying scheme to more rapidly classify a non-conforming roof-tile as being of the conforming class, instead of the opposite, which would be more advantageous.

Inclusion of an active noise cancellation scheme. Although the immunity to noise of the sound processing module has shown to be high, the inclusion of an active acoustic noise cancellation system would provide an improved performance in a factory, where rapid acoustic events of high amplitudes constantly occur. Perhaps this solution is less expensive than using high quality materials for the acoustic isolation of the sub-system.

Adaption to slowly varying production factors. Several production factors may influence the behaviour of the classifying system. For example, the temperature that the ovens submit to the clay may have an influence on the characteristics of the impulse responses of the produced roof-tiles. Another example is the variability caused by changing to a different clay supplier. These factors must be taken into consideration and their impact on the system must be evaluated.

Combined texture and contour analysis. The image analysis strategy that was developed uses only the contour information of the tiles. Because of that, the pattern matching stage can show difficulties in recognizing pieces that have been malformed or that have suffered from severe visible damages. Moreover, the system can get "confused" when tiles of different kinds share roughly the same contour. A combined analysis using texture and contour information could probably overcome this shortcoming. Moreover, the system could even detect those damages and assess their degree, automatically rejecting pieces with unacceptable visible defects.

The study of this kind of technology applied to other industry sectors can lead to possible similar research works. The industry of metal casting could be a strong candidate. The acoustic responses of typical metallic materials somewhat inspires less chaotic tonal characteristics than in the case of the roof-tiles. This fact would probably lead to more simple feature analysis processes.

A part of the work described in this dissertation has been reflected in an international conference paper [7].

Bibliography

[1] Iwasaki, Hideo, Izumi, and Mamoru. Method of Defect Evaluation for Ceramic Products. *United States Patent No. 4,562,736*, January 7th, 1986.

[2] Lapinski, Norman, Sather, and Allen. Process for the Detection of Micro-Cracks. *United States Patent No. 4,172,224*, October 23rd, 1979.

[3] Vahaviolos and John Sotirios. Methods and Apparatus for Detecting the Presence of Cracks in a Workpiece by the use of Stress Waves Emitted Therefrom. *United States Patent Application No. US1973000389414*, December 9th, 1975.

[4] Ronald P. Steiger, Peter K. Leung, Sugar Land, and Rudolf J. Stankovich. Apparatuses and Methods for Measuring Ultrasonic Velocities in Materials. *United States Patent No. 5,265,461*, November 30th, 1993.

[5] Miguel Falcão, José Vieira, and Aníbal Ferreira. Classificação Automática de Tijolos. *4th Meeting of the Portuguese National College of Electrical Engineering*, pages 15–20, 1999.

[6] Miguel F. M. Sousa. Caracterização Semântica de Sinais Acústicos: Aplicação à Classificação Automática de Peças Cerâmicas. Master's thesis, Faculty of Engineering of the University of Porto, Portugal – `http://telecom.inescn.pt/research/audio/tiles/index.html`, 2002.

[7] Vasco C. F. Santos, Miguel F. M. Sousa, and Aníbal J. S. Ferreira. Quality Assessment of Manufactured Roof-Tiles Using Digital Sound Processing. In *Proceedings of the First Iberian Conference on Pattern Recognition and Image Analysis*, pages 927–934, 2003.

[8] Abraham Silberschatz, Henry F. Korth, and S. Sudarshan. *Database System Concepts*. McGraw-Hill, 3rd edition, 1996.

[9] MySQL AB. *MySQL Reference Manual*. MySQL AB – `http://dev.mysql.com/doc/mysql/en/index.html`, 2004.

[10] Brian Hall. Beej's Guide to Network Programming: Using Internet Sockets. `http://www.ecst.csuchico.edu/~beej/guide/net/`, 2001.

[11] Aníbal J. S. Ferreira. *Spectral Coding and Post-Processing of High Quality Audio*. PhD thesis, Faculty of Engineering of the University of Porto, Portugal – `http://telecom.inescn.pt/doc/phd_en.html`, 1998.

[12] Anil K. Jain, Robert P. W. Duin, and Jianchang Mao. Statistical Pattern Recognition: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, January 2000.

[13] C. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, 1995.

[14] J. P. Marques de Sá. *Pattern Recognition: Concepts, Methods, and Applications*. Springer-Verlag, 2001.

[15] Sir Maurice Kendall and A. Stuart. *The Advanced Theory of Statistics*, volume 3: Design and Analysis, and Time-Series. Charles Griffin & Co Ltd, 1976.

[16] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. John Wiley & Sons, 2nd edition, 2001.

[17] P. Devijver and J. Kitler. *Pattern Recognition: A Statistical Approach*. Prentice-Hall, 1982.

[18] George H. John, Ron Kohavi, and Karl Pfleger. Irrelevant Features and the Subset Selection Problem. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 121–129, 1994.

[19] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Prentice Hall, 2nd edition, 2002.

[20] Nicolas Devillard. Fast Median Search: An ANSI C Implementation. `http://ndevilla.free.fr/median/`, 1998.

[21] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 2nd edition, 1992.

[22] Dana H. Ballard and Christopher M. Brown. *Computer Vision*. `http://homepages.inf.ed.ac.uk/rbf/BOOKS/BANDB/bandb.htm`, 1982.

[23] Sanjit K. Mitra. *Digital Signal Processing: A Computer-Based Approach*. McGraw-Hill, 2nd edition, 2001.

[24] FFTW. FFTW User Manual. `http://www.fftw.org/fftw3_doc/`, 2004.

[25] W. Richard Stevens. *Advanced Programming in the UNIX Environment*. Addison-Wesley, 2000.