CIMSA 2006 - IEEE International Conference on
Computational Intelligence for Measurement Systems and Applications
La Coruna, Spain, 12-14 July 2006

Genetic Techniques for Pattern Extraction in Particle Boards Images

Marco Gamassi, Vincenzo Piuri and Fabio Scotti , Manuel Roveri

Information Technologies Dept.
University of Milan, Milan, Italy
{gamassi,piuri,fscotti}@dti.unimi.it

Electronic and Information Dept.
Politecnico of Milan, Milano, Italy
roveri@elet.polimi.it

Abstract *- Time-to-market and high product quality standards are pushing the use of automatic visual inspection systems for defect detection in a wide broad of applications. The defect detection of particle boards requires the identification of all the printed and natural wood defects that can occur. The availability of information about the particle board to inspect (e.g. the pattern used to print the surface of the board) could increase heavily the defect detection capability of a quality assessment system. Nevertheless, most of the times the pattern is not available during the defect detection phase (i.e. when the pattern changes quickly or when printing and defect detection are not performed by the same company). We propose a novel approach for pattern extraction based on genetic techniques to identify the printing pattern that can be used in defect classification systems. Experimental results show the valuable pattern extraction capabilities of the proposed approach.*

Keywords *- Pattern Extraction, Surface Quality Assessement, Surface Defects Detection, Quality Control, Particle Boards*

I. INTRODUCTION

Particle boards (also known as wood-plastic composite boards) are a composite material lumber or timber made of plastic (sometime recycled) and wood wastes. Nowadays, the applications of such a material are extremely heterogeneous and its market is continuously growing. Particle boards are used indoor and outdoor to build deck floors, railings, fences, landscaping timbers, cladding and siding, park benches, molding and trim, window and door frames. The most common usage is for indoor furniture.

The basic structure of the boards consists of a inner layer of glued wood particles and two thinner layers made of plastic (high-density polyethylene, PVC, plastic) printed with a pattern (figure 1). The wood particles can by very heterogeneous: saw dust, other cellulose-based fiber fillers such as peanut hulls, bamboo, straw, etc. Other components can also be present in the final board, for example colorants, coupling agents, stabilizers, blowing agents, reinforcing agents, foaming agents and lubricants. During the production of boards, once the three layers have been composed, a single gluing process finalizes the board by melting the melaminic resin present in the board using high temperature and pressure.

Factories typically produce large boards (3-4 meters of length) for the furniture industries with different thickness and using a great variety of print patterns for the two surfaces (figure 1, right). The prints are typically made by a repetition of a basic image matrix, generally flipped (horizontally, vertically or both) so as to reproduce a wide broad of woods and solid colors. It is worth noting that a large factory can switch the print patterns tens of times a day.

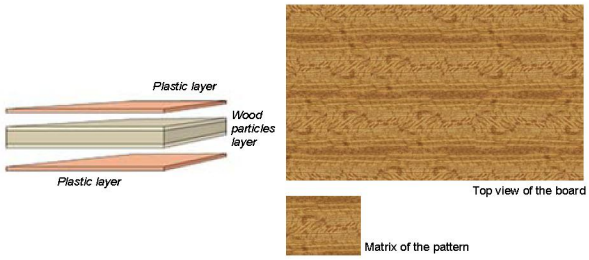


Figure 1. The structure of a board (left) and the top view of the board (right). The image of the upper surfaces shows how the print pattern is made by a repetition of a basic image matrix (vertically and horizontally flipped).

The detection of defects on the surfaces is critical: every defected board sent to the furniture producers causes an important economical cost to the board producer. For this reason great care is taken in the classification of the final board. Nowadays, this important task is performed by human operators which visually inspect the printed plastic surfaces looking for any defects. Visions systems for automated inspection are available, but only at prototype stage [1][2]. A board can have *printing defects* (defects related to the printing process of the melaminic covering paper) and *mechanical defects* (defects related to the mechanical process of the board processing such as tears, scratches and bumps).

In [3] we proposed an approach to the identification of defects in the images of the surfaces where the extraction of the basic printing matrix (supposed to not be available) is based on classical pattern-matching methods. In this paper we present an innovative method to extract the printing matrix based on the genetic techniques.

The paper is structured as follows. Section II describes the modeling of the problem and the creation of a dataset of test images to evaluate the proposed method. Section III presents the innovative approach based on genetic techniques to identify the printing matrix. Section IV describes the figures of merits we defined to evaluate the accuracy of the proposed methods and presents the experimental results.

II. PROBLEM MODELING AND DATASET CREATION

The extraction of pattern (p) from the panel (P) can be considered as a step in a more complex system which aims to recognize defects present in the surface of the panels [3]. In this

system, a real image of the panel (P) is acquired by a linear CDD scanner device in real time. An important point must be explained: if we know that p is the printing pattern of a panel P to be inspect, we could synthetically reconstruct an "ideal" version P' of the panel P. P' represents a version of the panel P that is printed without defect (scratches, bumps, blobs, bad alignments of the printing patters, etc.). The difference image (P'-P) should contain only defects and a negligible noise. Objects contained in this image can be classified better than defects in the original image (which is typically very complex in shapes, color and patterns). The system presented in [3] shows the effectiveness of this approach.

Respect to the pattern extraction methods present in the literature [4][5][6], the key point of our pattern extraction algorithm lies in the capability to manage the peculiar print of the panels. In fact, as briefly explained in the previous section, these panels are generally printed starting from a reference pattern. It is worth noting that pattern p can be present in different forms in the panel according to a set of predefined and a-priori known transformations. It can be present in its original form or it can be flipped in the horizontal direction (transformation T_H), flipped in the vertical direction (transformation T_V), or flipped in both horizontal and vertical directions (transformation $T_{HV} = T_H \bullet T_V = T_V \bullet T_H$). Thus each reference pattern p generates a family of patterns $F_p = \{p, p_H, p_V, p_{HV}\}$, where

$$F_p = \begin{Bmatrix} p \\ p_H = T_H(p) \\ p_V = T_V(p) \\ p_{HV} = T_{HV}(p) = T_H(T_V(p)) = T_V(T_H(p)) \end{Bmatrix}$$

It is worth noting that if p is the original pattern, p_H is the horizontally flipped pattern p, p_V is the vertically flipped pattern p, and p_{HV} is the pattern p flipped both horizontally and vertically. The considered transformations $T = \{T_H, T_V, T_{HV}\}$ have the peculiarity to be "self-inverting" ($f = f^{-1}$). Thus by considering:

$$\begin{Bmatrix} p = T_H^{-1}(p_H) = T_H(p_H) \\ p = T_V^{-1}(p_V) = T_V(p_V) \\ p = T_{HV}^{-1}(p_{HV}) = T_{HV}(p_{HV}) \end{Bmatrix}$$

we can state that the pattern extraction algorithm does not need to find exclusively the pattern p. In fact, we can accept also p_H, p_V or p_{HV} as a correct pattern since they are univocally derived by p.

A panel is organized in rows (called also staves) that are composed by tiled copies of randomly chosen patterns of $F_p = \{p, p_H, p_V, p_{HV}\}$. It is worth noting that the sizes of the panel are not a-priori fixed. Thus the panels are not generally composed by an integer number of staves; in particular the first and the last stave are generally taken only partially. Moreover, since the staring point of a stave in a panel is randomly chosen, the first and the last pattern of each stave are taken only partially. Thus, the occurrences of the patterns that lie at the borders of the panel are taken only partially.

The results of this complex printing phase is that we do not have any a-priori information about how many patterns are present in a panel and where are the patterns in the panel. Moreover the pattern can be present in the panel in four different configurations $\{p, p_H, p_V, p_{HV}\}$ that are randomly chosen. For these reasons the pattern extraction algorithms present in the literature do not suit our case [4][5][6].

We defined a dataset of 100 panels to evaluate the pattern extraction capabilities of our algorithm. These panels are generated from 4 different patterns representing pieces of common wood surfaces (see Fig. 2).

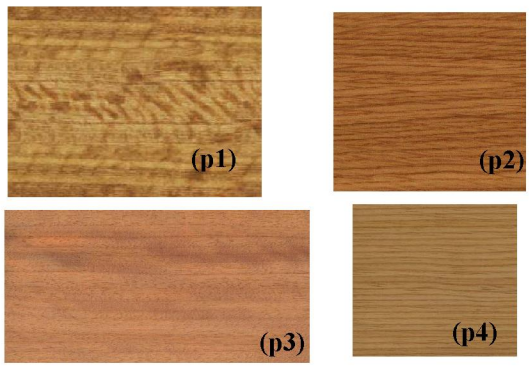
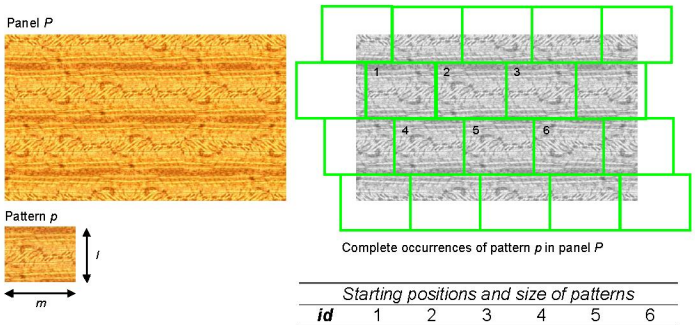


Figure 2. Set of reference patterns used to create the dataset

Each pattern generates 25 panels of 1000x1000 pixels each. In the panel, each occurrence of a pattern is randomly chosen from the pattern family F_p. In particular, for each integer occurrence of a pattern (i.e., the patter is fully present in the panel) we stored the values (x, y) of the upper-left corner of the pattern in the panel. An example of this procedure is presented in Fig. 3.

Thus, during the dataset creation, to allow a next verification phase, for each created panel P_i, we stored the values (x, y) of the n integer occurrences of the patterns $\{(x_1, y_1), (x_2, y_2), .., (x_n, y_n)\}$ and the size $\{m, l\}$ of the pattern in the tuple:

$$D_{P_i} = \{\{(x_1, y_1), (x_2, y_2), .., (x_n, y_n)\}, m, l\}.$$



Starting positions and size of patterns						
id	1	2	3	4	5	6
x	105	105	105	425	425	425
y	24	424	824	24	424	824
m	320	320	320	320	320	320
l	400	400	400	400	400	400
type	p_{hv}	p	p_{hv}	p	p	p_v

Figure 3. Creation of a panel P starting from a reference pattern p.

III. Pattern Extraction: An Innovative Approach

The classical approaches available in the literature to obtain the pattern p in panel P having only shape information about the pattern (for example squared, rectangular, etc.), are mainly based on the analysis of the autocorrelation function of the image [6]. Other examples of pattern extraction can be found in [4][5]. In particular, at the best of our knowledge, only in [3] the case of wood panels has been considered. In this section we describe a novel approach based on genetic techniques.

As described in the previous section, the problem can be described in two basic formulations:

1. Given the panel image P, find the pattern p as p=Crop(P, x, y, m, l); the solution can be represented by the tuple $\{x, y, m, l\}$.
2. Given the panel image P find the tuple

$$D_{P_i} = \{\{(x_1, y_1), (x_2, y_2), .., (x_n, y_n)\}, m, l\}.$$

The first formulation considers the pattern p as the result of a cropping operation from the original panel image P performed at the starting coordinates x and y and cropping an area of m x l pixels.

The second formulation is more complete since it stores each integer occurrence of the pattern. In addition, the number of occurrences of the printing matrix p can change for the same panel P when the values of parameters m and l change. This means that the number of parameters to be controlled during genetic optimization is not a-priori fixed in this formulation.

It is worth noting that it is possible to approximate the second solution given the first solution $\{x, y, m, l\}$ by calculating the points of the occurrences of the pattern as $x_{i+1} = x_i + m$ and $y_{i+1} = y_i + l$. Unfortunately, this approximation is not suitable in applicative applications as will be described in this section.

In the following we consider different techniques to directly solve the first formulation by means of genetic techniques.

The goal is to find the tuple $\{x, y, m, l\}$ (the *solution*) which is the one as closes as possible to *one* of the occurrences present in $D_{P_i} = \{\{(x_1, y_1), (x_2, y_2), .., (x_n, y_n)\}, m, l\}$ used to create the panel P starting with the pattern p. Our approach considers an individual as one possible solution $\{x, y, m, l\}$.

The fitness function of individuals can be calculated as how much the individual have enough information to reconstruct the whole panel P. There are many methods to express this capability of an individual and, hence, to express the fitness function. In this paper, we present and compare three different implementations of the fitness function. In each implementation the fitness function of an individual $i = \{x, y, m, l\}$ is achieved by the following steps: given the panel P,

1. produce the estimated patter p' by cropping the panel P (NxM pixel wide) starting at coordinates x and y for m rows and l columns.
2. using the estimated patter p' to build an estimated panel P' by properly tiling [12] p' in a blank area as wide as the surface of P.
3. Calculate the fitness function as a function of the difference pixel-by-pixel between P' and P, in our case $J = -\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\left|P'(i,j) - P(i,j)\right|$.

The three proposed methods differ in how P' is built in step (2). In all cases, if image P has not been corrupted by noise and if the evaluated individual i is a solution of the problem, the fitness function has a value equal to zero. If noise is present and/or the individual is not a correct solution, the fitness function has negative values.

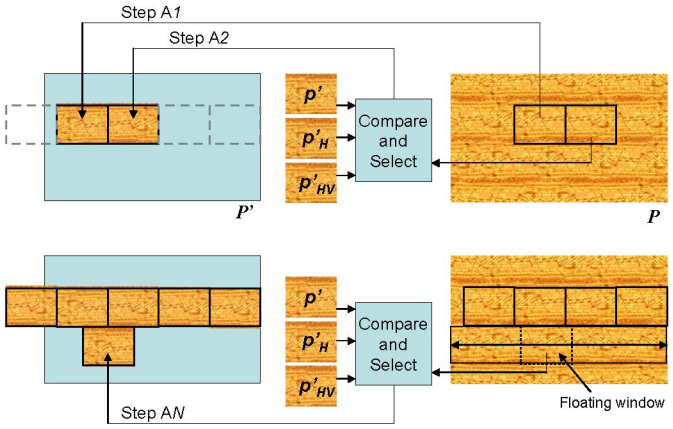


Figure 4. Method A: creation of the image P' from P and p' by simple tiling.

Method A: Simple tiling

This algorithm, proposed for the implementation of step (2), represents an operation of *simple tiling* of pattern p'. Figure 4 sketches the main steps of this method. The process is iterative and starts by placing the element p' (Figure 4, Step A1) in a blank image as large as P in the position (x, y). That identifies also the starting row.

Then, the algorithm starts to fill the blank image by placing copies of p' along the row (Step A2). In this case, all rotation of the element p' (p_H, p_{HV}, p_v, p) have to be checked. The algorithm places the elements p_H, p_{HV}, p_v or p which has the minimum difference pixel-by-pixel with respect to the original panel P in the corresponding position (the rectangle on the right in image P in the top-right subplot of Figure 4). By the iteration of this step the algorithm can complete one row of P'.

The next steps complete the tiling of the remaining rows. The height of a row is fixed by the parameter l in the individual i. First of all, a new starting point for tiling must be processed. At this stage, the algorithm compares the available pattern p_H, p_{HV}, p_v and p in a *floating window* as large as a row in image P (Figure 4, bottom-right subplot). The correct starting position for tiling the new raw will chosen by comparing pixel-by-pixel the available patter p_H, p_{HV}, p_v and p with respect to the sliding window in P. The position of the sliding window that ensures the minimum difference with respect to p_H, p_{HV}, p_v and p will be selected. The selected pattern will be placed in the row in P' in the corresponding matching position (Figure 4, StepA*N*). In the following, we refer to this operation with the term *localization with floating window*. This approach is very similar to the classical pattern matching technique based on the correlation function.

At this stage, step A*2* can be iterated in order to complete the current row. Iterations of StepA*N* can hence complete all the remaining rows in the image P'.

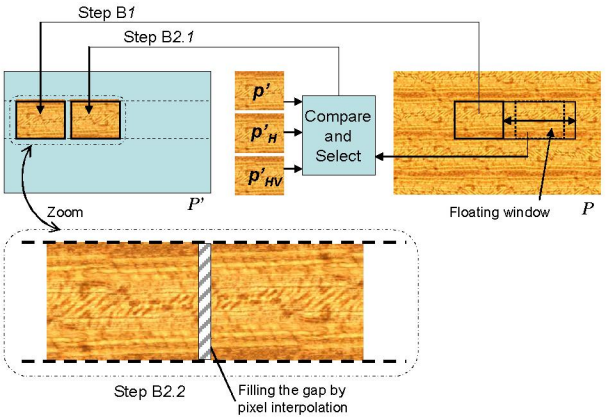


Figure 5. Method B: creation of the image P' from P and p' by adapting the tiling along the rows.

Method B: Adapting the tiling along the rows

Method A offers a quick solution to implement the fitness function but has an important drawback: if the parameters $\{x, y, m, l\}$ of the individual under evaluation are few pixel far from a solution, the fitness function J suddenly drops to large negative values (i.e., the reconstructed image P' is very different from the given P). This phenomenon is due to the effects of non correct alignments of the tiling (errors in x and y) and the repetitive errors that can happen if the estimated patter p' is larger or smaller then the correct one (errors in m and l). As further detailed, this effect of "sharp selectiveness" makes Method A is not suitable for the genetic engine since it cannot quick converge to the solution in real applications. Method A can be enhanced by the detection and correction of the small errors in the positioning the pattern p' along the rows of image P'.

Method B starts by placing the element p' in a blank image as large as P (Figure 5, Step B*1*) in the position (x, y). Then, the row of the image P' will completed by comparing the available pattern p_H, p_{HV}, p_v and p in P using the *localization with floating window* method (Figure 5, upper-right subplot, StepB*2.1*). The height of the sliding window is fixed by the row dimension (parameter l) and its width can be fixed by a threshold (in our experiment equal to $2m$).

Since in Method B the position of the patterns in P' is adapted, a gap along the rows can occur (Figure 5, zoom subplot). For this reason, it can be necessary a further processing step (StepB*2.2*) which fills the blank area between the two patterns by *pixel interpolation*. In our experiments we used a linear interpolation for each row between the last pixels close to the borders of the two patterns. Many other methods could also be applied such as pixel duplications, replications of portions, mirroring, and so on.

The tiling of remaining blank part of the image P' is achieved as described in Method A: a new row will be tiled by placing the new pattern using StepA.*N* and then the row is filled using again StepB2.1. This phase is iterated to fill the remaining rows.

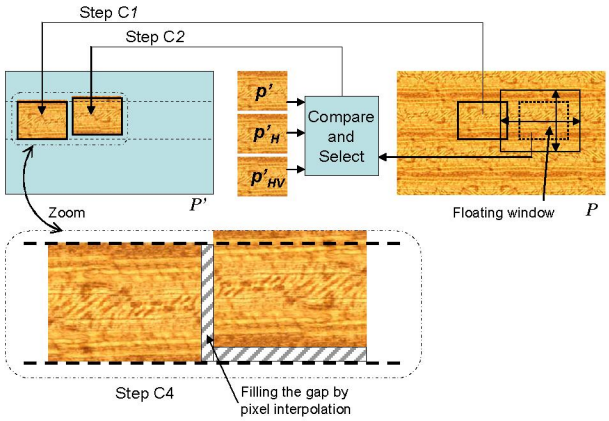


Figure 6. Method C: creation of the image P' from P and p' by fully adaptive tiling.

Method C: Fully adaptive tiling

Method C achieves a fully adaptation of the position of the pattern p_H, p_{HV}, p_v and p in P' along columns and rows.

Method C starts by placing the element p' in a blank image as large as P (Figure 6, Step C*1*) in the position (x, y). Then, the row of image P' will completed by comparing the available pattern p_H, p_{HV}, p_v and p in P using the *localization with floating window* method (Figure 5, upper-right subplot, StepC2). In this case, the floating window is larger than the row's width and can be fixed by two thresholds. In our experiments the size of the floating window is $2m$ x $2l$. This system can suitably compensate small differences in the solution's parameters x, y, m and l at the same time, producing as output a fitness value close to zero and that do not suddenly drop to large negative values as in the case of Method A.

The tiling of remaining blank part of the image P' is achieved as described in Method A: a new row will be tiled by placing the new pattern using StepA.*N* and then the row is filled using again StepB2.1. This phase is iterated to fill the remaining rows.

At this stage, the reconstructed *P'* can contain gaps also along the rows (Figure 6, zoom subplot). For this reason, the last step of Method C consists of filling the blank area between the patterns by pixel interpolation. In our experiments we used in Method C a mirroring operation: the gaps have been filled using mirrored portion of one of the contiguous non-blank portions of P' close to the current gap.

Other parameters of the genetic engine

Let us now define the remaining elements of the genetic algorithm. The selection function is the *roulette wheel*, and the genetic operators we used are *simple cross-over* and *binary mutation* [11]. Initial populations have been created by randomly initializing the chromosomes of individuals. The initial population of the genetic engine is a set of 50 individuals $\{x, y, m, l\}$ where the elements x, y, m, l are randomly initialized according to the bounds associated to the size of the input image P. The termination criterion chosen is the maximum number of generations. Since the fitting function has negative or zero values, the proposed problem is a maximization of the fitting of the individuals.

It is worth noting that an exhaustive approach is not feasible even for small image size since the exploration of solution's parameters $\{x, y, m, l\}$ requires an evaluation of the fitness function J for each parameters' combination. For example, using a panel image of 256x256 pixels produces more than $4\text{x}10^9$ evaluations of the fitness function.

IV. Pattern Extraction Accuracy and Experimental Results

To evaluate the pattern extraction capabilities of our algorithm we defined a set of figures of merit $\underline{\Delta} = \{\Delta_x, \Delta_y, \Delta_m, \Delta_l\}$. More precisely, the aim of these figures of merit is to evaluate the accuracy of the solution $\{x', y', m', l'\}$ proposed by the pattern extraction algorithm.

As explained in section II, multiple occurrences of a pattern are present in a panel. To discover the correct pattern occurrence identified by the algorithm, we initially evaluate the Euclidean distance between $\{x', y'\}$ and the points present in D_{P_i}. We take the point (from here called $\{x_{\min}, y_{\min}\}$) with the minimum distance by $\{x', y'\}$. The figures of merit for a panel P_i are computed as follows:

$$\underline{\Delta}_{P_i} = \begin{Bmatrix} \Delta_x = |x_{\min} - x'| \\ \Delta_y = |y_{\min} - y'| \\ \Delta_m = |m - m'| \\ \Delta_l = |l - l'| \end{Bmatrix}.$$

These figures of merits represent the errors of the recognition of the pattern extraction algorithm. In fact, they represent the distance (computed in pixels) of position and sizes between the real and the extracted pattern.

We computed $\underline{\Delta}_{P_i}$ for each panel in the dataset to evaluate the global behavior of the pattern extraction algorithm. In this experiment we consider a pattern as *found* is each component of the $\underline{\Delta}_{P_i}$ error vector is less than 3 pixels.

To validate our pattern extraction algorithm we initially evaluate its performance in an ideal experiment (the panels of the dataset are not corrupted by noise). The results of this ideal experiment are fully satisfactory since the algorithm was able to recognize the correct pattern for all the panels in the dataset (all $\underline{\Delta}_{P_i}$ are zeros) is enough epochs are given to the genetic algorithm. To provide more realistic results we corrupted the panels present in the dataset with a 2% additive uniform random noise (which is more than the typical noise of the linear CCD scanners). Figure 6.a shows the described behavior of the fitness function $J(x, y, m, l)$ in proximity of a solution $\{x_i, y_i, m_i, l_i\}$. considering $\Delta x/\Delta y$ (with $m = m_i$, and $l = l_i$) and $\Delta l/\Delta m$ (with $m = m_i$, and $l = l_i$) where $\Delta x = x - x_i$, $\Delta y = y - y_i$, $\Delta m = m - m_i$, $\Delta l = l - l_i$.

Results show that Method A (*simple tiling*) is not very effective for the genetic engine. It is very "selective" in the sense that it return high values of fitness only if the proposed individual is very close to be a solution. If the evaluated individual is just few pixels far from a solution, Method A returns very poor fitness values. That makes the performance of the genetic engine close to be a pseudo-random search. That can be clearly seen in Figure 6 (first rows of subplots) where a steep peak is present close to solution ($\Delta x = \Delta y = \Delta l = \Delta m = 0$).

Figure 6.b shows that method B produces a soft decrement of the fitness function J with respect to the l and y axis respectively. That shows that method B effectively compensates the selectivity of the fitness function J with respect small variations of the l parameter and the row positioning y of the pattern. Such a behavior of J enhance the speed of the convergence of the genetic engine: using 600 epochs we have 78% of found patterns (in the remaining 26% of the panels the algorithm was not able to find any pattern).

Figure 6.c shows the behavior of the fitness function J of the Method C. Remarkably, in the right subplot, Method C provides a very good smooth gradient in a convex surface, showing that it can suitably compensate small variation of m and l helping the genetic engine to better find the solution. Method C found a candidate pattern in 98% of the panels (in the remaining 2% of the panels the algorithm was not able to find any pattern) during 300 epochs. Unfortunately, Method C requires a computational time proportional to the area of the floating windows compared with respect to Method A. The presented methods are not jet ready to work in real-time operations: at the present stage of the work one evaluation of the J function of Method C takes about 1 minute on a Pentium Centrino 1.4GHz using Matlab Ver. R14. A complete optimization of the source code is hence required.

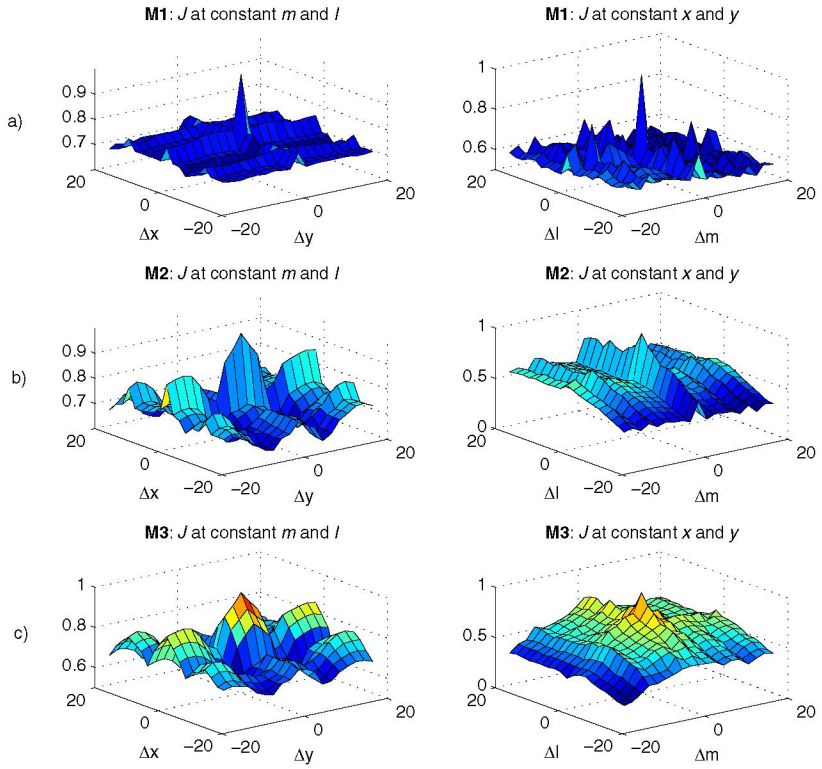


Figure 6. Values of the fittnes funcionf J processed for the panel $p1$ in Figure.2 in a soluzion point varing dx/dy and dm/dl respectively for the three considered methods

V. Conclusions

The detection of defects on the particle boards is critical since every undetected defected board can cause an important economical cost to board producers. The availability of the pattern used to print the surface of the boards to inspect could heavily increase the defect detection capability of the quality assessment systems.

The aim of this paper was to present a novel pattern extraction algorithm capable to extract pattern from panels with very peculiar characteristics (without knowing any a-priori information about the pattern - except its shape - and the pattern can be present in the panel in transformed forms). The pattern extraction algorithm we proposed is based on the genetic approach. We tested our algorithm on a dataset of real woods, showing its remarkable pattern extraction capabilities, but computational time are not jet suitable for a real-time applications.

References

[1] Radovan, S.; George, P.; Panagiotis, M.; Manos, G.; Robert, A.; Igor, D., "*An approach for automated inspection of wood boards*" Int. Conf. on Image Processing, 2001

[2] Ishimaru, I.; Hata, S.; Hirokari, M., "*Color-defect classification for printed-matter visual inspection system*" World Congress on Intelligent Control and Automation, 2002

[3] V. Piuri, M. Roveri, F. Scotti, "*Visual Inspection of Particle Boards for Quality Assessment*", International Conference on Image Processing ICIP, Genova Italy, 11-14 September 2005

[4] Valiente, J.M.; Albert, F.; Carretero, C.; Gomis, J.M. "*Structural description of textile and tile pattern designs using image processing*" Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on. Volume 1, pp 498 – 503, Aug. 2004

[5] Gil, F.A.; Gomis, J.M.; Valiente, J.M.; "*Reconstruction techniques in the image analysis of Islamic mosaics from the Alhambra*"; Computer Graphics International, 2004. Proceedings 2004 Page(s):618 - 621

[6] A.K. Jain, T. Kailath, *Fundamentals of Digital Image Processing*, Prentice Hall, 1988

[7] C Harris, M Stephens, "*A combined edge and corner detector*" the forth Alvey Vision Conference, pages 147-151, 1988

[8] Peter Kovesi, "*Phase Congruency Detects Corners and Edges*". The Australian Pattern Recognition Society Conference: DICTA 2003. December 2003. Sydney. pp 309-318

[9] M.A. Fishler and R.C. Boles. "*Random sample concensus: A paradigm for model fitting with applications to image analysis and automated cartography*". Comm. Assoc. Comp, Mach., Vol 24, No 6, pp 381-395, 1981

[10] Richard Hartley and Andrew Zisserman. "*Multiple View Geometry in Computer Vision*". pp 101-113. Cambridge University Press, 2001

[11] Koza, J., R. 1998. *Genetic programming*, Encyclopedia of Computer Science and Technology, volume 39, James G. Williams and Allen Kent 29-43.

[12] Grunbaum, Branko and G. C. Shephard. *Tilings and Patterns*. New York: W. H. Freeman & Co., 1987.