Intelligent processes for defect identification

Edson Pacheco Paladini

Abstract - This paper describes a knowledge-based system and other classical artificial intelligent techniques developed to identify imperfections or defects in industrial products. The defects we are studying used to appear on the piece external area (like spots, fractures, scratches, dark or white lines). The application of the system has been developed in wall or floor tiles factories and it has been showing itself adequate to its finality, as show its application results. The system works, basically, with codified information from the wall or floor tile faces. The piece of information is accessed by special devices which pick up the image and transform it in an array of numbers and codes. Therefore, the system behavior can be defined by these information pieces. Initially the system detects the existence of imperfections using a first group of computational programs; after that, s second group of programs defines the gravity level of each detected defect (for instance: if it implies to reject the piece). Finally, a third group of programs (the identification system) informs to its users what is the most probable kind of imperfection detected (defect identification). We show here the general ideas of the identification system and the structure and some results, what can be seen as a useful and interesting application of knowledge-based systems to quality control area.

Key-words: Artificial intelligent techniques, quality control, defect identification.

I. INTRODUCTION

The process of defect identification is one of the most complex problems when considering quality evaluation, mainly when this evaluation is made by attributes, it means, in a qualitative way. It is a relevant question and studies and proposals should be done in this area.

The theoretical basis of defect identification comes from Artificial Intelligence classical techniques, like syntactic pattern recognition [1] or threshold selection techniques [2].

Many studies have considered the defect identification problem in different areas – like design [3] or physical environments [4] and [5].

Recently, different resources have being used to different problems, like genetic algorithms [6], neural networks [7] or fuzzy inference [8]. Here, as it can be seen, we use a specific approach and other techniques.

When treating this question, this paper, fundamentally, considers some basic aspects:

(a) Imperfections detection problem is the *most important* in the evaluation by attributes. In fact, if a quality evaluation system can select defective pieces in an isolated way (without considering the production line) then it is possible to develop (with some easiness) a process to identify imperfect pieces using factory inspectors. Actually, it is an easy task to judge the kind of defect found when the piece had taken off the production line already. In this case, the analysis is made with few pieces, if compared with the amount which passes by the production line. And besides, the evaluation about the nature of the defect does not suffer pressure of the operation line, it is not urgent to make several analysis at the same time and, as a consequence, the level of inspectors' mistakes tend to decrease considerably. Maybe the biggest practical set of difficulties in identifying defects is (1) to organize the occurred failure historic; (2) to define which features must be observed in order to particularize a defect and (3) to select the actions to be taken in order to correct and prevent the occurrence of the detected imperfections;

(b) The identification becomes facilitated if it is possible to aggregate some devices to the detection system, which is able to give *preliminary defects analysis*, it means, if it is possible to inform some observed elements in the basic properties evaluation of each piece under inspection.

Considering these aspects, we have developed a detailed study about the most common imperfections in production lines of a ceramic products factory and, basing on that, it is proposed a system which allows, at the same time, to organize data related to the observed defects and determine, quickly, corrective and preventive actions to be developed in production lines.

Related to the second question, it is proposed an analysis method of information resulting from the programs which compose the system to detect imperfections. These programs can help in defects identification.

II. DEFECT DETECTION SYSTEMS

In order to detect and identify defects on a piece, an image capture and analysis set of devices is used. This image becomes an array of numbers and codes (for instance, a matrix of numbers and symbols). This array is the representation structure of the image. We use three systems that deal with those arrays (that represent the images of the pieces).

Manuscript received Febr, 28, 2007; Revised April 21, 2007

E. P. Paladini is with Departamento de Engenharia de Produção e Sistemas. Universidade Federal de Santa Catarina. Campus Universitário da Trindade - CP 476 - 88040-910 - Florianópolis – SC. BRAZIL. paladini@floripa.com.br; paladini@deps.ufsc.br.

Initially, the first system detects the existence of basic imperfections using a group of computational programs. Basic defect are those that make the piece useless - for instance, the piece is broken. If the piece has basic defects we do not make any other analysis and the piece is rejected. We will call it *System 1*. The second system works with pieces that do not have basic defects. This system detects defects comparing the pieces with specific patterns (to detect, for instance, differences in colors). The system detects defects and also defines the gravity level of each detected defect (for instance: if it implies to reject the piece because it is completely different when considering the patterns). We will call it *System 2*. Finally, a third system (the identification system or *System 3*) informs to its users what the most probable kind of imperfection detected is. Each system uses a set of computational programs. The arrays that represent the image of the pieces are the main inputs of the programs; the outputs are information about occurrence of basic defects; defects observed when comparing the pieces with some patterns; the gravity of each defect and its identification.

III. STRUCTURE OF THE SYSTEM

Essentially, the defects identification derives from the characteristics analysis that a given feature assumes in a piece judged as imperfect. So, it is intended to evaluate how this feature is observed at pixels level. The set of pixels will compose the image.

The computational programs of the first and the second groups use two kinds of analysis: global analysis (considering large areas of the pieces) and analysis at each pixel (punctual analysis). The programs making the global piece evaluation are relevant for imperfection detection, mainly in those cases where the feature occurrence does not depend on the place where it is observed. But it is important also to invest in programs which detect imperfections from the piece properties at the pixel level, mainly the ones that could make pixel statistic analysis referred to the areas of the pieces in which defects were found.

Imperfections identification processes should work in a comparative way. When comparing the image with the pattern it is possible to get information which make possible to define situations in which the defects probably occur. Unsupervised processes (that do not use patterns), like basic defects detection, are less useful here.

As an example, it can be observed what occurs with the edges matrix associated to the matrix which represents the image of a certain piece. Abrupt variations in gray levels in a monochromatic environment analysis use to indicate imperfections presence. If the spot is clear, the variation is very strong, and the edge level value in the borders of the defect occurrence area becomes largely affected. In this case there are sufficient indications to detect the imperfection. However, it is not possible to identify the defect. Nevertheless, any inspector can do that. It even allows us to consider if it is not less efficient and more expensive to create an imperfection automatic identification system...

Hence, it seems relevant to specify how the feature under study occurred in the piece. Since the evaluation will be done at pixels level, it is possible to detail the analysis in order to associate the information to the imperfections occurrence areas, providing the possibility of evaluating the (1) intensity and the (2) extension of the defects. These are basic parameters to the possible imperfections identification and two proposals are formulated for the defect identification:

(1) *Concerning the* ***intensity***: it is necessary to develop devices which can evaluate how much the pixel characteristic does not attend a specification;

(2) *Concerning the* ***extension***: it is necessary to cluster pixels that belong to the defect area, in order to know and delimit the affected area by the imperfection.

In the first case, it can be observed that all programs of the first and the second system have devices to execute a preliminary evaluation of the intensity of each pixel - also the pixels in the defect areas, of course. In fact, having fixed limits for the studied properties, the programs analyze the image representation structure and determine what pixels are below and what are above certain limits, and also the difference between the limit and the property level in the analyzed pixel.

As it was observed, even though not every defect detection system program analyses information at pixels level, it is possible to separate defected pieces and apply analysis the areas that have damages to evaluate the imperfection intensity.

Each element of the matrix A (that represents the image of a piece) represents a pixel of the piece. Then if $|A(i,j) - K| < L$, the pixel in the position (i,j) is considered acceptable for the L limit, considering an amplitude K. It means that, given two limits L_1 and L_2, with $L_1<L_2$, if $A(i,j) < L_1$, the defect is referred to the low intensity of the property occurrence and if $A(i,j) > L_2$ there is a high property incidence on the pixel, at intolerable levels.

Considering, as an example, in a monochromatic analysis, the RGB system, if f(x,y) is the gray levels function associated to the main diagonal of the cube RGB, it is possible to take $0 < f(x,y) < 255$. If f(x,y) is equal to zero, it can be noted a black point; if f(x,y) is equal to 255, it can be observed a clear point. If g(x,y) is the function associated to the same space, but having the axis direction inverted, it means, taking $g(x,y) = 255 - f(x,y)$, it is possible to generate a reverse graduation, from white to dark. It makes the system behaves in a similar way to the inspector does. Then its decisions are easier to understand. Practical experiences show that this new situation to evaluate the defects is much easier to implement. It allows determining what follows: If $g(x,y) < L_1$, the piece presents clear spots and the points (x,y) show this characteristic. The g(x,y) determines the pixel "brightness intensity". If $g(x,y) > L_2$, the piece presents dark spots and the points (x,y) also show this characteristic. The g(x,y) value determines the "darkness intensity" in that pixel.

In case of polychromatic analysis, f(x,y) can determine if a given property under study is above or below a limit value. In the example above, if $f(x,y) < L_1$, therefore the property intensity is very weak, it means, the analyzed characteristic is missing. It can mean, for example, that a certain tone more fainted than the normal or a very low saturation level (the color is very clear). If $f(x,y) > L_2$, the presence of the characteristic is very intense, having an excess of a chromatic tone or a saturation level so high that inhibits the predominance of any other color.

There is a specific program in the system 1 that determines the values of L_1 and L_2 but they can be fixed according to a given reference. For the most tested pieces in defect detection programs, involving monochromatic analysis, for instance, values like 40 for the minimum limit L_1 of the gray levels and 100 for L_2 were adequate, since the pieces have its gray levels oscillating between 60 and 90.

Computationally, the level evaluation operation according to some limits is very simple, quick and can be programmed in an efficient way. In some systems, this operation can be done directly in the board that captures the image.

It must be noted that, using this strategy, we aggregate to the evaluation by attributes, a typical characteristic of the evaluation by variables, by minimizing a natural restriction of evaluation by attributes (analysis subjective basis) and using a notable potential of the evaluation by variables (decision with quantitative basis).

Once defined each pixel situation, this information can be used to identify defects. It depends on each productive process, but it is possible to use, as a general rule, the notion that the incidence of a value below the limits identifies defects associated to the lack of a given characteristic in the property under study, while, when the superior limit is surpassed , there is excess of the characteristic under study. In this way, it is justified the utilization of the indicated function by g(x,y). For instance, low values of g(x,y) indicate that the wall tile has not received paint or enamel enough. Hence, there is “lack of material”. High values will indicate substances accumulation over the piece, i.e., “material excess”. In both cases, once pointed the defect, it is possible to associate more probable causes of occurrence for these situations and define corrective and preventive actions to the case under study.

IV. IDENTIFICATION PROCESS

The identification method concentrates much more attention when considering the extension of the defect. For this second case, it was developed a specific system, which determines information grouping of pixels presenting imperfections.

This method is more complicated than the first one (intensity analysis), but can indicate more precise results. We consider, at first, that many defects have very specific spatial characteristics that become perfectly defined if its real extension is known. For example, if we have a dark line in the piece (a kind of scratch). In this case, the extension of the defect is defined by horizontal, vertical or diagonal alterations in the piece representation structure. A hole on the external area of the piece, for example, will be restricted to a minor area, having a rounded shape. A spot will be characterized by a feature “spreading”. Hence, the basic question will be to define where the defect starts and where it ends. This aspect is not always easy to be solved.

There are two approaches which could be used in this problem. First, it can be considered a pixel as the imperfection “center” and analyze its borders, in order to verify until where it affects the piece. This method is similar to the thresholds determination. In fact, it is desired to determine pixels whose characteristics have values above a given threshold or below another threshold. In this specific case, we can use a local or dynamic threshold model [1], [2]. This method presents some restrictions. It can be observed, initially, that will be a problem to determine the defect “center”, and after that, develop an iterative method in order to observe the area under study. Besides this, it is necessary to develop analytical proceedings to evaluate where the function f(x,y) values were changed, where occurred inflections, local or global maximums can be required, and it is not usually available an expression to represent this function, but just punctual values, in each pixel.

A possibility to solve this question would be to determine values to the Weszka operator [2] T(x,y) which could define the interval in which f(x,y) was affected by the defect. Empirically tested, the method did not work, because the analysis mixes areas influenced by the imperfection incidence with piece tonality oscillations, which do not configure itself as a defect. So, as an example, if a clear piece has a printed image having clear tonalities, the Weszka operator considers it as a defect. Another observed problem the following: if there is an imperfection in the printed image, the system can not separate the area of that has no conformity from the rest of the image. We have worked with a “ladder function”, capable of analyzing tonality gradations and also fixing, to each gradation, what *is* and what *is not* conform. This method is not computationally efficient, since the function would have to be tested in a large number of intervals, with several operations repeating itself to each pixel.

Facing the imposed restrictions to this first method, it was tried a second proposal. Here, the idea was not to define “occurrence centers” but, on the contrary, to delimit the area where the imperfection was observed. This method eliminates immediately some restrictions imposed to the previous proceeding. Here it is not necessary to define imperfection areas centers neither to determinate an expression for the function f(x,y), which characterizes the image representation basic property, because it will be done punctually.

From the computational point of view, the method can be efficient if using results taken from the previous programs (detection system 1 e 2). The Weszka operator is still a restriction, because the T(x,y) can not be defined for the areas affected by the imperfection. However, in this case, it can be taken a simple alteration that solves the problem: instead of applying the threshold for the area affected by the imperfection, it is applied the threshold for the whole image, it means, it is determined S(x,y), in such a way that: (1) $S(x,y) = f(x,y)$ if $T_1(x,y) < f(x,y) < T_2(x,y)$; (2) $S(x,y) = -k$ if $T_1(x,y) \geq f(x,y)$; (3) $S(x,y) = k$ if $T_2(x,y) \leq f(x,y)$.

We will always have $k > 0$ if there is at least one point in the areas (2) or (3). It is possible to normalize the described area in (1) taking $S(x,y) = 0$, what creates an interesting symmetry in the image analysis. Therefore, it is desired to associate the function S(x,y) to each image point. From the values taken by S(x,y) it is defined the defect extension, what makes its identification easier.

Finally, it is necessary to determine an algorithm capable of mapping the image areas, attributing them values of S(x,y) and, mainly, grouping the pixels according to S(x,y) values, assumed by them. This process is very hard, given the variety of existing situations. The program described below solves this problem, and creates conditions (as it is seen in the results discussion) in order to make an imperfection efficient identification possible in an efficient way, besides providing statistic analysis of each identified clustering and pixels contained in it.

V. COMPUTATIONAL ALGORITHM FOR DEFECT IDENTIFICATION

The basic algorithm of the defect identification system is the following:

Objective: We want to determine the pixel clustering that present specific features (for instance: it has a basic defect or present nonconformity with some pattern). Such clustering set allows determining, exactly, the extension of the affected area by the occurrence of an imperfection in the studied piece, what facilitates the defect identification.

Basic utilization: In case of monochromatic analysis, the program aggregates points presenting superior or inferior gray levels compared to established limits, what configures the presence of spots and, consequently, the defects occurrence. In the case of polychromatic analysis, once fixed a basic parameter (such as intensity, saturation or chromatic tone), the program aggregates all pixels presenting superior or inferior values of these properties to well determined limits, what configures lack of conformity with a certain pattern. In both cases, it is made an analysis of the parameter under study, from results of imperfections detection programs. This program (system 3) always starts with results from the programs of the systems 1 and 2). Before utilizing the system 3, it is necessary to apply a preprocessing in the model results, in order to determine the function S(x,y) values: they define thresholds for the image representation structure of the piece under inspection.

General Characteristics*:* The program uses pieces representation matrixes and results obtained came from operations developed on these matrixes. In the monochromatic case, the matrix associates gray levels to each one of the pixels. Detected imperfections are described as incompatible gray level values (facing the considered patterns or even due to basic defects); in the case of polychromatic, the matrixes associate specific parameters of each pixel. Detected defects are described in terms of a certain property not attending the conformity relationship with the patterns. In general, for the polychromatic case, the more relevant parameters are chromatic tone, intensity, saturation and value, in the case of HSI and HSV systems; in the case of RGB system, the parameters are the primary colors that each pixel contains (the primary colors, in this case, are the green, the blue, and the red). If necessary, the system can operate with complementary colors, such as the cyan, magenta and yellow. It is just necessary to adapt the environment in which the analysis is done.

Specific characteristics: The program operates directly at the studied pieces. It is necessary the availability of results from the defects detection programs (systems 1 and 2), in order to identify areas containing basic defects or presenting no conformity with a certain pattern. The results must pass by a preprocessing, which can be done in the own clustering program or in an external program. Reference values for the clustering program will always be the same ones used in the defect detection program. The program provides the map of the whole area affected by imperfection, and allows determining the area borders and the imperfection shape. The program develops statistic analysis of each clustering and pixels that compose it.

Conceptual basis*:* The program utilizes the tested principles listed in the previous item, it means, it tries to apply the Weszka operator to determine the image segmentation thresholds. The operator, described as the function S(x,y), presents specific values. For instance, 0 denotes that there is no imperfection in any pixel of that piece area; k values denote pixels clustering, whose image property values under study are above the limit; -k values denote pixel clustering, whose image property values are below certain limits. Therefore, for example, in the monochromatic analysis, using the function g(x,y) = 255 - f(x,y) as image representation, the -1, -2 and -3 values for S(x,y) show that exist, respectively, 1, 2 and 3 clusterings of pixels whose are below the limit assumed in $L_1 = 40$, what means the occurrence of bright spots in the piece. Values 1, 2 and 3 for S(x,y) denote the presence of 1, 2 or 3 clusterings, respectively, of pixels whose are above the limit assumed (for example, $L_2 = 100$) and denote the occurrence of dark spots. It is important to observe that every operation of this program comes from results of the imperfection detection programs. Hence, it is observed that the whole individual conceptual support of the System 3 involves the theoretical basis used in the defects detection system, besides incorporating images segmentation concepts.

The model structure: The model structure that serves as a program basis is very hard, because it must consider every possibility of clustering formation of points which can be formed in the image. It is an exhaustive analysis that involves a considerable number of different situations. The model utilizes a main program, that accesses specific data, cluster them and generates the outputs. The data are referred to results from imperfections detection programs (mainly from system 2). The outputs will show the clusterings and generate statistics respecting each one of them. The clustering process uses to be difficult. Initially, the clustering matrix is initialized with specific values. A border analysis to each matrix pixel is then made, by using a systematized proceeding that includes lateral advance through columns, vertical advance through lines and a diagonal advance utilizing the main and secondary matrix diagonals. Once concluded the analysis, it is given a code to the clustering, according to the values assumed by the function S(x,y). The process then starts to the next neighbor. It seems to be reasonable to suppose that the selection of neighbors follows an extremely well defined and organized process in order to provide an efficient execution of the program and also to avoid the analysis exclusion or repetition of any point.

The program structure is composed by the following elements:

Inputs: A matrix whose inputs have the following characteristics: they are integer numbers between two specific values (for example, between 0 and 255). This matrix is the image representation. We use also the outputs of the imperfection detection programs (systems 1 and 2).

Processing: Preprocessing**:** It is applied the function S(x,y) to the program outputs of the systems 1 and 2. The preprocessing output, hence, will be the original matrix transformed to a matrix of integer values equal to 0; k or -k; Clustering: Every pixel having not null values associated to them is clustered in the several matrix lines and columns. The program identifies and counts them. Therefore, this is the program operation basic scheme: (a) Data from the original matrix are transformed in 0, 1 and -1 if the defect detection program considers the pixel conform the pattern (0), not conform by being above the reference value taken as superior limit (1) or below the inferior limit (-1); (b) The pixels are clustered according to the values assumed; (c) Counters like 1, 2, ... identify the clusterings above the limit, and -1, -2, ... for those pixels clusterings that are below the limit. The k values identify the clustering and count them; (d) once identified the groupings, the pixels that compose them are selected; (e) the statistic analysis of pixels and clusterings are made.

Outputs*:*

(a) Pixels clusterings that are above the reference value used by the respective imperfection detection program, occurred at rows and columns level;

(b) Clustering identification (Code k, k: 1, 2, 3, ...);

(c) The program informs also how many different clusterings occurred and how many pixels remained in each clustering (clustering size) as well identifies which pixels remained en each grouping (pair (x,y) that shows the pixels position in the matrix);

(d) It is repeated the proceeding for every pixel below the reference value used by the respective imperfection detection program, designating them with -k.

VI. APPLICATION EXAMPLES

The program was applied to 6,000 pieces, in several situations. It was used the following limits: L_1 = 40 and L_2=100. The program classifies correctly 94% of the situations and did not decide what to do in the rest of the cases. In a very simple way, the following examples are referred to the monochromatic pieces analysis, using gray levels, through the use of system 1 and report some situations found.

Example 1: Piece Nº 55:

Original Matrix (Output from System 2

191	181	180	189	090	080	087	089	099
190	191	160	091	099	080	082	087	088
190	080	085	084	095	082	084	089	087
190	088	087	071	079	087	086	085	081

Transformed Matrix
(Output from Preprocessing)

1	1	1	1	0	0	0	0	0
1	1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Clusterings above the limit: 1
Clusterings below the limit: 0
Clustering above the limit: Clustering: 1.
Number of elements: 9.
Elements: (1,1) (1,2) (1,3) (1,4) (2,1 (2,2) (2,3) (3,1) (4,1)

Clustering in the original matrix

1 1 1 1
1 1 1 0
1 0 0 0
1 0 0 0

The coded matrix

X X X X
X X X 0
X 0 0 0
X 0 0 0

Example 2: Piece Nº 75:

Original Matrix (Output from System 2

171	169	080	087	070	080	087	087	087
166	070	060	070	077	080	082	087	088
070	080	085	084	075	082	084	087	087
070	088	087	070	077	087	186	185	181

Transformed Matrix
(Output from Preprocessing)

1	1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1

Clusterings above the limit: 2
Clusterings below the limit: 0
Clustering above the limit: Clustering: 1.
Number of elements: 3.
Elements: (1,1) (1,2) (2,1)
Clustering above the limit: Clustering: 2.
Number of elements: 3.
Elements: (4,7) (4,8) (4,9)
Clustering 2 in the original matrix
0 0 0 0
0 1 1 1

The coded matrix
0 0 0 0
0 X X X

Example 3: Piece Nº 102:

Original Matrix (Output from System 2

091	081	080	189	090	080	087	089	099
090	091	060	091	199	080	082	087	088
090	080	085	084	195	182	084	089	087
090	088	087	071	179	187	086	085	081

Transformed Matrix
(Output from Preprocessing)

0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0
0	0	0	0	1	1	0	0	0
0	0	0	0	1	1	0	0	0

Clusterings above the limit: 1
Clusterings below the limit: 0
Clustering above the limit: Clustering: 1.
Number of elements: 9.
Elements: (2,5) (3,5) (3,6) (4,5) (4,6)

Clustering in the original matrix

1 0
1 1
1 1

The coded matrix

X 0
X X
X X

Example 4: Piece Nº 164:

Original Matrix (Output from System 2

030	035	080	087	070	080	087	155	157
031	070	060	070	077	080	082	087	159
070	080	085	084	075	082	084	087	087
070	088	087	070	077	087	186	185	181
027	088	089	090	091	092	094	095	099
029	028	088	087	086	082	080	166	167
025	021	081	082	086	091	090	160	162

Transformed Matrix
(Output from Preprocessing)

-1	-1	00	00	00	00	00	1	1
-1	00	00	00	00	00	00	00	1
00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00
-1	00	00	00	00	00	00	00	00
-1	-1	00	00	00	00	00	1	1
-1	-1	00	00	00	00	00	1	1

Clusterings above the limit: 2
Clusterings below the limit: 2

Clustering above the limit:
Clustering: 1.
Number of elements: 3.
Elements: (1,8) (1,9) (2,9)
Clustering 1 in the original matrix:
1 1
0 1
The coded matrix
X X
0 X

Clustering above the limit:
Clustering: 2.
Number of elements: 4.
Elements: (6,8) (6,9) (7,8) (7,9)
Clustering 2 in the original matrix
1 1
1 1

The coded matrix
X X
X X

Clustering below the limit:
Clustering: 1.
Number of elements: 3.
Elements: (1,1) (1,2) (2,1)
Clustering 1 in the original matrix:
-1 -1
-1 00

The coded matrix
X X
X 0

Clustering below the limit:
Clustering: 2.
Number of elements: 4.
Elements: (6,1) (56,2) (7,1) (7,2)
Clustering 2 in the original matrix
-1 -1
-1 -1

The coded matrix
X X
X X

VII. CONCLUSIONS

This is a simple program in its execution, in spite of being hard in its development, because there are many situations to be considered. Its utilization depends on the defect detection program executing and also on considering as valid and acceptable its results, as well depends on a preprocessing applied to those program results.

There is no restriction concerning the image quality: this question will depend on the kind of program used in the imperfection detection, as well it is not considered as relevant to analyze the defect occurrence dependence related to the piece area under study. The program will be more useful when the image analysis is not done at the global level, but particularizing the study for determinate situations.

A relevant feature is to observe the program flexibility. It can be used the limits judged necessary or convenient to each case. If there are no such limits, it is possible to utilize the limits proposed by some programs of the system 2. This decision must precede the imperfections detection program application whose results give the inputs for the system 3.

A result analysis shows that did not occurred errors in the program application. Once tested the most diverse situations, the program has structured and identified the pieces groupings correctly in every case. But in 6% of the cases, the program was not able to reach a result.

From the program results it is possible to create defects patterns according to the observed clusterings. The examples that follow show some patterns which could be created from the program:

Outputs from system 3

Case 1:
Characteristic: Just one affected pixel
Identified defect: Hole in the center.

0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	1	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Case 2:
Characteristic: Four isolated affected pixels
Identified defect: Holes in the different areas.

0	0	0	0	0	0
0	0	0	0	0	0
1	0	0	0	1	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	1	0	0	1
0	0	0	0	0	0

Case 3:
Characteristic: Three affected pixels
Identified defect: Hole in the border.

0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	1
0	0	0	0	0	1
0	0	0	0	0	1

Case 4:
Characteristic: Horizontal arrangement
Identified defect: Scratch type 1.

0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	1	1	1	1
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Case 5:
Characteristic: Crossed arrangement
Identified defect: Scratch type 2.

0	0	0	0	0	0
0	0	0	1	0	0
0	0	0	1	0	0
0	0	1	1	1	1
0	0	0	1	0	0
0	0	0	1	0	0
0	0	0	0	0	0

Case 6:
Characteristic: Vertical arrangement
Identified defect: Scratch type 3.

0	0	0	0	0	0
0	0	0	0	0	0
0	1	0	0	0	0
0	1	0	0	0	0
0	1	0	0	0	0
0	1	0	0	0	0
0	1	0	0	0	0

Case 7:
Characteristic: Undefined shape
Identified defect: Bright spot

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{matrix}$$

Case 8:
Characteristic: Undefined shape
Identified defect: Dark spot

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{matrix}$$

Case 9:
Characteristic: Matrix edges
Identified defect: Border defect

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{matrix}$$

Case 10:
Characteristic: Matrix corner.
Identified defect: crack in the corner of the piece

$$\begin{matrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

It is worth to observe that, once defined these patterns, it is possible to aggregate them to the program, what would facilitate even more the defect identification.

REFERENCES

[1] Fu, K. *Syntactic Pattern Recognition*. Englewood Cliffs, Prentice-Hall, 1982.

[2] Weszka, J. Survey of Threshold Selection Techniques. *Comput. Graphics Image Processing*. No. 7, 1978. pp. 259-265.

[3] Eierman, M. A.; Hungerford, B. C. Modeling Tool Impact on Defect Identification in IS Design. *Information Systems and Management*. Fall 2006: Vol. 21 No. 2.

[4] Woodward, J. H. Stains for the Determination of Paper Components and Paper Defects. *Microscopy Microanalysis.* 8 (Suppl. 2), 2002

[5] Zhang , D. *et al.* A New Method of Defects Identification for Wire Rope Based on Three-Dimensional Magnetic Flux Leakage. *Journal of Physics: Conference Series.* **48** 334-338. 2006.

[6] Engelhardt, M.; Schanz, M.; Stavroulakis, G.; Antes, H. Defect identification in 3-D elastostatics using a genetic algorithm. *Optimization and Engineering.* Vol. 7, N. 1, March 2006.

[7] Kwak. C.; Ventura J.; Tofang-Sazi K. A neural network approach for defect identification and classification on leather fabric. *Journal of Intelligent Manufacturing*. Volume 11, Number 5, October 2000, pp. 485-499(15)

[8] Kojima, F. Identification of crack profiles using genetic programming and fuzzy inference. *Journal of Materials Processing Technology*, Volume 108, Issue 2, Pages 263-267. 2006.