BINARY PATTERN PROCESSING FRAMEWORK FOR PERCEPTUAL FAULT DETECTION

M. Köppen and A. Soria-Frisch and Th. Sy

Fraunhofer IPK Berlin, Dept. Pattern Recognition
Pascalstr. 8-9, 10587 Berlin, Germany
{mario.koeppen|aureli|thorsten.sy}@ipk.fhg.de

ABSTRACT

This paper presents a framework for the binary processing of perceptual faults in texture analysis. The framework is composed of a chain of Binary Pattern Processing Modules (BPPM). Each BPPM is responsible for a class of perceptual faults. In a BPPM, several image operators are applied onto the preprocessed image. The result images are fused into a binary image, and a connected component analysis is performed for extracting fault indicating binary components. The whole image is described by a set of fuzzy features, and class memberships are derived by using the Choquet integral. Configurations of the framework for the perceptual fault classes strong-contrast faults (holes, cracks), broad-range faults and low-contrast faults are given, which are for all non-trivial cases based on the 2D-Lookup algorithm. The presented framework is able to work in the context of objective reproduction of subjective decisions about texture surface quality, and may be easily adapted to new technological needs by designing further components or by modifying existing ones.

1. INTRODUCTION

Traditionally, texture analysis for visual inspection is applied to the detection of functional faults in textured materials. Here, the task is to detect a fault, assign it to a fault class, and deliver processing signals to the superposed system. However, economic demands let the detection of perceptual faults become more and more important. The task now is to detect perceptuable anomalies of any kind, which might disturb the total visual appearance of the surface texture. Only part of such faults are of functional nature. This is especially related to the inspection of materials composed of organic substances, or to natural textures. Frankly spoken, the customer rejects materials with perceptual faults not due to malfunction but due to their "uglyness," with some relevance to a possible malfunction.

Wooden, metal, paper or carpet surfaces, to name a few, have been a long-termed research target in this direction [16]. However, a general attempt to treat this class of inspection problems has not been considered so far.

Perceptual fault detection is mainly characterized by the subjective nature of their interpretation. Different persons may either rely on the same bad or good impression of a texture or not. Also, perceptual faults are sticked on both, local and global image properties. While a floor tile might succesfully pass all functional fault checks by itself, it may fail in a pavement due to a perceptual fault (e.g. a large-scale regularity).

This paper presents a general approach to the detection and classification of perceptual faults. The approach is given as a framework, the components of which (Binary Pattern Processing Modules) are based on the combination of binarization approaches with fuzzy feature fusion in order to get fuzzy class membership values and to combine them in a whole texture processing framework.

While the presented framework fits well into the standard feature classification approach to texture analysis [7] [8], it solicits the binarization approach to texture analysis and reflects the subjective nature of perceptual fault detection.

This paper is organized as follows. Section 2 summarizes necessary prerequisites for the understanding of the presented framework, the binary texture processing, the 2D-Lookup algorithm and fuzzy fusion operators. Then, the framework is presented in section 3. The Binary Pattern Processing Modules components are presented in the subsections of section 3. Finally, special settings of the framework to cover a broad range of perceptual faults, as they have been used in industrial projects, are presented in section 4. This section may be considered as surrogat for the more common "Application example" section, but the design guidelines are the more favoured aspect here. The paper ends with a short discussion on how to measure the performance of such systems by providing a real-world example in section 5, the summary and the reference.

Besides of presenting a framework, which can be applied to a wide range of modern surface inspection tasks, this paper also presents the aspects of using fuzzy fusion as feature classification approach, and the new auto-lookup algorithm.

2. PREREQUISITES

This section recalls some concepts, which are of importance for the understanding of the presented Binary Pattern Pro-

cessing framework. The issue of binary texture processing will be shortly discussed in the following subsection. Then, the 2D-Lookup algorithm will be recalled. This algorithm is reused in several aspects of the framework according to the high degree of freedom for its configurational setting. Finally, aspects of fuzzy fusion are remarked. This is necessary, since fuzzy fusion approaches are not very often considered as fusion approaches in image processing applications.

2.1. Binary texture processing

Usually, textures are processed based on grayscale (or color) information (as exemplified by textbooks on image processing as [4]). However, images representing results of texture processing are of lower depth, i.e. this may be binary images indicating texture background by a grayvalue White, or label images indicating several texture classes.

Hence, the binary texture processing appears to be an alternative approach to grayvalue texture processing by performing one or more binarization procedures in parallel and evaluating the binary images in order to obtain a result image, or to get features for classification (see [19] [22] [23] for related approaches).

The advantage of using binary pattern processing for the detection of perceptual faults can be seen from the fact that the representation of a perceptual fault may depend on the chosen binarization procedure. Hence, such approaches do not rely on the grayvalue appearance of the fault alone, but also on the binary components (or binary patterns) obtained from the fault region by suitable binarization procedures.

2.2. 2D-Lookup

The 2D-Lookup algorithm stems from mathematical morphology [17], [18]. It was primarily intended for the segmentation of color images. However, the algorithm can be specified for its use on grayvalue images as well.

For starting off the 2D-Lookup algorithm, two operation images 1 and 2, which are of equal size, need to be provided. This means that for applying the 2D-Lookup algorithm, it is necessary to determine two image processing operations, which are applied to the original image. The 2D-Lookup algorithm goes over all common positions of the two operation images. For each position, the two pixel values at this position in operation images 1 and 2 are used as indices for looking-up the 2D-Lookup matrix. The matrix element, which is found there, is used as pixel value for this position of the result image. If the matrix is bi-valued, the resultant image is a binary image.

Let I_1 and I_2 be two grayvalue images, defined by their image functions g_1 and g_2 over their common domain $P \subseteq \mathcal{N} \times \mathcal{N}$:

$$\begin{aligned} g_1 : &\quad P \to \{0, \ldots, g_{max}\} \\ g_2 : &\quad P \to \{0, \ldots, g_{max}\} \end{aligned} \tag{1}$$

The 2D-Lookup matrix is also given as an image function l, but its domain is not the set of all image positions but the set of tupels of possible grayvalue pairs $\{0, \ldots, g_{max}\} \times \{0, \ldots, g_{max}\}$,

$$l : \{0, \ldots, g_{max}\} \times \{0, \ldots, g_{max}\} \to S \subseteq \{0, \ldots, g_{max}\}. \tag{2}$$

Then, the resultant image function is given by:

$$\begin{aligned} r &: P \to S \\ r(x,y) &= l(g_1(x,y), g_2(x,y)). \end{aligned} \tag{3}$$

In standard applications, every grayvalue is coded by eight bit, resulting in a maximum grayvalue of 255. Also, the domain of the image function is a rectangle. In this case, the 2D-Lookup is performed by the following (object-oriented) pseudo-code:

```
for x=0 to img width -1 do
begin
   for y=0 to img height-1 do
   begin
      g1 = g1(x,y)
      g2 = g2(x,y)
      out(x,y) = l(g1,g2)
   end y
end x
```

To give a simple example for the 2D-Lookup procedure, $g_{max} = 3$ is assumed in the following. Let

g_1 :

0	1	2
0	3	3

and g_2 :

2	3	1
2	3	2

be the two input images and the 2D-Lookup matrix be given by

l :

g_1 / g_2	0	1	2	3
0	0	0	1	1
1	0	1	2	2
2	1	2	3	3
3	2	3	3	2

Then, the resultant image is

r :

$l(0,2)$	$l(1,3)$	$l(2,1)$
$l(0,2)$	$l(3,3)$	$l(3,2)$

$=$

1	3	2
1	2	3

In case that the 2D-Lookup matrix is derived from the 2D histogram of the two images (with $l(g_1, g_2)$ being the number of simultaneous occurrences of grayvalue g_1 in image I_1 and grayvalue g_2 in image I_2 at the same location, normalized by a constant C), the 2D-Lookup algorithm is referred to as *Auto-Lookup.*

2.3. Fuzzy fusion approaches

Fuzzy fusion is a long-termed discipline of fuzzy set theory. Starting of with Sugenos work on fuzzy integrals [21], it has become a more and more regarded approach in fuzzy control theory.

Today, several fuzzy fusion operators have been studied. Major groups of such operators are

- Fuzzy integrals (e.g. Sugeno integral, Choquet integral [6]),
- Aggregation operators (e.g. Ordered Weighted Averaging [24], Weighted Min Aggregation [1]), and
- Triangular Norms and their co-norms [11].

The advantage of using them in image processing has been considered in [20] [2] [12] [14]. Three important aspects of using fuzzy fusion operators can be noted here: the admission of an infinity of operators; the possibility of different weightings for coalitions of features (this can not be adequately represented by linear weighting); and the flexible representation of subjective reasoning.

Fuzzy fusion operators are used in the presented framework for the derivation and evaluation of features, but for some of the binarization algorithms as well.

3. THE BINARY PATTERN PROCESSING FRAMEWORK

The Binary Pattern Processing Framework is basically composed of a processing chain of alternating Binary Pattern Processing Modules (BPPM), each of which having the same internal structure, and additional testing modules (see fig. 1). The texture analysis result is represented as a vector of class memberships. As an example, if there are two classes ("no perceptual faults" and "perceptual faults"), the output vector $(0.3, 0.7)$ will indicate a texture more belonging to the fault class.

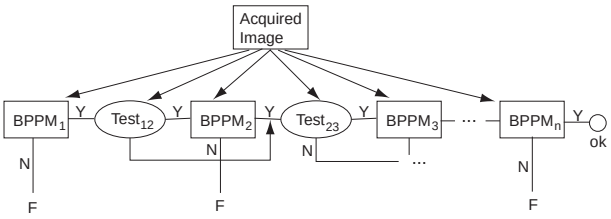

Fig. 1. The Binary Pattern Processing Framework.

Each BPPM has access to the acquired image and to the evaluation of the foregoing testing module. The evaluation (the module output) is given by the class vector, and it may be composed with the class vector of the foregoing module. Hence, the processing of each BPPM is independent of the processing of the others, but may refer to the results of the foregoing modules.

This indicates an ordering of the BPPMs: modules for the detection of the more frequent faults, or the more technical faults, or the more simple to detect faults should come first in the chain.

3.1. Binary pattern processing module

Each single BPPM is designed for a special fault class related to the faults appearance. A BPPM is composed of a Preprocessing submodule, a set of k (possibly binarization) procedures (those may use the same binarization algorithm, with differing numerical or structural parameters), a Binary Fusion submodule for fusing the k binary images into one single image, and a Fuzzy Feature Fusion submodule, for which fuzzy features derived from the connected components of the binary fused image are evaluated on base of the Choquet fuzzy integral (see fig. 2).

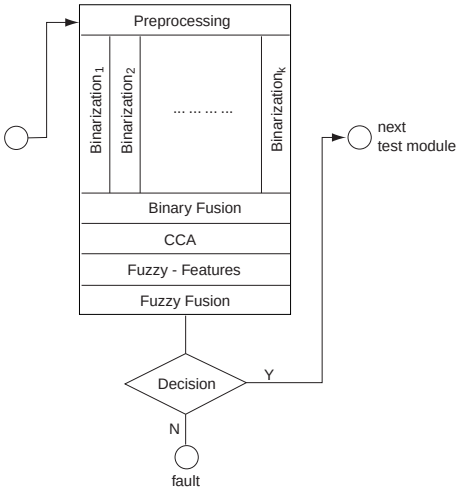

Fig. 2. Structure of a Binary Pattern Processing Module.

3.1.1. Preprocessing

Preprocessing is a necessary step of all image processing approaches. In this case, it has only to be noted that each BPPM may have its own preprocessing.

3.1.2. Binary fusion

The binary fusion is the "heart" of each BPPM. A set of k algorithms is performed in parallel, giving k result images of same size and resolution as the acquired and preprocessed image. Then, a fusion procedure derives a binary image from these k images. The foreground (or Black pixels) of that binary image are candidates for perceptual faults. Examples for the specification of a BPPM will be given below (see section 4).

3.1.3. Fuzzy feature extraction

The Fuzzy Feature Fusion submodule derives the evaluation from the binary fused image. At first, a Connected Component Analysis (CCA) is performed in order to select connected components from the binary fused image, which are perceptual fault regions, and to remove noise that remained from the binary fusion. Then, the remaining connected components are measured (see figure 3) and a set of absolute geometrical features is derived for each component (e.g. height, width, area, perimeter, roundness). Also, features are derived for the description of the component distribution (e.g. number of components, variation of geometrical component features).

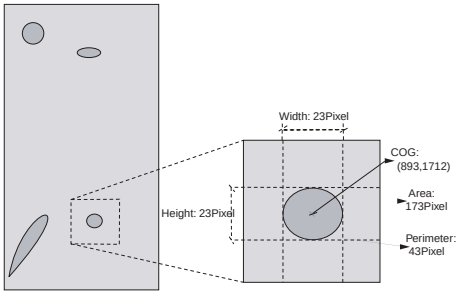

Fig. 3. Measuring of binary patterns after CCA.

Then, these fuzzy values are fused by using an Ordered Weighting Averaging (OWA) operator in order to get the description of a single meta-component instead of the various descriptions of all components. The goal is to obtain a global description of the elements under analysis instead of the individual fault descriptions. This description is then fuzzified when necessary by defining linguistic terms and assigning fuzzy membership values

To summarize: the set of derived features at the end are related to the binary fused image *as a whole*: either they describe the global distribution of the components, or they describe a meta-component derived from the fusion of the features of all single components.

3.1.4. Fuzzy feature fusion

Finally, for each perceptual fault class the fuzzy features are fused into a value from $[0, 1]$ by using the Choquet integral.

More formally: be a_i such a fuzzy feature, with $a_1 \leq a_2 \leq .. \leq a_n$ and $a_0 = 0$, then the Choquet integral is computed from the following formula:

$$(C)\int f d\mu = \sum_{i=1}^{n}(a_i - a_{i-1}) \cdot \mu(\{x \mid f(x) \geq a_i\}). \quad (4)$$

The fuzzy feature fusion uses several parameters, which have to be adapted to the specific goals of the BPPM: component filtering parameters, membership functions, OWA weights and the fuzzy measures for the Choquet integral. Since all those parameters have a "meaning" in the sense that the effect of their settings can be directly verified by the experienced user, the task of parametrization of the Fuzzy Feature Fusion submodule can be handled. However, adaptive procedures for the setting may be considered for future works. So far, some experiments were done for the automatic setting of the fuzzy measures for the Choquet integral by using an optimization technique based on quadratic programming [5]. However, those experiments did not give values as good as empirically adapted ones. This may lay on the necessity for the optimization technique to have a clear distinction between the memberships of the classes. In case of unclear definition of them the quadratic programming delivers trivial solutions.

3.2. Testing modules

As already noted, the testing modules, which link the consecutive BPPMs, are optional design components. Their main purpose is to bypass a BPPM computations, if there is no evidence for the faults, which are processed by that BPPM. This means, the testing module has to be implemented as a fast testing routine based on reduced information as e.g. histograms. The obvious advantage is to reduce the processing time for the whole framework. However, in some cases (as the low-contrast faults considered in sec. 4.3), there will be no easy design of such a procedure.

4. DESIGN GUIDELINES FOR THE FRAMEWORK

So far, three basic designs of a BPPM have been identified. Those designs differ only in the steps until CCA is performed, i.e. in the preprocessing, binarization and binary fusion steps.

4.1. Strong-contrast localized faults

The setup for a local fault detection module is refering to a strong contrast of the faults' appearance. The acquired images are preprocessed by using a highpass filter. Then, only one binarization operation is applied, which may be an intervall thresholding. Here, intervall thresholding means that grayvalue intervalls are identified, for which the pixels in the thresholded image are set to Black (0), and White (1) for all pixels with grayvalues out of those intervalls (see fig. 4).

Since there is just one binarization, no binary fusion is necessary.

This is supposed to detect perceptual faults in the texture, the appearance of which is remarkably different from the undisturbed texture (very dark or very bright). Also, the faults are local in the sense that only a small, connected part of the total image is covered by the fault area.

This kind of modules may cover the functional faults among the perceptual faults (cracks, holes) as well and should be used at the beginning of the processing chain.

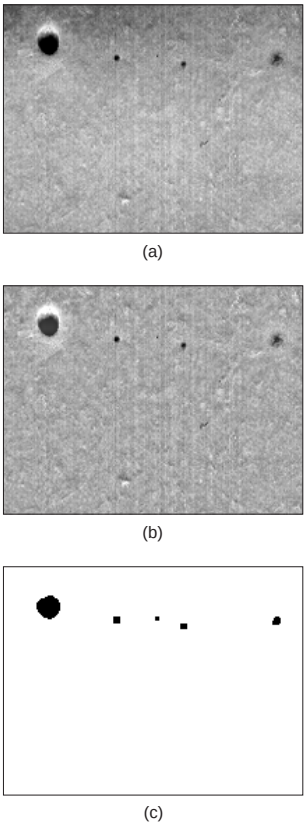

Fig. 4. Processing of a BPPM for detecting strong-contrast faults (holes in a texture): (a) acquired image, (b) highpass filtered image, (c) binary fusion and CCA of the preprocessed image.

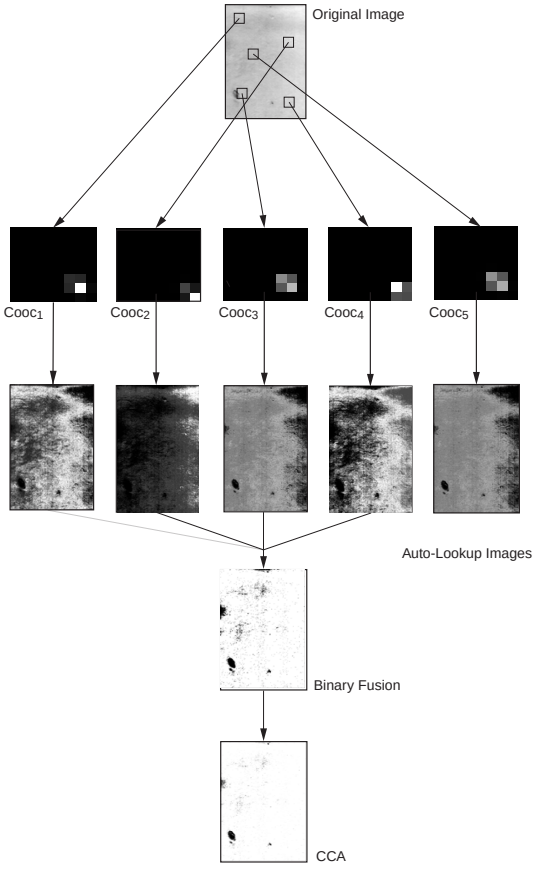

Fig. 5. Auto-Lookup procedure.

4.2. Long-range faults

An important class of perceptual faults are long-range faults. Here, the fault appearance is not related to a strong local contrast, but to a distortion within the global distribution of grayvalues within the image. Such faults may be detected by employing the auto-lookup procedure (see section 2.2) based on the co-occurrence matrix of a subset of pixels from the image (formally: I_1 is the original image, and I_2 is the original image shifted by the offset vector of the co-occurrence matrix). This can be qualified as a novelty filter, with dark regions indicating occurrences of grayvalue pairs, which are seldom in the image.

The procedure is illustrated in fig. 5. From the acquired image (no preprocessing is necessary), five co-occurrence matrices are derived at five randomly located windows. Then, for each co-occurrence matrix, the auto-lookup is performed on the whole image. Bright regions in the auto-lookup image stand for "typical" regions according to the grayvalue distribution within the window, and dark regions for "untypical" ones.

Now, all five images are fused into a single one. In case of the given example, the fusion procedure was simple: in case at least three out of the five images have a dark grayvalue at the same location, the pixel at that location in the binary fused image is set to Black, White otherwise.

The fusion is necessary, since the random location may lay on a fault boundary, thus giving ambiguous information in the auto-lookup image.

The effect of the CCA can also be seen in fig. 5.

4.3. Low-contrast faults

A third important class of perceptual faults are related to a very typical, but low contrast in the image grayvalues. In any cases, they are as local as the faults considered in sec. 4.1.

For their treatment, the already presented Lucifer2 framework may be used. This subsection will give a short summary of the Lucifer2 approach.

The purpose of the Lucifer2 framework is to design texture filters. Trained by user-provided examples, the adapted filters are able to separate a textured background from a foreground structure. Possible applications for these texture filters are: texture fault detection, texture border detection or handwriting extraction (on a bankcheck with textured background). These problems typically arise in fields like visual surface inspection on fabrics or optical document

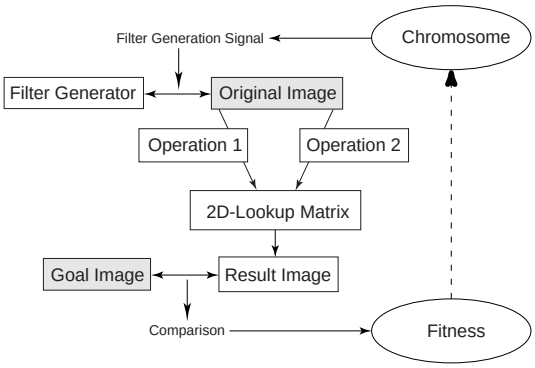

Fig. 6. The Framework for 2D-Lookup based texture filter generation.

preprocessing.

The framework (see fig. 6) is composed of (user- supplied) original image, filter generator, operation images 1 and 2, result image, (user-supplied) goal image, 2D-Lookup matrix, comparing unit and filter generation signal.

The framework can be thought of as being composed of three (overlapping) layers.

(1) The instruction layer, which consists of the user-supplied parts of the framework: original image and goal image. The user may also supply other components (operation 1, operation 2, 2D-Lookup matrix), for maintenance purposes.

(2) The algorithm layer performs the actual 2D-Lookup, once all of its components (original image, operation 1, operation 2 and 2D-Lookup matrix) are given.

(3) The adaptation layer contains all adaptable components of the framework (operation 1, operation 2, 2D-Lookup matrix) and additional components for performing the adaptation (comparison unit, filter generator).

For the instruction layer, the user interface has been designed as simple as possible. The user instructs Lucifer2 by manually drawing a (binary) goal image from the original image (by a photo retouching program as Photoshop). In this image, texture background is set to White and texture foreground (e.g. the texture fault, handwriting on a textured bankcheck background) to Black (see fig.7 for an example). Rest of the approach is data-driven. No special texture model has to be known by the user. There are no further requirements for the goal image.

The algorithm layer performs the 2D-Lookup algorithm. The algorithm decomposes the filter operation into a set of partial steps, each of which might be adapted to meet the user's instruction.

Adaptation is considered an optimization problem, and evolutionary algorithms [9] [3], esp. genetic programming [15], are used for performing this adaptation. The fitness

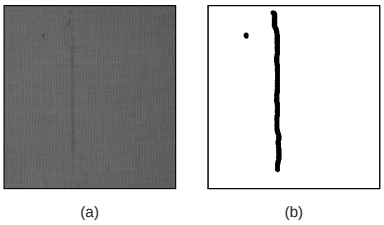

Fig. 7. Example for texture image containing fault (a) and goal image, as given by the user (b).

function is computed with the degree of resemblance between result image of an individual-specified 2D-Lookup and the goal image.

The Lucifer2 framework can be used to design the k binarization procedures within a BPPM. For doing so, an image part including the low-contrast perceptual fault area is cutted from the image, and a binary goal image is made by the user (note: those steps are done off-line and only once during the design phase of a BPPM).

Then, Lucifer2 may run $l > k$ times, and the k best results are selected as binarization procedures.

Fig. 8 gives an example for the design of algorithms for the detection of a class of contrast faults.

More details on Lucifer2 can be found in [10]. Generalization issues of the designed filters are studied in [13].

5. PERFORMANCE EVALUATION

In this section, the issues of performance evaluation of the proposed framework will be considered. Especially in this case, where subjective decisions are to be adopted, the question arises how to measure the quality of the adaptation. The basis of our study was a pool of 110 natural collagen sheets with different perceptual faults, which were inspected and subjectively evaluated by human experts.

What we did first was to separate the sheets as good as possible into the three classes. Afterwards we trained the modules. For the strong-contrast faults we took into account the number of faults, their location, density and dimension. In figure 9 a graphical representation of the results based on the fusion of these properties is given. The right, bright bar of each group indicates the number of elements in the computer-generated group that were wrongly classified. Independently of the mentioned problems, the correspondence between the human decision and the one of the computer is fair, but should become better by using a bigger training pool.

Another well known graphical representation is shown in figure 10.It shows the false acceptance and false rejection rate, also representing the strong-contrast faults. The crossing point of these two plots gives the equal error rate of about 21 %, which is a quite good result, if the complexity in the appearance of the faults is considered. This rate is widely used to demonstrate the performance of a system,

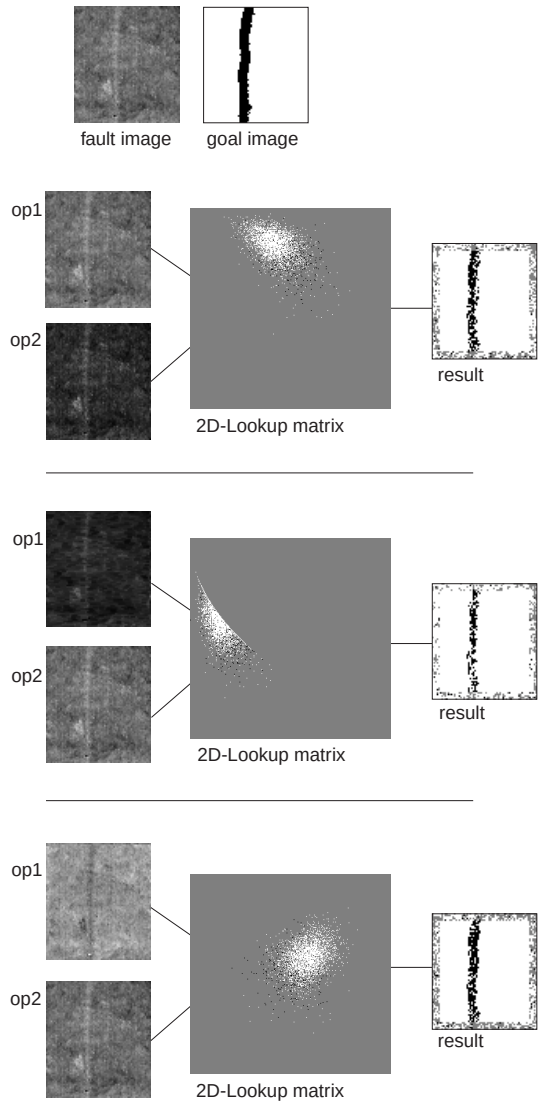

Fig. 8. The Lucifer2 framework applied three times to the design of a procedure for the detection of a low-contrast fault.

though neglecting the mentioned problems. So, this number doesn't e.g. represent the stability of the system.

6. SUMMARY

A framework was presented for the processing of perceptual faults and based on the binary texture processing paradigm. The framework is established from a chain of separately working binary pattern processing modules (BPPM). Each BPPM is suited for a special class of perceptual faults. Its processing starts from a preprocessed image and applies a set of k algorithms in parallel. Then, the processing result images are fused into one binary image. A connected component analysis solicits the patterns of the binary fused image, which are indicating perceptual fault locations. Then,

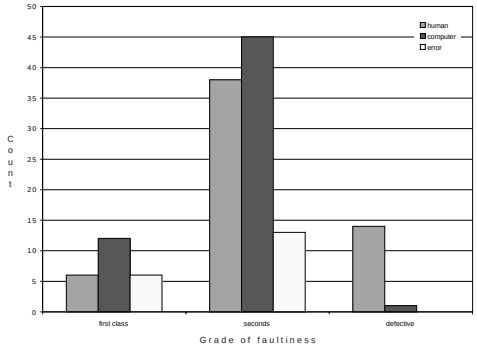

Fig. 9. Results of the fuzzy evaluation on the example of collagen sheets.

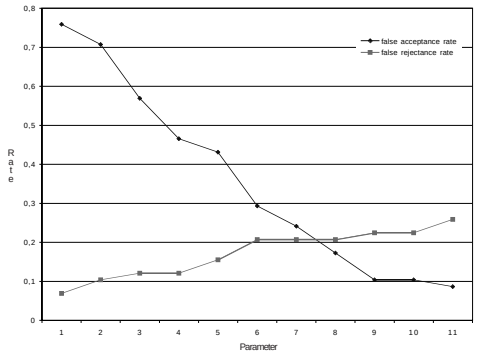

Fig. 10. Equal Error Rate.

a set of fuzzy features is derived for the joint description of global properties of the binary image, and the geometric description of a special structure, a pseudo-component with its features derived from aggregated features of all components of the binary fused image. Hence, there is one fuzzy feature set per image. These features are fused to a class membership degree by using the Choquet integral. There is one set of weights for the Choquet integral per class.

The class membership vectors are given to all following BPPMs in the processing chain of the framework. For reducing processing time, additional testing modules can be used.

The framework has been used for the evaluation of textured materials composed of organic substances. There, the special need for the processing of perceptual faults arose and gave the motivation for the presented design. The primary goal was to have an approach for an objective reproduction of various subjective material evaluations, and to provide a means for easy extension and reconfiguration of the system according to changes in production technology and perceptual fault importance.

The presented framework can be adapted to a broad range of perceptual fault classes. Three configurations for perceptual faults appearing as strong-contrasted, broad-ranged or low-contrasted ones were given. Besides of a unique fuzzy feature fusion approach, the BPPMs here are designed from a (simple) thresholding for strong-contrast faults, auto-

lookup for broad-range faults, and from the recently proposed Lucifer2 framework for the low-contrast faults. The suitability of those designs has been noted elsewhere.

Further work will concentrate on the partial use of adaptive procedures for the various parameter settings of the framework (besides of the one by genetic programming already in use for the Lucifer2 framework). First attempts were made on the setting of the fuzzy features subset weights for the Choquet integral.

Acknowledgments

A very special thank from the authors goes to the master degree students Rozbeh Alavi, Selma Redzebasic and Xiufen Liu from Technical University Berlin and Jing Zhang from Humboldt University Berlin for their technical help and their permanent support of the presented work.

7. REFERENCES

[1] Dubois, D., Fargier, H., Prade, H., "Beyond min aggregation in multicriteria decision: (ordered) weighted min, discri-min, lexmin", in: Yager, R. R., Kacprzyk, J. (eds.), "The ordered weighted averaging operators — Theory and applications", Kluwer Academic Publishers, Dordrecht a.o., 1997.

[2] Franke, K., Köppen, M., Nickolay, B., "Fuzzy image processing by using Dubois and Prade fuzzy norms", Proc. ICPR 2000, Barcelona, Spain, pp.518-521, 2000.

[3] Goldberg, D. E., "Genetic algorithms in search, optimization & machine learning", Addison-Wesley, Reading, MA, 1989.

[4] Gonzales, R. C., Woods, R. E., "Digital Image Processing", Addison-Wesley, Reading MA, 1993.

[5] Grabisch, M., Nicolas, J.-H., "Classification by fuzzy integral: Performance and tests", Fuzzy Sets and Systems (65), pp.255-271, 1994.

[6] Grabisch, M., Murofushi, T., Sugeno, M., "Fuzzy Measures and Integrals", Physica-Verlag, 1999.

[7] Haralick, R., Shanmugam, K., Dinstein, I., "Textural features for image classification", IEEE Trans. SMC, 3, (6), pp.610-621, 1973.

[8] Haralick, R., Shapiro, L., "Image segmentation techniques", Computer Vision, Graphics and Image Processing, 29, pp.100-132, 1985.

[9] Holland, J. A., "Adaptation in natural and artificial systems", MIT Press, Cambridge MA, 1975.

[10] Köppen, M., Teunis, M., Nickolay, B., "A framework for the evolutionary generation of 2D-Lookup based texture filters", Proc. IIZUKA'98, Iizuka, Japan, pp.965-970, 1998.

[11] Klir, G.J., Yuan, B., "Fuzzy Sets and Fuzzy Logic: Theory and Applications", Prentice Hall, NJ, 1995.

[12] Köppen, M., Franke, K., "Fuzzy Morphologies Revisited", Proc. IWSCI'99, Muroran, Japan, pp.258-263, 1999.

[13] Köppen, M., Zentner, A., Nickolay, B., "Deriving Rules from Evolutionary Adapted Texture Filters by Neural Networks", Proc. FUZZ-IEEE99, Seoul, Korea, pp.785-790, 1999.

[14] Köppen, M., Franke, K., Unold, O., "A survey on fuzzy morphology", Proc. PRIA-5, Samara, Russia, pp.424-427, 2000.

[15] Koza, J., "Genetic programming — On the programming of computers by means of natural selection", MIT Press, Cambridge, MA, 1992.

[16] Newman, T.S., Jain, A.K., "A survey of automated visual inspection", Computer Vision and Image Understanding, 61 (2), pp. 231-262, 1995.

[17] Serra, J., "Image analysis and mathematical morphology", Academic Press, London, 1982.

[18] Serra, J., "Image analysis and mathematical morphology. Vol. 2: Theoretical advances", Academic Press, London, 1988.

[19] Smith, G., Longstaff, D., "Using binary features in texture classification", Proc. DICTA95, Brisbane, Australia, pp.553-558, 1995.

[20] Soria-Frisch, A., Ruiz-del-Solar, J., "Towards a biological-based fusion of color and infrared textural image information", Proc. IEEE Workshop on Intelligent Signal Processing, Budapest, Hungary, 1999.

[21] Sugeno, M., "Theory of fuzzy integrals and its applications", Ph.D. thesis, Tokyo University, 1974.

[22] Topi, M., Timo, O., Matti, P., Maricor, S., "Robust texture classification by subsets of local binary patterns", Proc. ICPR 2000, Vol. 3, Barcelona, Spain, pp.947-950, 2000.

[23] Topi, M., Matti, P.,Timo, O., "Texture classification by multi-predicate local binary pattern operators", Proc. ICPR 2000, Vol. 3, Barcelona, Spain, pp.951-954, 2000.

[24] Yager, R. R., "On ordered weighted averaging aggregation operators in multi-criteria decision making", IEEE Trans. SMC, 18, pp.183-190, 1988.