A Block-Processing Approach Using Texture Analysis for Fabric Defect Detection

Luiz Antonio Buschetto Macarini
Universidade Federal de Santa Catarina – UFSC
Departamento de Automação e Sistemas
luiz.buschetto@posgrad.ufsc.br

Felipe Vieira Roque
Universidade Federal de Santa Catarina – UFSC
Coordenadoria Especial Interdisciplinar de Tecnologias da Informação e Comunicação
felipe.roque@posgrad.ufsc.br

Luan Casagrande
Universidade Federal de Santa Catarina – UFSC
Departamento de Computação
luan.casagrande@grad.ufsc.br

Tiago Oliveira Weber
Universidade Federal de Santa Catarina – UFSC
Departamento de Computação
tiago.weber@ufsc.br

Cristian Cechinel
Universidade Federal de Santa Catarina – UFSC
Coordenadoria Especial Interdisciplinar de Tecnologias da Informação e Comunicação
cristian.cechinel@ufsc.br

ABSTRACT

The quality control is an essential step in fabric industries. Detect defects in the early stages can reduce costs and increase the quality of the products. Currently, this task is mainly done by humans, whose judgment can be affected by fatigue. Computer vision-based techniques can automatically detect defects, reducing the need for human intervention. In this context, this work proposes an image block-processing approach, where we compare the Segmentation-Based Fractal Texture Analysis, Gray Level Co-Occurrence Matrix, and Local Binary Pattern in the feature extraction step. Aiming to show the efficiency of this approach for the problem, these results were compared with the same algorithms without the block-processing approach. A Support Vector Machine optimized by Grid-Search Algorithm was used to classify the fabrics. The database used, which is available online, is composed of 479 images from samples with defects and without it. The results show that this block processing approach can improve the classification results, achieving 100% in this work.

KEYWORDS

Fabric Defect Detection, Image Processing, Pattern Recognition, Machine Learning, Hyperparameter Tuning

1 INTRODUCTION

The industrial process' quality is becoming more important every day, where the cost reduction and process optimization are increasingly needed [1]. The application of image processing and machine learning techniques in the quality control process can lead to an improvement in the production system [2]. The number of works aiming to solve this problem is increasing, making this step more precise and improving the production process, where the main goal is to increase the quality of the products [3].

For example, various types of industries such as ceramic, glass, food and even fabrics are implementing these techniques to improve quality control [4]. Due to the high growth of the textile industry and the different manufactured products, it is necessary to create methods that guarantee the quality of the product in order to be competitive in the global market [5]. Textile products are manufactured for different purposes and with different technologies, where their classification varies according to standards. In many cases, this determines the quality of the product [6].

The Brazilian textile industry is the fifth-largest in the world. It is considered one of the most powerful and is placed into three more important sectors in the world's economy [7]. Furthermore, its success has a strong relationship with the capacity to reduce the defects in the textiles. The fabric is a product that is submitted to delicate processes, where it is difficult to get defect-free products. So, the quality inspection concerns to reduce the occurrence of defects in the early stages of production [8]. According to [9], defected textiles fabrics have their price reduced by 45% to 60% due to defects.

This inspection process needs to be done by specialized professionals [8]. However, the quality control can become exhausting, since the human visual system has its limitations and weaknesses [1]. Human beings can work in one activity like this during some limited time, getting tired in a couple of hours. Thus, the fatigue affects the human judgment [10]. Also, sometimes the inspector has to detect small defects located in a wide area that is moving through their visual field, resulting in a detection rate of approximately 70% [11].

In this context, this work proposes a comparative study of algorithms for fabrics defect detection. The methods Segmentation-Based Fractal Texture Analysis (SFTA), Gray-Level Co-Occurrence Matrix (GLCM) and Local Binary Pattern (LBP) with and without block-processing are compared. For classification, a Support Vector Machine (SVM) is employed. Furthermore, a new database is presented, being publicly available.

The work is organized as follows: Section 2 presents the works that are related to this. In Section 3 is presented the adopted methodology, describing the database and the algorithms used. Also, in

Section 4 the obtained results are presented, together with the discussions. Closing this work, in Section 5, the conclusion is presented, with future works.

2 RELATED WORKS

In [6], the authors made a comparison between SFTA and other algorithms to automatically select different patterned fabrics, yielding a good detection rate. Although they are also using SFTA, the goal in the present work is to detect defects, separating the good samples from the ones with defects. Robust results were presented, showing the efficiency of the methods in ten patterns of fabric.

In [12], the authors proposed an approach to detect fabric defects based on a statistical representation of patterns using Redundant Contourlet Transform (RCT), modeled by a finite Mixture of Generalized Gaussians (MoGG). The Bayesian Classifier was used to detect defective fabrics. The experiments were made with the TILDA database, where the approach showed good detection rates, with low false positives. Compared with other methods, the approach presented better results on several types of fabrics.

The authors from [13] also used the TILDA database. In their case, they applied Dual-Tree Complex Wavelet Transform (DTCWT) for defect detection in fabrics with Euclidean Distance Classifier in the classification step. The results showed that DTCWT overcomes the results obtained with Undecimated Discrete Wavelet Transform (UDWT). The highlight of this method is the approximate shift-invariance, an important property in pattern recognition applications.

In [14] the authors also used Euclidean Distance in the classification step to detect defects in yarn-dyed fabrics, but they proposed the use of autocorrelation function and GLCM to describe the features. The results showed that the approach can reach accurate results, where the highlight of this work is the adaptability of the algorithm.

In [15], the authors presented a scheme using local homogeneity to find defects in fabrics. In their work, a Discrete Cosine Transform (DCT) is applied to the H-image, that was divided into blocks (or windows). After, an Artificial Neural Network (ANN) is used to classify the defect accordingly to the training set. Although this work presented good results, the necessity of a window size definition is a limitation.

In [16] the authors developed a real-time machine vision system to detect defects automatically. The algorithm is based on the Wavelet Transform, double threshold binarization, and morphological operations. Besides the good results, another highlight of this work is the description of a complete system, including the acquisition hardware. An approach based on Visual Saliency Maps achieved good results in [17]. It was used a SVM in the classification step for a two-class problem. The highlight of this work is the adaptability of the approach and the low dimensionality of the feature vector, which is composed of two features only.

Comparing and combining two classical feature extraction approaches, GLCM and LBP, in [18] the authors presented robust results processing samples from TILDA database. The good accuracy was achieved using an ANN on the classification step. Their results show that in this case, a higher dimension of features was a better approach to achieve higher classification rates since the best result was obtained by the combination of GLCM and LBP. However, it increases the computation time considerably. Also comparing GLCM and LBP, alongside Histogram of Oriented Gradients (HOG), in [19] the authors proposed a method to automatically detect defects in circularly knitted fabrics. The first step is to enhance the defects in these fabrics. After this, it was tested different descriptors, where the Random Forest and SVM algorithms were compared. The best results were achieved using GLCM in one dataset and LBP in the two others, both classified using the Random Forest algorithm. The highlight of this work, besides the good classification rate, is the method to enhance the defects that happen in periodic patterns, since it can be hard to see even for humans.

3 METHODOLOGY

The methodology used in this work is similar to the one presented in [20], where the main differences are the *Segmentation* step exclusion and *Classifier Optimization* step addition. The adopted methodology is presented in Fig. 1.

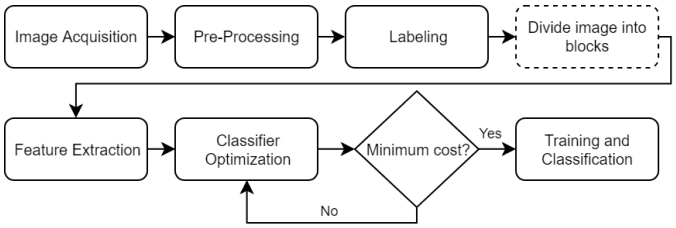


Figure 1: The methodology adopted in the work. The step named *Divide image into blocks* is presented with dotted lines because it is used only in the approach with block-processing.

The proposed algorithms were developed in C/C++ using the OpenCV 3.4 libraries. The execution was performed using a processing unit with an Intel Core i3-4030U CPU @ 1.90GHz × 2, with 8GB of RAM, a SSD Sandisk PLUS with 240 GB and an Intel Corporation Haswell - ULT Graphics Controller. The operating system used was Ubuntu 18.04 LTS 64-bit.

3.1 Image Acquisition, Pre-Processing and Labeling

The image acquisition is a fundamental step because if any problem happens, the whole process can be compromised. The database[1] used was created by the authors, using a camera with the resolution 3264 × 2448 pixels, 8-megapixel unit with a *f/2.0, 27mm-effective lens*. The sensor is a *Sony IMX179 1/3.2" CMOS* with *1.4 µm* pixels. Before the labeling and pre-processing steps, the acquired images were downsized to half of its original size (1624 × 1224 pixels). This process was necessary to reduce the final computational time.

Aiming to guarantee the quality of the acquired images, this process was executed in a controlled environment, with artificial illumination. The internal illumination system was positioned in a way to avoid creating shadows and/or reflecting light on the fabric.

[1]The database can be downloaded in: https://tinyurl.com/y74mksao

However, even in this controlled environment, some difficulties were faced in the analysis of the images, such as lack of focus, brightness change, and/or other imperfections. These problems could drastically compromise the training set composition.

Considering this fact, we decided to initially inspect the parts of the fabric with no imperfections, i.e., noiseless parts of the entire fabric image. The captured samples passed through a previous selection aiming to remove the images that had some noise. This initial classification was made by a professional from *Loop Jeans*. After, a label was defined for each selected image.

Since even in a controlled environment it was not possible to get rid of some imperfections in the image, only the center of the captured images were used to analyze the fabric. Therefore, the image was cropped to extract the features, which is the next step in the pipeline.

Table 1 shows the division between the patterns on the database. Classes *C*2, *C*3, and *C*4 will be treated as one class in this work, under the label "with defects". The database contains 282 samples of fabrics with defects and 197 without it. After the preprocessing steps, the images have a resolution of 919×571 pixels, since the borders were cropped due to shadows/noises introduced by the acquisition system.

Table 1: Division between the patterns in the database

Class	Type of Defect	Quantity
C1	Without Defects	197
C2	Hole	122
C3	Missing Yarn	86
C4	Oil Spot	74
Total		479

Fig. 2 shows examples of the classes in the database. The fabric used in this work is of the *plain* type. These database images also show that, even on samples with defects, a big part of the image contains no defects. Some images often contain excessive or distracting regions, where the test examples sometimes look different from the training ones [21].

3.2 Divide images into blocks

In [21], Boutell, Luo, and Gray stated that the performance of a classification system depends on the size and quality of the training samples. Also, the testing images can contain distracting or excessive foreground regions, sometimes making it different from the training set, causing a low detection rate. Also in [21], the authors presented an approach to solving this problem, dividing the images into blocks (or regions) and processing each one of them. For example, dividing an image into a 2×2 grid and applying the SFTA on each region will result in an usual number of features multiplied by 4 (the number of blocks). Fig. 3 shows an example of how the image can be divided.

3.3 Feature Extraction

3.3.1 Segmentation-Based Fractal Texture Analysis (SFTA). The SFTA algorithm [22] can be divided into two parts. In the first, occurs the decomposition of the input image (in grayscale) into a set of binary images using Two-Threshold Binary Decomposition (TTBD), also proposed in [22]. The TTBD takes an input image $I(x, y)$ and returns a set of binary images, using the Multi-Level Otsu Algorithm [23] applied recursively n_t times. This is a parameter defined by the user and influences the feature vector size and processing time. Then, for each resulting image, the fractal dimensions are computed from the boundaries of its region. This process returns $2n_t$ images.

In the second part of the algorithm, the pixel quantity and mean gray level are calculated from the binary images generated by TTBD. Also, the fractal dimensions of these images, along with the two features previously cited, will be placed into a vector. The fractal dimensions are obtained from a binary image $I_b(x, y)$ and it is represented as a border-image $\Delta(x, y)$. This border-image represents the region limits of each image and is computed as follows:

$$\Delta(x,y) = \begin{cases} 1, \text{if } \exists (x', y') \in N_8[(x,y)] : \\ \quad I_b(x', y') = 0 \wedge I_b(x,y) = 1, \\ 0, \text{otherwise.} \end{cases} \quad (1)$$

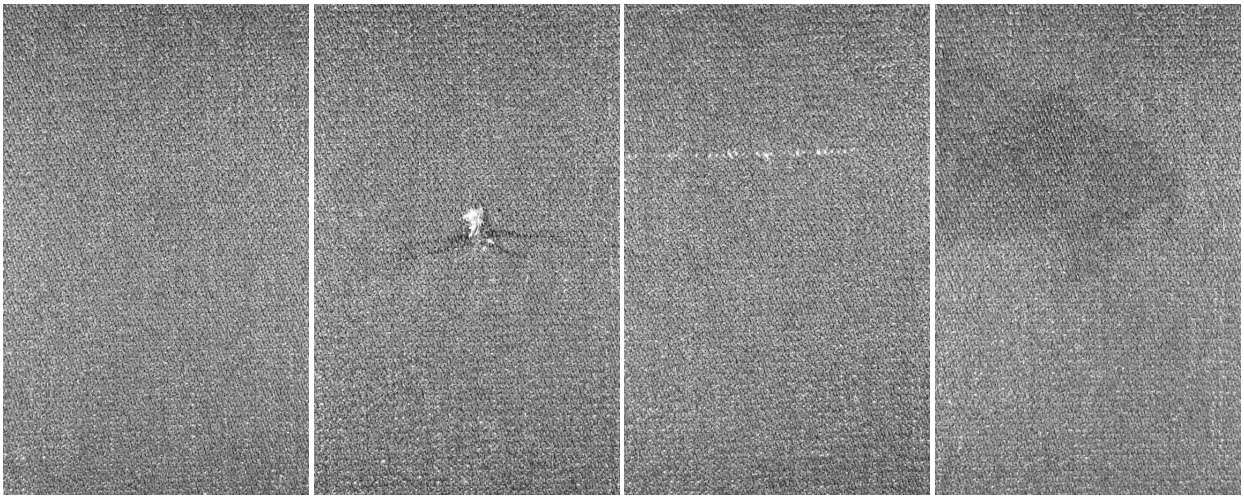
Where $N_8[(x, y)]$ represents the group of pixels that are 8-connected to (x, y). $\Delta(x, y)$ is 1 if $I_b(x, y) = 1$ and, at least, one of his neighbors is zero. Otherwise, $\Delta(x, y)$ is zero. The resulting borders are one-pixel wide.

Using the Box Counting Algorithm [24], the fractal dimensions can be computed in linear time. So, the asymptotic complexity of SFTA is $O(N \cdot |T|)$, where N represents the number of pixels of the grayscale image I, and $|T|$ is the number of different thresholds from multi-level Otsu Algorithm [22].

3.3.2 Gray Level Co-Occurrence Matrix (GLCM). The GLCM [25] is an approach proposed by Haralick, Shanmugam, and Dinstein. The elements of the co-occurrence matrix can reveal the relative frequency that pairs of determined gray level values occur, separated by a certain distance (or offset). These values are denoted by Δx, representing the number of columns between the pixel and its neighbors, and Δy, which corresponds to the number of rows. The offset is expressed in polar coordinates by the angle θ and distance ρ. If an intensity image doesn't have any texture, the resulting GLCM would be completely diagonal. The off-diagonal values become larger as the image texture increases [26].

The GLCMs are normalized and stored in a $N_g \times N_g \times N$ matrix. The N_g represents the number of gray levels used and N is the number of GLCMs calculated in different orientations and displacements. For example, a GLCM with a size of 8×8 means that it was created using 8 gray levels. The features used to describe the texture of the image are calculated using the GLCM previously constructed. In this work, besides the 14 characteristics proposed in [25], it was used another eight features. To see the description of each feature, please refer to [25], [27] and [28].

3.3.3 Local Binary Pattern (LBP). The LBP [29] is obtained by the difference between a central pixel and its circular neighborhood [30]. If the intensity of the pixel under analysis is greater than the central pixel, the corresponding cell in the binary code is 1. Otherwise, is zero. After this process, a matrix with $n \times m - 1$ bit number is created. The cells in the binary code matrix where the value is one are summed up to generate the LBP value for the central pixel [31]. The Equation 2 describes this process.



(a) Without Defects (C1) (b) Hole (C2) (c) Missing Yarn (C3) (d) Oil Spot (C4)

Figure 2: Patterns of fabrics and its defects present in the database

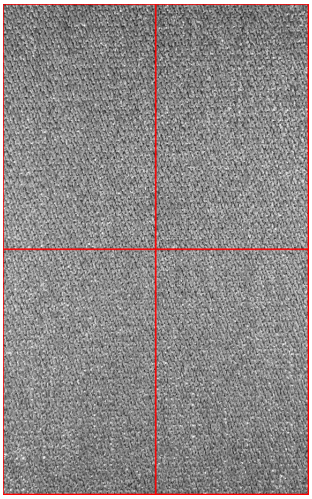


Figure 3: An example of division into blocks - 2×2

$$LBP_{n,r}(x_c, y_c) = \sum_{i=0}^{n-1} s(p_i - p_c)2^i \tag{2}$$

Where x_c, y_c is the position of the central pixel, (n, r) represents the pixel's neighborhood, n and r stand for sampling points on a circle of radius r, s denotes the sign function, p_c is the gray level of the center pixel, and p_i denotes the gray level of the neighbor pixel. Also, 2^i is the coefficient for each neighbor to involve textures in different proportions.

The *uniform pattern* can be considered as an extension to the original operator and it is denoted by $LBP^{u2}_{P,R}$. It is usually used to reduce the length of the feature vector, implementing a simple rotation-invariant property, since the original LBP is sensitive to noise. A local binary pattern can be considered uniform if the binary pattern contains, at least, two bitwise transitions from 0 to 1, or vice versa, when the bit pattern is traversed circularly. In the computation of the LBP labels, uniform patterns are used. Thus, there is a separate label for each uniform pattern and all the non-uniform patterns are labeled with a single label. After the LBP labeled image has been obtained, the LBP histogram can be calculated [32]. It happens because the code distribution over an image is used to describe the texture as a histogram of the image [31].

3.4 Optimization, Training and Classification

The SVM was used as classification algorithm. It is efficient, stable and can present a better performance in the majority of the problems [33]. Besides, SVM presents a good capacity of generalization in real situations, where it usually performs better than other classifiers, both in predictions and classifications [34].

SVM produces a model based on the training dataset, where it needs to be able to predict the class of a test sample using only its attributes [35]. Using a training set, organized as sample-class $(x_i, y_i),\ i = 1, ..., l$, where $x_i \in R^n$ and $y \in \{1, -1\}^l$, the SVM seeks for a solution for the optimization problem shown in Equation 3.

$$\begin{aligned} &\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \\ &\text{Subject to } y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i, \\ &\xi_i \geq 0. \end{aligned} \tag{3}$$

In this work, the Radial Basis Function (RBF) kernel was chosen, since it can deal with non-linear data [35]. There are two major RBF parameters, C and γ, and it must be set appropriately. The parameter C represents the cost of the penalty. The choice of this parameter influences the classification results. If it is too large, the classification accuracy rate can be very high in the training phase

and very low in the testing phase. If C is too small, the classification accuracy can be unsatisfactory. The γ parameter affects the partitioning outcome in feature space. If this value is too high, results may be overfitted. If it is too small, SVM can lead to an underfitting situation [36].

The optimization is an essential step because the wrong choice of these parameters can decrease the system accuracy [37]. To find the combination of parameters that brings the best accuracy, the Grid Search Algorithm [35] was used. This algorithm searches through a subset of the hyper-parameter space. The two most common types of this algorithm are the *Cartesian* and the *Random* grid-search. In this work, the Cartesian type was used. It tests each possible combination of hyper-parameter values, choosing the one with the best cross-validation accuracy [38].

The LIBSVM [38] was used for the implementation of the SVM. This library was chosen because it is easy to use and has a large amount of documentation. Also, integrate this library into the code is very simple.

It is important to scale the data before using the SVM. There are two advantages: avoid numerical difficulties during the calculation and avoid attributes in greater numeric ranges dominating those in the smaller ones [35]. This process consists in transforming the data between two limits, an inferior and a superior. In this work, the data were normalized by scaling each attribute to the range [-1 1].

4 RESULTS AND DISCUSSION

The dataset was randomly divided into training set (80%) and test set (20%). The SFTA algorithm was applied with $n_t = 3$. The blocks' (window) size was defined as 2×2. For the GLCM, the number of gray levels N_g was defined as 8, the angle $\theta = 0°$ and the distance $\rho = 1$. For the LBP, it was used a radius $r = 1$ and the number of neighbors $n = 8$. All these parameters were experimentally defined.

The analysis of the necessary computational time to process each image was divided into four steps: processing, optimization, train, and classification. In each process, the necessary time to run the algorithm was computed 25 times. From these values, the mean and standard deviation values were calculated. Table 2 presents the processing time obtained for each step of this work.

The optimization is the one that takes more time to be executed. However, since this step must be applied only when the fabric pattern changes, this performance is acceptable. The optimization time for the image processed in blocks usually is higher, since there are four times more features compared with the complete image. Also, the standard deviation shows that this step has a low variation in the execution time.

Accordingly to our experiments, SFTA is the algorithm that requires more processing time, both in the complete image and block processing approach. However, when the image was processed in blocks, the solution took less time if compared with the complete image. This was already expected because of the complexity of the algorithm, presented in Subsection 3.3. Also, the variation in time for this step is relatively low showing the stability of this algorithm.

GLCM achieved the second-best result related to processing time and the best result related to the optimization time. In the block processing approach, the processing and optimization time for GLCM increased significantly. However, its performance still higher than SFTA. The optimization time on this approach had almost folded up if compared with the complete image approach.

LBP achieved the best processing time among all algorithms for both cases: complete image and block processing. The computational time for the optimization step increased significantly, being three times bigger if compared with the other approach. It happens because this algorithm is the one that extracts the biggest number of features.

The training and classification times are small and almost don't influence the total time needed to execute the four steps. In every case, the training and classification time stays below 0.05 seconds. The only exception is for LBP in the block processing approach mainly by the number of features extracted.

The search space for Grid-Search Algorithm was defined from $C = 2^{-5}$ to $C = 2^{15}$ for the cost and from $\gamma = 2^{-15}$ to $\gamma = 2^{3}$ for the sigma [35]. The Table 3 shows the values obtained for each algorithm and approach.

Table 3: Optimal parameters obtained with Grid-Search Algorithm

	Complete Image		**Block Processing**	
	C	γ	C	γ
SFTA	2048	0.125	8	0.125
GLCM	128	2	8	0.5
LBP	8	0.5	2	0.125

In order to measure the system efficiency, accuracy, sensitivity, and specificity were calculated. The *Accuracy* represents the total number of samples that are correctly classified [39]. It is defined by Equation 4.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100 \quad (4)$$

Sensitivity is defined as the true positive rate. It represents the measure of samples that have defects and the system has classified them as "bad" [39]. Is defined by Equation 5.

$$Sensitivity = \frac{TP}{TP + FN} \times 100 \quad (5)$$

Specificity is defined as true negative rate. It shows the probability that the system classifies fabric with no defects correctly [39]. Equation 5 shows how this ratio is calculated.

$$Specificity = \frac{TN}{TN + FP} \times 100 \quad (6)$$

Fig. 4 shows the accuracy, sensitivity, and specificity achieved for the complete image approach.

In this case, SFTA achieved the worst classification rate (94,73%) among all the algorithms, classifying correctly 90 of 95 samples. LBP had the best accuracy, achieving 98,95% (94 of 95 samples were classified correctly). Also, GLCM achieved 97,89%, where 93 of 95 fabrics were correctly classified. The best sensitivity was achieved by GLCM (100%), followed by LBP and SFTA. However, the best specificity was achieved by LBP (also 100%), followed by SFTA and GLCM. In this case, it is better to have a good sensitivity since it is

Table 2: Processing time for each step

Complete Image						
	SFTA		**GLCM**		**LBP**	
	Mean (s)	Standard Deviation (s)	Mean (s)	Standard Deviation (s)	Mean (s)	Standard Deviation (s)
Processing	0.6289	0.0008	0.0707	0.0001	0.0537	0.0003
Optimization	7.1692	0.0205	6.3726	0.0404	11.7336	0.0528
Train	0.0385	0.0018	0.0166	0.0053	0.0393	0.0016
Classification	0.0045	0.0005	0.0040	0	0.0117	0.0008
Block Processing						
	SFTA		**GLCM**		**LBP**	
	Mean (s)	Standard Deviation (s)	Mean (s)	Standard Deviation (s)	Mean (s)	Standard Deviation (s)
Processing	0.6026	0.0002	0.2096	0.0003	0.0507	0.0002
Optimization	12.4651	0.0324	12.4181	0.0543	34.0989	0.0586
Train	0.0382	0.0017	0.0444	0.0026	0.1529	0.0035
Classification	0.0105	0.0007	0.0142	0.0014	0.0529	0.0014

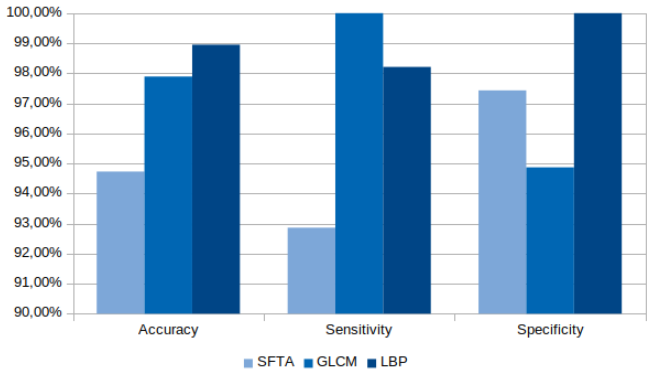


Figure 4: Accuracy, Sensitivity and Specificity - Complete Image

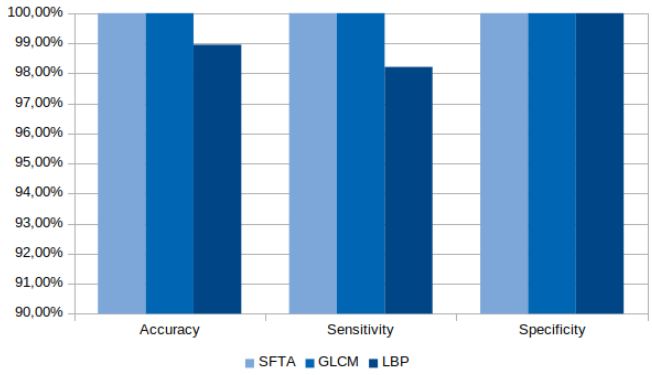


Figure 5: Accuracy, Sensitivity and Specificity - Block Processing

essential to find all defects because it decreases the quality of the product that reaches the customer.

Fig. 5 presents the results obtained with the block processing approach. It is important to show that, except by LBP, the other two solutions achieved a higher classification rate using the block processing approach.

Both SFTA and GLCM achieved 100% of correct classification showing that the block processing approach improves the results. Consequently, the sensitivity and specificity were also 100%. LBP had the same results obtained with the previous approach, classifying 94 of 95 samples correctly. This was expected since LBP proposes a local analysis where it observes the pixel value and compares it with its neighbors. The same is not true for GLCM, for example, where the results depend on the analysis made in the entire image to generate the co-occurrence matrix. However, in the LBP, when it is used the block processing approach, the influence of a pixel's gray level is higher (proportionally) than in the entire image.

Also, Cohen's Kappa (κ) [40] was calculated. It measures the agreement between two or more learners, taking into account the possibility of the agreement occurring by chance [41]. If the raters completely agree, $\kappa = 1$. If there is no agreement besides that what would be expected by chance (p_e), $\kappa \approx 0$ [42]. Table 4 shows the κ for each approach.

Table 4: Cohen's Kappa (κ) for each algorithm

	Complete Image	**Block Processing**
SFTA	0.8926	1
GLCM	0.9562	1
LBP	0.9783	0.9783

When the complete image was processed, the worst Kappa was achieved by the SFTA, followed by GLCM and LBP. Considering the LBP, the Kappa value in the two approaches were equal. However, in the block processing case it is considered the worst result. It happens because the other two algorithms achieved 100% of accuracy, and it reflects on the κ. For all these cases, it shows a substantial to almost perfect agreement, since all these values are above 0.8 [43].

As described, the detection rate achieved using SFTA and GLCM with block processing were higher than without it. This can be explained by the fact that sometimes the defects occupies a small

part of the image. So, when the whole image is processed once, there are many "distracting" parts, influencing in the feature extraction step. On the other side, with the block processing approach, these defects are evidenced.

Fig. 6 shows examples of samples that were misclassified by the SVM. Fig. 6a shows a false positive that was missed by both SFTA and GLCM using the complete image approach. In the same scenario, Fig. 6b shows another sample that was incorrectly classified. The defect in this image is really small, being difficult to see even by a professional. Fig. 6c shows a similar problem where the SFTA with the complete image approach could not describe correctly. Fig. 6d shows a small defect too, where LBP with the block processing approach was used. Fig. 6e shows another error, where SFTA with the complete image approach was used. Based on these samples, the defect which is more difficult to describe is the Missing Yarn. Also, all of the images containing the Oil Spot problem were correctly classified.

5 CONCLUSION

This work presented a comparative study to detect defects in fabrics using a block-processing technique. The SFTA, GLCM and LBP algorithms were compared for feature extraction, and SVM was used for classification. In order to validate the solution, the same algorithms were used to process the images without the application of the block-processing approach. The proposed approach demonstrated better results achieving 100% of accuracy using SFTA and GLCM for the tested dataset. The algorithm classifies the samples as *good* (without defects) or *bad* (with defects).

The proposed approach is the main contribution of this work since the division into blocks improved the results for SFTA and GLCM. Also, the SFTA was not previously used to detect defects in fabrics. The results show that the algorithm is appropriate for this type of application since, besides the high accuracy, the processing time fulfills the real-time needs. Another contribution is the database that is available online.

Even though SFTA is suitable for this type of application, GLCM proved to be a better option. The perfect accuracy (with the block processing approach) combined with a low processing time shows that this approach is the best option among the compared algorithms. LBP showed the lower processing time for the two approaches showing that it can be an option when this is more important than the accuracy. Also, the algorithm showed the best accuracy when the whole image is processed at once.

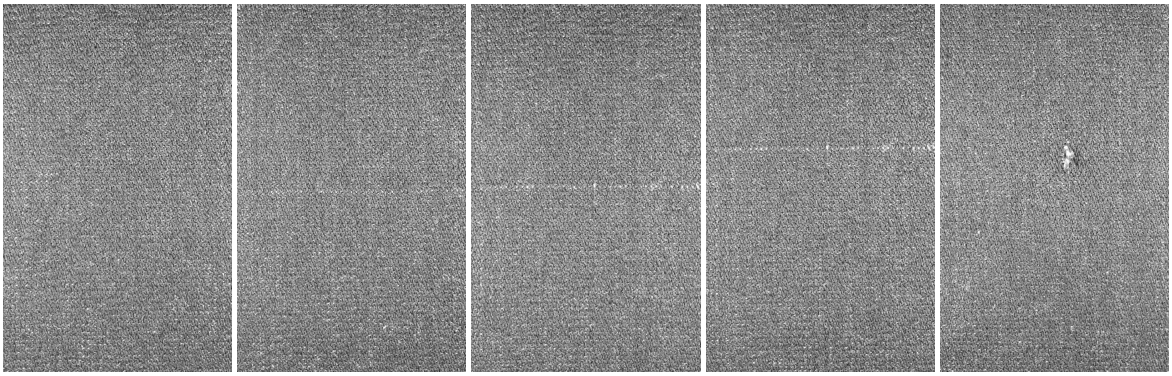
One limitation of this work is the acquisition system since it is quite simple. It impacts directly the quality of the acquired images. A more uniform illumination would most likely improve the detection of the defects. Also, the database is unbalanced, with 197 *good* samples and 282 of the *bad* ones. It was demonstrated that the excessive or distracting foreground regions in images can impact the results. So, using the block processing approach, the classification results can be improved, as well as processing time, depending on the algorithm's complexity.

Deep learning methods were not used in this work because there were only 479 samples available on this dataset. Usually, Convolutional Neural Networks (CNNs) and deep learning methods, in general, need a big amount of data to train a model [44]. We also opted for classical approaches due to limited hardware resources. The experiments were done on a personal computer, without any Graphics Processing Unit (GPU). According to the obtained results, the proposed solution presented the necessary requisites to solve this problem, both in accuracy and processing time.

Future works include improving the performance of the algorithms using a General Purpose Graphics Processing Unit (GPGPU) which supports Compute Unified Device Architecture (CUDA), for example. Some improvements in the acquisition system should be made as well, mostly in the illumination system. Furthermore, it is expected to use an algorithm that detects possible defects and classify them into the three classes present in the database. Also, a different number of blocks should be tested (e.g. 3×3 or 4×4 blocks).

REFERENCES

[1] Ajay Kumar. Computer-vision-based fabric defect detection: A survey. *IEEE transactions on industrial electronics*, 55(1):348–363, 2008.
[2] S. Ravikumar, K.I. Ramachandran, and V. Sugumaran. Machine learning approach for automated visual inspection of machine components. *Expert Systems with Applications*, 38(4):3260 – 3266, 2011. ISSN 0957-4174.
[3] Khaled Ragab and Nahed Alsharay. Developing parallel cracks and spots ceramic defect detection and classification algorithm using CUDA. In *Autonomous Decentralized System (ISADS), 2017 IEEE 13th International Symposium on*, pages 255–261. IEEE, 2017.
[4] Grasha Jacob, R Shenbagavalli, and S Karthika. Detection of surface defects on ceramic tiles based on morphological techniques. *arXiv.org - Cornell University Library*, 2016.
[5] Nelson Enrique Arias Gaviria and Jorge Andrés Ortiz Ruíz. *Análisis comparativo de descriptores para la clasificación de telas utilizando imágenes*. Universidad Tecnológica de Pereira. Facultad de Ingenierías Eléctrica . . . , 2016.
[6] Arley Bejarano Martínez, Andres Felipe Calvo Salcedo, and Carlos Alberto Henao. Descriptores espacio-frecuencia para identificación automática de patrones de textura en productos téxtiles utilizando aprendizaje supervisado. *Ciencia e Ingeniería Neogranadina*, 28(2), 2018.
[7] Lilyan Guimarães Berlin. A indústria têxtil brasileira e suas adequações na implementação do desenvolvimento sustentável. *ModaPalavra e-periódico*, 7(13): 15–45, 2014.
[8] Abdel Salam Malek. *Online fabric inspection by image processing technology*. PhD thesis, Université de Haute Alsace-Mulhouse, 2012.
[9] Krishnaswamy Srinivasan, PH Dastoor, P Radhakrishnaiah, and Sundaresan Jayaraman. FDAS: a knowledge-based framework for analysis of defects in woven textile structures. *Journal of the textile institute*, 83(3):431–448, 1992.
[10] Yadraj Meena and Ajay Mittal. Blobs and cracks detection on plain ceramic tile surface. *International Journal of Advanced Research in Computer Science and Software Engineering*, 3(7), 2013.
[11] Henry YT Ngan, Grantham KH Pang, and Nelson HC Yung. Automated fabric defect detection — a review. *Image and Vision Computing*, 29(7):442–458, 2011.
[12] Daniel Yapi, Mohand Saïd Allili, and Nadia Baaziz. Automatic fabric defect detection using learning-based local textural distributions in the contourlet domain. *IEEE Transactions on Automation Science and Engineering*, (99):1–13, 2017.
[13] Hermanus Vermaak, Philibert Nsengiyumva, and Nicolaas Luwes. Using the dual-tree complex wavelet transform for improved fabric defect detection. *Journal of Sensors*, 2016, 2016.
[14] Dandan Zhu, Ruru Pan, Weidong Gao, and Jie Zhang. Yarn-dyed fabric defect detection based on autocorrelation function and GLCM. *Autex research journal*, 15(3):226–232, 2015.
[15] Ali Rebhi, Issam Benmhammed, Sabeur Abid, and Farhat Fnaiech. Fabric defect detection using local homogeneity analysis and neural network. *Journal of photonics*, 2015, 2015.
[16] Hİ Çelik, LC Dülger, and M Topalbekiroğlu. Development of a machine vision system: real-time fabric defect detection and classification with neural networks. *The Journal of The Textile Institute*, 105(6):575–585, 2014.
[17] Hao Zhang, Jiajuan Hu, and Zhiyong He. Fabric defect detection based on visual saliency map and SVM. In *Computational Intelligence and Applications (ICCIA), 2017 2nd IEEE International Conference on*, pages 322–326. IEEE, 2017.
[18] Lei Zhang, Junfeng Jing, and Hongwei Zhang. Fabric defect classification based on LBP and GLCM. *Journal of Fiber Bioengineering and Informatics*, 8(1):81–89, 2015.
[19] Marcin Kopaczka, Hanry Ham, Kristina Simonis, Raphael Kolk, and Dorit Merhof. Automated enhancement and detection of stripe defects in large circular weft



(a) Without Defects **(b) Missing Yarn 1** **(c) Missing Yarn 2** **(d) Missing Yarn 3** **(e) Hole**

Figure 6: Examples of samples that were misclassified by the SVM

knitted fabrics. In *Emerging Technologies and Factory Automation (ETFA), 2016 IEEE 21st International Conference on*, pages 1–4. IEEE, 2016.

[20] Carsten Steger, Markus Ulrich, and Christian Wiedemann. *Machine vision algorithms and applications*. John Wiley & Sons, 2018.

[21] M Bautell, Jiebo Luo, and Robert T Gray. Sunset scene classification using simulated image recomposition. In *Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on*, volume 1, pages I–37. IEEE, 2003.

[22] Alceu Ferraz Costa, Gabriel Humpire-Mamani, and Agma Juci Machado Traina. An efficient algorithm for fractal analysis of textures. In *Graphics, Patterns and Images (SIBGRAPI), 2012 25th SIBGRAPI Conference on*, pages 39–46. IEEE, 2012.

[23] Ping-Sung Liao, Tse-Sheng Chen, Pau-Choo Chung, et al. A fast algorithm for multilevel thresholding. *J. Inf. Sci. Eng.*, 17(5):713–727, 2001.

[24] Caetano Traina, Agma Traina, Leejay Wu, and Christos Faloutsos. Fast feature selection using fractal dimension. 2000.

[25] Robert M. Haralick, Karthikeyan Shanmugam, and Its'Hak Dinstein. Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, (6):610–621, 1973.

[26] Manish H. Bharati, J.Jay Liu, and John F. MacGregor. Image texture analysis: methods and comparisons. *Chemometrics and Intelligent Laboratory Systems*, 72 (1):57–71, jun 2004. doi: 10.1016/j.chemolab.2004.02.005.

[27] L. K. Soh and C. Tsatsoulis. Texture analysis of sar sea ice imagery using gray level co-occurrence matrices. *IEEE Transactions on Geoscience and Remote Sensing*, 37(2):780–795, Mar 1999. ISSN 0196-2892. doi: 10.1109/36.752194.

[28] W. Gomez, W. C. A. Pereira, and A. F. C. Infantosi. Analysis of co-occurrence texture statistics as a function of gray-level quantization for classifying breast ultrasound. *IEEE Transactions on Medical Imaging*, 31(10):1889–1899, Oct 2012. ISSN 0278-0062. doi: 10.1109/TMI.2012.2206398.

[29] Timo Ojala, Matti Pietikäinen, and David Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 29(1):51–59, 1996.

[30] Qiang Wang, Bailin Li, Xiaoyan Chen, Jianqiao Luo, and Yun Hou. Random sampling local binary pattern encoding based on gaussian distribution. *IEEE signal processing letters*, 24(9):1358–1362, 2017.

[31] Afsoon Asghari Shirazi, Alireza Dehghani, Hasan Farsi, and Mehran Yazdi. Persian logo recognition using local binary patterns. In *Pattern Recognition and Image Analysis (IPRIA), 2017 3rd International Conference on*, pages 258–261. IEEE, 2017.

[32] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 24(7):971–987, 2002.

[33] Luan Casagrande, Luiz Antonio Macarini, Daniel Bitencourt, Roger Florzino, Felipe Vieira, Fabrício Ourique, and Gustavo Medeiros de Araujo. Sistema de controle de qualidade visual de pisos cerâmicos fundamentado em processamento de imagem e aprendizado de máquina. In *Simpósio Brasileiro de Automação Inteligente*, 2017.

[34] Kabiru O Akande, Taoreed O Owolabi, Ssennoga Twaha, and Sunday O Olatunji. Performance comparison of svm and ann in predicting compressive strength of concrete. *IOSR Journal of Computer Engineering*, 16(5):88–94, 2014.

[35] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification. 2003.

[36] Shih-Wei Lin, Kuo-Ching Ying, Shih-Chieh Chen, and Zne-Jung Lee. Particle swarm optimization for parameter determination and feature selection of support vector machines. *Expert systems with applications*, 35(4):1817–1824, 2008.

[37] Cheng-Lung Huang and Chieh-Jen Wang. A ga-based feature selection and parameters optimizationfor support vector machines. *Expert Systems with Applications*, 31(2):231 – 240, 2006. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2005.09.024. URL http://www.sciencedirect.com/science/article/pii/S0957417405002083.

[38] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, May 2011. ISSN 2157-6904. doi: 10.1145/1961189.1961199. URL http://doi.acm.org/10.1145/1961189.1961199.

[39] Alireza Baratloo, Mostafa Hosseini, Ahmed Negida, and Gehad El Ashal. Part 1: simple definition and calculation of accuracy, sensitivity and specificity. 2015.

[40] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

[41] Robert Gilmore Pontius Jr and Marco Millones. Death to kappa: birth of quantity disagreement and allocation disagreement for accuracy assessment. *International Journal of Remote Sensing*, 32(15):4407–4429, 2011.

[42] Louis Cyr and Kennon Francis. Measures of clinical agreement for nominal and categorical data: the kappa coefficient. *Computers in biology and medicine*, 22(4): 239–246, 1992.

[43] Anthony J. Viera and Joanne M. Garrett. Understanding interobserver agreement: The kappa statistic. *Journal of Family Medicine*, pages 360–363, may 2005.

[44] Andreas Kamilaris and Francesc X Prenafeta-Boldú. Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, 147:70–90, 2018.