Quality Control System for Ceramic Tiles using Segmentation-based Fractal Texture Analysis and SVM

Luiz Antonio Macarini
Universidade Federal de Santa Catarina
Araranguá, Santa Catarina, Brazil
Email: luiz.buschetto@grad.ufsc.br

Tiago Oliveira Weber
Universidade Federal de Santa Catarina
Araranguá, Santa Catarina, Brazil
Email: tiago.weber@ufsc.br

***Abstract*—The ceramic industry has a highly automated production system. The quality control, however, is still performed by humans, which limits its speed and precision. This work proposes a complete verification system for ceramic tiles based on image processing and machine learning. The system has four steps: image acquisition, pre-processing, feature extraction and classification. The feature extraction step uses Segmentation-based Fractal Texture Analysis (SFTA). A Support Vector Machine is employed to classify the ceramic tiles. The system is implemented using OpenCV libraries. In total, 783 ceramic tiles were used, being 80% for training and 20% to testing. The present work had reached the proposed objectives, both in processing time and accuracy, achieving 98.68% of detection rate.**

I. Introduction

At the end of the twentieth century, industries started to automatize their processes with the help of computation and automation [1]. These two areas are commonly employed in visual inspection of products. The allocation of humans to perform the visual inspection is therefore unnecessary if a proper computer vision system is implemented.

The production line of ceramic tiles is almost fully automatized. However, the quality control step requires human intervention yet. This procedure is hard, intensive and takes place in a heavy industrial environment, with high noise, temperature and humidity levels. The quality control process can be divided in: color analysis, dimensions verification and visual defects inspection. The defects on the ceramic tiles can occur due to chemical impurities or physical problems during the process [2].

The ceramic tiles inspection needs to be done by professionals. However, opinions about the presence or absence of defects can diverge. The human capacity depends on training, knowledge and experience [3].

The fact that quality control is done by humans brings several problems. People, in general, can work in this activity during some limited time and get tired in a couple of hours. Thus, the judgment is affected by fatigue [3]. Another negative factor is the output rate of the quality control being lower than the production rate.

In 2013, Asian continent was the leader in ceramic manufacture, producing 8315 million square meters (69.8% of the world's production). In the world manufacturing rank, China took the first place, producing 5700 million square meters, which represented 47.8% of the world's production. Brazil was the second, with 7.3% of the world's production (871 million square meters) [4].

The rank is repeated in consumption. Asia was the biggest consumer, using 7692 million square meters (66.5% of world's production). China and Brazil were responsible for consuming 39.4% (4556 million square meters) and 7.2% (837 million square meters) of world's production, respectively [4].

Defect detection on ceramic tiles has been a widely researched subject. For this reason, several approaches were used to try to solve this problem. Mohan and Kumar [5] proposed an automated system for detecting breaks in ceramic tiles using Discrete Wavelet Transform and Co-Occurrence Matrix. In Ghazvini et al. [6], the authors presented a solution to ceramic tile defect detection using 2D Wavelet Transform as well. However, in the classification step, the authors used a Multilayer Perceptron Neural Network. This approach's main goals were to keep high accuracy and decrease the computational time.

Furthermore, Jacob et al. [7] proposed an automated system of defect detection in tiles surface using techniques based on morphological operations such as dilatation and erosion, Simple Morphological Edge Extraction and edge detection techniques. In Mishra and Shukla [8], the authors presented a solution based on the use of a Probabilistic Neural Network (PNN). In the feature extraction step, techniques such as Mean Filter, Sobel Edge Detector, Roberts Detector and Histogram Calculation were employed.

Hocenski, Matić and Vidović [9] shows a computer vision station for real-time biscuit tile defect detection. It was developed in C++ using the OpenCV and CUDA libraries. Furthermore, Novak and Hocenski [10] presented a solution using Local Binary Pattern (LBP) for feature extraction. For the classification step, the k-nearest neighbor algorithm was used. The results show that this approach is efficient to this type of application. In Hanzaei et al. [11], a system for defect detection was proposed. It uses the Rotation Invariant Measure of Local Variance and a closing morphology operator was used

to fill and smooth the detected regions.

Some works had used the Segmentation-Based Fractal Texture Analysis (SFTA) for feature extraction. However, none of them had used this method applied to ceramic tiles. Arivazhagan et al. [12] presents an algorithm to detect cracks in railway tracks. El-Henawy et al. [13] proposed a model to Cattle Feature Extraction. Samiappan et al. [14] shows an algorithm to map the invasive *Phragmites australis* of a large wetland along the US Pearl River delta in southeastern of Louisiana. In this work, other texture analysis methods were used and the SFTA obtained the best performance among them.

This work proposes a complete control quality system for ceramic tiles based on image processing and machine learning. Segmentation-based Fractal Texture Analysis (SFTA) [15] is used for feature extraction. This algorithm consists in decomposing the input image into a set of binary images. Fractal dimensions of the resulting regions are extracted with the goal of describing the texture patterns on the image. For classification, the Support Vector Machine (SVM) technique is used [16], [17].

The main difference between this work and others is the use of SFTA to extract the texture information of the acquired images. Other works had used this method. However, none of them involves defect detection on ceramic tiles. The main objective is to decrease the processing time [14], making this approach applicable in a production line.

This work is organized as follows. Section II gives an overview of the system. Section III presents the prototype used for image acquisition. Furthermore, Section IV presents the algorithm used to pre-process the information and Section V shows the algorithm employed in feature extraction. Besides that, Section VI presents the classification method used in this work. The results are presented in Section VII. Finally, the conclusion and a discussion about future works is provided in Section VIII.

II. System Overview

An image processing system is made by a series of steps which are capable of producing a result from the domain of the problem [18]. The block diagram of the proposed method is shown in Figure 1.

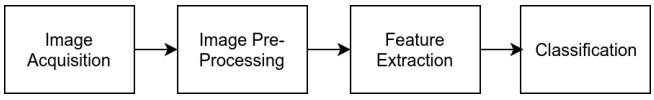


Fig. 1. Block diagram of the proposed method.

The first step consists in acquiring the image. It needs to be captured in a controlled environment with minimum possible outside influence. To this end, a low cost prototype was developed.

In the image pre-processing step, the information that is not useful for the application is removed. For instance, the captured image may contain parts of the floor and the illumination system. To decrease the amount of information to be processed, the background is removed, leaving only the tile.

On the feature extraction step, the SFTA technique is used. The necessary data is extracted and organized to be later classified by the next step, which consists in classification using Support Vector Machine.

III. Prototype for Image Acquisition

In order to validate the method, a low cost prototype for image acquisition was built. The goal is to create a stable environment with minimum external influence.

Fig. 2. Representation of the external view of the prototype.

Figure 2 shows a representation of the external view of the box-shaped prototype. The prototype internal part can be seen in Figure 3. On the bottom, it has a space to place the tile. After the image is captured, the tile can be removed.

The base itself is removable to allow adaptation to a production line. The system was created considering some necessary features that would be important in a real ceramic tile production, such as stable environment, a good illumination system and a dark background, aiming to create a bigger contrast between the prototypes floor and the tile in analysis.

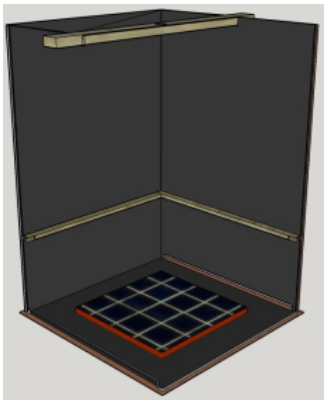


Fig. 3. Representation of the internal view of the prototype.

The prototype has a camera holder on the top. In a lower position, there is the fixation structure for the illumination

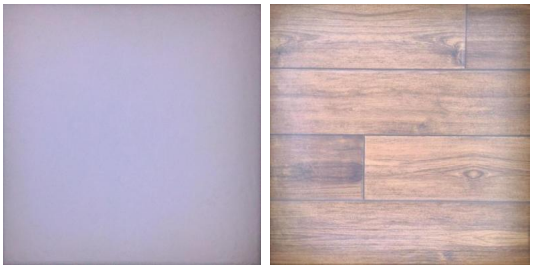
system. For this purpose, a LED string with 120 LEDs and 2 meters of length was used. It was placed in a lower part of the structure to avoid shadows and/or reflections.

The material used was Medium Density Fiber (MDF). The prototype has dimensions of 50 cm $\times$ 50 cm $\times$ 70 cm. Its interior was painted in matte black, aiming to create a higher contrast with the light tiles, the most common pattern in market. The tile under analysis has maximum dimensions of 48 cm $\times$ 48 cm.

The used camera has a T4K37 13-megapixel CMOS of 1/3.07" sensor. It was positioned at 69 cm of the tile, resulting in a Ground Sampling Distance (GSD) of 0.0206 cm. This sensor is not the best for this type of application. However, this was a good option, since one of the proposals of this work was to keep the system low-cost. However, with a better quality camera, the system is expected to achieve higher accuracy.

A. Database

To validate the proposed solution, a ceramic tile database was created using this prototype. Figure 4 shows the tile patterns used in this work. There are two types of tiles: with and without texture. The use of these two types of tiles is important to show how the algorithm behaves in different cases.



(a) Set I (b) Set II

Fig. 4. Tile patterns used in this work.

The database is composed by 783 images[1]. Table I shows how the database is divided. The tiles are classified as *good* (without defects) or *bad* (with defects). The classification of each tile contained in the database was based on the technical standard NBR 13818 [19].

TABLE I
DIVISION OF THE TILES PATTERNS.

Pattern	Good	Bad	Total
1	287	116	403
2	300	80	380
Total	587	196	**783**

IV. PRE-PROCESSING

The pre-processing step consists in removing all the information that is not useful for the application. A sample of an acquired image using the prototype is shown in Figure 5.

[1]The database can be found in: https://goo.gl/aTnBQE

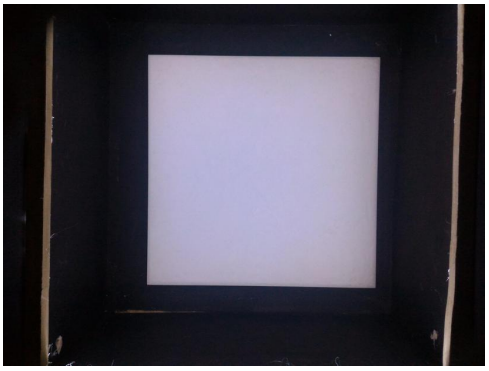


Fig. 5. Sample of acquired image from prototype.

In order to decrease the processing time, the first thing to do is re-size the image to 1/5 of its original size. As a result, the processing time can be decreased as a high resolution image is not needed for this application. After, the matrix image is reorganized in a vector for application of *k-means clustering* [20].

The k-means clustering algorithm is an iterative process and can be divided basically into three steps. The first one consists in choosing two centroids (or more, depending on the application) randomly or using a center initialization algorithm. Then, the distance between each point and both centroids are calculated. The data is labelled according to the nearest centroid. The last step is calculate the average of the data separately. Based on that, the new positions of the centroids are found. The second and third steps are iterated until the stop criteria is satisfied. It can be maximum number of iterations or when a specific accuracy is reached, for example.

The prototype floor is black to create a higher contrast with the ceramic tile. Therefore, the use of only two clusters is sufficient for this application. Equation 1 [21] defines the function used to cluster division.

$$\sum_{i} ||samples_i - centers_{labels_i}||^2 \tag{1}$$

kmeans++ center initialization [22] was employed for center initialization. Figure 6 shows the result of this step.

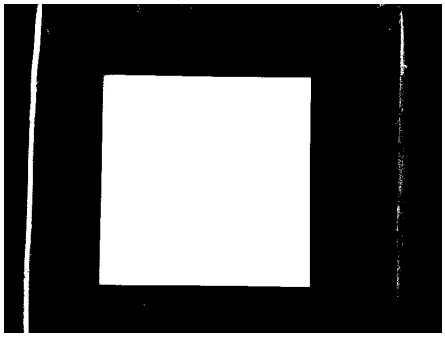


Fig. 6. Result after k-means clustering application.

After that, it is necessary to find the tile on the image. To this purpose, the algorithm searches for the area with the highest

number of pixels. Once this was found, the minimum square that fits the area is calculated. Figure 7 show the results.

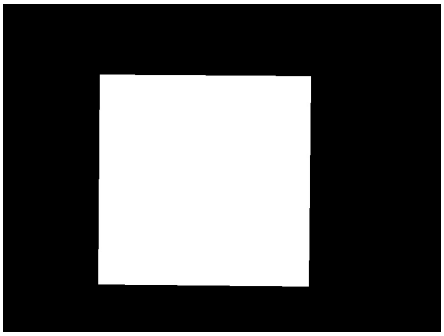


Fig. 7. Minimum square that fits the area.

Usually, the tile has a small rotation. To fix it, the angle between the two lower vertices is calculated. To remove the angle, an *Affine Transformation* [23] is applied in the image. Once the mask is done, it can be applied to the original image. The output of this process is shown in Figure 8.

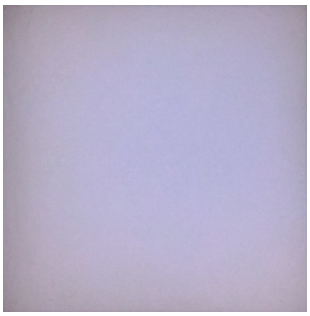


Fig. 8. Output from pre-processing step.

V. Feature extraction using SFTA

Texture gives us information about spatial color distribution or intensities in a complete image or a region. This information can be used for classification. However, the extraction of texture information consumes a lot of time [12]. It happens mainly in cases where the utilization of high resolution images is indispensable for high accuracy. Methods like Gray-Level Co-Occurrence Matrix can be used, but it needs a higher computational power [14]. With the goal of decreasing this processing time, this works proposes the utilization of SFTA for ceramic tiles classification.

This extraction algorithm can be divided into two parts. In the first one, the input grayscale image is decomposed into a set of binary images using Two-Threshold Binary Decomposition (TTBD) [15]. For each resulting image, the fractal dimensions are computed from its regions boundaries. Furthermore, the mean gray level is calculated and a pixel count is made.

The TTBD takes an input image $I(x, y)$ and returns a set of binary images. The first step is to calculate an interval T with several thresholds values. The values are obtained by selecting equally spaced gray level values. This can be achieved by the use of *Multi-Level Otsu Algorithm* [24] applied recursively n_t times. The number of times is a parameter defined by the user and has influence on the feature vector size. A binary image $I_b(x, y)$ is obtained applying the Equation 2.

$$I_b(x,y) = \begin{cases} 1, & \text{if } t_l < I(x,y) \leq t_u \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Where t_l represents the inferior limit of gray level and t_u the superior limit.

The set of binary images is obtained applying the Equation 2 on the input image, using all pairs of contiguous thresholds from $T \cup \{n_l\}$ and all pairs of thresholds $\{t, n_l\}, t \in T$, where n_l corresponds to the maximum possible gray level in $I(x, y)$. This process return $2n_t$ images.

In the second part of the algorithm, a vector containing the pixel quantity, mean gray level and the fractal dimensions of the binary images is created. The first two features are calculated using the binary images generated by TTBD. The fractal dimensions are obtained by a binary image $I_b(x, y)$ and it is represented as a border image $\Delta(x, y)$. This, represents the region limits of each image and is computed as follows:

$$\Delta(x,y) = \begin{cases} 1, \text{if } \exists (x',y') \in N_8[(x,y)] : \\ \quad I_b(x',y') = 0 \wedge I_b(x,y) = 1, \\ 0, \text{otherwise.} \end{cases} \tag{3}$$

$N_8[(x, y)]$ represents the set 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, its value is zero. The resulting borders are one-pixel wide.

The mean gray level and pixel count brings the advantage of complement the extracted information without increasing too much the computation time. The size of the features vector is three times the quantity of binary images produced by TTBD. This is due to the fractal dimension, mean gray level and size are extracted from each binary image.

The fractal dimensions can be efficiently computed in linear time by the use of *Box Counting Algorithm* [25]. Therefore, 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.

VI. Classification

The results of this work were obtained using a Support Vector Machine (SVM). It was chosen because it can present a better performance in most number of problems. Furthermore, the SVM is very efficient and stable [26]. This technique presents a good capacity of generalization in real situations, where it usually overcome others classifiers, both in predictions and classifications [27].

The objective of SVM is to produce a model based on the training data which is capable of predicting the class of the sample using only its attributes [28].

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 searches a solution for the optimization problem described as follows:

$$\begin{aligned} \min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \\ \text{Subject to } y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i, \\ \xi_i \geq 0. \end{aligned} \tag{4}$$

The training vector x_i is mapped to a higher dimensional space, maybe infinite, by the function ϕ. The SVM tries to find a hyperplane capable of separating the data linearly, with the maximum margin, in this new dimensional space. $C > 0$ is a parameter which describes the penalty according to the error.

Library LIBSVM [29] was used for the implementation of the Support Vector Machine. This library was chosen due to its ease of use and the large amount of documentation. Furthermore, it is easy to integrate it in the system's code.

To scale the data before applying the SVM is very important. The two main advantages are to avoid attributes in greater numeric ranges dominating those in smaller numeric ranges and to avoid numerical difficulties during the calculation [28]. This process consists in transforming the data between two limits: an inferior and a superior one. Therefore, the data is normalized scaling each attribute to the range [-1 1].

There are four basic types of kernel. To this application, the Radial Basis Function (RBF) was chosen as it can deal with non-linear data [28]. Equation 5 shows the kernel equation.

$$K(x_i, x_j) = exp(-\gamma ||x_i - x_j||^2), \gamma > 0 \tag{5}$$

There are two major RBF parameters: C and γ. They must be set appropriately. The parameter C represents the cost of penalty. The choice of this parameter has influence on classification results. If it is too large, the classification accuracy rate can be very high in training phase and very low in 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 in overfitting. If is too small, can leads to an underfitting [30].

Beforehand, this parameters are unknown. To find pair (C, γ) that brings the best performance, a parameter search known as *Grid Search Algorithm* [28] was used. It consists in trying various pairs (C, γ) and choosing the one with the best cross-validation accuracy [29]. It is an essential step because the wrong choice of these parameters can decrease the system accuracy.

VII. Results

In order to get the results, 80% of the images were used to train the classifier. The other 20% were unknown to the classifier and were used to validate the system. The division between training and testing images was made randomly.

In the first pattern, 322 images were used to train the classifier: 229 of them were from good tiles and 93 from bad tiles. To test it, 81 images were used, where 58 were from good tiles and 23 were bad tiles.

In the second pattern, 304 images were used to training: 240 of them were from good tiles and 64 images represents the bad tiles. To testing, 76 images were used, where 60 were from good tiles and 16 were from the bad ones.

Before training the classifier, the optimal values of C and γ need to be found using the Grid Search Algorithm. To the first tile pattern (Set I), C is 32768 and γ is 0.00048828125. To the second (Set II), the value found for C was 512 and to γ was 0.0125.

During the tests, processing time was measured for every tile. Table II shows the mean and the standard deviation related to the pre-processing step.

TABLE II
Mean and standard deviation - Pre-processing step.

Set	Mean (s)	Standard Deviation (s)
I	0.202	0.023
II	0.145	0.021

Table III shows the mean and standard deviation values obtained in processing step.

TABLE III
Mean and standard deviation - Processing step.

Set	Mean (s)	Standard Deviation (s)
I	0.480	0.025
II	0.453	0.012

In total, 157 tiles were used for prediction: 81 of the first pattern and 76 of the second. Table IV shows the detection results. The abbreviations in table are described below:

- TN: True Negative rate represents the number of tiles that don't have any defects and the system had classified it correctly;
- TP: True Positive rate represents the number of tiles that have defects and the system had classified it correctly;
- FN: False Negative rate represents the number of tiles that have some kind of defect and the system had classified it as "good";
- FP: False Positive rate represents the number of tiles that don't have defects but the system had classified it as "bad".

TABLE IV
Defect detection results.

Set	TN	TP	FN	FP
I	58	21	2	0
II	60	15	1	0
Total	118	36	3	0

Using the results shown in Table IV, some parameters can be calculated to measure the system efficiency. The *Accuracy*

represents the total number of tiles that are correctly classified and is defined by Equation 6.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100 \quad (6)$$

Sensitivity is defined as the true positive rate. It represents the measure of tiles that have defects and the system have classified them correctly. Is defined by Equation 7.

$$Sensitivity = \frac{TP}{TP + FN} \times 100 \quad (7)$$

Specificity is defined as true negative rate. Express the probability that the system classifies a tile with no defects as "good". Equation 7 shows how this ratio is calculated.

$$Specificity = \frac{TN}{TN + FP} \times 100 \quad (8)$$

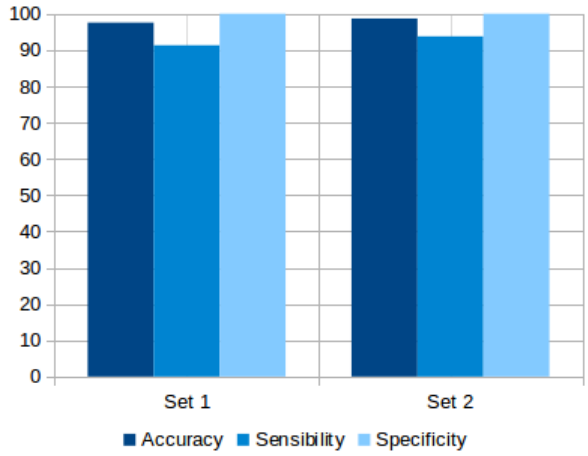


Fig. 9. Defect detection measures.

Figure 9 shows the defect detection results. To the first pattern, the system has achieved 97.53% of detection rate, classifying correctly 79 of 81 tiles. In the second pattern, the system classified 75 of 76 tiles correctly, achieving 98.68% of detection rate. In relation to sensibility, the system has achieved 91,30% for the first pattern and 93,75% for the second. Besides that, the specificity was 100% for both patterns.

About the processing time, good results were achieved. To process an image acquired from prototype, the system took less than 700 ms. The second tile pattern had textures. However, the processing time was lower than first pattern, which shows the algorithm efficiency to process textured images.

The achieved accuracy was satisfactory. Nevertheless, system sensibility was not so good. It happened because the database was quite unbalanced and affected directly the results.

A lot of works were done trying to solve this problem. However, the databases used on the other works are not public. Thus, was not possible to obtain the statistical parameters using this algorithm. Nevertheless, since the analyzed tiles and its acquiring method are different, the proposed system had obtained a satisfying accuracy.

The whole system was implemented in C++ programming language with OpenCV libraries. The processing unit has 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 operational system was Linux Mint 18.1, Cinnamon Edition, 64-bit with 4.4.0-53-generic Linux Kernel.

VIII. Conclusion

This work presented a complete system for automatic defect detection of ceramic tiles using image processing and machine learning. For the classification step, Support Vector Machine was used. Segmentation-based Fractal Texture Analysis (SFTA) was employed for feature extraction. The use of SFTA for ceramic tile classification is one of the main contributions of this work. The algorithm has already been used to other applications before. However, had never been employed on ceramic tiles defect detection.

The system was implemented in C++ using OpenCV libraries. A low cost prototype was built to acquire the images. The system classifies the ceramic tiles in two classes: *good* (no defects) or *bad* (with any kind of defect).

The achieved results were good both in processing time and accuracy. They can be compared to the results obtained on the works presented in Section I.

In many cases, mainly in works related to the industry, the authors don't make clear what methods are used for feature extraction. This makes it really difficult to compare this methodology with others proposed in literature.

However, although the achieved results are good, a better quality camera could improve the defect detection rate. Once more, it is important to remember that the system has a low cost purpose.

The present work can be applied successfully in ceramic tiles defects detection. Furthermore, the algorithm can be applied at ceramic industry, since the production line must work in real time and this work meet this requirement. However, the developed prototype needs some improvements.

The greatest drawback of this work is the simplicity of the image acquisition system prototype. However, as one of the goals of this work is keep the system low-cost, the prototype attended its objectives. Nevertheless, the methodology presented here has potential to be implemented in a production line, as the system can classify plain and texture tiles.

Future work will include improvement of the acquisition system, mostly on the illumination part and use a more appropriated camera. Besides that, this algorithm can be compared with other extraction algorithms. Code improvements can be performed using a more powerful hardware such as GPUs with CUDA support as OpenCV supports this type of optimization. Furthermore, the presented system strategy is expected to be implemented in real ceramic tile production.

REFERENCES

[1] G. Chryssolouris, D. Mavrikios, N. Papakostas, D. Mourtzis, G. Michalos, and K. Georgoulias, "Digital manufacturing: history, perspectives, and outlook," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 223, no. 5, pp. 451–462, 2009.

[2] H. Elbehiery, A. Hefnawy, and M. Elewa, "Surface defects detection for ceramic tiles using image processing and morphological techniques." *World Academy of Science, Engineering and Technology International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol:1, No:5, 2007*, p. 52, 2005.

[3] Y. Meena and D. A. Mittal, "Blobs & crack detection on plain ceramic tile surface," *International Journal of Advanced Research in Computer Science & Software Engineering*, vol. 3, no. 7, 2013.

[4] D. Stock, "World production and consumption of ceramic tiles," *Tile today*, vol. 73, pp. 50–58, 2011.

[5] V. Mohan and S. S. Kumar, "An automated tiles defect detection," *International Journal of Computer Applications*, vol. 109, no. 11, 2015.

[6] M. Ghazvini, S. Monadjemi, N. Movahhedinia, and K. Jamshidi, "Defect detection of tiles using 2d-wavelet transform and statistical features," *World Academy of Science, Engineering and Technology*, vol. 49, pp. 901–904, 2009.

[7] G. Jacob, R. Shenbagavalli, and S. Karthika, "Detection of surface defects on ceramic tiles based on morphological techniques," *arXiv.org - Cornell University Library*, 2016.

[8] R. Mishra and D. Shukla, "An automated ceramic tiles defect detection and classification system based on artificial neural network," *Int. J. Emerg. Technol. Adv. Eng*, vol. 4, no. 3, 2014.

[9] Ž. Hocenski, T. Matić, and I. Vidović, "Technology transfer of computer vision defect detection to ceramic tiles industry," in *Smart Systems and Technologies (SST), International Conference on*. IEEE, 2016, pp. 301–305.

[10] I. Novak and Z. Hocenski, "Texture feature extraction for a visual inspection of ceramic tiles," in *Industrial Electronics, 2005. ISIE 2005. Proceedings of the IEEE International Symposium on*, vol. 3. IEEE, 2005, pp. 1279–1283.

[11] S. H. Hanzaei, A. Afshar, and F. Barazandeh, "Automatic detection and classification of the ceramic tiles' surface defects," *Pattern Recognition*, vol. 66, pp. 174–189, 2017.

[12] S. Arivazhagan, R. N. Shebiah, J. S. Magdalene, and G. Sushmitha, "Railway track derailment inspection system using segmentation based fractal texture analysis." *ICTACT Journal on Image & Video Processing*, vol. 6, no. 1, 2015.

[13] I. El-Henawy, H. M. El Bakry, and H. M. El Hadad, "Cattle identification using segmentation-based fractal texture analysis and artificial neural networks," *Int. J. Electron. Inf. Eng*, vol. 4, no. 2, pp. 82–93, 2016.

[14] S. Samiappan, G. Turnage, L. Hathcock, L. Casagrande, P. Stinson, and R. Moorhead, "Using unmanned aerial vehicles for high-resolution remote sensing to map invasive phragmites australis in coastal wetlands," *International Journal of Remote Sensing*, vol. 38, no. 8-10, pp. 2199–2217, 2017.

[15] A. F. Costa, G. Humpire-Mamani, and A. J. M. Traina, "An efficient algorithm for fractal analysis of textures," in *Graphics, Patterns and Images (SIBGRAPI), 2012 25th SIBGRAPI Conference on*. IEEE, 2012, pp. 39–46.

[16] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, 1992, pp. 144–152.

[17] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[18] H. Pedrini and W. R. Schwartz, *Análise de imagens digitais: princípios, algoritmos e aplicações*. Thomson Learning, 2008.

[19] A. B. de Normas Técnicas, *NBR 13818 - Placas cerâmicas para revestimento - Especificação e métodos de ensaios*, April 1997.

[20] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[21] OpenCV 3.0.0 - dev documentation. (2014) Clustering. [Online]. Available: http://docs.opencv.org/3.0-beta/modules/core/doc/clustering.html

[22] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[23] G. Wolberg, *Digital image warping*. IEEE computer society press Los Alamitos, CA, 1990, vol. 10662.

[24] P.-S. Liao, T.-S. Chen, P.-C. Chung *et al.*, "A fast algorithm for multilevel thresholding," *J. Inf. Sci. Eng.*, vol. 17, no. 5, pp. 713–727, 2001.

[25] C. Traina, A. Traina, L. Wu, and C. Faloutsos, "Fast feature selection using fractal dimension," 2000.

[26] L. Casagrande, L. A. Macarini, D. Bitencourt, R. Florzino, F. Vieira, F. Ourique, and G. M. 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*, forthcoming.

[27] K. O. Akande, T. O. Owolabi, S. Twaha, and S. O. Olatunji, "Performance comparison of svm and ann in predicting compressive strength of concrete," *IOSR Journal of Computer Engineering*, vol. 16, no. 5, pp. 88–94, 2014.

[28] C.-W. Hsu, C.-C. Chang, C.-J. Lin *et al.*, "A practical guide to support vector classification," 2003.

[29] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 27:1–27:27, May 2011. [Online]. Available: http://doi.acm.org/10.1145/1961189.1961199

[30] S.-W. Lin, K.-C. Ying, S.-C. Chen, and Z.-J. Lee, "Particle swarm optimization for parameter determination and feature selection of support vector machines," *Expert systems with applications*, vol. 35, no. 4, pp. 1817–1824, 2008.