ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
UDC/UDK 658.562.4:666.646]:004.932

REAL TIME CONTOUR BASED CERAMIC TILE EDGE AND CORNER DEFECTS DETECTION

Tomislav Matić, Ivan Vidović, Željko Hocenski

Original scientific paper
In attempts to increase the production quality and to decrease the production costs in ceramic tile industry almost every phase in the production process needs to be automated. One way of decreasing the production costs is to detect the tiles with defects before the baking process (biscuit tiles). The separated tiles are returned to the input material mixture and the energy is not spent on baking the tiles with defects. In this paper methods for edge and corner defects detection of a biscuit ceramic tile are presented. Methods are implemented and tested on biscuit ceramic tiles using area scan camera and a prototype conveyor line. These methods are evaluated by the defect detection capabilities and the real-time application features. The results show that the methods could be used for real-time application and that the defect tiles detection percentage can reach the required value.

Keywords: *ceramic tile, corner defect, crude biscuit tile, edge defect, visual inspection*

Otkrivanje neispravnosti rubova i kutova keramičkih pločica u stvarnom vremenu

Izvorni znanstveni članak
Kako bi se povećala kvaliteta i smanjili troškovi proizvodnje u keramičkoj industriji, gotovo sve proizvodne faze se automatiziraju. Jedan od načina smanjenja troškova proizvodnje je otkrivanje neispravnosti na keramičkim pločicama prije procesa pečenja (sirova keramička pločica). Neispravne sirove keramičke pločice vraćaju se u ulaznu sirovinu i na taj način smanjuje se potrošnja energije. U ovom radu predstavljene su metode za otkrivanje neispravnosti rubova i kutova sirovih keramičkih pločica. Sve su metode implementirane i testirane na sirovim keramičkim pločicama korištenjem industrijske linije i kamere s matričnim osjetnikom. Metode su ocjenjivane na osnovu rada u stvarnom vremenu i mogućnosti otkrivanja neispravnosti. Rezultati pokazuju da se metode mogu koristiti za rad u stvarnom vremenu i da postotak otkrivanja neispravnosti može zadovoljiti zahtijevanu vrijednost.

Ključne riječi: *keramička pločica, neispravnost ruba, neispravnost vrha, sirova keramička pločica, vizualna inspekcija*

1 Introduction

The modern ceramic tiles production is highly automated applying the most sophisticated technologies adjusted to the requirements on environment protection [1 ÷ 3] and energy savings or sustainable development [4 ÷ 6]. One of the developing production stages is the classifying and visual inspection stage. It is based on detection of the anomalies or defects on ceramic tiles based on visual inspection. The tiles without defects belong to the first class and the other tiles belong to the second and third class depending on the defection degree [7 ÷ 9].

The visual inspection is done in biscuit tile stage and in the final product stage, where the classifying is done. Many of the factories in ceramic tile industry continue to use the human visual inspection using specially trained experienced workers to grade the quality of the tiles. Such visual inspection depends much on subjective circumstances due to the human and causing the varying and instability of the quality. These workers are often affected by problems like eye fatigue, lack of attention or sickness. It is important to detect the defect tile earlier in production process to spare the energy, the costs and the time. If the tiles were inspected before the kiln, production cost would be decreased and the number of defective tiles that comes to human visual inspection by workers would be reduced [10 ÷ 12].

The ceramic tile production is a complex process. The ceramic tiles used for wall and floor covering are produced in mono-firing technology. The manufacturing process consists of a series of sub-processes or production stages like: preparation of raw materials, preparation of granules for pressing, shaping, drying of crude tiles, glazing, firing and packaging [13].

The production starts with the preparation of raw materials by making a specific mixture of various materials that are pressed to obtain the crude tile. The mixture contains several components like clay, granite and dolomite. Clay is exploited by the mine inside the factory area. After aging the clay is taken to a warehouse. The glazing material, frit and pigments, are delivered by suppliers from Italy, Spain and other countries. The next production stage is the preparation of granules for pressing. The mixture of components is ground and wetted by a process (by water and electrolyte) with the addition of ceramic wastes in a ball mill. After milling and sieving the liquid mixture is atomized into granules kept in the silo from which they are fed to the presses.

So, the next stage is pressing making the crude tiles by the usage of hydraulic high power presses with high productivity. The charge granulate is pressed under specific high pressure making several crude tiles simultaneously. The crude tiles go to drying stage after the pressing. The pressed tiles are introduced into the dryer. The crude tiles are dried by gradual warming. The drying time depends on the type of tiles (for wall or floor covering), and the moisture content of crude pressed tiles. After drying the hot crude tiles go to glazing and decorating stage. The fragile crude biscuit tiles are transported along the production line where several enamelling and silk-screening processes are applied, according to the tile type that is being produced (plain, textured, grained, etc.). The glazed tiles are assorted by robotic manipulator to the transport car that brings tiles to the kiln. The several cars from several production lines are controlled by automated transport system. At these stages, several types of defects can appear: geometrical defects (edge defects, corner defects, etc.), surface spots and colour anomalies [14, 15].

After the silk-screening process the tiles are baked in a kiln at temperatures in the range 1050 ÷ 1200 °C. This process transforms the biscuit tile into a rigid tile and reveals the authentic colours. In this stage some failures could appear as small pin holes, surface cracks, glazing coverage irregularity, texture printing misalignment or misprint, wrong colour and texture composition. The last production stage is classification and packing. The tiles are visually inspected and classified into categories by human trained workers. The tiles are packed into the carton boxes by type and class and asserted on wooden pallets wrapped with foil.

Research has been made on the topic of ceramic tile inspection, sorting and colour grading by Boukouvalas and his group [16, 17]. In [18] authors implemented a real-time system for biscuit tile inspection based on Hough transform that is focused only on corner defects. Edge defect detection based on Radon transform and boundary analysis using fuzzy thresholding is proposed in [14]. Surface and edge defect detection based on ceramic tile image registration has been researched in [19 ÷ 21].

Our methods propose to test the biscuit tile for corner and edge defects based on contour tracing analysis [22-24]. Methods are tested for real time application and defect detection rate. Future work will include inspection of biscuit tile surface defects (scratches, bumps, pits, texture and glazing errors, etc.).

The rest of the paper is organized as follows. Section 2 describes the biscuit tile prototype inspection setup. Detailed explanation of ceramic tile contour detection is given in Section 3. Section 4 describes the detection of the edge and corner biscuit tile defects. Experimental results are discussed in Section 5, and Section 6 concludes the paper.

2 Biscuit tile inspection setup

Prototype inspection system consists of two parts: PC based processing module and a conveyor belt production line with area scan camera.

Processing module uses Intel Q6600 processor based PC and a digital I/O card for sensor-actuator communication. Based on the data from the photo sensors the black and white image is captured and processed. If the tile has a defected edge or corner it is rerouted for recycling by a pneumatic actuator.

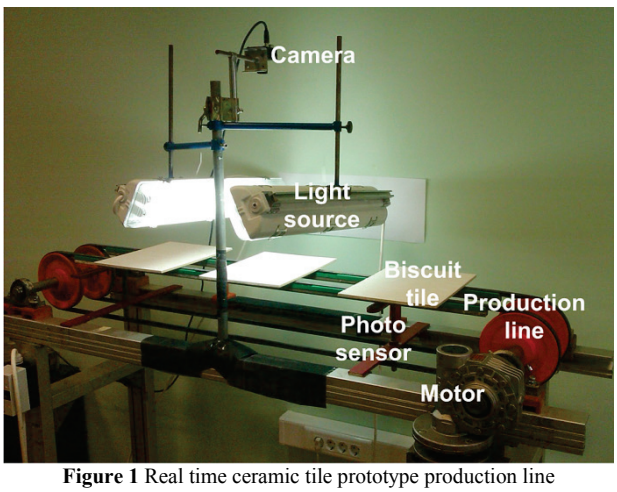
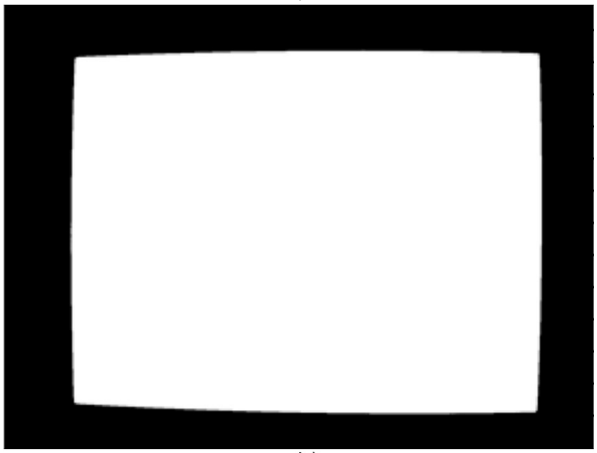


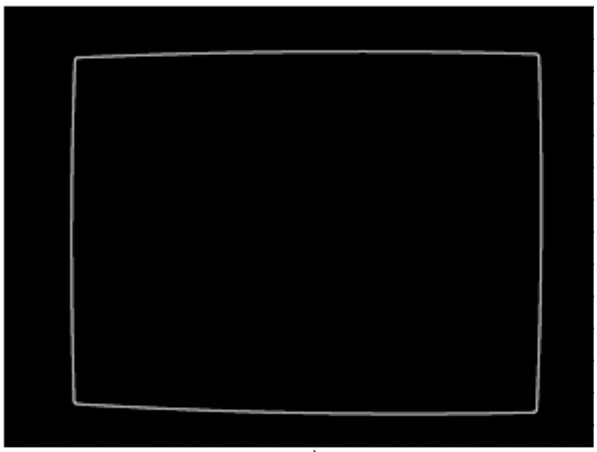
Figure 1 Real time ceramic tile prototype production line

Prototype production line uses a conveyor belt to transport the tiles. The tile is illuminated with two fluorescent tubes and the image is captured by the Basler A102FC camera c. f. Fig. 1. Camera uses firewire protocol for communication with the processing module.

a)



b)



c)

Figure 2 Biscuit tile a) input image, b) thresholded image, c) detected edge image

3 Ceramic tile contour detection

The input image is a black and white 1388 × 1038 pixel image c. f. Fig. 2a). Image is thresholded to get a binary image where ones represent a ceramic tile, and zeroes represent the background Fig. 2 b). Thresholding is applied using the Otsu method. The method works well when the image to be thresholded has clear histogram

peaks and valleys. It works for images whose histograms show clear bimodal distributions [25]. This is also the case with the ceramic tile image shown in Fig. 2a). Histogram of Fig. 2a) is depicted in Fig 3. It is clearly visible that the histogram is bimodal and thus the Otsu method can be applied. The Otsu method returns a threshold value that is applied to get the binary image. For every captured image the Otsu method is applied and new threshold value is calculated.

Thresholded image is used as an input for the edge detection method. Canny edge detection method was used for processing. It is one of the standard edge detection methods and is widely used in research. Canny's Edge Detector is optimal for a class of edges known as step edges (that is the case with the thresholded image shown in Fig. 2b)). The method detects edges as close as possible to the real edge [26]. Also Canny's Edge Detector has several optimal C++ implementations that can be used for real-time application. Result of the Canny edge detection method is depicted in Fig. 2c).

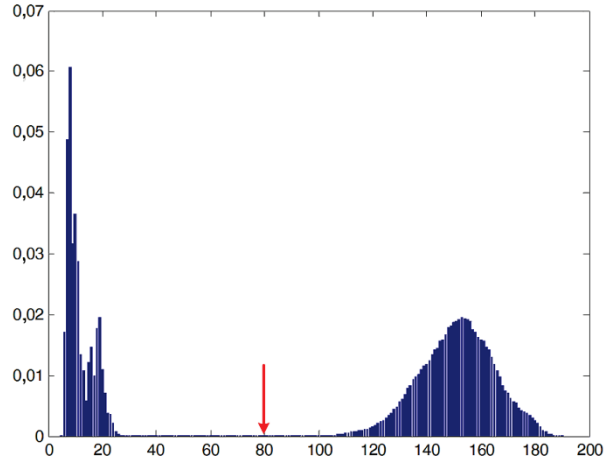


Figure 3 Normalized histogram of the ceramic tile image (red arrow marks the Otsu method threshold value)

Ceramic tile contour detection method uses the detected edge image as an input c. f. Fig. 2c). Process starts at the centre of the input image and moves pixel by pixel to the right until it finds the first white pixel (white pixels mark the edge). The pixel is marked as a starting point of the contour and a search direction $\alpha[0]$ ($\alpha[0]$ represents the absolute angle of the contour point) is chosen ($\alpha[0] = 0°$ for search direction up, $\alpha[0] = 180°$ for search direction down). Next contour point is found based on the current $\alpha[0]$ and the relative angle window. Fig. 4a) gives an example of the relative angle window where $\alpha[0] = 0°$. In the figure P1 to P5 mark the search direction priorities, P1 has the highest priority, P5 has the lowest priority. Based on the before stated priorities the next contour point is found. Absolute angle $\alpha[1]$ is calculated using (1) where $\beta[0]$ represents the relative angle according to $\alpha[0]$ relative angle window. The algorithm stops when it reaches the starting contour point or when the next contour point is not found.

$$\alpha[1] = \alpha[0] + \beta[0] \tag{1}$$

Figs. 4b) ÷ 4d) represent an example of the contour detection for four edge pixels. In Fig. 4b) contour point has an absolute angle $\alpha[0]=0°$. Based on the relative angle window and (1) absolute angle of the next contour point is $\alpha[1] = 0° - 45° = -45°$ as it is depicted in Fig. 4c). Absolute angles of the next two contour points are $\alpha[2] = 0°$ and $\alpha[3] = 45°$ c. f. Fig 4d).

Requirements for the above algorithm are:

- binary input image,
- one pixel edge width,
- continuous edge.

First two requirements are met using the thresholded image and the Canny edge detection method. The third requirement is fulfilled when there are no abrupt changes in the thresholded image.

The algorithm returns a vector $\alpha[i], i = \{0, \dots, n-1\}$ of all absolute angles of the contour points, where n represents the number of found contour points. Calculated vector is used in the next phase of the solution.

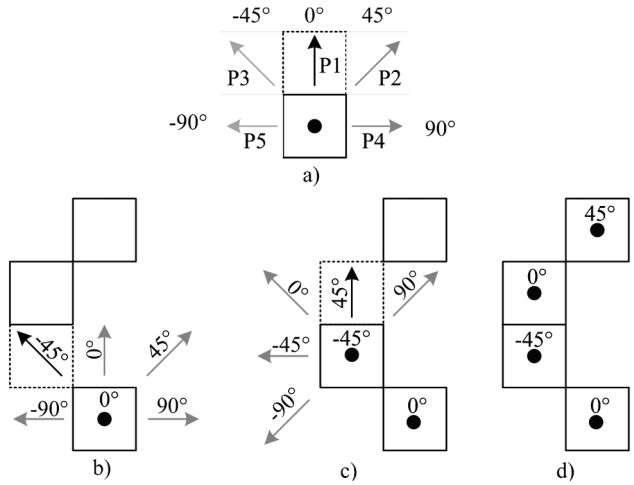


Figure 4 Apsolute angle calculation (square marks a pixel, dashed square marks the next pixel): a) relative angle window and direction priorities, b) step one absolute angle calculation, c) step two absolute angle calculation, d) step three and four absolute angle calculation

4 Edge and corner defect detection

For the edge and corner defect detection we used two methods. First step is to find corners of the ceramic tile and detect edge defects between corners. For this purpose a new method was created called MASD (**M**oving Average **A**bsolute Angle **S**tep **D**ifference). The second method CMASD (**C**orner **MASD**) detects only corner defects and it is a modification of the before mentioned MASD method. In the following subsections both methods are explained.

4.1 MASD

MASD method uses the moving average filter to smooth the absolute angle values of the given contour. Moving average value is calculated by (2) where W represents the moving average window size [27].

$$\bar{\alpha}[i] = \frac{1}{W}\sum_{j=0}^{W-1} \alpha[i \cdot W + j], i = \{0, \dots, n-1\}. \tag{2}$$

The smallest defect that can be detected depends on the moving average window size and image resolution. If the average window size is too small the algorithm also detects as defects the errors caused by lens distortions (in our case barrel effect), if the average window size is too large the algorithm smoothens out the smallest defects and does not detect them.

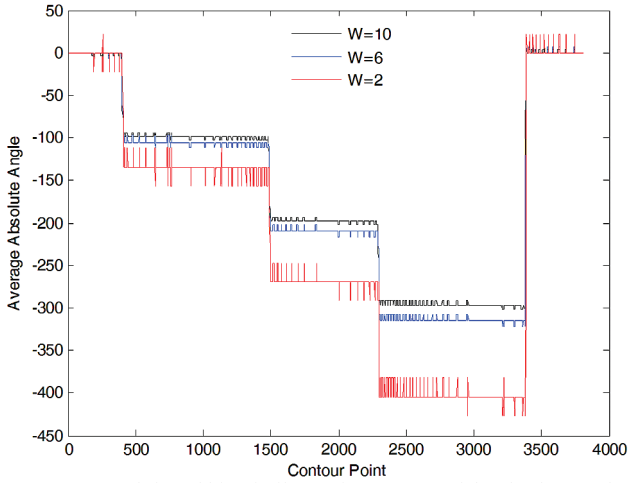


Figure 5 Nondefected biscuit tile moving average of the absolute angle for window sizes: 2, 6, 10

Fig. 5 shows the average absolute angle values for three window sizes of nondefected biscuit tile. In the image a step function represents a corner (four step functions mark the position of the four corners of the biscuit tile). Also it is clearly visible that distortion errors are more emphasized for $W = 2$ than for $W = 10$.

The smallest defect is defined by (3) where *SD* represents the number of error pixels of the smallest defect and *DE* represents the maximum number of distortion error pixels. Based on the experimental results for the biscuit tile size 30 × 30 cm and image size 1388 × 1038 pixels, *DE* equals 3 pixels and *SD* equals 6 pixels. From *SD*, biscuit tile size and image size smallest defect size length in millimetres can be calculated based on (4) or (5) where *a* and *b* represent height and width of the tile in mm and px_a, px_b represent height and width of the tile in pixels. If the defect is found on the left or the right side of the image equation (4) is used, otherwise if the defect is found on the top or the bottom side of the image equation (5) is used. For square biscuit tiles and square image sizes (4) = (5).

$$SD = 2 \cdot DE \tag{3}$$

$$SD_{la} = \frac{a}{px_a} \cdot SD \tag{4}$$

$$SD_{lb} = \frac{b}{px_b} \cdot SD \tag{5}$$

$$DIFS_{\bar{\alpha}}[i] = \bar{\alpha}[i+S] - \bar{\alpha}[i], i = \{0, \ldots, n-1\}. \tag{6}$$

Vector $\bar{\alpha}[i]$ is used in the next step of the MASD method. Difference of the two average absolute angle values is calculated by (6) where *S* represents the distance (step) between two contour points used in the calculation. Different step values give similar results but the main goal is to maximize the difference between the $DIFS_{\bar{\alpha}}$ value of the edge defect and the $DIFS_{\bar{\alpha}}$ of the distortion error (to emphasize the defect).

Fig. 6 depicts the $DIFS_{\bar{\alpha}}$ values for three step sizes of the biscuit tile image with one edge defect (defect size is 6 pixels). Four corners of the tile are clearly visible from all of the three graphs (points on the graphs marked from 1* to 4*). Edge defect is located near the third corner (3*). From the magnification area of Fig. 6 it can be seen that the edge defect creates greater fluctuations in the graphs than the lens distortion errors.

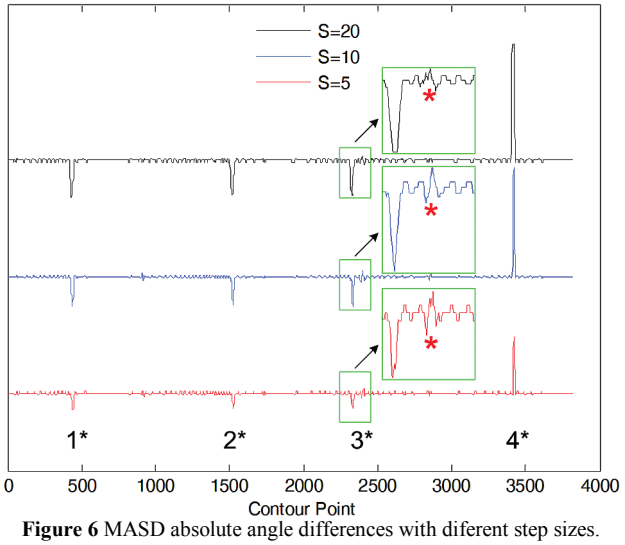


Figure 6 MASD absolute angle differences with diferent step sizes. Green rectangles mark the magnification area, red star marks the edge defect, *n** marks a corner.

```
1:  thresh=t1;
2:  corner_number=0;
3:  error=0;
4:
5:  for i=0 to n-1
6:     cond=difsa[i]>thresh OR
       difsa[i]<-thresh;
7:     if(cond)
8:        i0=i;
9:        while(cond)
10:       i++;
11:       end while;
12:       corner_number++;
13:       c_coordinates[corner_number]=
          coordinates(i0,i);
14:    end if;
15: end for
16:
17: if(corner_number!=4)
18:    error=1
```

Figure 7 Thresholding algorithm for edge defect detection

Based on the $DIFS_{\bar{\alpha}}$ value a thresholding algorithm is used to find the edge defect c. f. Fig. 7. The thresholding algorithm counts the number of found corners and returns the coordinates of the corners. The algorithm finds the first contour point i_0 of a corner or a defect (contour point must fulfil the condition in line 6). The last contour point of the current corner (defect) is found after the execution of the while loop (lines 9 to 11). Based on the first and the last contour point, coordinates of the corner (defect) are calculated (line 13). Because the tile is rectangular the

resulting *corner_number* should be four or else the algorithm returns an error.

If *corner_number* = 4 next step in the edge and corner defect detection process is the CMASD method.

4.2 CMASD

CMASD (Corner MASD) is a modification of the earlier explained MASD method and it is used for corner defects detection.

Inputs for CMASD are four corner coordinates $c\_coordinates[i]$, $i = \{1, 2, 3, 4\}$ (calculated by the MASD method) and absolute angle values of contour points around the corners.

Number of contour points (n_c) around the centre of the corner ($c\_coordinates[i]$, $i = \{1, 2, 3, 4\}$) that are used for CMASD algorithm depends on tile and image size. For tile size $a \times b$ mm and tile image size in pixels $px_a \times px_b$, n_c is calculated by (7). In (7) l_{pxc} represents the length of the corner (without defect) in pixels and one represents the centre contour point of the corner.

Absolute angles for all four corners are smoothed using moving average window (8) where W_c represents corner moving average window size.

Calculating the difference of two average absolute angle values is the next step in the algorithm (9). In the equation S_c represents the distance (step) between two contour points used in the calculation. Value of S_c that maximizes $DIFS_{\bar{\alpha}_c}$ value of the nondefected corner is used in the calculation.

```
1:  thresh=t2;
2:  corner=0;
3:  error=0;
4:
5:  for i=0 to nc-1
6:     cond=difsac[i]>thresh OR
       difsac[i]<-thresh;
7:     if(cond)
8:        while(cond)
9:        i++;
10:       end while;
11:       corner++;
12:    end if;
13: end for
14:
15: if(corner!=1)
16:    error=1
```

Figure 8 Thresholding algorithm for corner defect detection

$$n_c = 4 \cdot \left(\frac{a+b}{px_a + px_b} \right) \cdot l_{pxc} + 1 \tag{7}$$

$$\bar{\alpha}_c[i] = \frac{1}{W_c} \sum_{j=0}^{W_c-1} \alpha[i \cdot W_c + j], i = \{0, \ldots, n_c - 1\} \tag{8}$$

$$DIFS_{\bar{\alpha}_c}[i] = \bar{\alpha}_c[i + S_c] - \bar{\alpha}_c[i], i = \{0, \ldots, n_c - 1\}. \tag{9}$$

Thresholding algorithm for finding corner defects based on $DIFS_{\bar{\alpha}_c}$ values is shown in Fig. 8. The algorithm is executed for all four corners. If condition *cond* is true, a corner is found. If the resulting corner number (*corner*) is not equal to one (line 15) defect is found on the specific corner. The algorithm is repeated for other three corners.

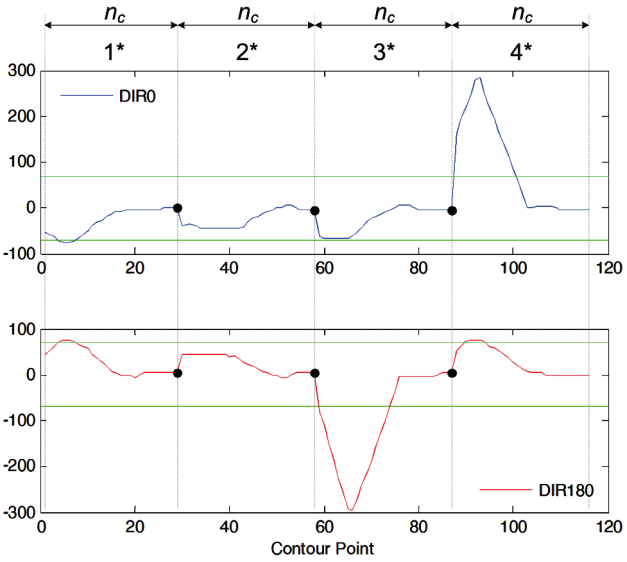


Figure 9 CMASD absolute angle differences in direction 0° (DIR0) and 180° (DIR180) for W_c=10 and S_c=10.

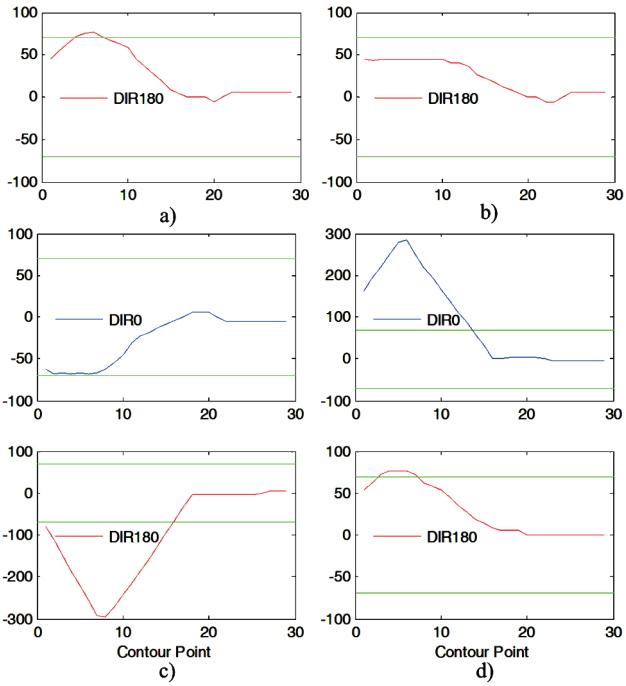


Figure 10 Magnified CMASD absolute angle differences for four corners: a) corner 1*, b) corner 2*, c) corner 3* and d) corner 4*.

Fig. 9 depicts the $DIFS_{\bar{\alpha}_c}$ values for a biscuit tile with two corner defects (corner 2* and 3*) for both directions (DIR0 – search direction up and DIR180 – search direction down). All four corners of the biscuit tile are displayed. Black dots mark the borders between the corners and green lines mark the threshold level for the given example. From Fig. 9 it can be seen that corner 4* in DIR0 direction has higher $DIFS_{\bar{\alpha}_c}$values than other three corners. This is the result of the absolute angle value change from –270° to 0°. Because of this it is impossible to detect defects on all four corners with the same threshold value. To avoid this problem we calculated $DIFS_{\bar{\alpha}_c}$ values for both search directions (DIR0 and DIR180). In that case higher values of $DIFS_{\bar{\alpha}_c}$ appear on

different corner and the same value of the threshold can be used for all four corners. In Fig. 9 higher $DIFS_{\bar{\alpha}_c}$ value is shifted from corner 4* in DIR0 to corner 3* in DIR180. For direction DIR180 corner 3* has no defect, but for the DIR0 defect on the corner is found.

Fig. 10 shows $DIFS_{\bar{\alpha}_c}$ values for both search directions of all four corners. It is clearly visible that corners in Fig. 10 b) and 10 c) have corner defects ($DIFS_{\bar{\alpha}_c}$ value is between threshold values, green lines) and corners in Fig. 10 a) and 10 d) have no defects ($DIFS_{\bar{\alpha}_c}$ value crosses the threshold value). Result of the CMASD method for the given example in Fig. 9 and Fig. 10 would be: corner 1* - OK, corner 2* - ERROR, corner 3* - ERROR, corner 4* - OK.

5 Experimental results

Experimental results are obtained with Intel Core2 QUAD 6600 processor based PC. PC is running Windows 7×64 OS using 6 GB of RAM. MASD and CMASD methods are developed in Visual Studio 2008 C++ programming language [28]. Implementation is sequential, only one core of the processor is used. OpenCV2.1 library is used for image and result visualization [29]. All experiments are carried out on the prototype line described in section II (c.f. Fig. 1).

First experiment is used to test the real time performance of the proposed methods. Test was executed using two different image sizes: 1388 × 1038 pixels (1,4 MP) and 2776 × 2076 pixels (5,7 MP). Average times of 100 tests were measured for two execution modes (debug mode and process mode) and two proposed methods. Debug mode is used for parameter testing and evaluation. In this mode all intermediate values are saved to the hard drive for validation of the results. Hard drive data transfer speed of the processing module has a great influence on execution time in this mode (with the increase of the data transfer speed execution time decreases). Times in Tab. 1 and Tab. 2 do not include camera grabbing time. Average camera grabbing time for image size 1388 × 1038 pixels is 65 ms and for image size 2776 × 2076 pixels is 260 ms.

Table 1 Debug mode average execution time for two tile sizes

Method	Time / ms	
	Image 1388 × 1038	Image 2776 × 2076
MASD	149,81	650,12
MASD + CMASD	152,56	679,32

Table 2 Process mode average execution time for two tile sizes

Method	Time / ms	
	Image 1388 × 1038	Image 2776 × 2076
MASD	46,15	185,94
MASD + CMASD	46,35	186,15

Table 3 Worst case execution time for two tile sizes

Method	Time / ms			
	Image 1388 × 1038		Image 2776 × 2076	
	Debug	Process	Debug	Process
MASD	187	47	1469	188
MASD + CMASD	233	52	1453	203

Table 4 MASD + CMASD results for 100 tiles (%)

Parameters	Correct	False positives	False negatives
params 1	90	4	6
params 2	90	1	9
params 3	70	3	27

Factory statistics show that it is necessary to process maximum of 60 tiles in one minute. This means that execution time of all algorithms used for tile inspection cannot be longer than 1000 ms. Tab. 1 and Tab. 2 show that MASD and CMASD methods fulfil these requirements and can be used for real time biscuit tile inspection. Instead of average time, maximum execution time of 100 tests can be compared (Tab. 3) with real time requirements. Measured times for the process mode c.f. Tab. 3 show that real time requirements are also met for worst case execution time ($DIFS_{\bar{\alpha}_c}$ values calculated for both search directions).

Second experiment is used to test the percentage of defect detection for proposed methods. Experiment is executed for different window sizes (W and W_c) and different threshold values (t_1 and t_2). Threshold values t_1 and t_2 for all runs are determined using 10 *learning tiles*. These tiles must be without edge and corner defects.

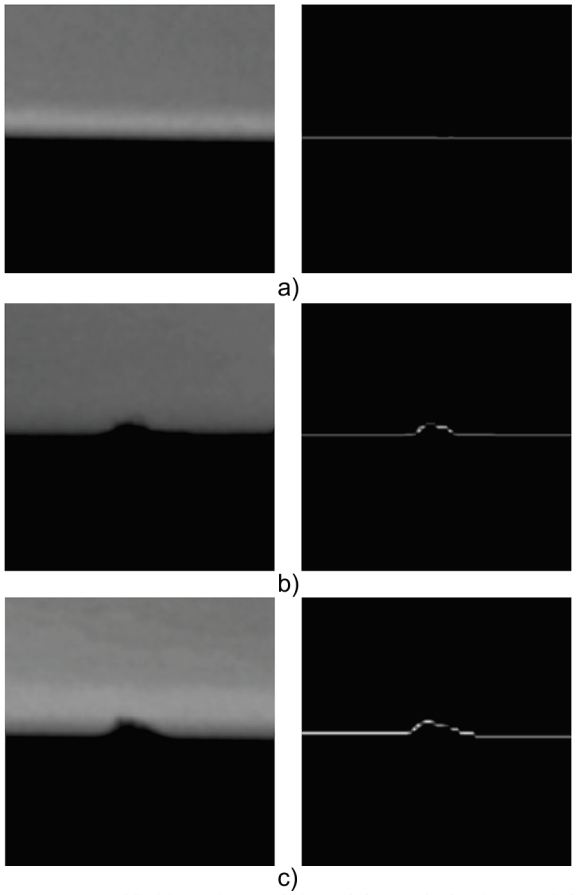


Figure 11 Magnified input image (left) and detected edge image (right): a) edge without defect, b) small edge defect, c) large edge defect.

In the first run (params 1) window sizes were set to 6, threshold values were $t_1 = 17$, $t_2 = 65$ and step sizes S and S_c were set to 6. In the second run (params 2) window sizes were changed to 10, threshold values were changed to 10 and 75 respectively and step sizes were changed to 10. The third run (params 3) is executed with window sizes $W = W_c = 20$, threshold values $t_1 = 7$, $t_2 = 40$ and step sizes $S = S_c = 20$. For all runs 100 different biscuit tiles were used. 40 used tiles were without defects and 60 tiles had one of defects shown in Fig. 11 and Fig. 12. These defects are most common edge and corner defects that occur in the factory during production. Fig. 11a)

shows magnified input image (left image) and magnified detected edge image (right image) for tile without edge defects. Two examples of edge defects are shown in Fig. 11b) (small edge defect) and Fig. 11c) (large edge defect). Fig. 12a) depicts magnified tile corner (input image and detected edge image) without defects while Fig. 12b) and c) shows examples of small and large corner defect respectively.

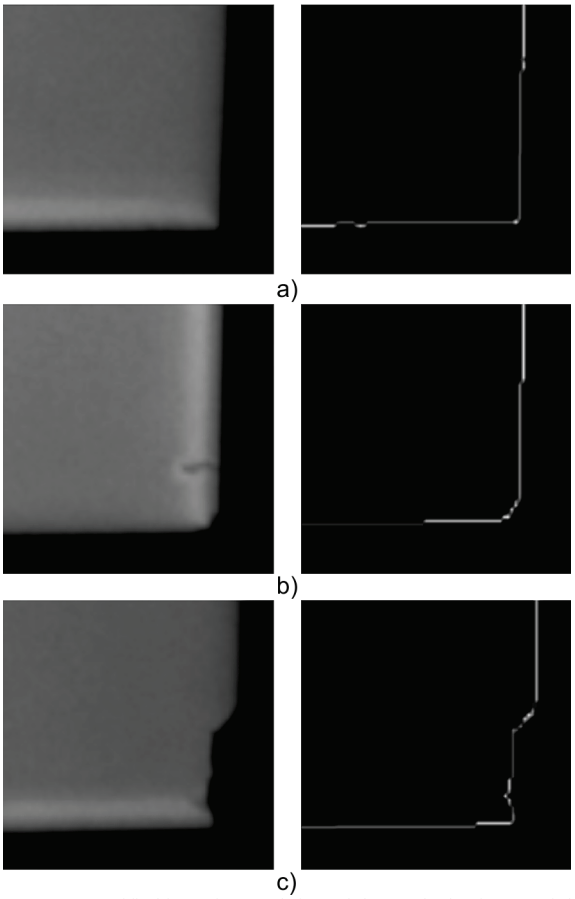


Figure 12 Magnified input image (left) and detected edge image (right): a) corner without defect, b) small corner defect, c) large corner defect.

Results of the experiment, divided in three categories, are shown in Tab. 4. The meaning of each category is as follows:

- Correct – tile did not have any defects and the methods did not detect any defect; tile had one or more defects and the methods detected all of them.
- False positive – tile did not have any defect but the methods falsely detected one or more defects; tile had defects but the methods additionally detected one or more defects.
- False negative – tile had one or more defects but the methods did not detect one or more of the defects.

From Tab. 4 it can be seen that the first run of the experiment has an increased percentage of "False Positives". Third run has a high percentage of "False Negatives" (majority of "False Negatives" are large factory unacceptable defects) and the second run gives the best results. The results are expected because of the fact that for small window sizes edge distortion errors are detected as defects while for large windows sizes small defects are smoothed (c.f. Section III.A.). Percentage of "False Negatives" for second run is acceptable because undetected defects are very small defects (defects smaller than depicted in Fig. 11b) and Fig. 12b)) which are tolerated for factory second or third quality class of ceramic tiles. Low percentage of "False Positives" is significant because correct tiles are not declared as false and mistakenly recycled. With this taken into account the methods can reach 99 % of detection rate for three quality classes.

6 Conclusion

Edge and corner defects are common defects in ceramic tile industry. If defects are detected on biscuit tile before the entry into the glaze kiln there would be no need for grinding of the backed tile. The defected biscuit tile can immediately be reused in material mixture for pressing. Detection of these defects before kiln will reduce production cost and will save time for visual inspection after the tile is backed.

In this paper we proposed methods for biscuit tile edge and corner inspection, MASD and CMASD respectively. MASD method is used for edge defects detection while its modification CMASD is used for corner defects detection. The methods must confirm with the real time requirements of the factory production line (60 tiles in one minute) and must have a high percentage of unacceptable defects detection (specified by the factory). The methods were tested on a real time ceramic tile prototype production line and a PC based processing module. For real time requirements experiment two image sizes of 100 biscuit tile samples were used. Defect detection experiment used 100 biscuit tile samples with different edge and corner defects.

Results of the first experiment show that MASD and CMASD fulfil the real time requirements for both image sizes (1000 ms to inspect a single tile). Also the results show that there is sufficient time for additional surface inspection methods (797 ms remains in worst case time scenario for 2776 × 2076 image). Second experiment shows that proposed methods, when used with the appropriate parameters, can correctly detect 90 % of defects. If the factory produces second and third ceramic tile quality class, for appropriate parameters, "False Negatives" can be declared as correct. Therefore the percentage rate of correctly inspected tiles increases to 99 %.

Future work will include different real time surface inspection methods. If new methods, including MASD and CMASD, do not fulfil real time requirements MASD and CMASD methods can be parallelized (multithreaded implementation).

Acknowledgements

Authors thank Keramika Modus d.o.o. Orahovica on biscuit tile samples and experimenting on production line.

7 References

[1] HRN EN 14 411: 2004 Keramičke pločice-Definicije, razradba, značajke i označavanje (EN 14 411:2006), 2nd ed. Dec. 2008., HZN.

[2] HRN ISO 9001: Sustavi upravljanja kvalitetom-Zahtjevi (ISO 9001: 2008; EN ISO 9001: 2008), 5th ed., April 2009., HZN.
[3] HRN EN ISO 14001: Sustavi upravljanja okolišom-Zahtjevi sa uputama za primjenu (ISO 14001: 2004, EN ISO 14001:2004), 3rd ed, April 2009., HZN.
[4] Breedveld, L.; Fregni, A.; Timellini, G.; Casoni, G.; Busani, G. Application of Eco-efficiency in the context of the IPPC Directive. The case of fabric filter in the Italian ceramic tile industry, in: Eco-efficiency for sustainability. // Proceedings of the International conference on Eco-efficiency for sustainability / Leiden, 2004, pp. 35 - 36.
[5] European Commision Reference Document on Best Available Techniques in the Ceramic Manufacturing Industry. 2007. URL: http://eippcb.jrc.ec.europa.eu/reference/BREF/cer_bref_0807.pdf.
[6] Hocenski, V.; Hocenski, Ž.; Vasilić, S. Application of Results of Ceramic Tiles Life Cycle Assessment due to Energy Savings and Environment Protection. // Proceedings of the IEEE International Conference on Industrial Technology, IEEE ICIT 2006 / Mumbai, India, 2006, pp. 2972-2977.
[7] Costa, C. E.; Petrou, M. Automatic registration of ceramic tiles for purpose of fault detection. // Machine Vision and Applications, 11 (2000), pp. 225-230.
[8] Hocenski, Ž.; Keser, T. Failure detection and isolation in ceramic tile edges based on contour descriptor analysis. // Proceedings of the Mediterranean Conference on Control & Automation, 2007. MED '07 / Athens, 2007, pp. 1-6.
[9] Valiente, J. M.; Acebron, F.; García, F. An Automatic Visual Inspection System for Ceramic Tile Manufacturing Defects. Proceedings of the IASTED International Conference on Signal Processing and Comunications (1988)
[10] Hocenski, Ž.; Keser, T.; Baumgartner, A. A Simple and Efficient Method for Ceramic Tile Surface Defects Detection. // Proceedings of the 2007 International Symposium on Industrial Electronics, Vigo, 2007, pp. 1606-1611.
[11] Keser, T.; Hocenski, Ž.; Hocenski, V. Intelligent Machine Vision System for Automated Quality Control in Ceramic Tile Industry. // Strojarstvo, 51 (2010), pp.101-110.
[12] Keser, T. Automated intelligent system for ceramic tiles classification, University J. J. Strossmayer in Osijek, Faculty of Electrical Engineering, dissertation, Osijek, 2009.
[13] Hocenski, V. New approach to decrease the influence of ceramic industry to environment based on neural networks, dissertation, University in Zagreb, Faculty of chemical engineering and technology, Zagreb, 2012.
[14] Mansory, M.; Tajik, H.; Mohamedi, G.; Pashna, M. Edge Defect Detection in Ceramic Tile Based on Boundary Analysis Using Fuzzy Thresholding and Radon Transform. In: Proceedings of the IEEE International Symposium on Signal Processing and Information Technology, 2008 (ISSPIT), pp. 58-62.
[15] Matić, T.; Hocenski, Ž. Parallel Processing with CUDA in Ceramic Tiles Classification. // Proceedings of the 14th international conference on Knowledge-based and intelligent information and engineering systems KES'10 / Cardiff, 2010, pp. 300-310.
[16] Boukouvalas, C. ASSIST: automatic system for surface inspection and sorting of tiles. // Journal of Materials Processing Technology, 82, (1998), pp. 179-188.
[17] Boukouvalas, C.; Kittler, J.; Marik. R.; Petrou, M. Automatic color grading of ceramic tiles using machine vision. // IEEE Trans. on Industrial Electronics, 44, (1997), pp. 132-135.
[18] Valiente, J.; Acebron, F.; Lopez, F. A ceramic tile inspection system for detecting corner defects. // Proceedings of the IX Spanish Symposium on Pattern Recognition and Image Analysis / Castellón, 2001, pp. 213-218.
[19] Costa, C. E.; Petrou, M. Automatic registration of ceramic tiles for the purpose of fault detection. // Machine Vision and Applications, 11, 5(1999), pp. 225-230.
[20] Valiente, J.; Lopez, F.; Acebron, F. An image registration method for ceramic tile inspection purposes. // Proceedings of the 2001 Internacional Conference on Quality Control by Artificial Vision / Gatlinberg, 2001.
[21] Novak, I.; Hocenski, Ž.; Slišković, D. Using Pixel Pairs Difference for Visual Inspection of Ceramic Tiles. // Tehnicki vjesnik-Technical Gazette, 12, 3-4(2005), pp. 3-9.
[22] Hocenski, Ž.; Keser, T. Failure Detection and Isolation in Ceramic Tile Edges Based on Contour Descriptor Analysis. // Proceedings of the MED '07. Mediterranean Conference on Control & Automation / Athens, 2007, pp. 1-6.
[23] Miyatake, T.; Matsushima, H.; Ejiri, M. Contour representation of binary images using run-type direction codes. // Machine Vision and Applications, 9, 4(1997), pp. 193-200.
[24] Wagenknecht, G. A Contour Tracing and Coding Algorithm for Generating 2D Contour Codes from 3D Classified Objects. // Pattern Recognition, 40, 4(2007), pp. 1294-1306.
[25] Hui-Fuang, N. Automatic thresholding for defect detection. // Proceedings of the 2004 IEEE First Symposium on Multi-Agent Security and Survivability / Philadelphia 2004, pp. 532-535.
[26] Canny, J. A computational approach to edge detection. // IEEE Trans on Pattern Analysis and Machine Intelligence, PAMI-8, 6(1986), pp. 679-698.
[27] Ya-lun, C. Statistical analysis. Holt, Rinehart and Winston, Toronto, 1969.
[28] Soulié J. C++ Language Tutorial, cplusplus.com, 2008.
[29] Bradski, G. The OpenCV Library // Dr. Dobb's Journal of Software Tools, 25, 11(2000), pp. 122-125.

Authors' addresses

Tomislav Matić, research assistant, dipl. ing.
Faculty of Electrical Engineering in Osijek
Kneza Trpimira 2b, 31000 Osijek, Croatia
tmatic1@etfos.hr

Ivan Vidović, research assistant, mag. ing. comp.
Faculty of Electrical Engineering in Osijek
Kneza Trpimira 2b, 31000 Osijek, Croatia
ividovi2@etfos.hr

Željko Hocenski, full professor, PhD.
Faculty of Electrical Engineering in Osijek
Kneza Trpimira 2b, 31000 Osijek, Croatia
zeljko.hocenski@etfos.hr