The algorithm for automatic detection of the calibration object

Cite as: AIP Conference Proceedings **1836**, 020070 (2017); https://doi.org/10.1063/1.4982010
Published Online: 05 June 2017

Kruglov Artem, and Ugfeld Irina

The Algorithm for Automatic Detection of the Calibration Object

Kruglov Artem[1, a)] and Ugfeld Irina[1, b)]

[1]*Ural Federal University, Russian Federation, Russia*

[a)]Corresponding author: avkruglov@yandex.ru
[b)]irinaugfeld@gmail.com

Abstract. The problem of the automatic image calibration is considered in this paper. The most challenging task of the automatic calibration is a proper detection of the calibration object. The solving of this problem required the appliance of the methods and algorithms of the digital image processing, such as morphology, filtering, edge detection, shape approximation. The step-by-step process of the development of the algorithm and its adopting to the specific conditions of the log cuts in the image's background is presented. Testing of the automatic calibration module was carrying out under the conditions of the production process of the logging enterprise. Through the tests the average possibility of the automatic isolating of the calibration object is 86.1% in the absence of the type 1 errors. The algorithm was implemented in the automatic calibration module within the mobile software for the log deck volume measurement.

INTRODUCTION

The purpose of the software for the log deck volume measurement is computing of the quantity and volume of the stacked logs via image analysis, i.e. photogrammetry. The software is developed for tablets and other mobile devices under Android OS and adapted for manipulating with images. The execution sequence of the user is the following:

- Loading of images. The number of images depends on the availability of the sidewalls of the deck for photograph. The measurement could be carried out via one or two projections of the log deck.
- Calibration of each image. Calibration means the determination of the inner and outer parameters of the camera. There are manual and automatic calibration modes in the software.
- Automatic search of the log cuts in the image.
- Manual editing. It is necessary to correct the results of the automatic isolating, i.e. to eliminate the 1 and 2 type errors and adjust log cut edges.
- Achievement and analysis of the results. Software output is saved as a detail report in the local database for further editing or data export into delivery note in PDF format.

This Paper is devoted to the image automatic calibration. According to the selected procedure of the image capture, the camera is arranged opposite the log deck sidewall and the log deck itself is determined as a set of the coplanar log cuts. In view of this assumption the problem of the calibration is reduced to the determination of the scale coefficient to move beyond pixels to millimeters. Thus the result of the calibration is two values which determine the vertically and horizontally costs of the pixel unit ([X,Y] mm/pix)

The common matching approach was chosen for the calibration. It assumes the comparison with the object which geometry is known in advance. The calibration object in the concerned task is the equilateral triangle with red markers in the corners. The distance between markers is 50 cm. (Fig. 1).

The workflow of the automatic calibration could be divided into several stages:

- Image preprocessing in order to decrease noise and remove irrelative artifacts
- Detection and isolation of the calibration object contours

- Detection and isolation of the contours of the calibration object markers

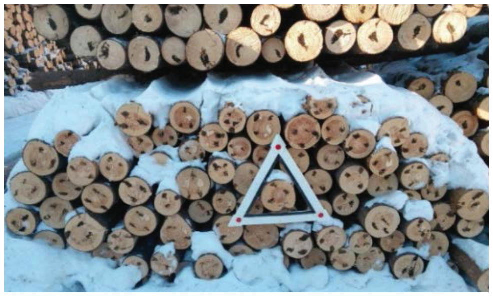

FIGURE 1. The calibration object in the processed image

PREPROCESSING

The isolated triangle has white color, whereas logs are much darker . Thus binarization with a threshold closed to the white color has to emphasize the edges of the triangle. The binarization has two stages:

1. Image conversion into gray scale. At this stage three-channel image (model RGB) is converted into one-channel according to formula:

$$y(i,j) = 0,299 \cdot R(i,j) + 0,587 \cdot G(i,j) + 0,114 \cdot B(i,j) \tag{1}$$

where i, j – pixel coordinates

2. Appliance of the threshold filter, that leads to black-and-white output image.

There are lights, shadows and reflections of neighbor objects on the calibration triangle. Thus, the value of the binarization threshold is not equal to the white color of the implemented color model, it should differ by the some value. It has been discovered by practical consideration that the optimal threshold value for the given task is 0.74 of the lightest color value in the image. From there the threshold is executed according to the equation (2):

$$y(i,j) = \begin{cases} 0, & if\ x(i,j) < 0,74 * maxGrVal \\ 1, & if\ x(i,j) \geq 0,74 * maxGrVal \end{cases} \tag{2}$$

where i, j – pixel coordinates.

In order to exclude irrelative artifacts from the image, such as fallen leaves in the foreground, log cut texture, etc. the median filter is implemented with aperture 5. As far as the input image is in the black-and-white mode, filtering is applied to one channel. The result of the filtering is presented in Fig. 2.

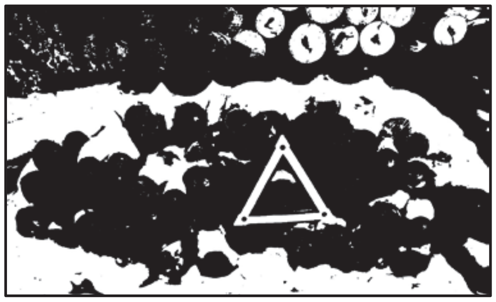

FIGURE 2. Image after threshold and median filtering implementation.

DETECTION AND ISOLATION OF THE TRIANGLE CONTOURS

Canny edge detector is used to detect the contours of the objects and target triangle in particular.

Each detected edge has to be represented as a set of parameters and analyzed in order to select the contours of the target object. Function *findContours* from OpenCV was used to obtain the list of the contours in the image where each contour is represented as a sum of the points. However, the detected triangle edges are not straight lines due to the overlapping objects and other clutters. Thus the approximation is implemented on the next stage.

Approximation

Approximation is performed by the Ramer–Douglas–Peucker algorithm. The idea of this method is in the following:

1. The sum of the points forming curve and distance ε > 0 are the inputs of the algorithm
2. Algorithm divides curve recursively. The first and the last points remain intact. Then the algorithm finds the point most distant from the obtained segment. If the point located closer than ε then the rest points from the sum are excluded from the analysis and the obtained line approximate polygonal curve with accurate to ε.
3. If the distance is bigger than ε, the algorithm callbacks for the subsets from the first to the selected point and from selected to the last point. The selected point is added to the final list.
4. After completing the algorithm recursive iterations the output polygonal curve is on the basis of the listed points.

As far as triangle contour is also a polygonal curve with 3 vertexes, the ε value has to be selected in the way that these 3 vertexes will remain intact whereas other kinks will be smoothed. It was practically found that the optimal value of ε is 20 for the given task.

Metrics

To make a decision whether the contour is a triangle or not a number of the metrics was implemented:

1. The amount of contour kinks is equal to 3, as far as triangle has three corners.
2. The perimeter of the triangle has to be no less than 1/70 of the image area. This assumption allows to exclude from the analysis small image artifacts which could have the triangle shape.
3. The analyzed contour is matched with the equilateral triangle contour by the matchShapes function from OpenCV library. This function compares two contours and returns the value of the similarity according to (3). The less the value, the more the similarity. The idea of the function is based on the H-U moments computing. It provides the invariance to the scale, rotation and reflection.

$$I\left(A,B\right)=\sum_{i=1..7}\left|m_i^A-m_i^B\right|,\quad \begin{array}{l} m_i^A=sign\left(h_i^A\right)\cdot\log h_i^A \\ m_i^B=sign\left(h_i^B\right)\cdot\log h_i^B \end{array} \tag{3}$$

where h_i^A, h_i^B – H-U moments of the A and B contours respectively,
$I(A,B)$ – the similarity value.
For the purpose of the triangle contour validity the constraint $I(A,B) < 0{,}3$ is used.

ISOLATION OF THE CALIBRATION OBJECT

This stage involves the isolation of the contours which are the contours of the triangles. The several triangle contours in the image are possible, but only one of them is a contour of the inner triangle of the calibration object.

As far as calibration object contour consists of the contours of the inner and outer triangles and three contours of the circle markers which are located between inner and outer triangles, and the proportions of the sizes of these elements are known, it is possible to determine the calibration object as following:

1. The search of the contour of the inner triangle is the first step. There are several contours have been detected at the previous stage, so the coordinates of the probable outer triangle contour are computed for each of them according to the formula:

$$L_o/L_i=1.45 \tag{4}$$

where L_o – the side length of the outer triangle,

L_i – the side length of the inner triangle .

2. The circle markers are located within triangles. They are detected identical to the triangle search expect the fact that applied metrics are comparison with minimum perimeter and matching with circle contour.
3. For each circle the condition (5) is checked. The side length of the inner triangle to the circle diameter ratio in millimeters is known. The same ratio is computed in pixels and the obtained proportions are compared. The circles which fail to meet the (5) are eliminated. If the three circles are left then they and triangle contours compose the calibration object.

$$0,6 < \frac{L_i(mm) * D(pix.)}{D(mm) * L_i(pix.)} < 1,5 \qquad (5)$$

where $\frac{L_i(mm)}{D(mm)}$ – the inner triangle to the circle diameter ratio in millimeters,

$\frac{L_i(pix.)}{D(pix.)}$ – the inner triangle to the circle diameter ratio in pixels.

4. The operations 1-3 are performed for each contour, thus they are parallelized to speedup algorithm.
5. In case of the more than one calibration object in the algorithm output, the one with the largest area is selected.

Fig. 3 shows the example of the detected calibration object.

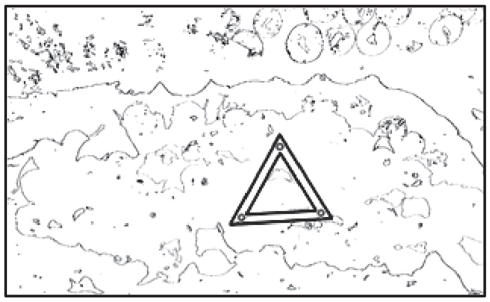

FIGURE 3. Detected calibration object.

ALGORITHM PRIMARY TESTING AND MODIFICATION

The algorithm was tested on the set of 20 photos captured in the autumn and winter seasons. The testing has shown that the bottleneck of the algorithm is threshold filtering. Different images require different threshold values because:

- Calibration object could be located in the shadows, whereas there are some much lighter regions of the image In that case binarization with high threshold value will assign calibration object to the background. Thus the proper threshold value is 50-60% of the lightest pixel in the image.
- Otherwise, if the calibration object located in the lights, the flecks of light could appear on its black parts so the applying of the small threshold value will lead to edge blurring and pinched on algorithm performance.

In the view of above it was decided to test the algorithm with a range of threshold values from 0.5 to 0.8 of the lightest image pixel intensity in increments of 0.05. Testing report is given in the Table 1. As can be seen there is no universal threshold value that allows to detect calibration object in any image. Thus the decision is to apply several threshold values from 0.5 to 0.8 in increments 0.05.

In other words, the entire sequence of the algorithm operations is performed for each threshold until calibration object will be detected or all values will be attempted. The computations are performed in the parallel mode.

The additional optimization that was introduced into the algorithm after primary testing is sorting of the thresholds according to their efficiency instead of values. The efficiency is a metric which describes the total amount of the detected calibration object from the testing set at the given threshold value. The efficiency of threshold values is presented in Table 2.

Thus the threshold values are applied in the following order: 0.65; 0.7; 0.55; 0.75; 0.6; 0.5; 0.8.

TABLE 1. Testing report.

coeff	recognized									
	img 1	**img 2**	**img 3**	**img 4**	**img 5**	**img 6**	**img 7**	**img 8**	**img 9**	**img 10**
0,50	false	true	false	false	false	false	true	true	true	true
0,55	false	true	false	false	false	false	true	true	true	true
0,60	false	true	false	false	true	false	true	false	true	true
0,65	false	true	false	false	true	false	true	false	true	false
0,70	false	true	false	false	true	false	true	false	true	false
0,75	false	false	false	true	true	false	true	false	true	false
0,80	false	false	false	false	true	false	false	false	true	false
coeff	**recognized**									
	img 11	**img 12**	**img 13**	**img 14**	**img 15**	**img 16**	**img 17**	**img 18**	**img 19**	**img 20**
0,50	false	true	false	false	true	false	false	true	true	true
0,55	false	true	true	false	true	false	true	true	true	true
0,60	false	true	true	true	true	false	false	false	true	true
0,65	false	true	true	true	true	true	true	false	true	true
0,70	true	true	true	true	true	true	true	false	true	false
0,75	true	false	true	true	true	true	true	false	true	false
0,80	true	false	false	false	false	true	false	false	true	false

TABLE 2. Threshold values efficiency.

Threshold value	The amount of the detected calibration objects		Total
	Autumn-season images	Winter-season images	
0,5	4	6	10
0,55	5	7	12
0,6	4	7	11
0,65	6	6	12
0,7	5	7	12
0,75	5	6	11
0,8	2	3	5

ALGORITHM FINAL TESTING

Figure 5 illustrates the output of the calibration object automatic detection module as a component of the software for the roundwood volume measurement.

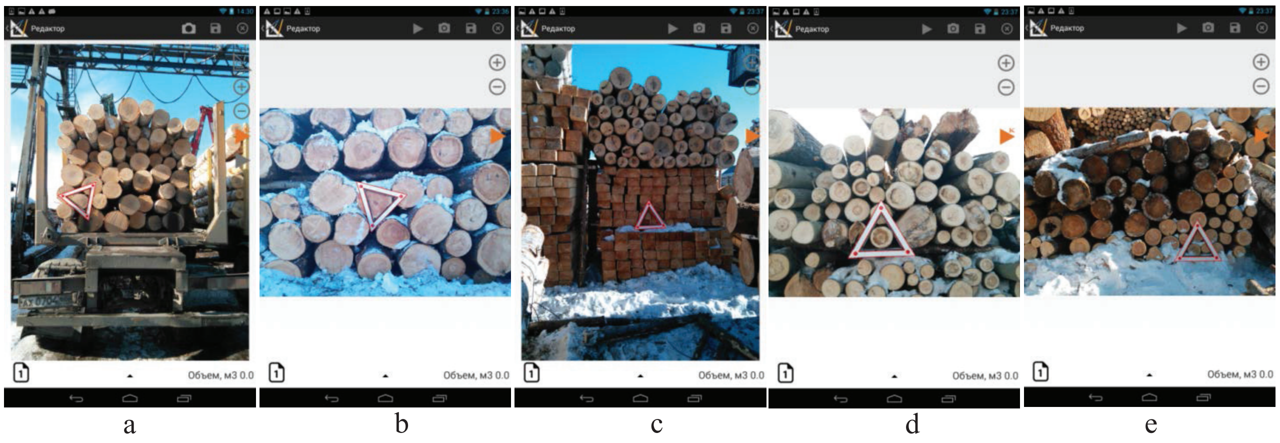

FIGURE 4. Successful operation results of the automatic calibration module.

After the algorithm modifications the final testing was carried out. Testing device is the tablet ASUS Nexus 7 with 4-cores NVIDIA Tegra 1200 MHz, 1Gb DDR2. Testing results are presented in Table 3.

TABLE 3. Algorithm testing results.

Parameter	Value
Minimum average processing time, ms	1,2
Maximum average processing time, ms	13
Probability of the type 2 error, %	86,1
Probability of the type 1 error, %	0

CONCLUSIONS

The algorithm final testing has pointed out the reasonably high efficiency of the calibration object isolation – 86.1%. Moreover, there are a substantial potential for further development and improvement of the automatic calibration module not only in performance but also in detection probability. The possible directions of the further researching are:

- Search of the optimal color combinations for the calibration object to provide its maximum contrast in the image.
- Implementation of multithread computing.
- Applying of the complex mathematical models of the image calibration.

At the moment the developed automatic calibration algorithm is implemented in the automatic image processing module of the “FoRest” mobile software for the log deck volume measurement via photogrammetry. This software is used successfully at the logging enterprises of the Ural Federal District.

REFERENCES

1. A. Maltcev, 2014. A few words on image recognition. URL: https://habrahabr.ru/post/208090.
2. Road signs recognition (opencv), 2009. URL: https://habrahabr.ru/post/61048/210.
3. W. Mokrzycki, M. Samko, 2012. Canny edge detection algorithm modification, Computer Vision and Graphics, 535-540.
4. D. Vaishnavi, T. S. Subashini, 2015. Image tamper detection based on edge image and chaotic arnold map, Indian Journal of Science and Technology, № 8 (6), pp. 548-555
5. OpenCV step by step, 2011. URL: http://robocraft.ru/page/opencv.
6. OpenCV, OpenCV Documentation, URL: http://docs.opencv.org/.
7. S. Suzuki, K. Abe, 1985. Topological Structural Analysis of Digitized Binary Images by Border Following. CVGIP conf. proceedings, vol. 30 of 1, pp. 32-46.
8. M. Romadi, R. Oulahhajthami, R. Romadi, R. Chiheb, 2014. Detection and recognition of road signs in a video stream based on the shape of the panels. In: 9th International Conference on Intelligent Systems - Theories and Applications (SITA), Morocco.
9. W. Chien-Chung, H. Jyun-Jie, 2013. The study of android parallel programming based on the dual-core cortex-A9. In: 9th International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), Beijing Univ Technol, Beijing, pp. 477-480.
10. S. Senthamilarasu, M. Hemalatha, 2013. A genetic algorithm based intuitionistic fuzzification technique for attribute selection, Indian Journal of Science and Technology № 6(4), p.4336-4346
11. P. Balaji, 2015. Programming Models for Parallel Computing (Scienti_c and Engineering Computation), The MIT Press.
12. A. Gorelic, U. Barabash, O. Krivosheev, S. Epshtein, 1990 Selection and detection on the basis of the location data, Radio and connection.