WROCLAW UNIVERSITY OF
SCIENCE AND TECHNOLOGY

Introduction to Computer Vision in Quality Control

Free parking slots recognition

TEAM MEMBERS:

Martyna Poślednik 209842

Adrian Strugała 209227

SUPERVISOR:

Dr. inż. Łukasz Jeleń

WROCLAW, 2017

Spis treści

1. Scope of the project 3
2. Solution 3
3. Implementation 4
4. Results 5
   Example 1 5
   Example 2 8
   Example 3 8
   Example 4 9
   Example 5 9
   Example 6 10
   Example 7 10
5. Results discussion 12
6. Conclusion 12

1. Scope of the project

We introduce the solution to find free parking slots from static images using computer vision based algorithms. In this project, we implement some of the suggested approaches and some improvised methods to classify the state of a parking space.

Project aims to focus on computer vision relating to monitoring quality control, hence the idea of the project is connected with such a helpful solution as finding free parking slots. Finding free parking slot can be irritating as well as might encourage drivers to left a car in improper way and even threatening other's safety or blocking fire escape. That's why solutions aiming to simplify either driving itself or parking are more and more desirable and popular.

A system that can automatically identify empty parking spaces and guide users to it will save a lot of time, money and effort of the driver and passengers and also will bring comfort and safety to users. Many solutions engaging sensors and Artificial Intelligence for this purpose have been suggested. Computer vision based techniques on the other hand have the potential to provide a cost-effective solution to this problem because they can use existing infrastructure such as security cameras to capture images which can be processed to classify the state of the parking space. Finding free parking slots using computer vision techniques is promising solution and its exemplary implementation is proposed in this project.

2. Solution

The proposed solution is obtained with a use of image preprocessing techniques like contouring, thresholding, negative. Then Hough Transform is used as a clue of obtaining rectangles on the image. Hough Transform is described with equation representing a single line:

$$r = x\cos\theta + y\,\sin\theta.$$

If the values of x and y are known it's possible to calculate all the values of r by going through all possible values of θ. Plotting these values of r on a graph for every value of θ a sinusoidal curve is obtained. The Hough Transform works by looking at a number of such x,y coordinates, which are usually found by some kind of edge detection. Each of these coordinates is transformed into an r, theta curve. This curve is discretised so we actually only look at a certain discrete number of theta values. The accumulator space is plotted rectangularly with theta on one axis and r on the other. Each point in the array represents an (r, theta) value which can be used to represent a line using the formula above. Once all the points have been added should be full of curves. The algorithm then searches for local peaks in the array. The higher the peak the more values of x and y crossed along that curve, so high peaks give good indications of a line.

3. Implementation

The solution was designed and further implemented in Java environment. The application was built with a use of WindowBuilder with interactive menu to choose its functionalities. The functions of the program are image processing functions. Few of them is not correlated directly to the scope of project but provides useful additional functions.

After loading the desired image, the program can:

1) prepare the image before Hough Transform (contour, threshold, negative)
2) apply Hough Transform
3) find points to be connected to find a rectangle using Hough Transform
4) mark the rectangles of all parking slots on the picture
5) find the empty place on the picture

The main window of the program is shown in the following figure (Figure 1).

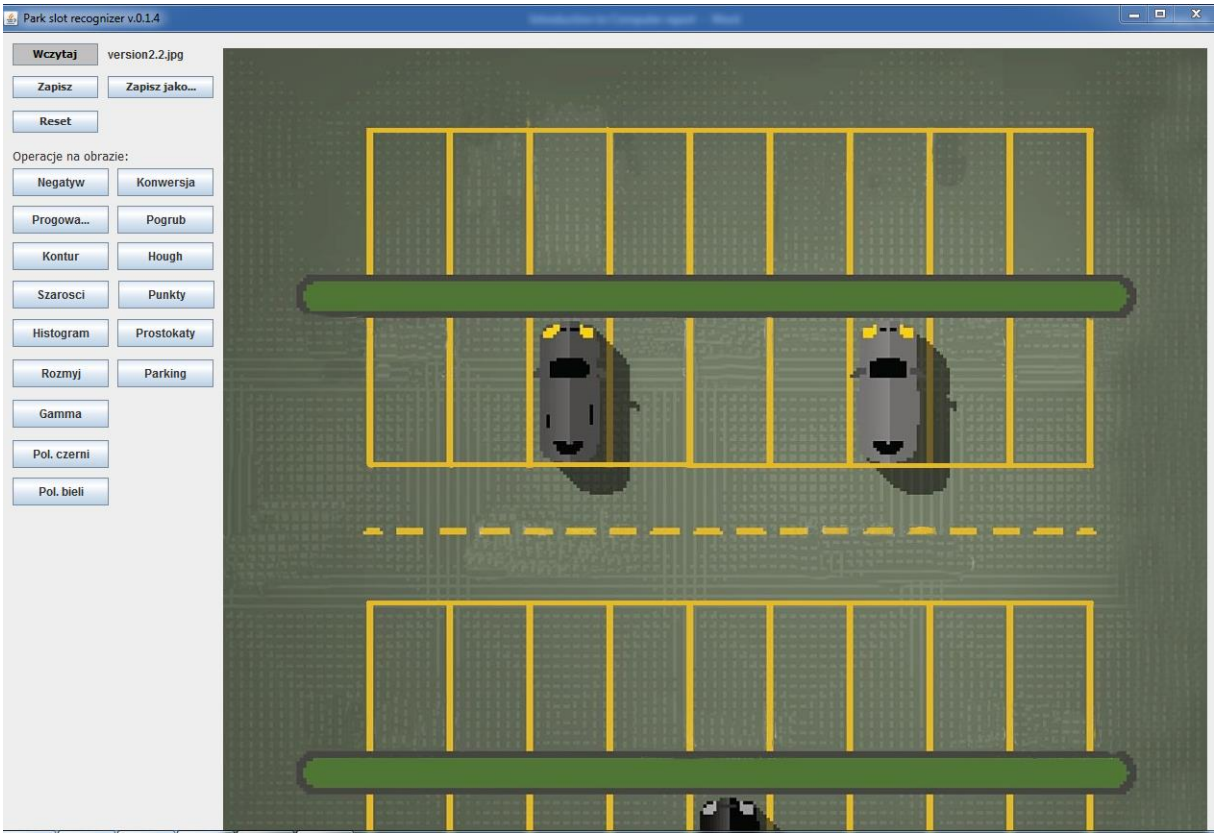

Figure 1 The main view of the program after loading the image.

4. Results

The program was tested on few different parking pictures to show how it works. The classification of parking places is not yet perfect, but obtained result definitely in major cases it is close to what was expected. The following section presents some of examples of results our algorithm obtained. Table (Table 1) presents numerical results on ten tested cases.

Example 1

Figure 2 Step 1: image prepared for Hough Transform.

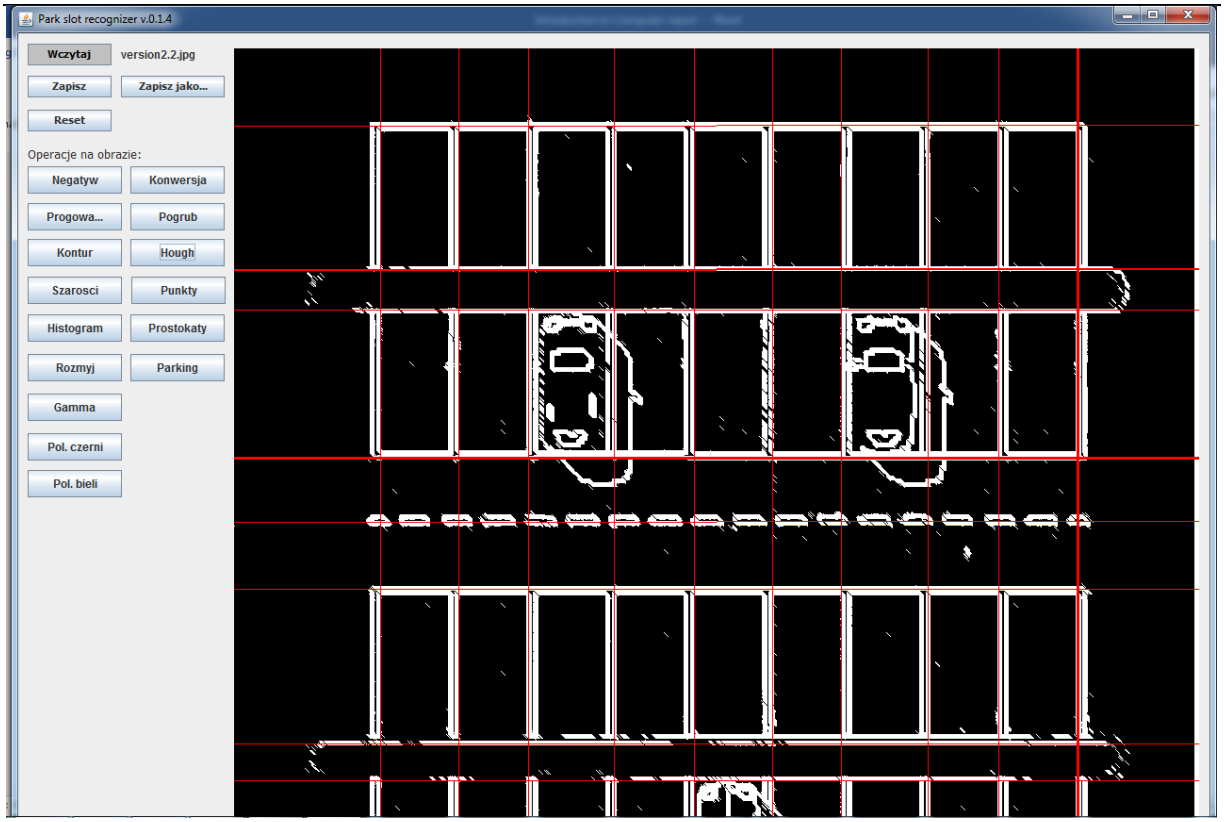

Figure 3 Step 2: Hough Transform lines applied on the image.

Figure 4 Step 3: Crossing points (green) found to lead lines on the image.

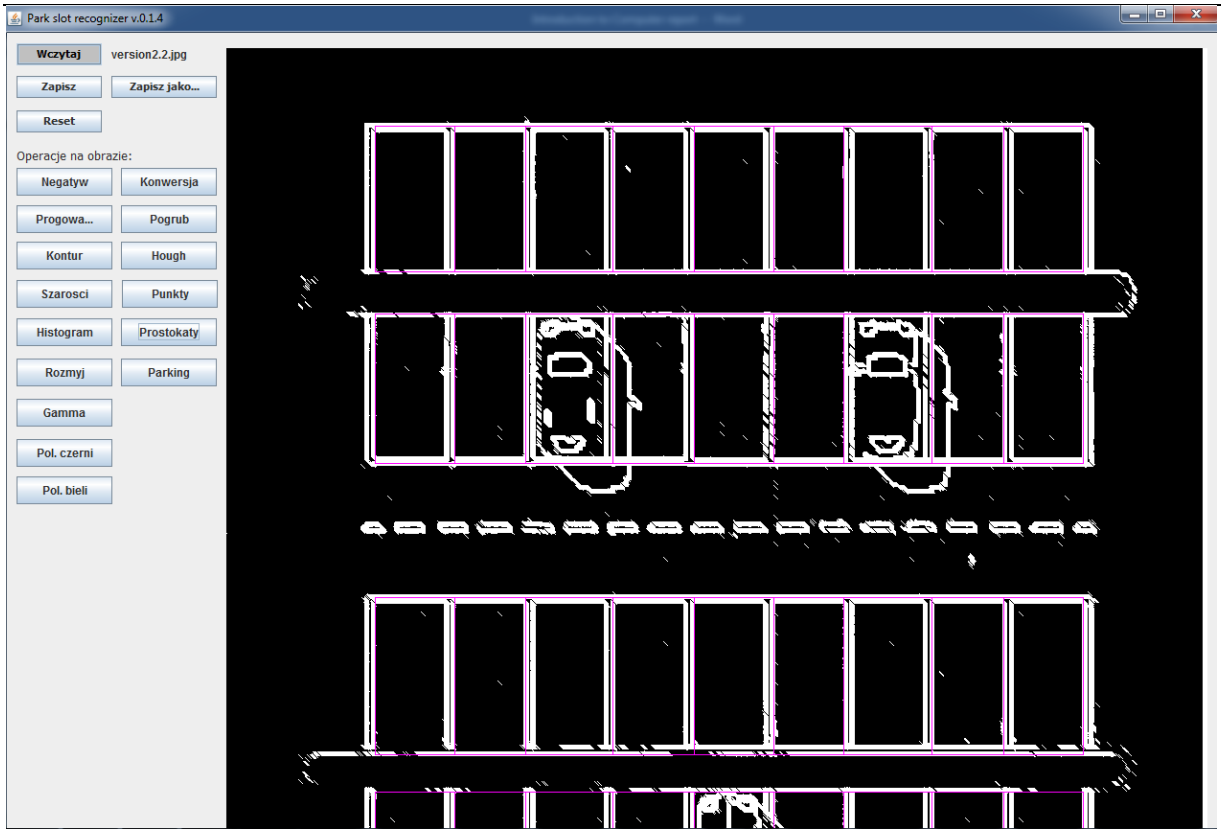

Figure 5 Step 4: Rectangles (purple) found on the image.

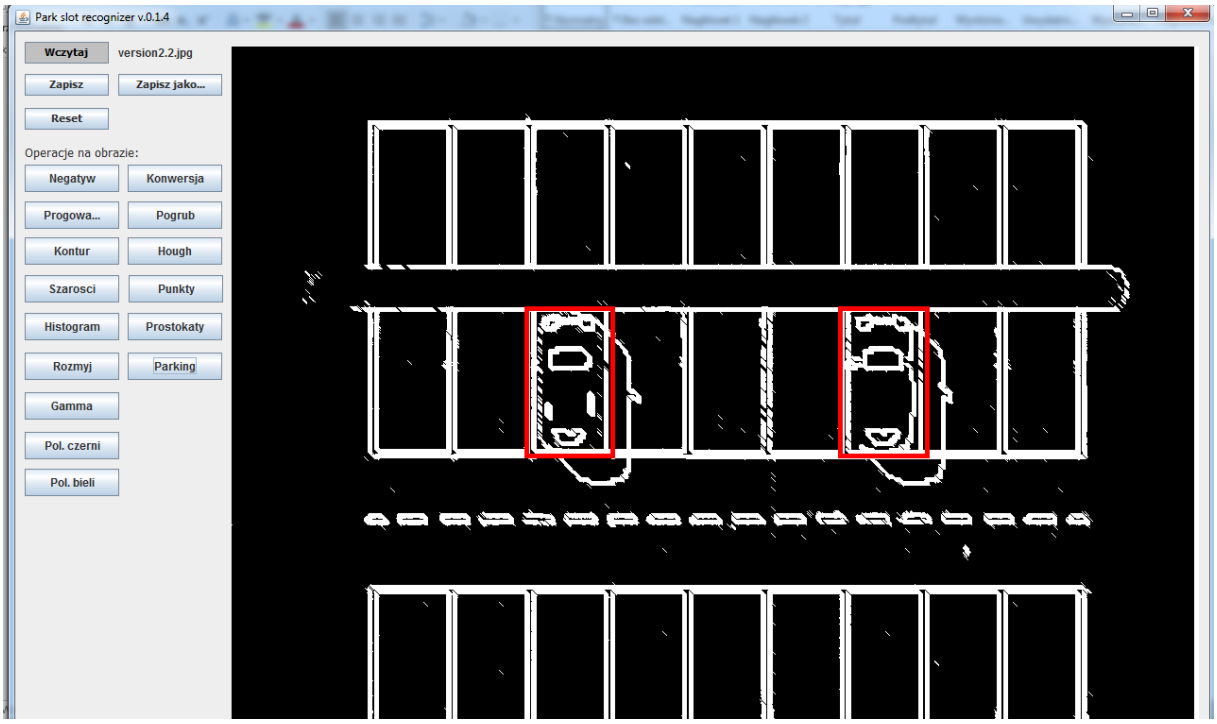

Figure 6 Step 5: Occupied places (red) found.

Example 2

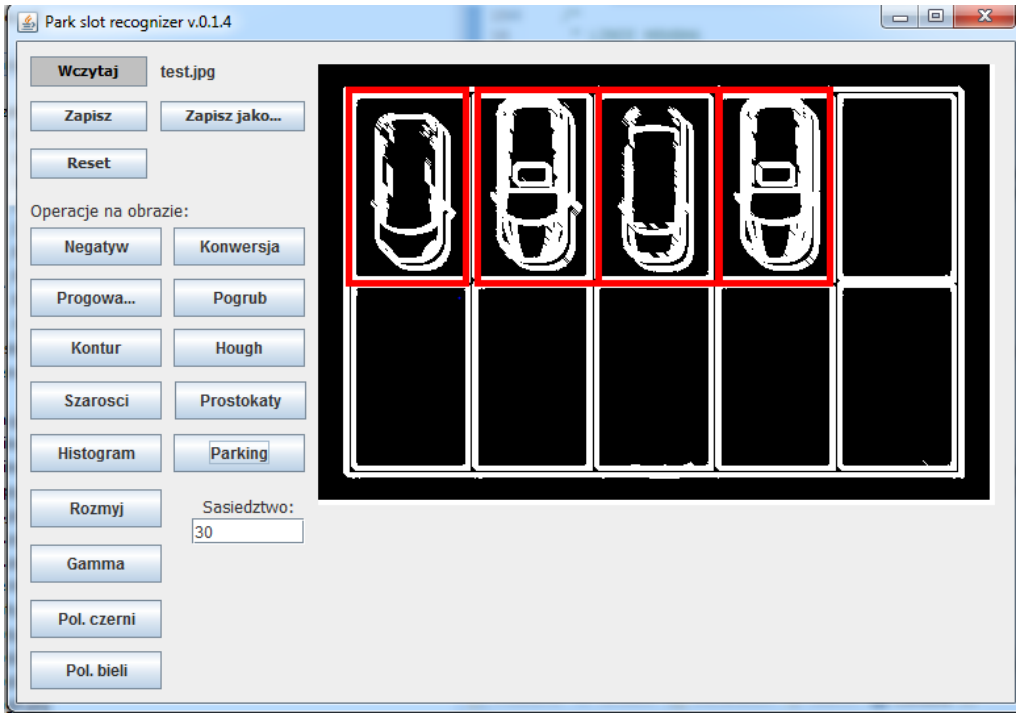

Figure 7 Example of successfully found occupied parking space.

Example 3

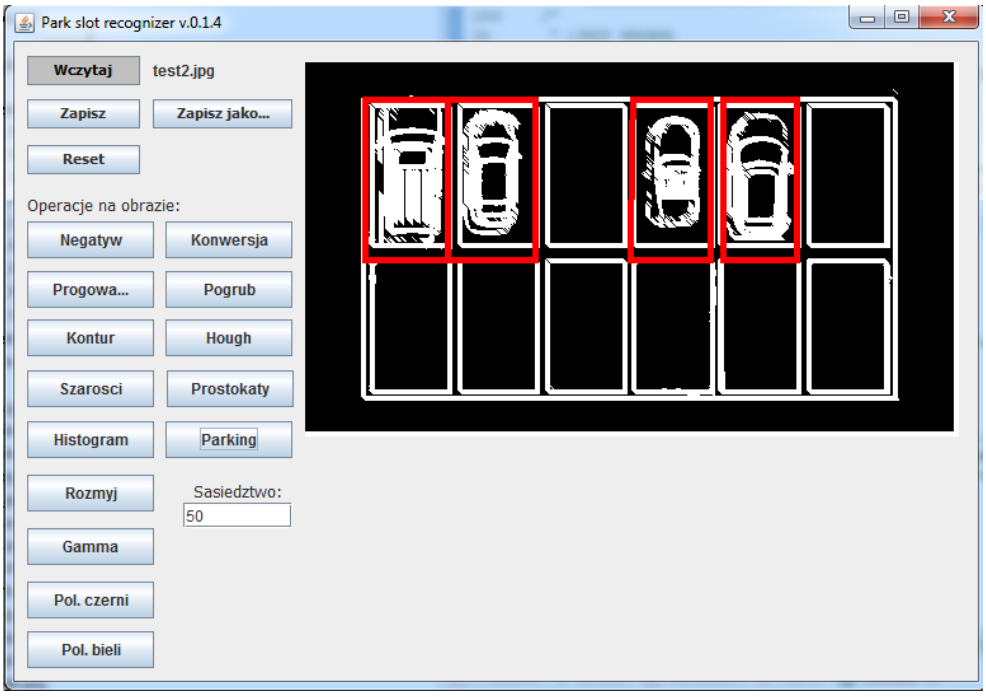

Figure 8 Another example of successfully found occupied places on different set.

Example 4

Of course, the algorithm isn't perfect and there are still cases that recognition doesn't go the right way. Example on figure 9 shows that some places were found, but some of them were missed also some places were classified as occupied while they weren't. However, in many tested cases it was satisfactory.

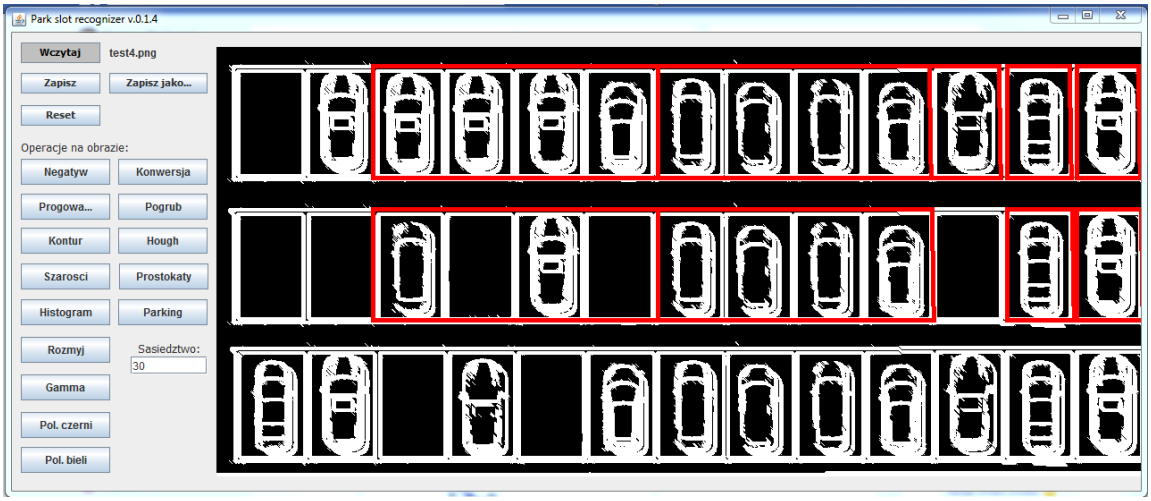

Figure 9 Misleading effect of program.

Sometimes the effect of wrong classification is caused by some important issues with the testing image. In this case it was observed that Hough transform didn't catch the last line of the parking so it misled it there is no parking there.

Example 5

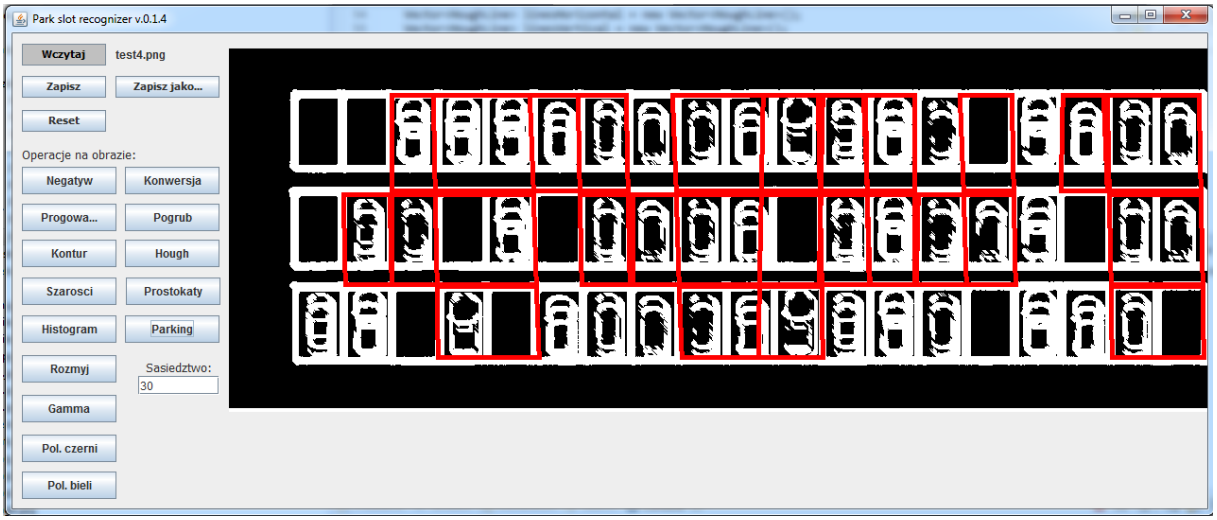

Figure 10 Another example of mistakes.

Example 6

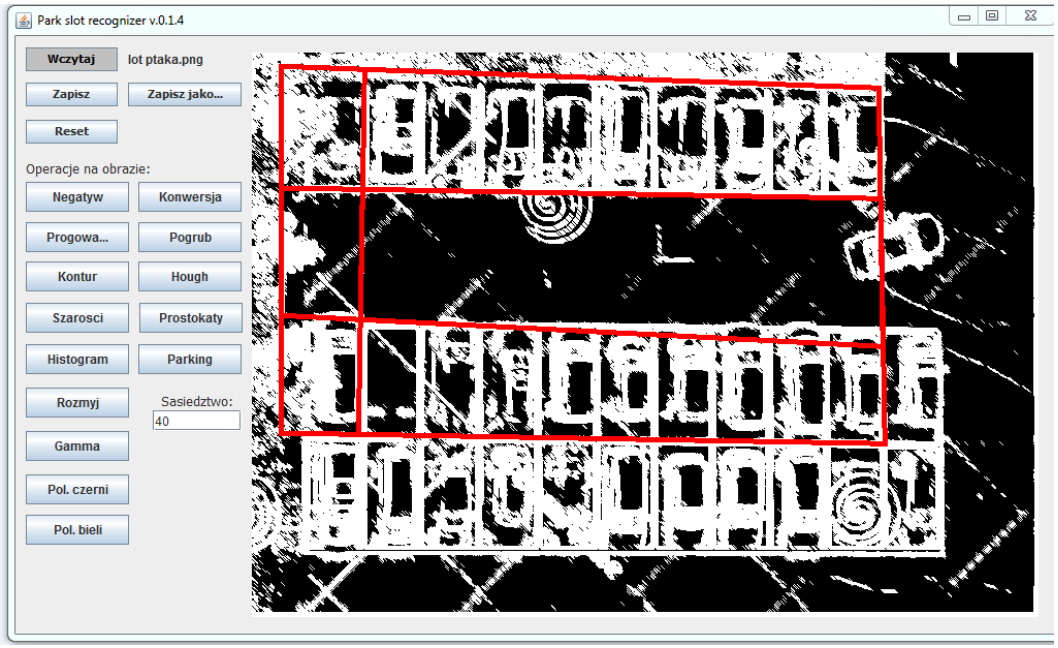

Figure 11

Example 7

Sometimes there are some additional horizontal lines, however the result is satisfactory. The car that parked on two places brought additional lines which algorithm had to manage with.

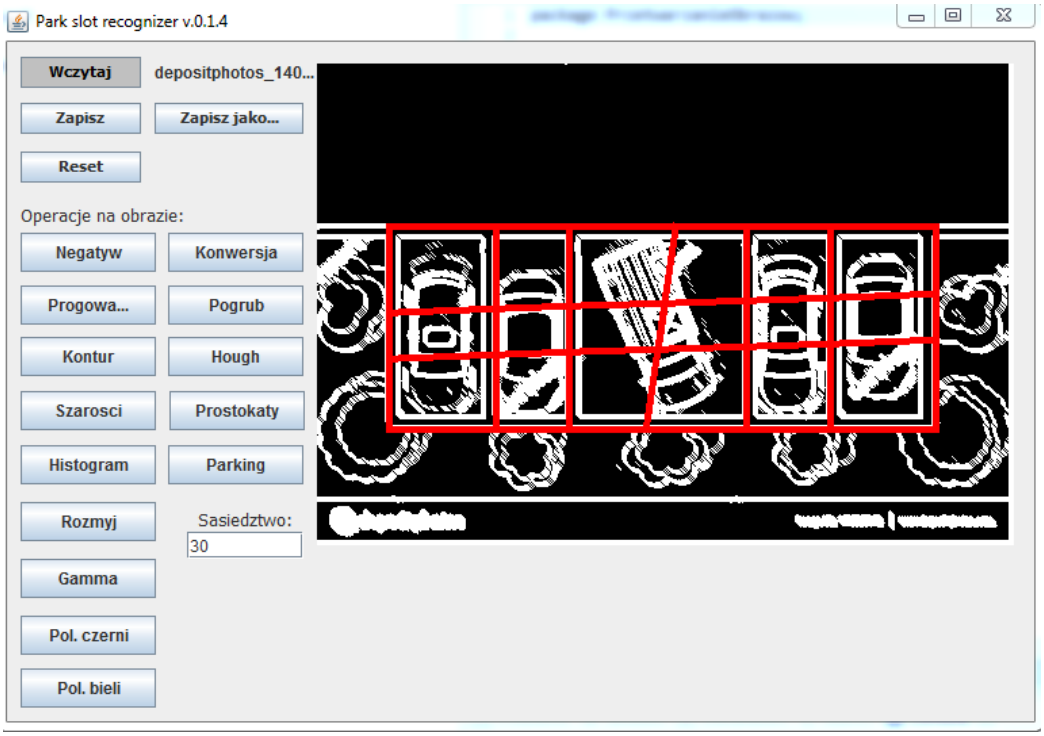

Figure 12

Table 1 Results of the tests.

	Image name	All	Occupied	True Positive	False Positive
1.	paking_yellow_lines.jpg	36	3	3	0
2.	aerial-view.jpg	33	32	19	1
3.	three_rows.jpg	15	9	1	8
4.	test2.jpg	12	4	4	0
5.	car-park-clipart.jpg	10	10	4	6
6.	lines_in_between.jpg	22	13	2	11
7.	123rf.jpg	36	36	36	*found places in the background of the car park
8.	parking2.jpg (with stock sign)	36	3	0	0
9.	car-on-two-places.jpg	6	6	6	* Additional horizontal lines
10.	paking_yellow_lines(2).jpg	36	6	3	3
11.	lines_in_between(2).jpg	22	13	1	12
12.	thee_rows(2).png	15	9	0	9
13.	aerial-view(2).jpg	22	21	10	11
14.	test2(2).jpg	12	1	1	11
15.	car-park-clipart(1).jpg	10	7	6	1
16.	123rf(2).jpg	36	18	6	12
17.	car-on-two-places(2).jpg	6	2	1	5
18.	empty.jpg	6	0	0	6
19.	version2.jpg	36	3	3	0
20.	test3.jpg	6	5	3	2

5. Results discussion

The table (Table 1) presents results of the test. Tests of the program were taken on 20 different images of parking lot. The table presents numeric information about: the number of parking places, number of occupied places on the parking and two important factors: true positive and false positive. True positive denotes the number of parking places correctly classified by the algorithm as occupied. It is desired that this number is equal to number of occupied places. Usually, it was higher than 50% of occupied places which is still satisfactory. Another factor is false positive, which indicates the number of places that weren't find by algorithm even if they were occupied.

If the classification is effective, we can get known from the ratio $\frac{occupied}{true\ positive}$. In most cases it was above 50%, so the algorithm for this test cases worked good. The best classification was for the image which we used to model the solution "paking_yellow_lines.jpg". The algorithm finds all three occupied places and no other additional. In other cases, it is sometimes mistaken however in most cases the ratio $\frac{occupied}{true\ positive}$ was above 50%, so the algorithm for this test cases worked good. Other fully successfully classified images were e.g. no 4, 9,14,19 but in this case, it is hard to say if other pictures' quality was just worse or maybe there was some noise on the picture etc. By now, the pictures taken to analyze must be well prepared, because the program does just a simple preprocessing. In further steps, it would possibly take any picture and successfully find the place.

6. Conclusion

To summarize, by now the project obtained results as expected, however it still needs tests and improvements. More various test cases should be taken into consideration (varying sizes of parking lots, different orientation of the image etc.). The desired tests should reach satisfactory level. This project can surely be helpful in quality control, for example managing parking lots like the one on Wroclaw University of Technology. Preliminary results indicate that the proposed method can successfully detect rectangles showing good performance when applied to images. Future work might concentrate on making the solution more flexible and reduce false detection which wasn't so much of but still.