Pothole Detection and Tracking in Car Video Sequence

Pothole Detection and Tracking in Car Video Sequence

Ionut Schiopu*, Jukka P. Saarinen†, Lauri Kettunen‡, and Ioan Tabus*
*Department of Signal Processing, Tampere University of Technology, Tampere, FINLAND
†Nokia Technologies, Tampere, FINLAND
‡Department of Electrical Engineering, Tampere University of Technology, Tampere, FINLAND

***Abstract*—In this paper, we propose a low complexity method for detection and tracking of potholes in video sequences taken by a camera placed inside a moving car. The region of interest for the detection of the potholes is selected as the image area where the road is observed with the highest resolution. A threshold-based algorithm generates a set of candidate regions. For each region the following features are extracted: its size, the regularity of the intensity surface, contrast with respect to background model, and the region's contour length and shape. The candidate regions are labeled as putative potholes by a decision tree according to these features, eliminating the false positives due to shadows of wayside objects. The putative potholes that are successfully tracked in consecutive frames are finally declared potholes. Experimental results with real video sequences show a good detection precision.**

***Keywords*—Pothole detection; pothole tracking; reflection in windshield; region of interest; Euclidean distance mapping.**

I. Introduction

In the developed countries the infrastructure of the complex roadway network needs prompt maintenance. Pavement degradation such as potholes and cracks appear as a result of climate conditions (e.g., water lying on the road, temperature changes, icing) or weight pressure of heavy vehicles.

There are many existing approaches for pothole detection. In the *3D laser scan approach* [1], the system is using an LED linear light, which rays vertically on the road surface, and the 3D projection transform for image analysis. In the *reconstruction based approach*, in [2] a complex system is using a Kinect sensor and a camera, while in [3] a stereo vision method is using a surface fitting algorithm to estimate the road plane in the disparity map and applies thresholding for detection. In the *vision based approach*, the image segmentation is first generated using a histogram based thresholding method; in [4] the pothole detection is based on spectral clustering; in [5] a pothole shape approximation method is proposed, and the pothole detection is done using the texture comparison of the areas inside and outside the pothole, the same method was improved in [6] and the video tracking feature was added; the method in [7] is using histogram shape based thresholding, candidate region extraction, and ordered histogram intersection. In the *vibration based approach*, accelerometers are used for pothole detection [8]; in [9] a smart-phone application was developed for processing the sensor's output.

The research was supported by the Data to Intelligence (D2I) research program funded by Tekes and a consortium of companies. The test video sequence was collected by Prof. Lauri Kettunen.

We introduce a pothole detection approach, applicable to crowd-sourcing road conditions, consisting in: a threshold algorithm to generate a set of candidate regions, a pothole detection algorithm based on region analysis, and a pothole tracking algorithm.

The paper is organized as follows: in Section II we propose the pothole detection algorithm, in Section III we describe the pothole tracking algorithm, in Section IV we discuss the experimental results, in Section V we draw the conclusions.

II. Pothole detection

The proposed method for pothole detection is based on the idea that the potholes are represented in the intensity image using high values. It starts by using a threshold-based algorithm to generate a set of candidate regions by selecting areas of the image containing pixels with high intensity values. Each candidate region is analyzed and the regions containing potholes are distinguished from the regions containing object shadows (also represented with high intensity values) by checking the following region properties: pixel size, regularity, estimated depth, contour length and shape; and the appearance of a pothole in consecutive frames.

Next we discuss: the selection of the area inside the image where the potholes are best visible; the generation of the set of candidate regions; the detection and removal of the reflections of the objects found inside the car; the detection and removal of the regions containing shadows; and the labeling rules for setting the 'pothole' label to a candidate region.

A. Selecting the Region of Interest

Let us denote F the current video frame of size $n_r \times n_c$. The algorithm is searching for potholes in the area of the image representing the road that lies ahead of the moving car. This area is denoted the *region of interest* (ROI), and is selected by an off-line procedure using one selected frame (containing a straight road) based on the following facts: *(a)* in an image, two parallel lines marked on the road are intersecting in a point called *vanishing point* and denoted V; *(b)* the best area to search for potholes is the area starting from just above the car hood and ending at a distance in front of the car where the smallest potholes are still visible. In Fig. 1, V was found using

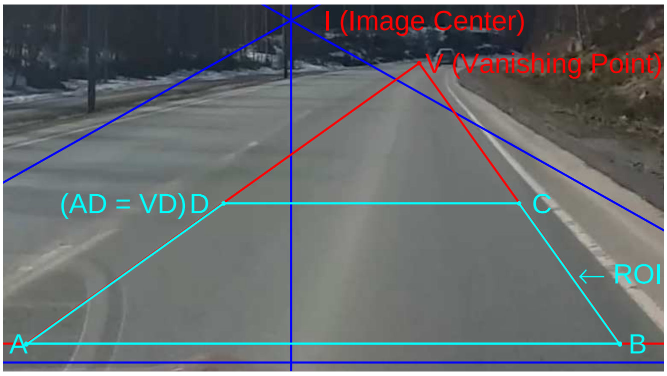

Fig. 1. The selection of the *region of interest* (ROI). The boundary pixels of ROI are marked with cyan and are forming the trapezium $ABCD$. $ABCD$ is obtained using the vanishing point, V, and the row index of the maximum position of the car hood, which is marked with a horizontal blue line. The image center, I, is at the intersection of the two diagonals (marked with blue).

the intersection of the road markings, while the segment AB is set at a distance of $2\% n_r$ above the car hood line (marked with blue). The middle point of the segment VA is denoted D, and of the segment VB is denoted C. ROI is selected as the area inside the trapezium $ABCD$.

B. Generating candidate regions

Although ROI selects an area above the car hood where the potholes should be visible, sometimes the car gets too close to the wayside and the objects outside the road are included in ROI. Here, a threshold algorithm is used to remove from ROI the pixels representing the wayside by computing the intensity image Y of the current frame F and by searching for the pixels inside ROI with the intensity value lower than a threshold. The threshold is denoted T and is set as

$$T = \max(90, \bar{y} + \sigma_Y), \tag{1}$$

where $\bar{y} = \sum_{k=1}^{N_{ROI}} Y(x_k, y_k)$, σ_Y is the standard deviation of the N_{ROI} intensity values $\{Y(x_k, y_k)\}_{k=1,2,\ldots,N_{ROI}}$ of ROI. Let us denote $Mask_{ROI}$ the pixel positions for which $Y(x_k, y_k) < T$. $Mask_{ROI}$ selects from Y the pixel positions used to estimate the plane of the road, denoted $P(x, y)$. We denote $D(x, y)$, the depth matrix containing the differences between the intensity values, $Y(x, y)$, and the heights in the road plane, $P(x, y)$, at each pixel position $(x, y) \in Mask_{ROI}$, i.e. $D(x, y) = Y(x, y) - P(x, y)$ measures the distances between the road plane and each pixel's intensity value.

The method selects from D the darkest pixels, i.e. those having the depth $D(x, y)$ smaller than $minDepth = -15$, and collects them in a *pool of pixel positions*. Each candidate region is created from the pool of pixel positions by collecting neighboring pixels and by checking if the region is big enough to contain a pothole, i.e. by constraining it to contain at least $minSize = 100$ pixels.

C. Detecting and removing object reflections

In the bottom-left of Fig. 1, one can notice that the frame is corrupted with the reflexions in the windshield of the objects placed inside the car. The areas corresponding to object reflections are removed from ROI by applying an off-line procedure to a set of consecutive frames. The set was selected so that the frames contain a straight road and do not contain any shadows or potholes, and hence only the object reflections are visible.

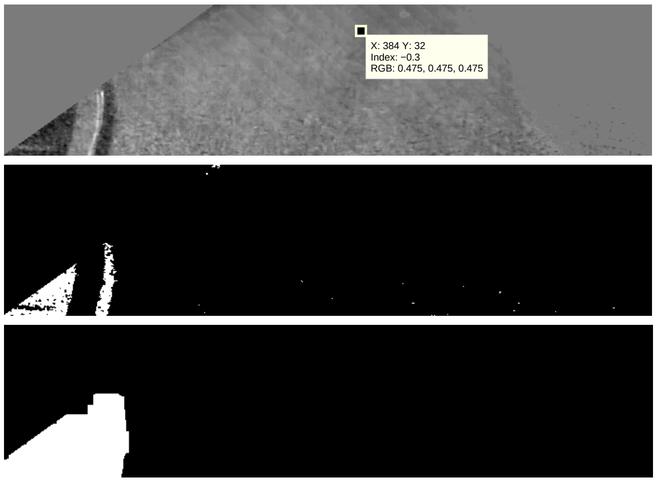

Fig. 2. *(Top)* The depth matrix D_M computed for ROI. *(Middle)* The pool of pixel positions selected from D_M is marked with white. *(Bottom)* The removed area containing object reflections is marked with white.

Fig. 3. A light-pole shadow detected in ROI. The boundary pixels are marking with: (cyan) ROI, (red) the light-pole shadow, (black) object reflexions.

The algorithm first computes the mean intensity image Y_M and the depth matrix D_M, and collects the positions for which $D_M(x, y) < 3\sigma_D$ in a *pool of pixel positions*, where σ_D is the standard deviation of the 'depth' values $D_M(x, y)$. The candidate regions are generated using $minSize = 100$. The reflection's positions depends on the position of the car regarding the Sun, therefore a few neighboring pixels are included in the regions. The found regions, now containing object reflections, are removed from ROI, see Figs. 2 and 3.

D. Shadow detection

The objects placed next to the road (e.g., traffic signs, trees, buildings) or other passing cars are creating shadows which may possibly be mistakenly labeled as potholes. Fig. 3 shows a correct detection of a light-pole shadow. Our method distinguishes the potholes from the shadows by checking the following region properties:

(a) *Region's model.* An object shadow has a regular shape, while a pothole has an irregular shape. Based on this idea, an

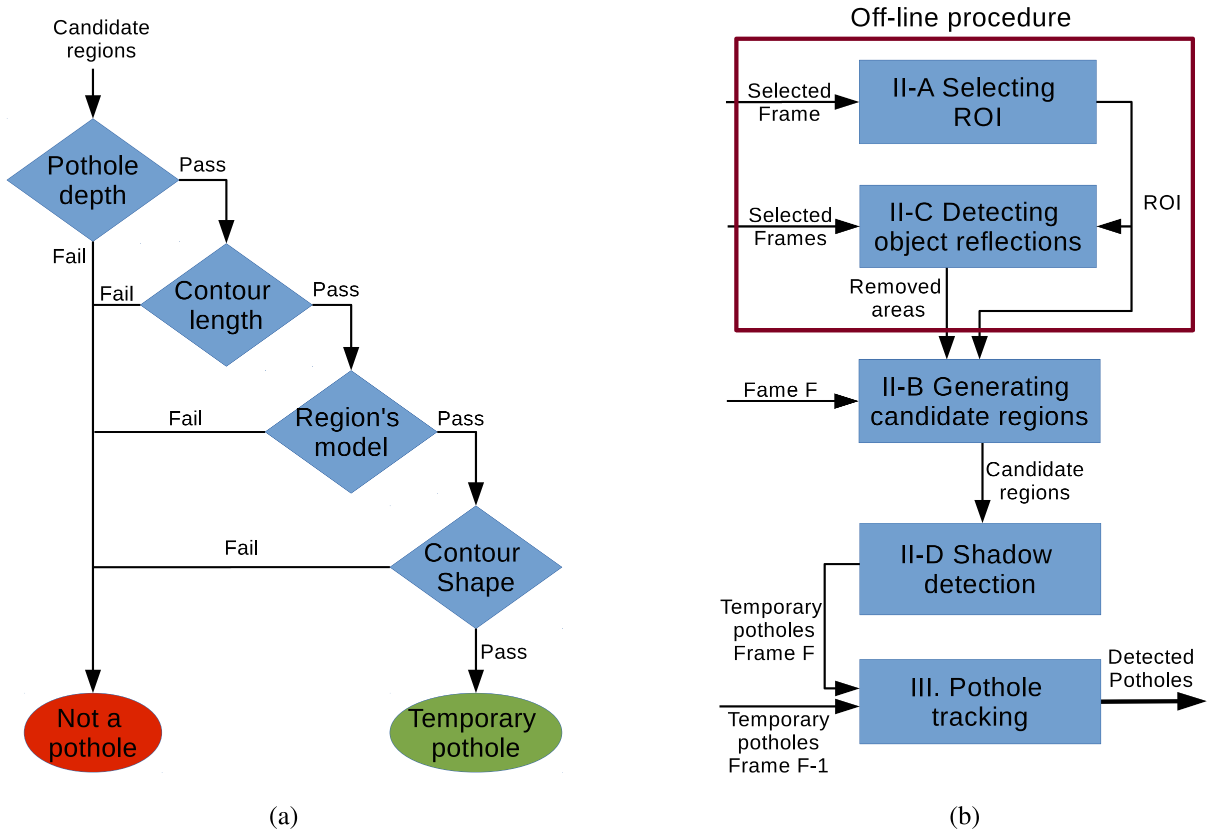

Fig. 4. *(a)* The decision diagram for the shadow detection step introduced in Section II-D. *(b)* The block diagram of the proposed algorithm.

estimation of an order-2 model is computed for each region

$$M_2(x, y) = ax^2 + by^2 + cxy + dx + ey + f, \quad (2)$$

where (x, y) is the pixel position in the candidate region and $M_2(x, y)$ is the value of the order-2 model $\theta_2 = [a\ b\ c\ d\ e\ f]^T$. The model is evaluated using MSE and the region is labeled as shadow if $MSE < minMSE$, where $minMSE = 50\ dB$ is the minimum MSE value accepted for a pothole region.

(b) *Pothole depth*. A region should be deep enough to be labeled pothole, i.e. the region must contain dark enough pixels. Since the road may present small cracks, the regions are constrained to have the minimum average depth smaller than $minAvgDepth = \min(-25, -1.75\sigma_Y)$.

(c) *Contour length*. The small cracks are further removed by checking if the number of boundary pixel is less than $\frac{2}{3}$ of the total number of pixels in the candidate region.

(d) *Contour shape*. The contour of a shadow is straighter (more regularly) than the contour of a pothole. The contour of each candidate region is evaluated using the Three-OrThogonal (3OT) representation [10] by computing the percentage of symbols: 0 ('go forward'), 1 ('turn left/right'), and 2 ('go back'), denoted p_0, p_1, and p_2, respectively. The imposed criterion has two parts: *(i)* $p_2 > 8.5\%$, i.e. the pothole's contour should have a lot of left/right turns; *(ii)* $p_0 < 60\%$, i.e. the pothole's contour should have only a few straight lines.

Fig. 4.(a) shows the used decision scheme, where the order in which the region properties are checked was decided based to the computational complexity. If a candidate region successfully checks a property (the "Pass" branch), the next property in order is checked, else the candidate region can not be a pothole and is removed from the tests (the "Fail" branch).

E. Labeling the pothole regions

The remaining candidate regions which passed the criteria in Section II-D, are labeled as temporary pothole regions. The temporal attribute is removed if the pothole is detected again (at least) in the next frame. There are cases when only a part of a shadow is selected by ROI and the 'incomplete' shadow region may pass the criteria introduced in Section II-D, therefore the shadow region can be easily mistaken as a pothole region. This type of regions can be found at the margins of ROI, especially in the bottom-right part.

Fig. 4.(b) shows the block diagram of the proposed algorithm, where the regions labeled as temporary potholes after the shadow detection stage are tracked in consecutive frames using the algorithm presented in the next section.

III. Pothole Tracking

In video tracking, a moving object is located in different frames. In our case, we are tracking the pothole's position in consecutive frames as long as it is visible in ROI.

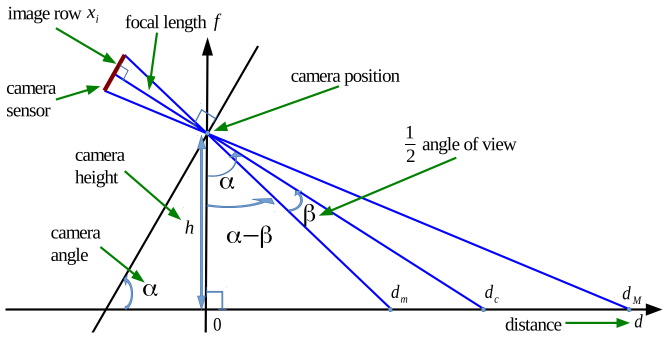

Fig. 5. Camera geometry. The projection of the area of the road, between the distances x_m (minimum) and x_M (maximum) from the car, to the camera sensor. The camera is placed at height h and at an angle α.

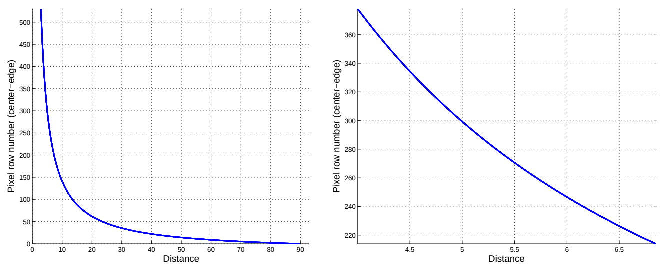

Fig. 6. *(Left)* The mapping of the distance x (from Fig. 4) to the pixel row number counted from the image center to the image edge. *(Right)* The mapping of the ROI's row number.

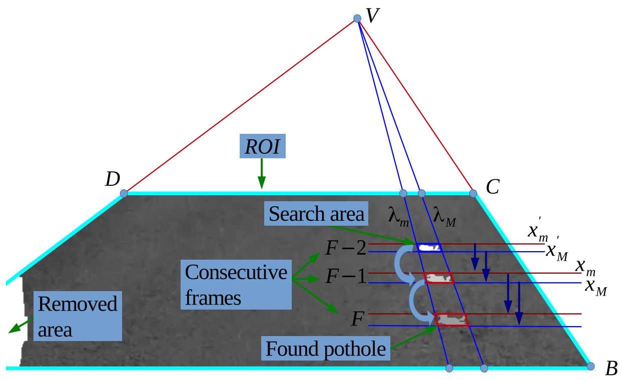

Fig. 7. Pothole tracking in consecutive frames. The pothole's shape from the previous frames, $F-2$ and $F-1$, are overlapped with the current frame F.

After its detection in a frame, a pothole can be traced using $d_2 = d_1 - st$, where d_1 is the pothole's first position (distance from the pothole to the car), s is the speed of the car, and $t = \frac{1}{30}\ sec$ for 30 frames per second rate. If the speed is known, then the previous or future positions (d_2) can be computed. However, the speed is usually not known and it is estimated using the pothole's detection in two consecutive frames.

The pothole tracking algorithm is mapping the distance visible in the image between the car hood and the vanishing point to the image row index; more exactly the distance visible in ROI. Fig. 5 shows the camera geometry based on the following notations: h is the height of the camera; α is the angle of the camera with a horizontal line; 2β is the camera angle of view; f is the focal length; $\mathrm{d}_v \times \mathrm{d}_h$ is camera sensor size; $p_v = \frac{\mathrm{d}_v}{n_r}$ is the vertical pixel size; and $\rho = \frac{\mathrm{d}_v}{n_r f}$ is the pixel density. Let us consider that an object is placed at the distance d_i from the camera, the camera captures the object at an angle of view β_i, and the object is represented in the image at the row number x_i. Using Fig. 5 we can write the following equations:

$$\tan(\alpha - \beta_i) = \frac{d_i}{h}, \tag{3}$$

$$\tan \beta_i = \frac{x_i \frac{\mathrm{d}_v}{n_r}}{f} = x_i \rho. \tag{4}$$

Let us denote $a = \tan\alpha$, change the variables as: $d' = \frac{d_i}{h}$, $x' = x_i\rho$, and use (3) to rewrite (4) as:

$$d' + ad'x' + x' - a = 0. \tag{5}$$

Fig. 6 shows the mapping of the *Euclidean distance* to the image's *row index*, using the camera specifications of the front camera of the *Samsung S4* smart-phone. In the experimental setup, the camera was placed at height $h = 1.23\,m$ (which varies and the precision is about $1\,cm$) and at camera angle $\alpha = 89.21°$. The mapping *distance* $\leftrightarrow$ *row index* ($d \leftrightarrow x$), with the mathematical form (5), is used by the pothole tracking algorithm to find the pothole's search area in a different frame.

Fig. 7 shows an example of a pothole detected in the current frame F and in the previous frame $F-1$. Using this two consecutive detections, the car speed is estimated and the pothole is tracked in the previous frame $F-2$ or in the next frame $F+1$, see Figs. 7 and 10. The bounding box of the pothole detected in $F-1$ is defined by the following elements: *(a)* the minim, x_m, and the maximum, x_M, row index of the pothole region; *(b)* the minim slope, λ_m, and maximum slope, λ_M, of the two lines traversing V and the boundary pixels of the pothole region. Let us consider that the car is moving in a straight line, and define the pothole's search area in frame $F-2$ using: *(a)* the two lines with the slopes λ_m and λ_M; *(b)* the estimated position of the minimum row index, x_m', and maximum row index x_M' of the pothole region.

The minimum row index x_m' is computed as follows: *(i)* compute the distance d_m corresponding to the row index x_m using (5); *(ii)* subtract from (add to) d_m the distance traveled by the car and estimate the previous (next) position d_m'; *(iii)* find the row index x_m' corresponding to d_m' using (5). A similar procedure is used to estimate x_M'.

IV. Experimental Results

The experiments were carried out using the front camera of the *Samsung S4* smart-phone placed inside a car, while driving in Finland on the route: *Hervanta*, road 309, European route $E12$, *Kalvola*. The video sequence contains 34 minutes of footage, at a full HD resolution 1080×1920 and a frame-rate of 30 frames per second (see Table I for more camera specifications), on dry condition and a sunny sky.

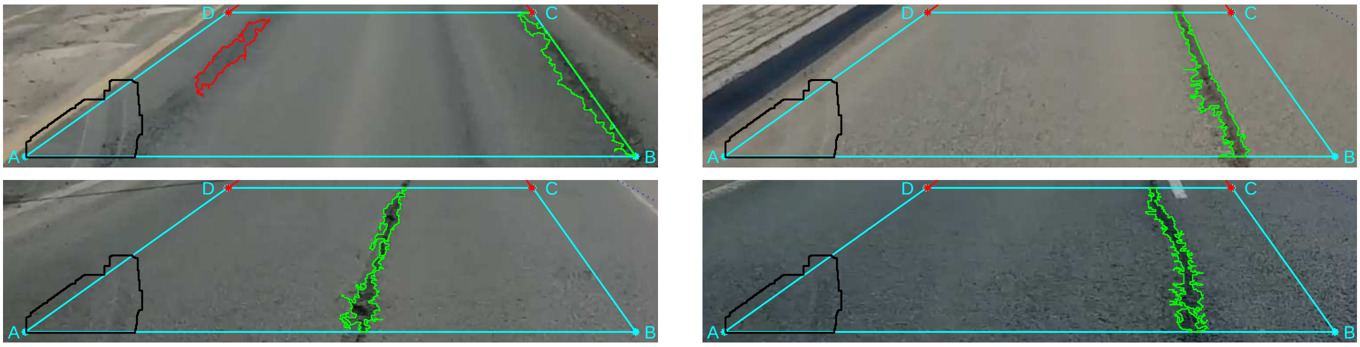

Fig. 8. Examples of detected potholes. The boundary pixels are marking with: (green) potholes, (red) shadows, (cyan) ROI, (black) object reflexions.

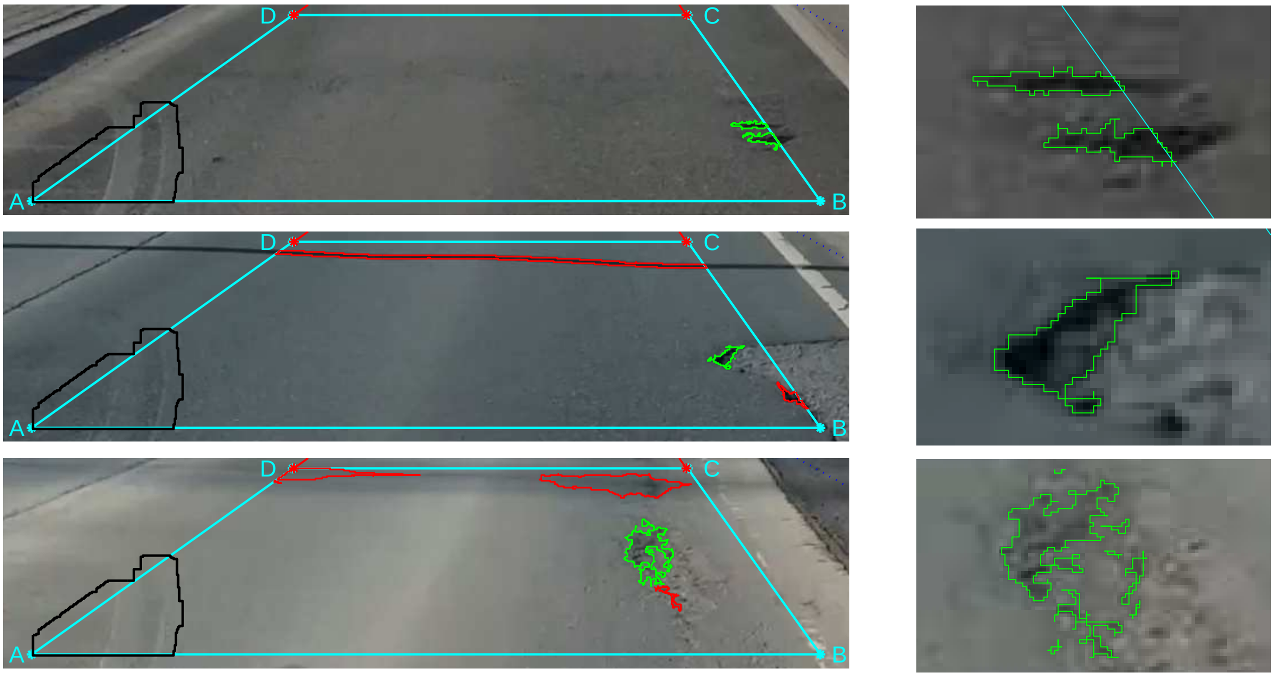

Fig. 9. *(Left column)* Examples of detected potholes. *(Right column)* Zoom in at the detected pothole. The boundary pixels are marking with: (green) the detected potholes, (red) the shadows, (cyan) the ROI, (black) the objects reflexions.

TABLE I. SAMSUNG GALAXY S4 PRIMARY CAMERA

Camera specification	**Value**
Sensor model	Sony IMX135 Exmor Rs
Sensor type	CMOS
Sensor size	4.69×3.52 mm
Pixel size	≈ 1.136 μm
Crop factor	≈ 7.38
Aperture	$f/2.2$
Shutter speed	$1/14 - 1/10000$
Focal length	≈ 4.2 mm

The algorithm[1] is implemented in MATLAB and it is processing off-line the test video sequence. In the first stage of the method, the *region of interest* of every frame in the sequence is analyzed and the list of detected potholes is reported using each pothole corresponding sequence of frame numbers, see Figs. 8 and 9. In the second stage, every pothole is tracked in the previous frames or in the next frames and their corresponding frame sequence is updated, see Fig. 10.

Fig. 8 shows four examples of detected potholes having an elongated shape, which might be sometimes refer to as ‘big cracks’. The boundary pixels of the area from the image containing object reflexions are marked with black. The boundary pixels of the detected potholes are marked with green and of the detected shadows are marked with red.

Fig. 9 shows three examples of detected potholes, together with a zoom in at the detected pothole. In this case, the ellipsoidal shape of the pothole can be easily noticed.

Fig. 10 top row shows two examples of pothole tracking: *(left)* tracking in the next frames; *(right)* tracking in the previous frames. In the second case the pothole’s positions

[1]For the algorithm implementation see www.cs.tut.fi/~schiopu/Potholes

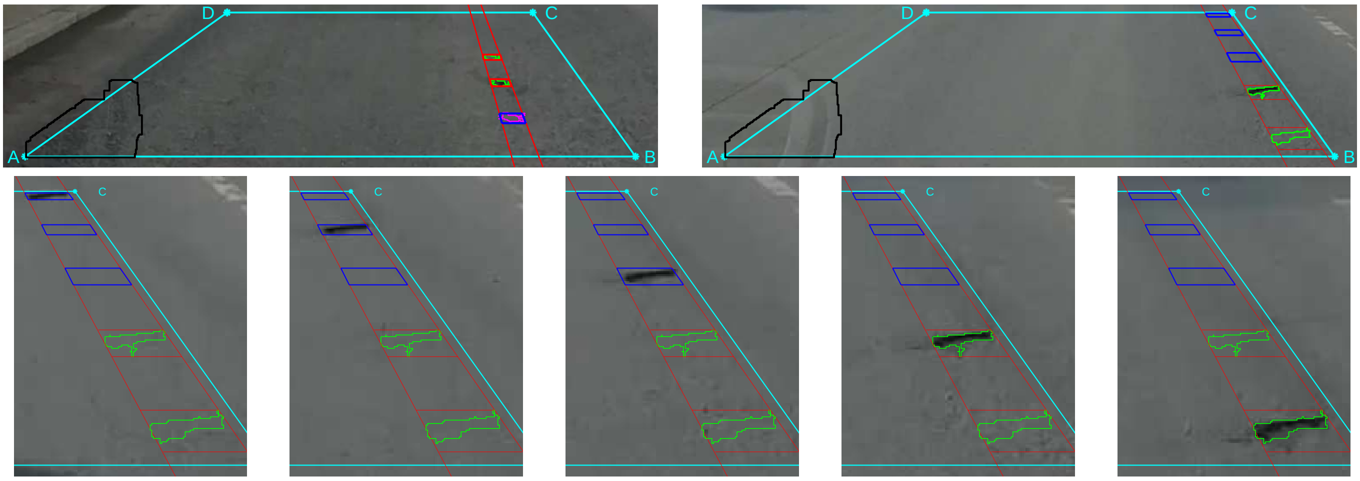

Fig. 10. *(Top)* Two examples of pothole tracking. *(Bottom)* Frame sequence in pothole tracking. The boundary pixels are marking with: (green) the detected potholes, (cyan) the ROI, (black) the object reflexions, (magenta) a pothole detected in the search area. Blue rectangles are marking the potholes tracked in the three previous frames; red rectangles are marking the potholes that were detected in two consecutive frames.

TABLE II. MEASURED PERFORMANCES OVER 61200 FRAMES

Performance		Value
Total Runtime (seconds)		639.90
Runtime (seconds) checking	pothole depth	0.40
	contour length	3.73
	region's model	0.71
	contour shape	22.40
True Positives (TP)		55
False Positives (FP)		6
False Negatives (FN)		0
Precision		90%
Recall		100%

from two consecutive frames (marked with green) are used to find its position in previous frames. The blue rectangle is marking the pothole's search area in the previous frames.

Table II shows the measured performances for the tested video sequence, which contains a number of $34\,(minutes) \times 60\,(seconds) \times 30\,(frame\_rate) = 61200$ frames. The algorithm detects correctly a number of 55 potholes (and cracks) on the road. There are 6 cases of false positives, were some shadows are labeled as potholes. These regions are placed at the edge of ROI and are representing the shadows of the trees or light-poles found in a time slice of a few minutes. Table II shows also the runtime for checking each of the four properties from Fig. 4.(a), in the shadow detection stage (see Section II-D). With a precision of 90%, a recall of 100%, the algorithm is achieving good results within a small runtime.

V. CONCLUSION

The paper proposed an algorithm for pothole detection and tracking. The region of interest (ROI) was selected off-line and candidate regions were generated using a threshold based algorithm. In a test video sequence of 34 minutes, thousands of candidate regions were analyzed by checking the region's size and model, the pothole's intensity variation, the region's contour length and shape, and the region's consistent detection in consecutive frames. The algorithm is detecting and tracking potholes with a good precision and a small runtime.

REFERENCES

[1] H. Youquan, W. Jian, Q. Hanxing, Z. Wei, and X. Jianfang, "A research of pavement potholes detection based on three-dimensional projection transformation," in *Proc. International Congress on Image and Signal Processing*, Shanghai, vol.4, pp. 1805–1808, Oct. 2011.

[2] D. Joubert, A. Tyatyantsi, J. Mphahlehle, and V. Manchidi, "Pothole tagging system," in *Proc. Robotics and Mechanics Conference of South Africa*, pp. 1–4, 2011.

[3] Z. Zhang, X. Ai, C.K. Chan, and N. Dahnoun, "An efficient algorithm for pothole detection using stereo vision,," in *Proc. International Conference on Acoustics, Speech and Signal Processing*, Florence, pp. 564–568, May 2014.

[4] E. Buza, S. Omanovic, and A. Huseinovic, "Pothole Detection with Image Processing and Spectral Clustering," in *Proc. International Conference on Information Technology and Computer Networks*, Antalya, pp. 48–53, Oct. 2013.

[5] C. Koch and I. Brilakis, "Pothole detection in asphalt pavement images," *Advanced Engineering Informatics*, vol. 25, no. 3, pp. 507–515, Aug. 2011.

[6] S. C. Radopoulou and I. Brilakis, "Patch detection for pavement assessment," *Automation in Construction*, vol. 53, pp. 95–104, May 2015.

[7] S.-K. Ryu, T. Kim, and Y.-R. Kim, "Image-Based Pothole Detection System for ITS Service and Road Management System," *Mathematical Problems in Engineering*, vol. 2015, pp. 1–10, 2015.

[8] F. Seraj, B.J. van der Zwaag, A. Dilo, T. Luarasi, and P. Havinga, "RoADS: a road pavement monitoring system for anomaly detection using smart phones," in *Proc. International Workshop on Machine Learning for Urban Sensor Data*, Nancy, pp. 1–16, Sep. 2014.

[9] A. Mednis, G. Strazdins, R. Zviedris, G. Kanonirs, and L. Selavo, "Real time pothole detection using Android smartphones with accelerometers," in *Proc. International Conference on Distributed Computing in Sensor Systems and Workshops*, Barcelona, pp. 1–6, 27-29, Jun. 2011.

[10] H. Sanchez-Cruz and R. Rodriquez-Dagmino, "Compressing bilevel images by means of a three-bit chain code," *Optical Engineering*, vol. 44, no. 9, pp. 097004–097004-8, Sep. 2005.