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 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 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 × 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

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.
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

$$T = \max(90, \bar{y} + 3\sigma_Y), \tag{1}$$

where $\bar{y} = \sum_{k=1}^{N_{ROI}} Y(x_k,y_k)$, $\sigma_Y$ is the standard deviation of the $N_{ROI}$ intensity values $\{Y(x_k,y_k)\}_{k=1,...,N_{ROI}}$ of ROI. Let us denote $\text{Mask}_{ROI}$ the pixel positions for which $Y(x_k,y_k) < T$. $\text{Mask}_{ROI}$ 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 $(x,y) \in \text{Mask}_{ROI}$, 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 $\text{minDepth} = -15$, 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 $\text{minSize} = 100$ 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.

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 $\sigma_D$ is the standard deviation of the ‘depth’ values $D_M(x,y)$. The candidate regions are generated using $\text{minSize} = 100$. 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...
estimation of an order-2 model is computed for each region

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

(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 \(\text{MSE}\) and the region is labeled as shadow if \(\text{MSE} < \min\text{MSE}\), where \(\min\text{MSE} = 50\) dB is the minimum \(\text{MSE}\) 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 \(\min\text{AvgDepth} = \min(-25, -1.75\sigma_Y)\).

(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.

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.(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).

### 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.
After its detection in a frame, a pothole can be traced using
\( d_1 = \frac{d_i}{f} \), and at an angle \( \alpha \).

Let us denote \( a = \tan \alpha \), change the variables as: \( d' = \frac{d_i}{f} \),
\( x' = x_i \rho \), and use (3) to rewrite (4) as:
\[
d' + ad'x' + x' - a = 0.
\]

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 smartphone. In the experimental
setup, the camera was placed at height \( h = 1.23 \text{ m} \) (which
varies and the precision is about 1 cm) and at camera angle
\( \alpha = 89.21^\circ \). 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, \( \lambda_m \), and maximum slope,
\( \lambda_M \), 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 \( \lambda_m \) and \( \lambda_M \);
(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 smartphone 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 x 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

<table>
<thead>
<tr>
<th>Camera specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor model</td>
<td>Sony IMX135 Exmor Rs</td>
</tr>
<tr>
<td>Sensor type</td>
<td>CMOS</td>
</tr>
<tr>
<td>Sensor size</td>
<td>4.69 × 3.52 mm</td>
</tr>
<tr>
<td>Pixel size</td>
<td>≈ 1.136 µm</td>
</tr>
<tr>
<td>Crop factor</td>
<td>≈ 7.38</td>
</tr>
<tr>
<td>Aperture</td>
<td>f/2.2</td>
</tr>
<tr>
<td>Shutter speed</td>
<td>1/14 − 1/10000</td>
</tr>
<tr>
<td>Focal length</td>
<td>≈ 4.2 mm</td>
</tr>
</tbody>
</table>

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
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) × 60 (seconds) × 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.

Table II. Measured Performances over 61200 frames

<table>
<thead>
<tr>
<th>Performance</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Runtime (seconds)</td>
<td>0:03:30</td>
</tr>
<tr>
<td>Runtime (seconds) checking</td>
<td>0:30</td>
</tr>
<tr>
<td>pothole depth</td>
<td>0.40</td>
</tr>
<tr>
<td>contour length</td>
<td>0.75</td>
</tr>
<tr>
<td>region’s model</td>
<td>0.71</td>
</tr>
<tr>
<td>contour shape</td>
<td>22.30</td>
</tr>
<tr>
<td>True Positives (TP)</td>
<td>55</td>
</tr>
<tr>
<td>False Positives (FP)</td>
<td>0</td>
</tr>
<tr>
<td>False Negatives (FN)</td>
<td>0</td>
</tr>
<tr>
<td>Precision</td>
<td>90%</td>
</tr>
<tr>
<td>Recall</td>
<td>100%</td>
</tr>
</tbody>
</table>

REFERENCES