DETECTING POTHOLES USING SIMPLE IMAGE PROCESSING TECHNIQUES AND REAL-WORLD FOOTAGE

S NIENABER*, MJ BOOYSEN* AND RS KROON**

*Department of E&E Engineering, Stellenbosch University, Private Bag X1, Matieland, 7602; Tel: 021 808-4013; Email: mjbooysen@sun.ac.za
**Computer Science Division, Stellenbosch University, Private Bag X1, Matieland, 7602 Tel: 021 808-9375; Email: kroon@sun.ac.za

ABSTRACT

Potholes are a nuisance, especially in the developing world, and can often result in vehicle damage or physical harm to the vehicle occupants. Drivers can be warned to take evasive action if potholes are detected in real-time. Moreover, their location can be logged and shared to aid other drivers and road maintenance agencies. This paper proposes a vehicle-based computer vision approach to identify potholes using a window-mounted camera. Existing literature on pothole detection uses either theoretically constructed pothole models or footage taken from advantageous vantage points at low speed, rather than footage taken from within a vehicle at speed. A distinguishing feature of the work presented in this paper is that a thorough exercise was performed to create an image library of actual and representative potholes under different conditions, and results are obtained using a part of this library. A model of potholes is constructed using the image library, which is used in an algorithmic approach that combines a road colour model with simple image processing techniques such as a Canny filter and contour detection. Using this approach, it was possible to detect potholes with a precision of 81.8% and recall of 74.4.%.

1 INTRODUCTION

Potholes in road surfaces are mostly caused by water (CSIR, 2010) and regular road maintenance is vital to prevent the decaying process. An example of a road with potholes is shown in Figure 1. The manner in which a pothole forms is dependent on the type of bituminous pavement surfacing. The volume of traffic and the axle load experienced by the road are example factors that lead to fatiguing of the road surface, resulting in the formation of cracks. These cracks allow water to seep through and mix with the asphalt. When a vehicle drives over this area, this water will be expelled through the crack with some of the asphalt, and this will slowly create a cavity underneath the crack. Eventually the road surface will collapse into the cavity, resulting in a visible pothole. If regular road maintenance is neglected, the aforementioned cracks are not repaired before they cause substantial damage to the road.

For road maintenance to take place, it is obvious that the entity responsible for the road in question must know where the pothole or decaying road section is located and an automated process could assist with this. Potholes are also problematic for drivers as they can cause a lot of damage to their vehicles. Currently, there is no device/system available to drivers that would allow them to avoid potholes. There is thus a potential market for such a solution.

Figure 1. Pothole example

The first step of the solution requires developing a device that is attached to a vehicle and will continually scan the road surface. If a pothole is detected, it will alert the driver in time and enable the driver to avoid the pothole. The second key aspect of the solution is to enable the device to log the position of the pothole via GPS (Global Positioning System). The GPS data can be uploaded via a GPRS (General Packet Radio Service) module to a network system incorporating mapping software such as Google Maps or OpenStreetMap. The data in the system can be made available to the general public as well as municipalities and road maintenance agencies. A higher awareness of the location of potholes will result in road users being more careful on certain roads or even less usage of an affected road which would reduce the additional formation of potholes. This paper focuses on the pothole detection task in the first step. The later steps will be considered in future work.

The physical attributes of a pothole, and the unique way it is perceived from a moving vehicle, allows for the identification of various technologies that could possibly detect potholes. These technologies include the use of LASER (Light Amplification by Stimulated Emission of Radiation), cameras and even vibration. Each technology type identifies at least one aspect of encountering a pothole.

This paper discusses a method for detecting potholes (via image processing) that does not require any machine learning, but is based on an algorithmic approach that is rooted in the fundamental properties of potholes. Due to the visual nature of the approach, it is self-evident that the solution is dependent on lighting conditions, obstructions in the line of view, rain and any other factors that visually impair the ability to see potholes. The work presented is therefore limited by this aspect and so does not provide an all-encompassing solution. The next section will briefly summarize the current pothole detection literature. Thereafter, the proposed methodology will be discussed in detail. An implementation including preliminary results will be presented and a clear conclusion on the effectiveness will be drawn from the results.

2 RELATED WORK

To contextualize this work, several previous studies are outlined on the detection of potholes.

In Danti et al. (2012), a system incorporating image processing algorithms was used as the main detection and classification mechanism. The study aimed to detect a variety of road-related objects such as lanes, road signs and potholes. The paper states that a pothole is

normally characterized by a distinctive black color in the road, and that this can be the predominant characteristic used to detect a pothole. To successfully segment the pothole region, a black and white threshold was applied to the image that would highlight the pothole area. The method does not rely on the use of any machine learning techniques to detect a pothole. The conclusion of the study indicated that the algorithm did not deliver desirable results and segmented many undesirable areas within an image and that a more effective filtering method would need to be used to improve the accuracy of the approach.

The work presented in Karuppuswamy et al. (2000) modeled potholes as circular obstacles with a white color and 2 feet diameter. A standard imaging board was used to detect the potholes which led to the use of an EPIX PIXCI SV4 imaging board. The software utilized by the board was configured to detect the "simulated" potholes i.e. to retrieve the centroids of the white shapes in the image. The board was connected to a camera and these components were attached to a robot and driven on a track. The image processing method used an image histogram to calculate the image threshold necessary to detect the pothole. The white color of the pothole is in a stark contrast to the black color of the road surface and it was concluded that a large peak near the lighter pixel bins would represent the pothole. After the threshold was performed on the image, the result was fed into a Robert's edge detector to highlight the pothole edge. Lastly, blobs were detected by using a contour detection algorithm and the size of the blob detected was calculated to determine if it fell within the 2 feet diameter scope. No results or analysis was given in the paper but due to the nature of the setup (physical deviations from real-world potholes – size and color) this approach can be seen as an image processing exercise that is not directly applicable to real-world pothole detection.

Another method for the detection of potholes proposed segmenting a road based on its defect and non-defect regions (Koch & Brilakis, 2011a). Similarly to the previous approaches, it utilized image thresholding and accomplished this by using a histogram shape-based thresholding algorithm. A pothole is detected in a defect region by the use of morphological thinning and elliptic regression. Morphological thinning is an algorithm that is used on binary images to separate foreground pixels from the background pixels (Kaehler & Bradski, 2008). The paper assumes that a pothole is elliptically shaped and therefore implements an elliptic regression algorithm which aims to determine how well the collection of data points (pixels) fit an elliptical shape. All of the images processed by the algorithm were obtained by placing a high-speed camera on a robot. The placement of the camera simulates being placed on the back of a moving vehicle with the camera mounted in such a manner that it would be tilted towards the road surface. Images were processed in two stages. The first stage operated in real-time and identified frames with defect regions. The second stage would pass only the frames that had defects to the more complex pothole detection algorithm. The system parameters were obtained by using a training set of 50 pothole images. It was deployed on 70 images and was found to have a precision of 81.6%.The images that were used in the paper were mostly acquired via Google Images. These images were taken by different people with different cameras at a variety of angles to the road surface. Also, some of the photos were taken up close to the potholes whilst others were not.

Once a pothole is detected, it can also be tracked so that the same pothole in subsequent frames is only counted as a single pothole occurrence. This is useful when it's necessary to accurately determine the number of potholes detected. Koch & Brilakis (2011a) continued with further research that led to the work in Koch & Brilakis (2011b) where a method for tracking the potholes between frames was discussed. This paper disregards this aspect as the focus is warning a drivers of potholes, so that it is only necessary to classify whether a section of road has potholes or not.

Pothole detection via spectral clustering is another option and is presented in Buza et al. (2013). Buza et al. (2013) based their work on Koch & Brilakis (2011a) and the thresholding algorithm was slightly modified to use Otsu's image thresholding instead which will automatically determine the correct threshold value necessary per frame using internal algorithms (OpenCV, 2014). Images were obtained from Google and the algorithm was tested on these images.

As has been previously stated, it is possible to exploit a variety of physical properties in order to detect potholes. One such an attempt is presented in Joubert et al. (2011) where it was attempted to utilize the various InfraRed sensors built into a Microsoft Kinect (OpenKinect, 2011) that was attached to the back of a vehicle. Initial results were given but the utilization of the system in broad daylight is problematic due to the overwhelming presence of InfraRed radiated by the sun as was demonstrated by Drexel Autonomous Systems Laboratory (2013).

3 METHODOLOGY

In this paper, a visual approach is proposed that does not require any machine learning algorithms in the same fashion as the related work presented in the previous section. Thus, the system will not require any prior training as is required in, for example, a HAAR cascade classifier (Viola & Jones, 2001). A selection of images was withheld from the input test images to tune the various constants necessary for the different types of algorithms used, such as the Canny threshold (Kaehler & Bradski, 2008).

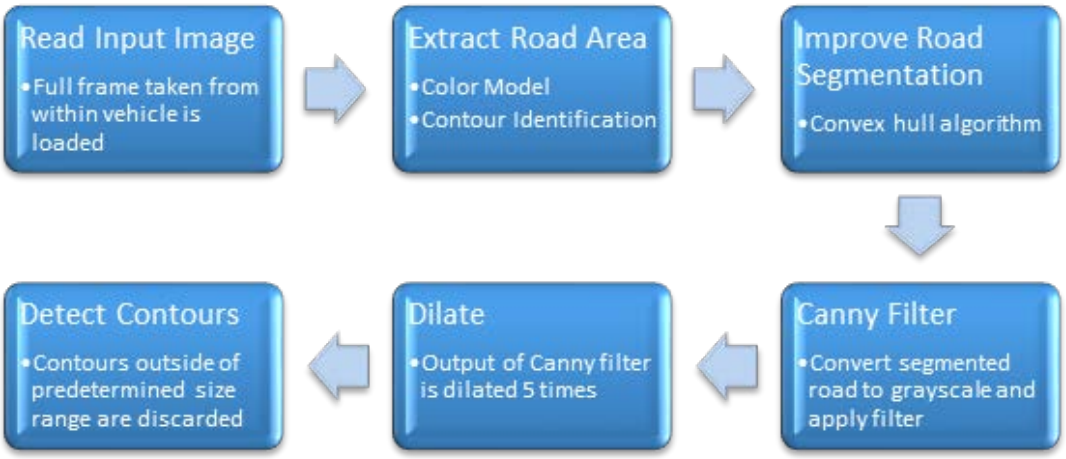

Figure 2. Overall algorithm block diagram

To obtain the images used, a camera was mounted inside the vehicle on the front windscreen. The reason for this is that it more accurately reflects the scenario of developing a device that can be fitted to a vehicle for commercial use. An image library was created by driving around and creating an image library that includes 48913 images. To sidestep any issues that normally occur when taking photos from a moving vehicle such as blurring of the image, a GoPro camera was used. This also frees up some of the processing power required to run the developed software. The GoPro camera was set to a 0.5 second time lapse mode and using this setting it was possible to achieve stable images for speeds up to 60 km/h.

A Road model

In preliminary work it was found that irrelevant information in a frame, such as foliage, can lead to false positives. Therefore, it was decided to first extract the entire road surface

automatically in the software and scan only the extracted area for potholes. This is reasonable since the only region of interest with regards to potholes, is the road surface directly in front of the vehicle. Additionally, it eases the classification process as all other objects and foliage outside of the road has been removed from interfering with the classification.

Extraction of the road was achieved by using a small rectangular region of interest within the image boundary just above the hood of the vehicle. This method assumes that this section will always contain a part of the road if the driver maintains a safe following distance from a vehicle in front. The mean and standard deviation per color channel of the this region of interest is calculated and in order to robustly address the issue of road color variation within the image, the road color is modelled as lying between three[1] standard deviations below and above the mean for each channel. The contours within the frame that describe the specific road color sections are then determined. In a given frame, it is possible that the average color model obtained from the selected region of interest does not describe the road accurately for the entire road surface of that frame (especially when large patches of sand are present). Therefore, to extract the road surface even more accurately, a convex hull algorithm was applied to the extracted contours. A convex hull algorithm constructs a convex contour around all the furthest points of several points of interest. An example of this algorithm is shown in Figure 3. Each of the indentations of the hand are marked (A-H) and indicate the gaps in the figure with respect to the outlying points. These gaps are included in the convex hull because the convex hull algorithm will connect each of the furthest points together with straight lines and would encompass the entire hand. Similarly, when sections of the road cannot be extracted via its colour, the convex hull aids in extracting an entire section.

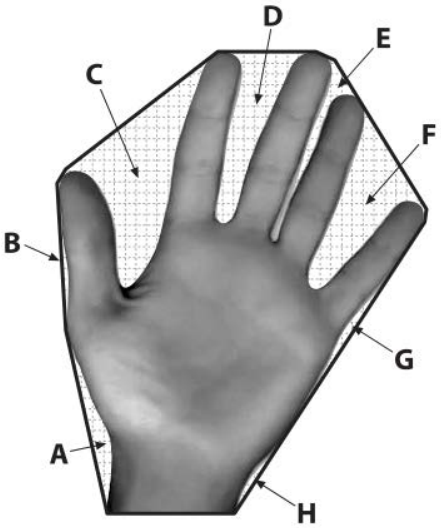

Figure 3. Example Convex Hull Algorithm (Bradski & Kaehler, 2008)

B Pothole model

The pothole model is derived from the assumption that any strong dark edge within the extracted road surface is deemed a pothole edge if it adheres to certain size constraints. By inspecting Figure 1, it can be seen that one of the characteristics describing the potholes is a large dark shadow area. At this point, potholes that do not have dark edges

[1] The paper accepted for presentation at SATC 2015 erroneously reported this value as one. This version of the paper correctly states this value as three.

and only have different color variations within them like sand or dirt are disregarded and will be studied in future work. The size constraints were obtained using the selection of images withheld for parameter tuning. Any shape of contour that meets these conditions is deemed a pothole by the algorithm.

C Algorithm working

The proposed method for detecting a pothole starts by converting the extracted road section image to a grayscale image. To clear up this image and remove noise, a Gaussian filter was applied to the grayscale image. A simple differentiation-based edge detection algorithm (Canny edge detection) is then performed on the extracted road surface. The Canny edge detector produces a black-and-white image. An example output of this edge detection is shown in Figure 4, where the input photograph is shown on the left and the output from the edge detection is shown on the right. Unwanted edges, especially around the outer boundaries of the road, are created by shadows of branches and leaves in trees and are more often worsened by sections of light shining through. Additionally, other vehicles on the road also create unwanted edges.

Figure 4. Example Canny edge detector output (Bradski & Kaehler, 2008)

From the figure it is clear that all of the edges detected by the Canny filter are white lines and are depicted on a black background. The dark regions found within a pothole are usually small in the frame, and often a single pothole will not produce a single edge, but many small edges that are not connected together. Another problem encountered is that the extracted road surface sometimes has sharp and unwanted unconnected edges on its outer boundaries. These are created by the convex hull algorithm and are dependent on the road colour and lighting conditions. Due to the approximate shape of the road visible to the camera, grass and dirt alongside a road is also sometimes extracted as part of the road section (due to the convex hull algorithm) and can lead to false positives. To solve this problem, it was determined that the unwanted edges close to the outer boundaries can be removed by dilating the Canny output image several times. Performing dilation on an image increases the area of the lighter pixels. As a result, when dilation is performed, the unwanted edges close to the outer boundaries become absorbed into the outer boundary leaving only the boundary contour visible. Another advantage of this approach is that it can be used to aid the removal of other vehicles in the frame. Other vehicles on the road are normally found close to the outer boundaries of a frame and sufficient dilation would mean that they too are absorbed into the outer boundary.

The last contour detection is then applied to the dilated image to find the potholes within the road section. The contours are filtered and those that do not meet the size constraints of the pothole model are discarded. This last step filters out any small defects in the road that are not classified as potholes as well as the larger contours found on the outer boundary of the extracted road.

4 IMPLEMENTATION AND RESULTS

The setup comprised a GoPro camera that was connected to the front windscreen of a car with the setting on 0.5 s time lapse mode and it provided footage that required no de-blurring. The footage was processed off-line on an Intel Pentium core i7 3.5 GHz with 8 GB of RAM. By timing the duration of the program execution, preliminary observations can be made to determine whether or not the algorithm could be deployed in real time. The software used to create the algorithms was Microsoft Visual C++ 2012 with the OpenCV open source library.

A selection of 53 images containing 97 potholes from the newly created pothole image library were selected as input for the algorithm. This amount of images were chosen to determine the feasibility of the project and to test the preliminary work presented in this paper. The selection included images from various scenarios, such as driving whilst facing the sun, having the sun on the right of the vehicle, and potholes near trees with large shadows. The input footage was obtained while driving at a speed of approximately 40 km/h, which resulted in subsequent frames showing a single pothole at various distances from the vehicle. It was also found that the lines that indicate the lane separation were either faded or not apparent at all. The appearance of potholes and fading of lines usually coincide, since they both arise as a result of poor maintenance. If the lines were more clear they would have created problems with the algorithm and in future work this aspect will be addressed to ensure a more complete real-world solution.

A full-sized example of the footage that was taken from within the vehicle using the camera mounted to the windscreen of the vehicle is given in Figure 5. It can clearly be seen that there is a lot of information within the image (such as foliage) that is not relevant to pothole detection. This full-sized image is then fed into the algorithm which first extracts the road before any potholes are detected.

Figure 5. Example input frame to algorithm

Two examples of the step-by-step outputs of the algorithm are illustrated in Figure 6 - the example on the right hand side corresponds to the input image shown in Figure 5.

The top images in Figure 6 show the output of the road extraction algorithm. The extracted road is then converted to a grayscale image and a Gaussian filter is applied to it to yield the next set of images. The Canny edge detector is then applied, yielding the middle image

set. The second last image set is obtained by the dilation process, which clearly increases the area of all of the white pixels in the image. Lastly, the contours within a dilated image are found and those within certain size constraints are classified as potholes.

These results are back-projected onto the initial image which clearly indicates the potholes detected in the footage. In the more complex example on the right hand side it can be seen that there were potholes present as well as an oncoming vehicle. From the dilation process it can be seen how the wipers and the oncoming vehicle are absorbed into the larger outer boundary and are therefore "filtered" out by this process.

Usually, the first point of failure of this approach was the Canny filtering being too strict in certain cases and not including certain edges. The second point of failure, which was expected given the approach used, was that potholes on the outer edges of the road would be absorbed into the segmented road boundary.

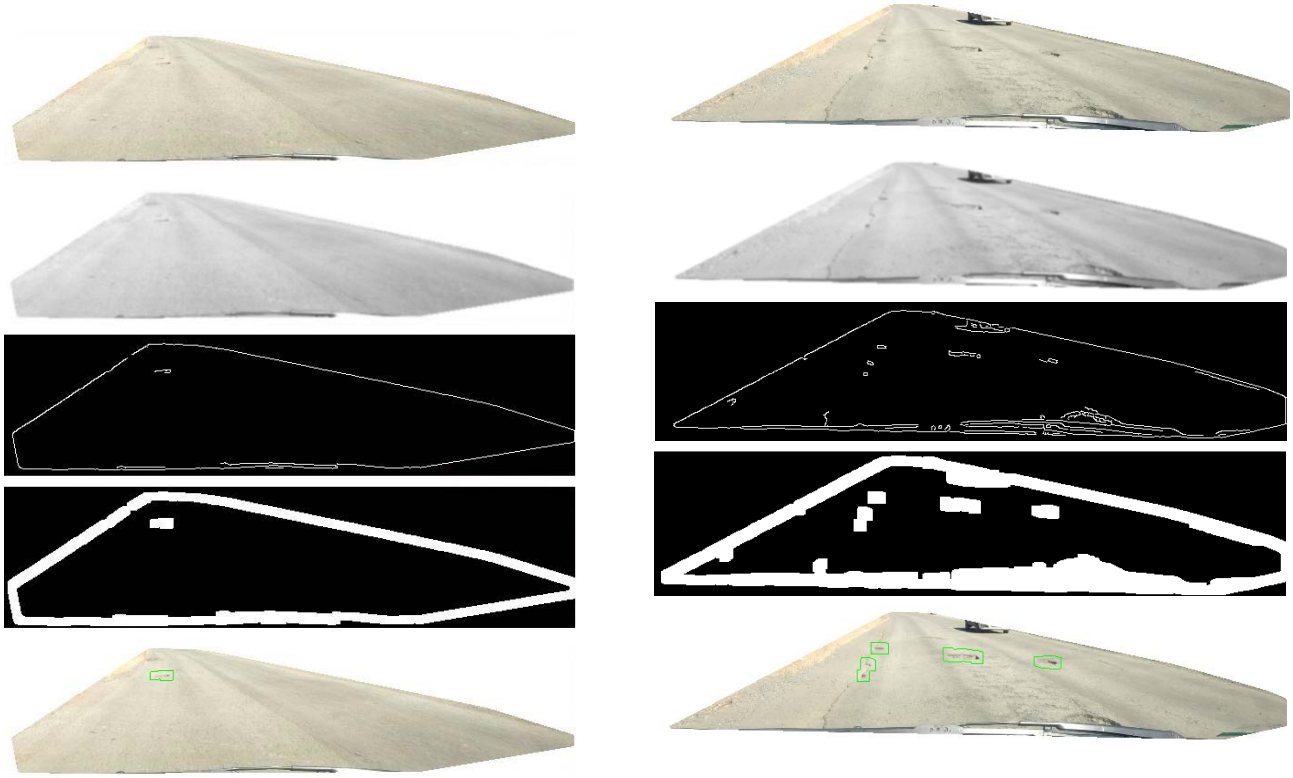

Figure 6. Example output of algorithm. Left side gives a simple example whilst right side shows a more complex example

The key performance measures for this study are presented in Table 1. The sample study indicated a precision of 81.8% and recall of 74.4%. It was also found that the algorithm rejected vehicles on the road 80% of cases (5 frames contained vehicles, and a vehicle was classified as a pothole in only 1 frame). It is possible to increase this accuracy by increasing the dilation process by several factors; however, this will lead to a loss of pothole detection, as more potholes likely merge with the outer road contour.

Table 1. Results of algorithm

True Positives (TP)	False Positives (FP)	False Negatives (FN)	Precision (%)	Recall (%)
72	16	24	81.8	74.4

The formulas to calculate the precision and recall are given in Equations 1 and 2 (Szeliski, 2011).

$$\text{Precision} = TP/(TP+FP) \quad (1)$$

$$\text{Recall} = TP/(TP+FN) \quad (2)$$

Due to the nature of the algorithm it was also determined that in the event that two potholes are closely spaced together, they would be grouped together and seen as a single contour. This was mostly the case with potholes that were further away from the vehicle. As the vehicle moved closer, the potholes eventually separate far enough from each other that they can be identified as individual potholes.

An important aspect with respect to the applicability of the algorithm is the time it takes to run the algorithm. Per frame, it was found that it takes approximately 0.148 seconds to process the road extraction algorithm and 0.037 seconds to detect potholes within the extracted road frame. The total time to complete the algorithm is ≈ 0.2 seconds. This time does not include optimizations at this point and the code was deployed in debug form. The resulting computational time to complete the detection indicates that it is likely that the algorithm can be deployed on a computer on a vehicle driving at a normal speed and warn the driver ahead of time to avoid the pothole.

The algorithm does have certain limitations such as the range in which it will detect a pothole in front of the vehicle. From the frames, it is evident that the algorithm will only yield accurate TP results for potholes between approximately 2 m and 20 m ahead of the vehicle. The lower limitation is due to the dilation process as the pothole merges with the lower lines of the outer boundary of the road whilst the upper limitation has to do with visibility of the pothole through the camera lens. In certain instances, the dilation process will also allow potholes to be absorbed into the outer border if a strong shadow is present across the road, like a bridge or tall tree and the pothole is in close proximity (within $\approx$ 1 m) to the shadow. The upper limitation can be adjusted via the Canny filter parameters, but if it is adjusted to include more potholes at a further distance, it will also currently, include more unwanted edges such as small cracks closer to the vehicle that are not classified as potholes.

5 CONCLUSION

The paper presented a good preliminary method for pothole detection using a single optical camera that can detect potholes within a range of ≈ 2 m - 20 m and does not rely on training any models. It was found that the method works relatively well in discarding other vehicles although further research must be performed to improve this aspect. By measuring the time it takes to perform the algorithms, it was found that the algorithm execution speed is adequate given a vehicle speed of less than 60 km/h although the actual maximum distance at which the pothole can be detected needs to be improved to account for the driver reaction time. The algorithm is successful in the detection of potholes and an attempt will be made to upgrade it to include potholes with no visible edges (due to sand or dirt) in future research.

ACKNOWLEDGEMENTS

The authors would like to thank MTN for their continued financial support through the MTN Mobile Intelligence Lab.

REFERENCES

Bradski, G. & Kaehler, A., 2008. In: *Learning OpenCV.* California: O'Reilly Media Inc., p. 154.

Buza, E., Omanovic, S. & Huseinovic, A., 2013. *Pothole Detection with Image Processing and Spectral Clustering.* Antalya, Turkey, 2nd International Conference on Information Technology and Computer Networks.

CSIR, 2010. *Potholes: Technical guide to their causes, identification and repair.* [Online] Available at: http://www.csir.co.za/pothole_guides/docs/Pothole_CSIR_tech_guide.pdf [Accessed 15 April 2015].

Danti, A., Kulkarni, J. & Hiremath, P., 2012. An Image Processing Approach to Detect Lanes, Pot Holes and Recognize Road Signs in Indian Roads. *International Journal of Modeling and Optimization,* 2(6), pp. 658-662.

Drexel Autonomous Systems Laboratory, 2013. *KINECT in direct sunlight.* [Online] Available at: http://dasl.mem.drexel.edu/wiki/index.php/KINECT_in_direct_sunlight

Joubert, D., Tyatyantsi, A., Mphahlehle, J. & Manchidi, V., 2011. *Pothole Tagging System.* Pretoria, 4th Robotics and Mechatronics Conference of South Africa.

Kaehler, A. & Bradski, G., 2008. *Learning OpenCV.* 1st ed. California: O'Reilly.

Karuppuswamy, J., Selvaraj, V., Ganesh, M. & Hall, E., 2000. Detection and Avoidance of Simulated Potholes in Autonomous Vehicle Navigation in an Unstructured Environment. *Intelligent Robots and Computer Vision XIX: Algorithms, Techniques, and Active Vision. Proc. SPIE 4197,* pp. 70-80.

Koch, C. & Brilakis, I., 2011a. Pothole detection in asphalt pavement images. *Advanced Engineering Informatics,* August, 25(3), pp. 507-515.

Koch, C. & Brilakis, I., 2011b. *Improving Pothole Recognition through Vision Tracking for Automated Pavement Assessment.* Enschnede, Netherlands, The 18th EG-ICE Workshop on Intelligent Computing in Engineering, pp. 1-8.

OpenCV, 2014. *The OpenCV Reference Manual Release 2.4.9.0.* [Online] Available at: docs.opencv.org/opencv2refman.pdf [Accessed 18 July 2014].

OpenKinect, 2011. *OpenKinect Hardware Info.* [Online] Available at: http://openkinect.org/wiki/Hardware_info

Szeliski, R., 2011. *Computer Vision - Algorithms and Applications.* London: Springer.

Viola, P. & Jones, M., 2001. Rapid Object Detection using a Boosted Cascade of Simple Features. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* pp. 511-518.