ISBN: 978-628-95207-8-1. ISSN: 2414-6390. Digital Object Identifier: 10.18687/LACCEI2024.1.1.694

Application of artificial intelligence and object detection to determine failures in flexible pavements

R. Muñoz, Bachelor of School of Civil Engineering[1], J.P. Saldaña, Bachelor of School of Civil Engineering[1], M. Silvera, Master of School of Civil Engineering[1], F. Campos, Master of School of Civil Engineering[1] D. Palacios-Alonso, Ph.D. degree in advanced computation[2]

[1] Peruvian University of Applied Sciences, Lima - Peru, u201913366@upc.edu.pe, u201911019@upc.edu.pe, manuel.silvera@upc.edu.pe, pccifcam@upc.edu.pe
[2] Rey Juan Carlos University, Madrid, daniel.palacios@urjc.es

***Abstract*—** According to the Association of Traffic Accident Victims (AVIACTRAN), on average there are 10 potholes per kilometer of flexible pavement in the city of Lima. This is due to the lack of timely road maintenance by government authorities, as they do not have a system that allows them to identify in real time the different pavement defects to make decisions. To address this problem, this paper proposes the use of the Cascade Trainer GUI algorithm and Python programming to determine defects in flexible pavements. The proposal consists of training the algorithm with images of different pavement defects using mobile phone cameras or drones for data collection and evaluation of the condition of the pavement. The implementation of the model provides a saving of 60% in the detection time of functional defects of the pavement compared to the traditional method. The methodology detects 5 types of defects (crocodile cracking, edge cracking, block cracking, potholes and wear) with an accuracy of 70%. This innovative approach offers an efficient and fast solution for management road infrastructure in urban and rural environments.

Keywords*— *Cascade Trainer GUI, failure detection, Python, drone, pavements.

I. INTRODUCTION

Flexible pavements are prone to frequent failures due to their inherent nature. This vulnerability implies the need to quickly identify these failures to preserve the integrity of the pavement and guarantee the safety of users. However, the detection of these failures is commonly done manually, which is laborious and time-consuming. In this context, the need arises to develop more efficient, faster and less expensive detection methods, in order to keep the roads in optimal conditions and ensure the safety of those who travel on them.

Digital Object Identifier: (only for full papers, inserted by LACCEI).
ISSN, ISBN: (to be inserted by LACCEI).
DO NOT REMOVE

Traffic congestion in developing countries has caused budget disputes that affect the maintenance and durability of pavements directly or indirectly [1]. Additionally, asphalt mix pavements are vulnerable to early damage, such as rutting and potholes, exacerbated by heavy traffic and adverse weather conditions [2][3][4]. To address this challenge, faster ways of pavement assessment are needed and advanced approaches employing machine learning and automated data collection technology, such as in-vehicle cameras, drones, and laser scanning, are being explored [5]. Furthermore, the accuracy of 3D laser scanning systems has been demonstrated in the evaluation of asphalt pavements [6]. Other recent research focuses on commercial technologies, such as object detection and deep learning, for automated pavement evaluation [7]. Innovative methods to detect pavement failures are also explored, such as the use of computer vision with Yolo v3-SPP, achieving an accuracy of 82.1% [8]. Others propose the use of a GoPro camera in vehicles to inspect cracks in roads, using the "ConnCrack" method with accuracies between 81% and 97% [9]. Also the improvement of object detection algorithms such as YOLO v7, called YOLO-SAMT, demonstrates an accuracy of 96.67% in detecting various failures [10]. The use of YOLOv3 in deep learning shows a correct detection rate between 91.0% and 97.3%, outperforming traditional methods in pavement failure detection [11]. However, the YOLO program only works as long as it is connected to a Wi-Fi network, which makes it difficult to use in rural areas where there is no stable connection and makes it unviable for civil engineering projects. Another innovative proposal is the use of the Cascade Trainer GUI software, which despite having limited use only for Windows, has been shown to present greater ease of training compared to other detection methods [12].

That said, an innovative and commercial computational tool is used to identify failures in flexible pavements, which does not require an internet connection for its operation unlike other programs. The methodology involves using Python programming to process images of pavement failures, thereby creating an actionable database. Subsequently, an algorithm is trained with Cascade Trainer GUI to generate a neural network and thus be able to detect failures in an automated manner. This technology, in addition to detecting failures, also determines the area and lengths presented as long as there is a reference point. This proposal, once trained, does not present any major difficulty for detection since taking photos would be enough for measurement and detection, making it less expensive since only a drone and an operator would be enough.

II. MATERIALS AND METHODS

A. *Failure Detection Using the Cascade Trainer GUI Program*

Object detection is a technological advance that can be used for various functions, in this case the use of the Cascade Trainer GUI program is used. This program can work in areas where there is no stable signal since it does not require internet connectivity, making it essential to detect functional failures in pavements in any area where there is no coverage.

The Cascade Trainer GUI software makes it easy to train cascade classifiers, which are object detection techniques that can work through images or videos. This program, in addition to being trained, allows you to test and improve waterfall classifier models. It presents a graphical interface in which you can adjust the parameters and easily use the OpenCV tools for training and testing. The training is done through Python, a versatile, powerful and easy-to-use language.

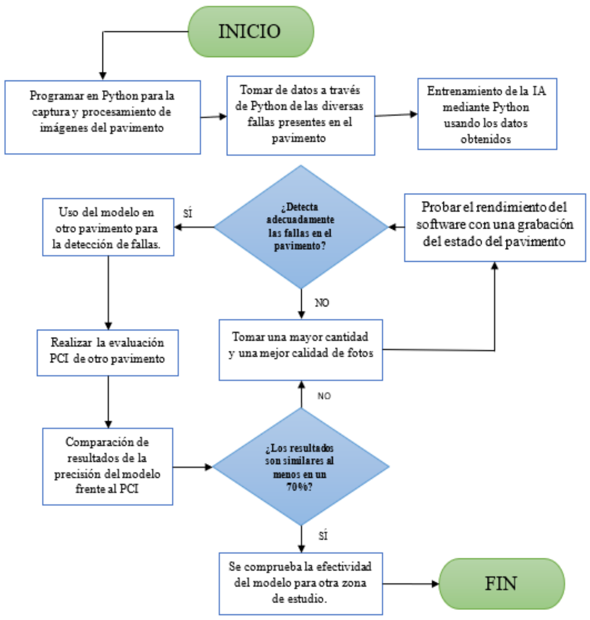


Fig.1. Methodology flowchart

B. *Python Programming Language*

Python is the most widely used programming language, which works for purposes such as data analysis, automation, web development, and software creation. In this case, its use is used to automate the detection of pavement failures.

This language is used to train the Cascade Trainer GUI software so that it can automatically detect and classify the types of failures present in different areas.

C. *Camara*

Currently it is possible to carry out studies of the properties of pavements using images captured with different types of cameras, the resolution of said camera being the fundamental key.

The cameras that can be used for the study, which are compatible with the program, are:

- Drone camera: these are located in the front of the drone and send the image instantly in real time to the pilot, imitating first-person vision. These types of cameras are small and lightweight.
- Cell phone camera: currently all cell phones have a high-resolution camera implemented with different additional features implemented.
- Webcam: This type of digital camera is small and allows you to capture and transmit images and videos over the Internet to the connected computer or laptop.

D. *Pavement Condition Index (PCI)*

The Pavement Condition Index is a comprehensive approach used to evaluate and rate pavements, both flexible and rigid, within the context of current Road Management systems.

This methodology stands out for its simple implementation and does not require the use of specialized tools.

To calculate the PCI, a visual inventory is carried out that allows to evaluate the current condition of the pavement. This process determines the typology, severity and extent of the damage observed on the pavement surface.

TABLE I.

PCI CLASSIFICATION

Rank	Classification
100 - 85	Excellent
85 - 70	Very good
70 - 55	Good
55 - 40	Regular
40 - 25	Bad
25 - 10	Very bad
10 - 0	Failed

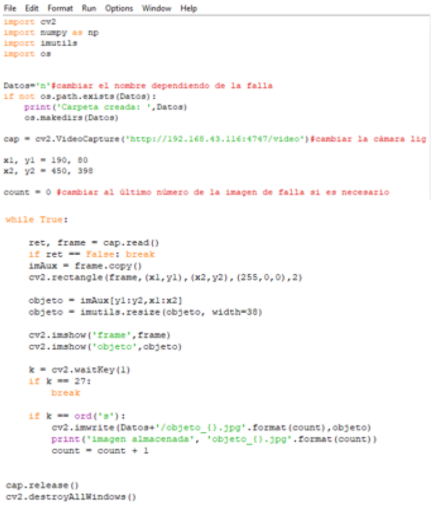
Table 1 shows the PCI classification, which varies on a scale that goes from 0 to 100. The value of 0 represents a pavement that is in poor condition or has completely failed, while a value of 100 indicates that the Pavement is in optimal condition.

III. DEVELOPMENT

A. *Data collection*

The classification of failure types depends on the data taken in the field. In this case, 5 types of failures in flexible pavements were captured (crocodile skin, edge cracking, block cracking, holes and weathering).

Taking photos must be done through Python, since they must have a small size so that the processing is more fluid and does not have problems with the capacity of the laptop on which the programming is carried out, this depends on the hardware of the laptop. The Python program allows to connect any camera to the laptop through the same WiFi network and take photos with it, in this way the program reduces the size of the photo to optimize the training.



```
File Edit Format Run Options Window Help
import cv2
import numpy as np
import imutils
import os


Datos='n'#cambiar el nombre dependiendo de la falla
if not os.path.exists(Datos):
    print('Carpeta creada: ',Datos)
    os.makedirs(Datos)

cap = cv2.VideoCapture('http://192.168.43.116:4747/video')#cambiar la cámara lig

x1, y1 = 190, 80
x2, y2 = 450, 398

count = 0 #cambiar al último número de la imagen de falla si es necesario

while True:

    ret, frame = cap.read()
    if ret == False: break
    imAux = frame.copy()
    cv2.rectangle(frame,(x1,y1),(x2,y2),(255,0,0),2)

    objeto = imAux[y1:y2,x1:x2]
    objeto = imutils.resize(objeto, width=38)

    cv2.imshow('frame',frame)
    cv2.imshow('objeto',objeto)

    k = cv2.waitKey(1)
    if k == 27:
        break

    if k == ord('s'):
        cv2.imwrite(Datos+'/objeto_{}.jpg'.format(count),objeto)
        print('imagen almacenada', 'objeto_{}.jpg'.format(count))
        count = count + 1


cap.release()
cv2.destroyAllWindows()
```

Fig.2. Typing the code in Python to take photos of the failures

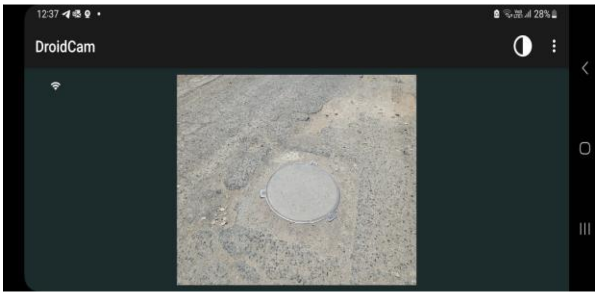


Fig. 3. Screenshot of the cell phone used to take photos

Fig. 4. Taking photos from the cell phone for training

More than 200 photos are taken for each type of failure, classifying them into different folders to identify them. It is important to center the images and have a good resolution so that the software can work more accurately.

When taking the pavement images, 2 folders will be created. The first, with positive images, which alludes to the detected failure, and the second, with negative images, made up of images of the environment, the pavement in good condition and other types of failures present. This must be done with each failure present. The purpose of taking these images is so that the algorithm recognizes when there is a failure and when it does not.

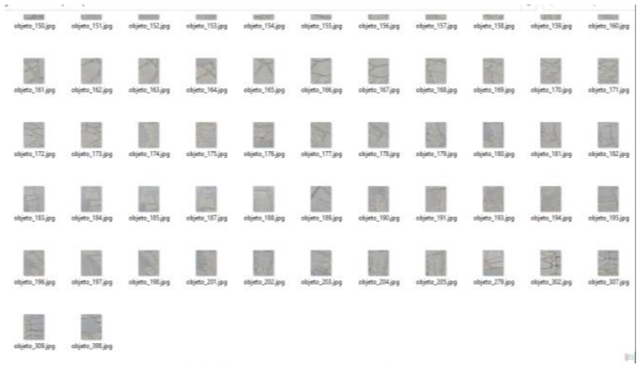


Fig. 5. Creation of folder "p" to store positive images

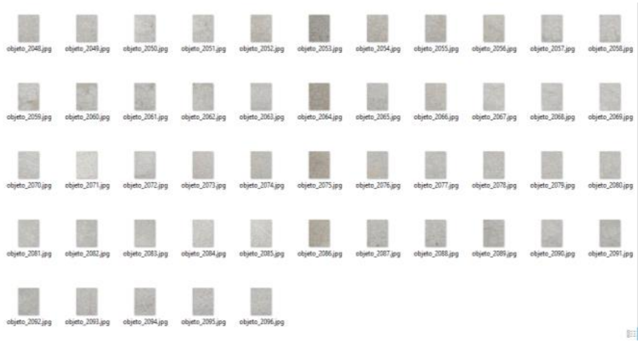


Fig. 6. Creation of folder "n" to store negative images

Finally, after training the algorithm, a recording of the study area must be made so that the algorithm can detect functional failures later. The recording can be done either with a cell phone camera or with a drone, in this case it will be done with a drone.

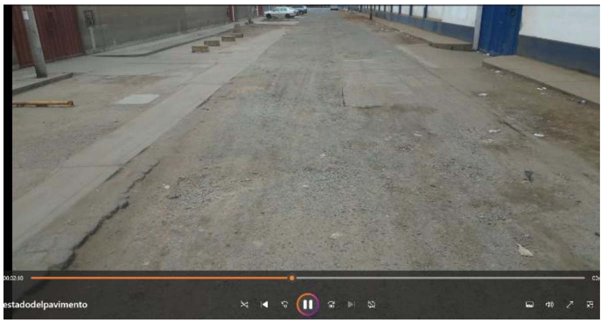


Fig. 7. Recording of the study area with a drone.

B. *Training using Python*

Programming is carried out using Python after training the neural networks in the Cascade Trainer GUI software, this is done with the purpose of detecting the types of failures existing in the pavements. This must be done for all existing failures. The codes implemented are to choose the image to detect, the program to use (Cascade Trainer GUI) and the resizing of the image.

```
import cv2
import numpy as np

img = cv2.imread('objeto1.jpg')
rows,cols = img.shape[:2]
img = cv2.resize(img,None,fx=0.13,fy=0.13,
                 interpolation=cv2.INTER_CUBIC)

pruebaClassif = cv2.CascadeClassifier('cascade.xml')

gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

toy = pruebaClassif.detectMultiScale(gray,
scaleFactor = 5, minNeighbors = 91, minSize=(70,78))

for (x,y,w,h) in toy:
    cv2.rectangle(img, (x,y),(x+w,y+h),(0,255,0),2)
    cv2.putText(img,'Agrietamiento en bloque',(x,y-10),2,0.7,(0,255,0),2,cv2.LIN

cv2.imshow('frame',img)
```

Fig.8. Python code for failure detection

Then the codes in Python must be organized to be able to detect all the neural networks simultaneously and implement a green square that locates and mentions the type of failures.

```
import cv2

cap = cv2.VideoCapture('estadodelpavimento.mp4')

fallaClassif = cv2.CascadeClassifier('3.xml')
fallaClassif1 = cv2.CascadeClassifier('7.xml')
fallaClassif2 = cv2.CascadeClassifier('1.xml')
fallaClassif3 = cv2.CascadeClassifier('13.xml')
fallaClassif4 = cv2.CascadeClassifier('19.xml')

while True:

    ret,frame = cap.read()
    cols,rows = frame.shape[:2]
    frame = cv2.resize(frame,None,fx=0.3,fy=0.3,
                       interpolation=cv2.INTER_CUBIC)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    toy = fallaClassif.detectMultiScale(gray,
    scaleFactor = 5,
    minNeighbors = 91,
    minSize=(70,78))

    for (x,y,w,h) in toy:
        cv2.rectangle(frame, (x,y),(x+w,y+h),(0,255,0),2)
        cv2.putText(frame,'Agrietamiento en bloque',(x,y-10),2,0.7,(0,255,0),2

    toy2 = fallaClassif.detectMultiScale(gray,
    scaleFactor = 5,
    minNeighbors = 91,
    minSize=(70,78))

    for (x1,y1,w1,h1) in toy2:
        cv2.rectangle(frame, (x1,y1),(x1+w1,y1+h1),(0,0,255),2)
        cv2.putText(frame,'Grieta de Borde',(x1,y1-10),2,0.7,(0,0,255),2,cv2.I
```

Fig.9. Python code for multiple failures detection

Once the code has been entered, the previously recorded video is analyzed to detect the main functional failures.

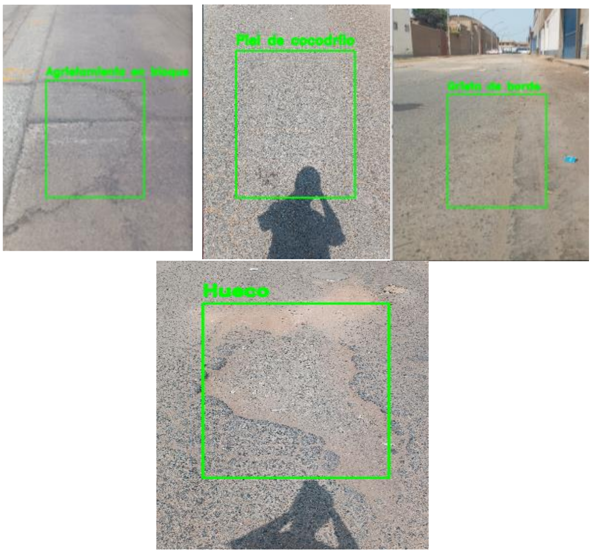


Fig.10. Simultaneous detection of existing failures

C. *Software performance*

The complexity of the proposed method for failure detection lies in the software training; on average, 200 positive images and between 2000 to 2100 negative photos are needed, which make up the environment.

The positive images are used for the program to compare the types of class failures according to the type of failure, in this way the program classifies the new photos captured by comparing them with the positive images, while the negative images are used so that the program knows which ones are not that type.

Detection performance varies depending on the quantity and quality of the images used for training. The same number of positive images were used for each type of failure, but in the case of negative images they vary.

After training and coding, the following results were obtained:

TABLE II.

PERFORMANCES OF THE ALGORITHM WHEN DETECTING DIFFERENT FAILURES

Type of failure	Positive Images	Negative Images	Performance
Crocodile skin	200	2200	90%
Edge cracking	200	2200	80%
Block cracking	200	2100	85%
Hole	200	2100	70%
Weathering	200	2100	85%

As seen in Table 2, the model presents different performances depending on the type of failure being analyzed. This is due to the complexity of one failure compared to another. However, on average the model performance is 82%.

It is considered a good percentage, so the model is implemented in another study area.

Then, the PCI evaluation of the pavement of the other study area is carried out in order to compare the results of the traditional method against the proposed method.

After performing the PCI evaluation, the following failures were found:

TABLE III.

FAILURES LOCATED IN THE STUDY AREA

N°	Type of fault
1	Crocodile skin
13	Hole
9	Unevenness
3	Block cracking
7	Edge cracking

Table 3 shows the number of failures according to the PCI manual and the types of failures identified in the PCI manual evaluation.

Finally, the proposed model will be applied to this new study area and the result obtained will be compared with the result of the manual PCI method.

IV. RESULTS

In this section, the analysis of the main indicators and the comparison between the traditional PCI method and the proposed method will be carried out. First, the precision of the methods will be analyzed and compared to verify their reliability and finally, the indicators of precision, time and cost will be compared.

TABLE IV.

ACCURACY COMPARISON BETWEEN PCI METHOD AND OBJECT DETECTION WITH PYTHON AND CASCADE TRAINER

Type of failure	PCI Accuracy	Precision Cascade Trainer	Difference
Crocodile skin	94%	75%	19%
Hole	94%	70%	24%
Unevenness	90%	55%	35%
Block cracking	94%	80%	14%
Edge cracking	94%	70%	24%

As seen in Table 4, there is an average difference in accuracy of 23.2%, which is a considerable difference and must be fixed.

However, it should be noted that the model took 15 minutes to detect the present failures. This value represents 40% of the total time spent in the PCI method, so the model demonstrates greater speed.

A. Failures detection

The program successfully detects the types of failures seen through the drone's camera. Also shown in previous images (Fig. 10) is the real-time operation using the cell phone camera.

Failure detection through drone recording is more complicated than detection via cell phone, since the video captured by the drone must be processed. Processing time would vary depending on the computer hardware where the program is located.

Fig.11. Block Cracking Failures Detection via Drone Camera

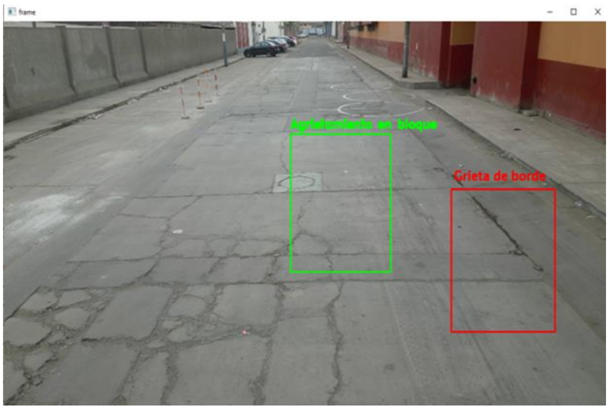


Fig.12. Block Edge Cracking and Cracking Failure Detection via Drone Camera

B. *Comparison between the proposed method and the traditional method*

When comparing the proposed method with the traditional method, it is observed that failures detection using the Cascade Trainer GUI software has lower precision, greater speed and lower cost.

TABLE V.

COMPARISON BETWEEN THE RESULTS OF THE PCI METHOD AND OBJECT DETECTION WITH PYTHON AND CASCADE TRAINER

Proposed Method	Traditional Method	Unit of Measure
70	96	%
15	233	min
570	1000	soles

Table 5 shows the comparison of precision, time and cost of the traditional method versus the proposed method. From these results the following is concluded:

Traditional Method (PCI):

- Accuracy: The Pavement Condition Index is 96% efficient for measuring pavement failures.
- Speed: The time taken to measure section 1 is 45 minutes, in the case of section 2 it took a total of 1 hour and 16 minutes, in the case of section 3 an approximate of 44 minutes was obtained, for the Section 4 took 27 minutes, in the case of section 5 it took approximately 18 minutes and for section 6 it took a total of 23 minutes to evaluate using the PCI method.
- Cost: For the total cost, expert engineers in PCI measurement were consulted. These experts gave an approximate budget of 1000 soles, including the equipment, professional staff and printed map of the area. Highlight that it was consulted for the evaluation of 215 linear meters of pavement.

Proposed method:

- Accuracy: The model has an accuracy of 70% because it detects several false positives. However, this can be improved by increasing the amount of images used in training, as well as using better equipment for data collection.
- Speed: The drone was rented for a period of 1 hour, but it took approximately 15 minutes to record the 6 sections studied at a height of 1.9 meters.
- Cost: The drone was rented, which includes the drone and the operator, at an approximate price of 270 new soles plus a soil engineer to verify the failures, which would cost approx. 300 new soles.

Comparing the traditional (PCI) method with the proposed method reveals significant differences in terms of accuracy, speed, and cost, each with important implications for pavement evaluation.

The PCI method demonstrates a remarkable accuracy of 96%, outperforming the proposed method that achieves only 70%. However, it is possible to improve this value through model refinement strategies. Despite this, the proposed method is 60% faster than PCI and offers significant cost savings, making it an attractive option for resource-limited pavement evaluations.

V. Conclusions

The algorithm used for the detection of functional failures in the pavement offers significant advantages in terms of cost and measurement time compared to the traditional method (PCI). The use of a drone and an operator for the recording, together with the corroboration of a soil engineer, results in a saving of 60% in the time of detecting functional failures since only the recording of the pavement that lasts approx. 15 minutes and with a total cost of 570 soles (270 soles for the drone and the operator + 300 soles for a soil engineer).

However, the accuracy of the proposed method, which reaches 70%, suggests areas for future improvements. It is possible that the lack of positive images during training, as well as the quality of the cameras used, may have contributed to this relatively low accuracy. Additionally, the lack of initial experience in object detection with Python may have been a limitation.

For future research, it is recommended to explore acquiring higher quality cameras and seeking additional training in object detection to improve the accuracy of the algorithm. These improvements could lead to more accurate and reliable results in the detection of functional failures in the pavement.

References

[1] Liu, S.-S., Arifin, M. F. A., & Chen, W. T. (2022). An Integrated Optimization Model for Life Cycle Pavement Maintenance Budgeting Problems. IEEE Access, 10, 13884-13900. Recovered from: https://ieeexplore.ieee.org/document/9695458

[2] Chen, W., Hu, G., Han, W., Zhang, X., Wei, J., Xu, X., & Yan, X. (2022). Research on the Quality of Asphalt Pavement Construction Based on Nondestructive Testing Technology. Coatings, 12, 379. Recovered from: https://doi.org/10.3390/coatings12030379.

[3] Leblouba, M., Tarabin, M., & Zahri, M. (2022). Probabilistic analysis and simulation of crack propagation in concrete pavements and surfaces. Scientific Reports, 12, 14157. Recovered from: https://doi.org/10.1038/s41598-022-18060-8.

[4] Liu, S.-S., Budiwirawan, A., Arifin, M. F. A., Chen, W. T., & Huang, Y.-H. (2021). Optimization Model for the Pavement Pothole Repair Problem Considering Consumable Resources. Symmetry, 13, 364. Recovered from: https://doi.org/10.3390/sym13030364.

[5] Sholevar, N., Golroo, A., & Roghani Esfahani, S. (2022). Machine learning techniques for pavement condition evaluation. Automation in Construction, 136, 104190. ISSN 0926-5805. Recovered from: https://doi.org/10.1016/j.autcon.2022.104190.

[6] Wang, Y., Yu, B., Zhang, X., & Liang, J. (2022). Automatic extraction and evaluation of pavement three-dimensional surface texture using laser scanning technology. Automation in Construction, 141, 104410. ISSN 0926-5805. Recovered from: https://doi.org/10.1016/j.autcon.2022.104410.

[7] Qureshi, W. S., Hassan, S. I., McKeever, S., Power, D., Mulry, B., Feighan, K., & O'Sullivan, D. (2022). An Exploration of Recent Intelligent Image Analysis Techniques for Visual Pavement Surface Condition Assessment. Sensors, 22, 9019. Recovered from: https://doi.org/10.3390/s22229019.

[8] Bučko, B., Lieskovská, E., Zábovská, K., & Zábovský, M. (2022). Computer Vision Based Pothole Detection under Challenging Conditions. Sensors, 22, 8878. Recovered from: https://doi.org/10.3390/s22228878.

[9] Mei, Q., & Gül, M. (2020). A cost-effective solution for pavement crack inspection using cameras and deep neural networks. Construction and Building Materials, 256(10), 119397. Recovered from: https://doi.org/10.1016/j.conbuildmat.2020.119397.

[10] Yang, Z., Ni, C., Li, L., Luo, W., & Qin, Y. (2022). Three-Stage Pavement Crack Localization and Segmentation Algorithm Based on Digital Image Processing and Deep Learning Techniques. Sensors, 22, 8459. Recovered from: https://doi.org/10.3390/s22218459.