AI-based technology to prognose and diagnose complex crack characteristics of railway concrete sleepers

Kaewunruen, Sakdirat; Adesope, Abdullah Abimbola ; Huang, Junhui; You, Ruilin; Li, Dan

DOI:
10.1007/s42452-024-05880-8

Document Version
Publisher's PDF, also known as Version of record

Citation for published version (Harvard):
Kaewunruen, S, Adesope, AA, Huang, J, You, R & Li, D 2024, 'AI-based technology to prognose and diagnose complex crack characteristics of railway concrete sleepers', *Discover Applied Sciences*, vol. 6, no. 5, 217. https://doi.org/10.1007/s42452-024-05880-8

Discover Applied Sciences

Research

AI-based technology to prognose and diagnose complex crack characteristics of railway concrete sleepers

Sakdirat Kaewunruen[1] · Abdullah Abimbola Adesope[2] · Junhui Huang[3] · Ruilin You[4] · Dan Li[5]

Received: 20 January 2024 / Accepted: 8 April 2024
Published online: 19 April 2024

Abstract
Railway concrete sleepers are key safety-critical components in ballasted railway tracks. Due to frequent high-intensity impact loadings from train-track interaction over irregularities together with hostile environmental conditions, complicated characteristics of various crack patterns can incur on railway concrete sleepers, which will decrease their durability and service life overtime. Early warning of those cracks can help railway engineers to plan and schedule for renewal and maintenance timely and effectively. This study thus explores the artificial intelligence application of YOLOv5OBB (YOLOv5 with Oriented Bounding Box output) in the identification and classification of cracks in railway sleepers into three distinct types: longitudinal, transverse, and inclined, based on their specific crack angles, which have not been investigated in the past. The identification of crack angles is the novelty of this study. Recognising the various types of cracks is critical, given their varying causes and degrees of severity. Current corrective maintenance methods pose considerable safety risks to workers and exhibit low efficiency, underscoring the need for a more autonomous and efficient solution. This study marks a significant stride towards revolutionising railway maintenance, evidenced by an impressive mAP (Mean Average Precision) of 0.72 for crack detection and a 92% accuracy rate for angle detection. These promising results substantiate our study's potential to pioneer advancements in railway infrastructure maintenance.

Keywords Railway concrete sleepers · Damage detection · Cracks · Machine learning · YOLO · Track maintenance

Abbreviations

YOLO	You Only Look Once
OBB	Orientated Bounding Box
HBB	Horizontal Bounding Box
mAP	Mean Average Precision
AP	Average Precision
GPU	Graphical Processing Unit
CPU	Central Processing Unit
CSL	Circular Smooth Label
IOU	Intersect Over Union
RAM	Random Access Memory

✉ Sakdirat Kaewunruen, s.kaewunruen@bham.ac.uk; Abdullah Abimbola Adesope, AXA1664@alumni.bham.ac.uk; Junhui Huang, JXH596@student.bham.ac.uk; Ruilin You, youruilin0731@126.com; Dan Li, danli1992@sit.edu.cn | [1]Department of Civil Engineering, School of Engineering, University of Birmingham, Edgbaston, Birmingham B15 2TT, UK. [2]Department of Mechanical Engineering, School of Engineering, University of Birmingham, Edgbaston, Birmingham B15 2TT, UK. [3]Birmingham Centre for Railway Research and Education, University of Birmingham, Edgbaston, Birmingham B15 2TT, UK. [4]Railway Engineering Institute, China Academy of Railway Sciences, Beijing 100082, China. [5]School of Urban Construction and Safety Engineering, Shanghai Institute of Technology, Shanghai 201418, China.

NMS Non-Max Suppression
CNN Convolutional Neural Network

1 Introduction

Globally, the dominant type of trackforms is still based on ballasted railway tracks since they are very cost-effective, environment-friendly, and very resilient. They have been installed and used for urban, suburban, metro, and freight rail networks. The sleeper-ballast track systems are commonly applied to cater rail operations up to a train speed around 250–270 km/h. It is well-known in rail industry practice that higher train speeds above the range will cause excessive ballast dilation and track settlement, requiring excessive track maintenance (such as excessive ballast tamping and resurfacing). The ballast track structure consists of various key components that guide and facilitate the safe, cost-effective, and smooth ride of trains. Railway sleepers are a critical element to redistribute train load to ballast and formation. The sleepers also play a key role in enhancing lateral track stability and maintaining the rail gauge. Among a number of material choices, prestressed concrete sleepers are the most common type of railway sleepers in use globally [1].

Railway sleepers have been a critical part of the railway track structure since they were first introduced in the nineteenth century. Their main functions include: the distribution of the dynamic track loads across the ballast; alignment of the track; maintaining track gauge; and the reduction of vibrations and stresses in the structure [2]. Initially, timber was used as a railway sleeper because of its simple assembly. However, due to its susceptibility to termite infestation, fungal decay, and repetitive damage from the use of, the then newly introduced, high-speed trains, their use was replaced with concrete sleepers [3]. Concrete has the required material properties to surpass the life span of timber by 30 years and to withstand the dynamic loads exerted on the track [4], and yet, due to the cycle of tension and compression of the upper face of the sleeper, cracks do occur over time. Efforts have been made to decrease the frequency at which these types of cracks occur through the latest development of prestressed concrete sleepers, consisting of steel rods running through their cross-section. Nevertheless, the occurrence of cracks in the concrete via various other mechanisms of "wear and tear", remains inevitable. As such, requisite track maintenance is performed to detect, monitor, and evaluate them.

There are a few typical damage forms, each with their respective causes as shown in Fig. 1. For the scope of this study, the following three main damage forms, occurring on the upper surface of the sleeper, will be at the focal point: transverse, longitudinal, and inclined, cracks.

(i) Transverse: these are cracks that form "perpendicular to the long axis of the concrete sleeper" and occur in 2 places: the rail seat, and the central section of the sleeper. Both locations are caused by the cycle of positive and negative bending moments, experienced during service. Narrow transverse cracks are commonly observed, with little-to-no adverse effects, but wider cracks can lead to failure of sleeper function [5].

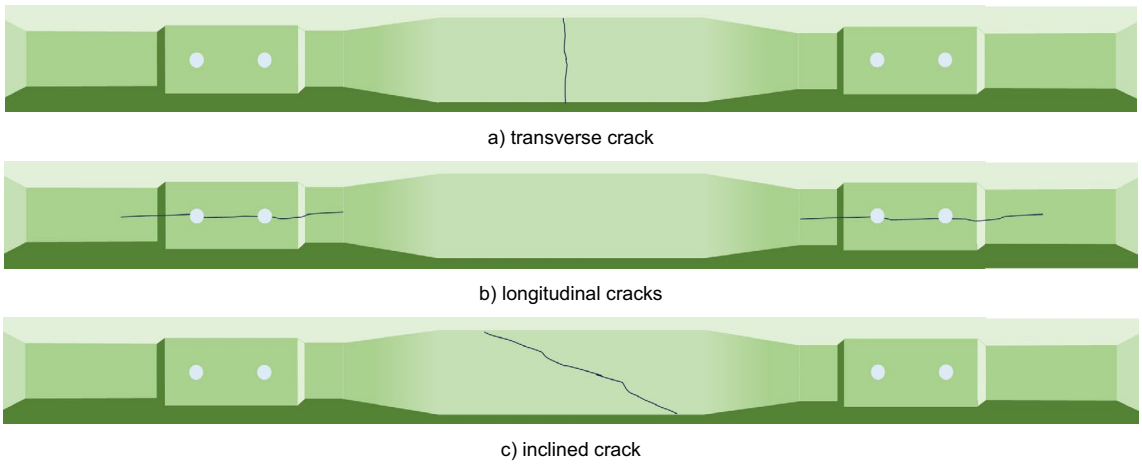


Fig. 1 Typical severe cracks in railway concrete sleepers

(ii) Longitudinal: longitudinal cracks are characterised as fissures that develop parallel to the long axis of the sleeper. In prestressed concrete sleepers, the steel reinforcement bars are pulled to high tensions, inducing stress. Over time, this stress results in longitudinal crack formation. Manifestation of these cracks greatly contributes to diminishing the strength of sleepers, thus compromising their ability to sustain the correct track gauge [5]. Therefore, addressing them becomes a high priority in the maintenance work.

(iii) Inclined: improper tampering of the ballast bed results in inclined cracks forming on the top face. The sooner these are detected, the sooner the problem can be rectified to prevent the crack from propagating further [5].

Current methods of detecting cracks involve the use of imageries obtained from tamping vehicles, visual inspection, and track inspection vehicle (see Fig. 2) for each sleeper via cameras. In practice, subsequent tasks, such as the measurement of crack length, width, and angle, are completed manually by workers, on-foot, which poses a safety risk (when working on live tracks) and has been proven to be timeconsuming [6]. Other similar methods include: Acoustic Emission [7], developed by Thompson et al. who used ultrasound to detect the location of the cracks; as well as the Infrared Thermography method [8] by which Sham et al. used light, and the reflected radiation from the cracks, to locate them. Both of these methods were limited by their inability to detect the quantitative location of the cracks or the crack angle that are incurred in reality.

There are similar research investigations into computer visions, which utilise machine learning for structural conditions of concrete structures, of which the most relevance includes Golding et al. [9], and Yu et al. [10]. Golding et al. [9] investigated the use of a pre-trained CNN (Convolutional Neural Network), the VGG16 architecture, to autonomously detect cracks in concrete structures. They modified the model using various image processing techniques, such as the Sobel filter, RGB, Greyscale and Otsu method, in order to determine which produced the best results. Their results were impressive, with a detection accuracy of greater than 98% for all the techniques used. In addition, an image segmentation visual output was used which made it easier to evaluate the detected cracks. Yu et al. [10] uses the YOLO V5s deep learning algorithm to also detect cracks in concrete. YOLO (You Only Look Once) is an algorithm that splits the image into grids and scans the image in one pass, outputting results faster than alternative CNNs. The algorithm produced results of 84.37% average precision (AP) with the visual output being horizontal bounding boxes (HBB) around the detected cracks [11]. Although these projects are similar to that discussed in this study, in terms of the methodology, the absence of the output of the crack angle from both projects, results in them facing the same limitation in terms of their inability to classify the cracks by type.

This study aims to develop a new technique capable of detecting and classifying complex crack characteristics that can be observed on the railway concrete sleepers. The key research objectives are (i) to utilise the YOLOv5OBB (YOLOv5 with orientated bounding box output) model to detect the three main types of cracks that occur on the top face of railway sleepers; (ii) to determine the crack angle with respect to the long axis of the sleeper; and (iii) to classify the crack type; longitudinal, transverse, or inclined crack depending on the calculated crack angle. The oriented bounding box model, as shown in Fig. 3, is the novelty of this study for railway concrete sleepers. The insights derived from this study

Fig. 2 Track Inspection Vehicle (so-called 'AK Car' in Australia), fully equipped with axle box accelerometers, laser sensors (for track geometry), 360° laser scanners (for clearance and transit space), gyroscopes, cameras (front, rear and on track surface), GPS, inertia, impact force detection, cord-based and corrugation systems. In addition to track geometric parameters and other sensing data, the inspection vehicle provides imageries of track surface including condition of rails, sleepers, fastenings and ballast. Courtesy: Sydney Trains, Transport for NSW (photo taken by Kaewunruen)

will help rail track engineers by offering new alternative solutions to enable the predictive and preventative maintenance of railway track infrastructures.

2 Materials and methods

2.1 Dataset

Collecting a comprehensive dataset required for this study proved to be quite difficult, with many aspects to consider, such as: ascertaining the position, orientation, and height at which the images would be captured; procurement of a sufficient number of sleepers with surfaces that exhibited structural fissures; as well as the safety and legal aspects of capturing the images on live train tracks (e.g. carrying the possibility of a £1,000 fine) [12]. The most straightforward strategy for this study would have been to capture images directly from an operational railway track. A request for the desired images was initially made through the FOI (Freedom of Information) department of the railway maintenance team. Unfortunately, this avenue resulted in an extremely small amount of data, of precisely 10 images, all of which were taken from non-specific angles, and at varying distances from the sleepers. With the insufficient quantities, and inconsistencies in the quality of the dataset, another roadblock was encountered.

The solution to all this was for the images to be personally captured in a controlled full-scale laboratory environment. A full-scale sleeper of dimensions 2.5 m by 0.22 m by 0.23 m, was sourced. An iPhone XR camera with a 12MP wide camera was used to capture all comprehensive 121 images, outputting them as 4032×3024 pixels. A camera stand, securely suspending the iPhone directly above the sleeper, was positioned to maintain a constant height of 1.9 m, from which the images were taken, thus enhancing the reliability and accuracy of the generated dataset. Artificial cracks using fabricated on the sleepers were used and mounted in arbitrary locations and orientations across the top face of the sleeper as shown in Fig. 4. In doing this, the aim was to try and emulate the random nature by which cracks form on sleepers that are in active service, in order to reduce the amount of bias from this chosen method of data acquisition. Cracks of varying widths, with the smallest being 1 mm, were fabricated to produce a comprehensive dataset. The decision for the smallest width to be set at 1 mm was for it to align with the minimum threshold width for concerning cracks, thus ensuring satisfactory practical function of the software with regards to its application in railway maintenance [13]. In practice, the crack will propagate much further prior to failure. The detection of this crack size is applicable to practical track inspection and monitoring. Note that our AI-based technology developed can actually detect smaller cracks if needed.

ROBOFLOW is the software used in this study; a modern tool, commonly used in machine learning as shown in Fig. 5. It expedites the process of annotating and labelling sets of images with their respective class labels. Furthermore, its collaboration function allows for a team to be created, and for images to be split and allocated to the different members, resulting in increased efficiency of the image labelling process. The procedures adhered to, in the course of employing ROBOFLOW for this study, are systematically implemented in the following order:

1. Importing the dataset onto ROBOFLOW
2. Splitting the dataset between the created team
3. Maintaining a consistent zoom of 350% for each image
4. Locating and carefully drawing around the cracks in each image using the 'Polygon tool'

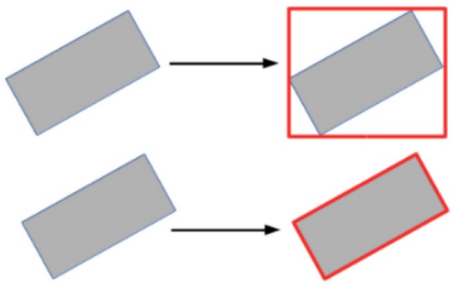
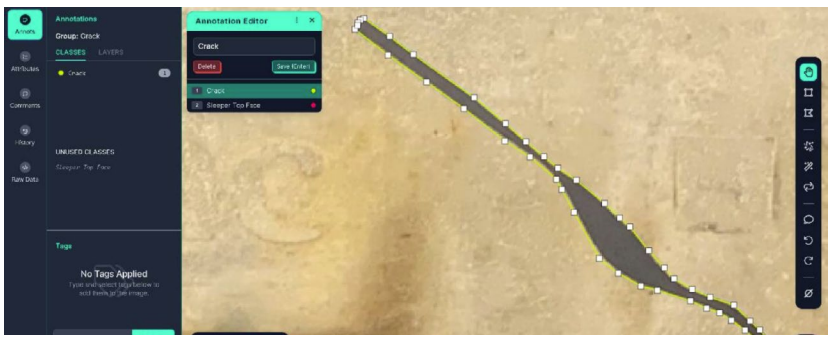


Fig. 3 Difference between a horizontal bounding box (top) and an orientated bounding box (bottom). Traditional YOLO algorithm is based on the horizontal bounding box, which cannot properly quantify the maximum crack width. This study has modified the algorithm to re-orient the box frame to align with the crack pattern, enhancing the quantification of diagonal crack width

a) Inclined cracks

b) transverse cracks

Fig. 4 Example of fabricated cracks on railway concrete sleepers in laboratory environment



a) zoom-in localised crack identification

b) global crack identification

Fig. 5 Schematic illustration of crack identification using ROBOFLOW Interface

5. Labelling the polygons with the class name, 'Crack'
6. Preprocessing and augmenting the dataset
7. Splitting the dataset into, Training, Validation, and Testing
8. Generating and exporting the new increased dataset in the YOLOv5OBB.txt format

The format used for YOLOv5OBB consists of ten parameters: the four boundary point coordinates (×1, y1, ×2, y2, ×3, y3, ×4, y4) of the OBB (Orientated Bounding Box) around the detection; the class label for the bonding box; and the visual difficulty in seeing the detected object, as shown in Fig. 5. This is listed in a.txt file for each image, with each of the lines representing instances within the image [14].

For the preprocessing step, various image resize options were experimented with, but 1280 × 1280 pixels was the optimum size that was able to detect the minute cracks in the image. White borders were used as padding, to resolve the fact that the input images were not square. Imitation of the irregularities observed with the cracks on railway sleepers in real-life scenarios, was achieved via the implementation of various augmentation techniques on the dataset. The imagery implementation as demonstrated in Fig. 5 includes (i) 5% noise—imitates the possible rocks or dirt that could be present around the cracks; and (ii) 25% lighter and darker—considers the different lighting environments and times of day. With the addition of these augmentations, the total number of images within the new dataset increased from 121 to 283. A training, validation and testing split of 87%, 8% and 5% respectively, was used on the dataset. The split was training heavy as it was a custom dataset and the object in focus was very small in comparison to the image size. Note that the industry data was also fused into the data pool to improve the data integrity and AI robustness.

2.2 Crack detection

YOLO (You Only Look Once) integrates a single neural network that has the capacity to predict bounding boxes and assess class probabilities directly from images in one evaluation. This innovative technique offers a more efficient and direct approach to image analysis [10]. A neural network is a combination of multiple convolutional layers, with each layer processing and learning information from the input image (i.e. w × h × 3—image size multiplied by the three colours of an image, RGB) [15]. Each layer then feeds this information to the subsequent layer, resulting in an output layer with the prediction. The YOLO model is trained to detect an object within an image and proceeds to output the image with a bounding box around the object of interest, along with its probability, and class name. To prevent the drawing of multiple boxes around the same object, NMS (Non-Max Suppression) is used. This selects the bounding box with the highest probability and discards the remaining bounding boxes with an IOU (Intersect Over Union) less than a set threshold value. The bounding box with the next highest probability is then selected and the process is repeated.

Image Segmentation is a sophisticated technique used to group objects in an image into categories that are homogeneous [16]—the two main types being semantic and instance segmentation. The former splits every aspect of the image into groups of broad categories, whilst the latter splits the image based on each instance a category is detected. As YOLOv8 was the latest version of YOLO at the time of this study, it was trialled on the collected dataset with the instance segmentation function; one difference, in comparison to the use of YOLOv5OBB, was the change in export format from ROBOFLOW. This method outputted outstanding results for the detection of the cracks in the image, and it was able to highlight the locations of the cracks and print HBB (Horizontal Bounding Box) around them. However, it could not predict the quantifiable angle of the crack, and hence could not automatically predict the crack type, limiting its capability to fully meet the aims of this study.

YOLOv5OBB is a modified YOLOv5 repository designed by a GitHub user, hukaixuan19970627 [14]. It embodies the use of CSL (Circular Smooth Label) as the OBB detection model which assesses images from a classification point of view, rather than a regression point, to produce higher precision in the predicted OBB output [17]. YOLOv5 resembles the YOLOv4 version. However, the structural change of the YOLOv5 model resulted in better detection results. YOLOv4 is structured with a backbone of CSPDarknet53, a neck of SPP block which separates the significant features of the image, PANet, and a YOLOv3 head. Whilst Yolov5 has its own head and backbone as seen in the model.yaml file from Ultralytics [18], with another difference between the two versions being the use of the PyTorch framework for YOLOv5, and the Darknet framework for YOLOv4 [19].

Even with it being a slightly older version of YOLO compared to YOLOv8, YOLOv5OBB provides the OBB output in the format (×1, y1, ×2, y2, ×3, y3, ×4, y4) for the four boundary points (as shown in Fig. 6), the class category, and the confidence score. The boundary point values were then extracted from YOLO and inputted into a code, to enable the output of the crack angle (see Sect. 2.3).

The YOLOv5OBB model was trained using the collected dataset. It undertook an initial run on Google Colab, which had free Tesla T4 GPU (Graphical Processing Unit). The role of the GPU being to speed up the processing time of the model

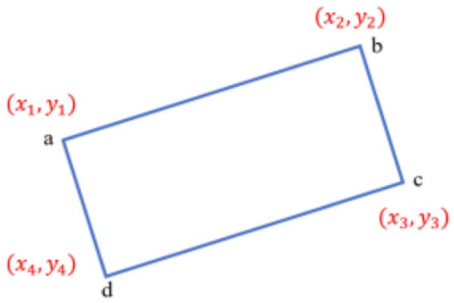


Fig. 6 Orientated bounding box with the boundary point coordinates

[20]. An additional advantage gained with the use of Google Colab was the large 12 GB cloud RAM and 78 GB cloud disk space. This enabled the training with larger batch sizes, and a decrease in training time compared to using a CPU alone with less RAM and disk space. A challenge faced in the process of collecting results was caused by the upgrade of Google Colab's Python from version "3.9" to "3.10". This prevented the YOLOv5OBB model from running, due to the incompatibility of PyTorch regarding the operation of the upgraded Python software. This resulted in the model being ran locally on a 4 GB RAM, CPU-only Laptop, using a Python IDE (Integrated Development Environment), PyCharm. PyCharm provided the luxury of being able to download specific versions of Python and the required libraries to match the requirements stated in the Yolov5OBB GitHub repository [14]. The Dataset was organised into the right folders, as per the repository's requirements, to enable a successful training process. The model was then trained on the dataset, with an initial batch size (images per training iteration) of 4, and an epoch (number of training iterations) of 20. After a training time of 30 min, zero detections were made in the validation step. In response to this, the batch size was increased to 16, however, the program crashed due to the small amount of RAM that was available. Trailling multiple combinations of batch sizes and epochs eventually brought about the final values of 2 and 400, respectively. This large number of epochs allowed for the model to be trained thoroughly at the expense of a training time of 18 h, which was due to the small 4 GB RAM and the absence of a GPU.

2.3 Crack type and angel detection

The main aim of the study has been consequently executed employing a unique code, authored specifically for this study's purposes. It involved the use of two Python libraries, namely math and cv2, from OpenCV to: import the OBB coordinates from YOLOv5; calculate the angle of the bounding box, with respect to the 'y = 0' of the image frame; and output the crack type prediction, based on an angle threshold value of ten degrees.

Extracting the bounding box coordinates from the YOLOv5OBB output was accomplished using the code in Appendix A. The code imported the first 8 values of the YOLOv5OBB format into an empty array named "Coordinates". These 8 values of the "Coordinate" array (bounding box coordinates) were subsequently assigned to one of the following boundary points coordinates: "×1", "y1", "×2", "y2", "×3", "y3", "×4", "y4", to be used in the calculations explained below.

The main body and function of the code can calculate the length of each side of the bounding box, using a rearranged version of Pythagoras' theorem: $a^2 + b^2 = c^2$ where the values of a and b were calculated using the value of absolute change in the x and y values of the coordinates respectively, and the value of c being the gradient length. The length of each boundary point pair was calculated clockwise and stored as "L12" (length between points 1 and 2), "L23" (length between points 2 and 3), etc. Trigonometry, particularly the inverse tangent function, was then used to calculate the angle of each side, with respect to the y = 0 axis (the x-axis of the image frame) which was then stored as "CA12" (crack angle of the side formed by the boundary point pair of points 1 and 2), "CA13", etc. for each boundary point pair.

The angle between an inclined line and the x-axis using arc-tangent can be determined by:

$$angle = tan^{-1}\left(\left|\frac{\Delta y}{\Delta x}\right|\right) \quad (1)$$

The longest side of the bounding box was used to determine the angle of the crack, as this was taken to be the main chord length of the crack. Using 'if statements', the angle of the bounding point pair with the longest side was used to define the "CrackAngle" in degrees.

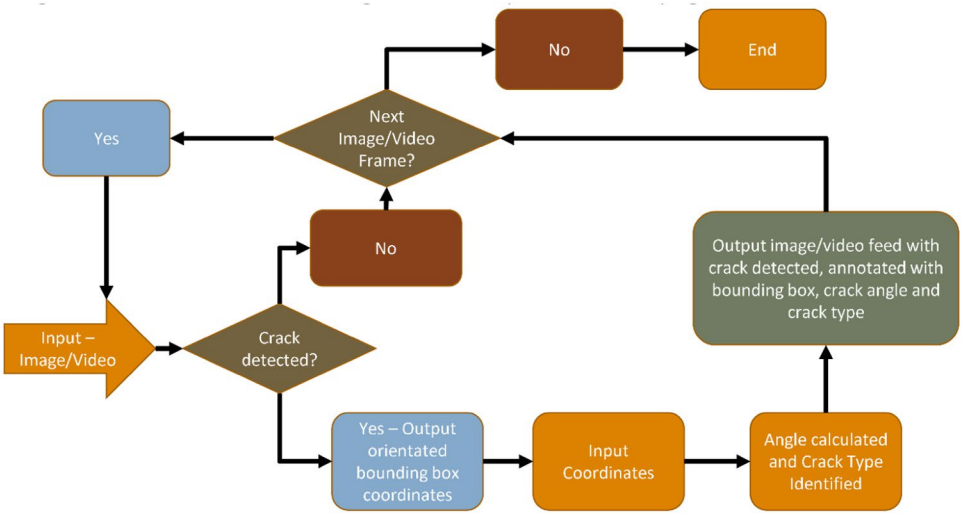


Fig. 7 Flow Chart depicting the processes of the customised algorithm in this study

Table 1 Comparison of the prediction results for YOLOv8 and YOLOv5OBB trained with the same dataset

Algorithm	Batch size	Epochs	Training time (hr)	mAP	Angle detection accuracy
YOLOv8	16	150	0.75	0.995	99%
YOLOv5OBB	2	400	18	0.72	92%

Classification of the crack can be completed using an 'if statement' and outputted as "CrackType". The longitudinal and transverse crack types were each given a 10 degrees range within which their crack angle could fall. The range of crack angles within which cracks would be classified as longitudinal was 0–10 degrees (including), with the range for transverse cracks being 80–90 degrees (including), and inclined cracks being anything else. Because the crack angle detection function output angles in the range of 0–90 degrees (including), inclined cracks were classified as such, if their "CrackAngle" fell within the range of 10–80 degrees (excluding). The bounding box coordinates were then drawn onto the image file using the cv2 library, with the "CrackAngle" and "CrackType" text printed on the image file at a set distance from the first corner of the bounding box.

2.4 Flow diagram of the algorithm

The diagram in Fig. 7 represents the process involved in this study's algorithm, beginning at the "Input–Image" arrow, on the left, and concluding and the "End" process, in the top right. We have developed a unique code to modify YOLO algorithm as shown in Appendix. Note that, according to EU' AI trustworthiness framework, AI robustness requires a machine learning system to perform without failure even when test examples are slightly off the training distribution. This flowchart thus demonstrates the capability to detect any adversary data; and then we can flag the adversary data and report it to experts for further review. This action will enable experts (e.g. rail engineers, inspectors, maintainers) to manage and override the data disagreement (e.g. when an imagery is fouled or when the sensor is broken in reality). In addition to extensive data testing and model validation, this process significantly improves the trustworthiness and robustness of our AI technology by engaging with real-world experts.

3 Results

Commonly, object detection models are evaluated, based on their accuracy and performance, by a metric termed mAP (Mean Average Precision) or nowadays, generally referred to as AP (Average Precision). This is done in the validation step and is performed by comparing the values of the predicted bounding boxes to the ground truth box coordinates using

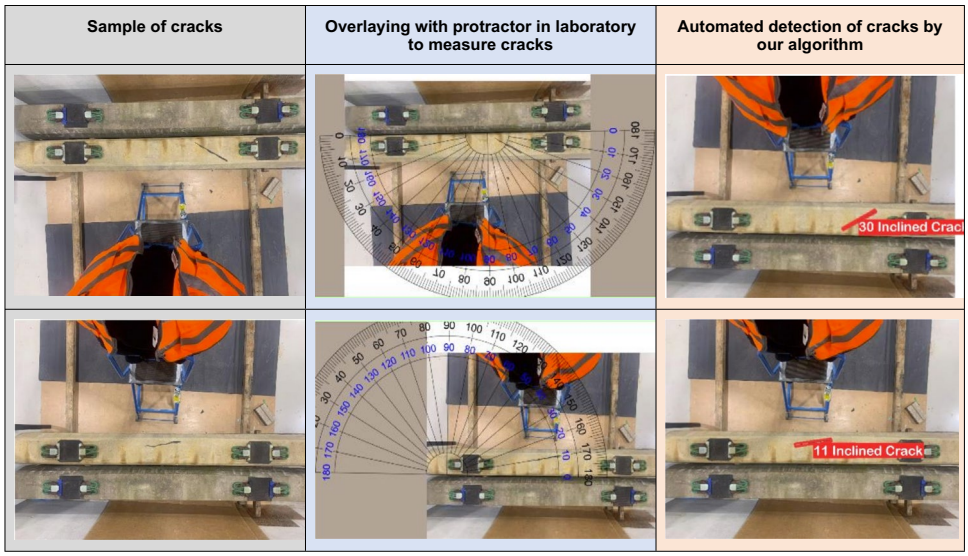


Fig. 8 Flow of the automation processes to detect and classify cracks on railway concrete sleepers using YOLOv5OBB

the IOU (Intersect Over Union). The IOU is calculated by how much of the predicted bounding box overlaps with the total area of the predicted and ground truth bounding box. A threshold IOU is set and applied, to filter out predictions that would count as false positives. Table 1 shows the comparison of prediction results between the data-driven methods adopted in this study including YOLOv8 and YOLOv5OBB.

As shown in Table 1, the crack detection aspect of this study's algorithm produced a mAP value of 0.72, after being trained on 400 epochs with a batch size of 2. To determine the accuracy of the angle detection, which depended on the predicted OBB, an online protractor tool (modified from [21]) was used to measure the actual crack angle with respect to the sleeper. This has been done for the images in the validation set and an accuracy of 92% was achieved. However, this does not include the images with false positives. Figure 8 demonstrates automated detection of cracks on railway concrete sleepers. The samples of cracks are shown on the left. Their respective input images have been overlayed with protractor manually to identify and estimate cracks (shown in the middle). The results of the automated crack detection are illustrated in the right column. The alignment of crack derived from the algorithm is in excellent agreement with the measurements.

For comparison, we have adopted and modified YOLOv8 to improve the crack detection system. In our study, YOLOv8 has also been used to train the dataset, but for the sole purpose of performing crack detection along with instance segmentation. The model has been trained for multiple epochs to batch size ratio, with the arrival at an optimum ratio of 150 epochs to a batch size of 16 (see Table 1 for detailed comparison of algorithmic performance). This task has been conducted on Google Colab and taken a total of 45 min to train. This time resource requirement has reduced significantly using this new technique.

Interestingly, the results produced by YOLOv8 can be illustrated in Fig. 9 (left column is the original samples; the middle column is the manual measurement; and the right column is the output from YOLOv8). Surprisingly, after the training and validation process, the accuracy of testing is close to perfect, with a mAP of 0.995. The result comparison for solely object detection can be seen in Table 1 where YOLOv8 can be seen to outperform YOLOv5OBB in both precision, and training time. It is noted that YOLOv8 facilitates the oriented frame, enabling the crack path to be defined correctly (nearly 100%). This performance is very promising for applications in practice.

4 Discussion

As aforementioned, the results in Table 1 clearly support the findings that YOLOv8 outperforms YOLOv5OBB in all aspects, apart from the aspect of the main objective of this study. YOLOv8 is able to successfully detect cracks on the concrete sleepers with an mAP of 0.995 compared to YOLOV5OBB's mAP of 0.72. Based on [19], the architectural changes of YOLOv8, when compared to the YOLOv5 model, could have resulted in the higher precision of crack detection observed.

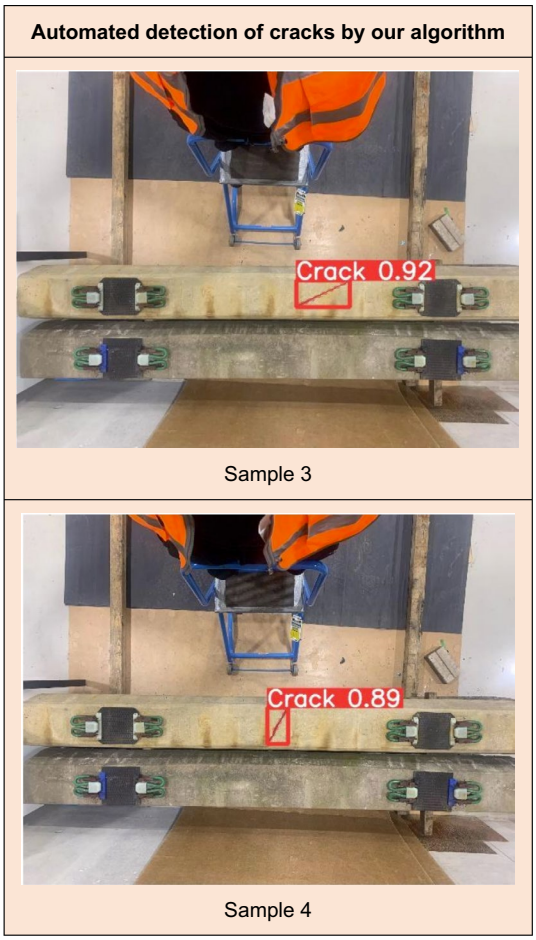


Fig. 9 Detection of cracks on railway concrete sleepers using YOLOv8

The YOLOv8 results (in Figs. 9) demonstrate the algorithm output as a segmentation with a HBB (Horizontal Bounding Box), hence the type of crack could not be predicted using the code. It can however be detected visually, which does improve the current methods of railway sleeper maintenance, as discussed in Mohammed et al.'s study [6], as it eliminates the manual work involved in the initial process of detecting cracks. Nevertheless, analysis of the results from the detection process would still require manual input, for cracks to be classified by type.

The training time used for YOLOv8 is significantly less than that of YOLOv5OBB. This could be due to several factors, one of which being the initial use of Google Colab to train the former. The added GPU issued an advantage, as the former model was capable of training within a much shorter time period in comparison to the latter model, which was trained using a CPU. Despite this, the YOLOv5OBB model was able to achieve crack detection with an impressive mAP value of 72% (0.72) compared to previous work done by Yu et al. [11] with mAP of 84.37%, demonstrating its suitability for use as an initial crack detection model, for this study's algorithm. Even so, all algorithms and models have limitations and possible areas of improvement, which could result in greater precision.

During the image annotation phase, potential discrepancies in annotations may have served as a pathway to lower performance metrics. The annotations for this study were performed by a team who were not provided set annotation instructions. Consequently, the decisions to include shadows, tightness of the polygons, etc., were left to the annotators' individual discretion, possibly resulting in differences in annotations. Such inconsistencies may have resulted in ambiguity during the model's training phase, potentially hindering optimal learning and performance. Establishing guidelines for a benchmark annotation, thereby ensuring a consistent and reliable standard for annotations, would have been sufficient to address this.

Focusing on the crack detection section of this study's algorithm, a dataset containing only fabricated cracks was used during the training of the model. This could hinder the accuracy of the model in detecting cracks when applied to real-world scenarios involving actual cracks. Using information sourced from a larger dataset would improve the model's ability to learn different crack location scenarios, improving its predictions, thus increasing its mAP. Whilst considering

efforts being made to explore avenues to obtain images of actual sleepers with cracks to no avail, there is the possibility that this study could be taken up by the railway infrastructure managers, who would be able to utilise a larger dataset, based entirely on real-life scenarios, with their available resources.

With regards the angle detection section, its accuracy depended on a complete parallel existing between the camera's image frame and the top face of the sleeper in focus. In the event of a shift in the angle between the image frame and the upper face of the sleeper, the resulting calculated angle would be an inaccurate representation of the actual crack angle. This could lead to an error in the classification of the crack. Continuing with the discussion on the angle detection section, this study's algorithm detects the angle of the crack in reference to the x-axis of the image. In the scenario whereby the x-axis does not align parallel to the long axis of the sleeper, the output angle would be inaccurately calculated, thus opening up to the greater probability of a flawed crack classification.

The YOLOv5OBB is a modification of the YOLOv5 model, differing in the integration of the CSL (Circular Smooth Label) function, to the former, which enabled output of the OBBs. A suggested improvement to the concept of this study would be the implementation of the YOLOv8 model as the base model for the OBB (Orientated Bounding Box), potentially resulting in better performance metrics.

As initially expressed, the primary aim of this study was to develop an innovative crack detection and classification algorithm. The goal was to transform the existing track maintenance procedures; to introduce a new level of efficiency and precision in railway infrastructure management. Therefore, possible future applications of this algorithm were explored, namely the use of drones or a mini autonomous rolling stock. Singularly manned drones could be mounted with a camera, tilt sensors, and altitude sensors. These would be flown above the track, ensuring that the camera frame was parallel to the upper face of the sleepers, with the height above the sleepers maintained by the pilot. The images taken from the drone would be ran through this study's algorithm, and the results would be outputted. As a result, the current crack detection phase of the sleeper maintenance procedure would shift from requiring an entire team of workers, to being completed by a single pilot.

A self-driving rolling stock with a camera affixed at a set height, and adjusted so that the captured image frame meets the requirements of this study's algorithm, would be deployed onto the railway track. A different object detection algorithm, YOLO (You Only Look Once) could be used again, would be used to detect the presence of a sleeper at a specific location in the image frame. This would then trigger the camera to take a picture of the sleeper in view, and the image would be fed into this study's algorithm to output the results. To further elaborate on how the sleeper detection would operate in practice, YOLOv8 has an integrated object tracking function. This function could be used to track the movement of a sleeper as it crosses a specific line drawn onto the image frame, triggering the camera to take a picture as the long edge of the sleeper aligns with this line. For effective application of this method, the speed of the rolling stock would have to be controlled, to ensure that the correct balance between the production of a clear image, and the efficiency of the process, is achieved.

5 Conclusions

This study has embarked on the task of pioneering a novel approach towards railway track maintenance procedures, through the creation of an innovative complex crack detection and classification algorithm, with the use of YOLOv5OBB integrated with a trigonometry-based code, and YOLOv8. YOLOv5OBB is a very unique model modified to enable the capability to predict OBBs for objects of interest, from an image input. Comprehensive full-scale datasets have been collected and validated with detailed measurements. The datasets have been fused between full-scale laboratory and field measurements to improve data integrity. AI robustness has been fully demonstrated by engaging the AI system with real-world experts. Producing promising results of a 0.72 mAP (mean average precision) and a 92% angle detection accuracy, places this study into consideration for the future of improved railway maintenance. In comparison, the OBB technique has been embedded in a new version YOLOv8. Interestingly, YOLOv8 can outperform YOLOv5OBB in terms of speed, resource consumption and accuracy. YOLOv8 enables an alternative crack detention with mAP of 0.995. This is very promising for applications in practice. The use of automated crack detection in railway concrete sleepers can help rail and track engineers to enable predictive and preventative track maintenance. The early warning results can help the engineers to plan better for value-based asset management practices.

Acknowledgements The authors wish to gratefully acknowledge the Japan Society for Promotion of Science (JSPS) for JSPS Invitation Research Fellowship (long-term), Grant No. L15701, at the Track Dynamics Laboratory, Railway Technical Research Institute and at Concrete Laboratory, the University of Tokyo, Tokyo, Japan. The JSPS financially supported this work as part of the research project entitled "Smart and reliable railway infra-structure". Special thanks are given to the European Commission for H2020-MSCA-RISE Project No. 691135 "RISEN: Rail Infrastructure Systems Engineering Network". Partial support from and by European Commission and UKRI Engineering and Physical Science Research Council (EPSRC) for the financial sponsorship of Re4Rail project (Grant No. EP/ Y015401/1) is acknowledged. In addition, the sponsorships and assistance from PCAT, Transport for London (TfL), China Academy of Railway Science, and RSSB (Rail Safety and Standard Board, UK) are highly appreciated.

Author contributions Conceptualisation, S.K., A.A.A., J.H., R.Y. and D.L.; methodology, S.K., A.A.A., J.H., R.Y. and D.L.; software, S.K. and A.A.A.; validation, S.K., A.A.A. and J.H.; formal analysis, S.K., A.A.A., and J.H..; investigation, S.K., A.A.A., J.H., R.Y. and D.L.; resources, S.K.; data curation, A.A.A.; writing—original draft preparation, A.A.A.; writing—review and editing, S.K. and J.H.; visualisation, J.H.; supervision, S.K.; project administration, S.K.; funding acquisition, S.K. All authors have read and agreed to the published version of the manuscript.

Funding This research was funded by H2020 Marie Curie Action, grant number 691135; and by European Commission and UKRI Engineering and Physical Science Research Council (EPSRC) for the financial sponsorship of Re4Rail project (Grant No. EP/ Y015401/1). The APC has been kindly sponsored by the University of Birmingham's Open Access Fund.

Data availability Data can be made available upon a reasonable request.

Declarations

Competing interests The authors declare no competing interests.

Appendix A: Angle detection and classification algorithm

A1: Extraction

```
#open the txt file with the predictions and read in the first 8 values into an array for crack.
with open('file.txt', 'r') as file:
coordinates = [list(map(float, line.split()[:8])) for line in file]
import math


# Assign the values from the array to the variables using indexing
x1 = coordinates[0]
y1 = coordiantes[1]
x2 = coordinates[2]
y2 = coordinates[3]
x3 = coordinates[4]
y3 = coordinates[5]
x4 = coordinates[6]
y4 = coordinates[7]
```

A2: Angle detection

```
# Lengths of the sides of the bounding box, clockwise

L12 = math.sqrt(((abs(x1 - x2))**2) + (((abs(y1 - y2))**2)))

L23 = math.sqrt(((abs(x2 - x3))**2) + (((abs(y2 - y3))**2)))

L34 = math.sqrt(((abs(x3 - x4))**2) + (((abs(y3 - y4))**2)))

L41 = math.sqrt(((abs(x4 - x1))**2) + (((abs(y4 - y1))**2)))

# Angle of the corresponding sides of the bounding box with respect to y=0
```

```
CA12 = math.degrees(math.atan(abs(y1 - y2) / abs(x1 - x2)))

CA23 = math.degrees(math.atan(abs(y2 - y3) / abs(x2 - x3)))

CA34 = math.degrees(math.atan(abs(y3 - y4) / abs(x3 - x4)))

CA41 = math.degrees(math.atan(abs(y4 - y1) / abs(x4 - x1)))

# Calculate crack angle if L12 > L23 and L12 > L34 and L12 > L41:

CrackAngle = CA12

elif L23 > L12 and L23 > L34 and L23 > L41:

CrackAngle = CA23

elif L34 > L12 and L34 > L23 and L34 > L41:

CrackAngle = CA34

else:

CrackAngle = CA41
```

A3: Classification

```
# Define Crack Type depending on the crack angle

if CrackAngle > (90 - 10):

CrackType = 'Transverse'

elif CrackAngle < (0 + 10):

CrackType = 'Longitudinal'

else:

CrackType = 'Inclined Crack'
```

A4: Display

```
import cv2

# Load the image from the file

image = cv2.imread('image.jpg')

# Define the bounding box coordinates

bounding_box = coordinates

# Draw the bounding box

for i in range(4):

cv2.line(image, bounding_box[i], bounding_box[(i + 1) % 4], (0, 255, 0), 2)

# Define the text to display

angle_text = f"Crack Angle: {CrackAngle:.2f}"

type_text = f"Crack Type: {CrackType}"

# Choose the text position

text_position = (int(x1), int(y1) - 20)

# Draw the crack angle and crack type text on the image

cv2.putText(image, angle_text, text_position, cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 1)

cv2.putText(image, type_text, (text_position[0], text_position[1] - 20), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 1)

# Show the image with the bounding box, crack angle, and crack type

cv2.imshow('Crack Analysis', image)

cv2.waitKey(0) cv2.destroyAllWindows()
```

References

1. Kaewunruen S, Sresakoolchai J, Huang J, Zhu Y, Ngamkhanong C, Remennikov AM. Machine learning based design of railway prestressed concrete sleepers. Appl Sci. 2022;12:10311. https://doi.org/10.3390/app122010311.

2. You R, Li D, Ngamkhanong C, Janeliukstis R, Kaewunruen S. Fatigue life assessment method for prestressed concrete sleepers. Front Built Environ. 2017;3:68. https://doi.org/10.3389/fbuil.2017.00068.
3. Jabu MA, Alugongo AA, Maube O, Nkomo NZ. A review of the effectiveness of different types. Int J Eng Trends Technol. 2021;69(10):193–9.
4. Kaewunruen S, Fu H, Ye C. Numerical studies to evaluate crack propagation behaviour of prestressed concrete railway sleepers. Eng Fail Anal. 2022;131: 105888. https://doi.org/10.1016/j.engfailanal.2021.105888.
5. You R, Wang J, Ning N, Wang M, Zhang J. The typical damage form and mechanism of a railway prestressed concrete sleeper. Modeling and Analysis of Damage and Failure of ConcreteLike, Brittle and Quasi-brittle Materials. 2022.
6. Mohammed HA. Practical methods for measuring crack depth in concrete members in site. 2019.
7. Thompson R, Buck O, Spittzig W. A New ultrasonic methods for measuring deformation and fracture related material states. Metall Mater Trans. 1989;20A:611–8.
8. Sham FC, Chen N, Long L. Flash thermography. Surface crack detection by flash thermography on concrete surface. 2008;50:240–3.
9. Golding VP, Gharineiat Z, Munawar HS, Ullah F, Crack detection in concrete structures using deep learning. In: Sustainable construction management and computer simulation; 2022.
10. Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 779–788;2016.
11. Yu Z. YOLO V5s-based deep learning approach for concrete cracks detection. In: SHS Web of Confrences; 2022.
12. Network Rail. Warning—It is a criminal offence to trespass on the railway. Network Rail, 16 Jun 2004. [Online]. https://www.networkrailmediacentre.co.uk/news/warning-it-is-acriminal-offence-to-trespass-on-the-railway. Accessed 16 Jan 2023.
13. Lay S. Lateral seperation cracks in concrete track slab. Eur Rail Rev. 2005;11(2).
14. hukaixuan19970627. Github. 17 July 2022. [Online]. https://github.com/hukaixuan19970627/yolov5_obb/blob/master/docs/GetStart.md. Accessed 27 January 2023.
15. Nielsen M. Neural Networks and Deep Learning. San Francisco: Determination Press; 2015.
16. Chakraborty D, Sen GK, Hazra S. Image Segmentation Techniques, Saarbruken. Deutchland: LAP LAMBERT Academic Publishing; 2012.
17. Yang X, Yan J. Arbitrary—oriented object detection with circular smooth label. Comput Vis ECCV. 2020;2020:677–94.
18. Jocher G. YOLOv5 Backbone. 1 July 2022. [Online]. https://github.com/ultralytics/yolov5/issues/8430. Accessed 12 Apr 2023.
19. Terven JR, Cordova-Esparaza DM. A comprehensive review of yolo: from yolov1 to yolov8 and beyond. Under review in ACM computing surveys, 2 Aug 2023.
20. Ramani S, Desai V, Karia K. GPU-graphics processing unit. Int J Innov Res Comput Sci Technol (IJIRCST). 2015;3(1).
21. Ginifab, Online Protractor. [Online]. https://www.ginifab.com/feeds/angle_measurement/. Accessed 1 May 2023.