Research Article

Piotr Bojarczak* and Piotr Lesiak

UAVs in rail damage image diagnostics supported by deep-learning networks

https://doi.org/10.1515/eng-2021-0033
Received Oct 21, 2020; accepted Jan 03, 2021

Abstract: The article uses images from Unmanned Aerial Vehicles (UAVs) for rail diagnostics. The main advantage of such a solution compared to traditional surveys performed with measuring vehicles is the elimination of decreased train traffic. The authors, in the study, limited themselves to the diagnosis of hazardous split defects in rails. An algorithm has been proposed to detect them with an efficiency rate of about 81% for defects not less than 6.9% of the rail head width. It uses the FCN-8 deep-learning network, implemented in the Tensorflow environment, to extract the railhead by image segmentation. Using this type of network for segmentation increases the resistance of the algorithm to changes in the recorded rail image brightness. This is of fundamental importance in the case of variable conditions for image recording by UAVs. The detection of these defects in the rail head is performed using an algorithm in the Python language and the OpenCV library. To locate the defect, it uses the contour of a separate rail head together with a rectangle circumscribed around it. The use of UAVs together with artificial intelligence to detect split defects is an important element of novelty presented in this work.

Keywords: Unmanned Aerial Vehicles, Split Defect in Rail, Deep-Learning Networks

1 Introduction

An increasing number of railway network managers are striving to create an accurate and dynamic system to identify actual and potential damage to railway infrastructure. This involves obtaining the highest quality data, and thus preventing accidents. In addition, decisions are made on planning and prioritization for the maintenance, repair and renewal of railway lines.

In recent years, new techniques using, for example, GPS and Unmanned Aerial Vehicles (UAVs) are increasingly being used to diagnose objects. The work [1] presents the use of satellite measurements to evaluate the railway track geometry. UAVs, on the other hand, can be used to diagnose building facades [2] and bridges [3]. UAVs, which can monitor crucial sections of railway lines in real time, have become a very successful tool to support these tasks. They perform visual inspections of railroad infrastructure in sight, VLOS – Visual Line of Sight, or out of sight (control by means of UAV image transmission), BVLOS – Beyond Visual Line of Sight [4–7]. The main advantage of such a solution compared to traditional inspections performed with measuring vehicles is the elimination of decreased train traffic and, as a result, trains can run with optimal capacity.

There are UAVs on the market with the option to equip them with the highest quality imaging equipment and various sensors, allowing users to record data not only of practical use, but also to conduct scientific research work [8]. The measurement is made with an accuracy from fractions of to single millimetres [9], thanks to high optical resolution and special on-board flight stabilization systems [10]. The precise localisation of UAVs is also important here and is ensured by the GPS system.

Once recorded, overlapping aerial images are processed in photogrammetry software to produce an accurate 3D-point cloud [11]. By comparing with the reference point clouds from previous inspections, changes in infrastructure can be monitored.

Various defects of the railroad structure [12], including those of the rolling surfaces of rails, such as rail burns, squat, head checking or splits [13], can be displayed by the UAV imaging system and further analysed. They are the result of the dynamic wheel interaction on the rail, which results in rail fracture and train derailment.

A study on the classification of rail defect images recorded by classical non-destructive methods used by a specialized rail vehicle have already been conducted [14]. Moreover, some research projects also concerned visual methods of detecting squat defects in rails [15].

*Corresponding Author: Piotr Bojarczak: University of Technology and Humanities, Radom, Poland; Email: p.bojarczak@uthrad.pl
Piotr Lesiak: University of Economics and Innovation in Lublin, Faculty of Transportation and Computer Science, Poland; Email: piotr.lesiak@wsei.lublin.pl
The article focuses on video detection of particularly hazardous split defects, those practically preventing normal train traffic. In standard flaw detection systems, ultrasonic, magnetic, eddy current or laser methods are generally used to detect different types of rail defect [16]. The presented approach uses the imaging method with the systems installed on board the UAVs.

Split defects are vertical defects in the rail head edge and, in extreme cases, horizontal defects in the entire head. It seems natural to use one of the standard edge detection algorithms, or more complex ones like [17, 18]. The effectiveness of the above methods in terms of their use for split defects was also checked, but they proved to be ineffective.

In the next step, image segmentation methods were applied, which would allow the effective extraction of the rail head from the image, and thus detect its edge defects. The standard image segmentation methods are based on texture analysis [19]. However, the texture representing the rail head is subject to great variability due to corrosion, a whole range of surface defects or various types of dirt. This is also a significant impediment to this method.

Therefore, in the article, the authors attempted to apply, for the extraction of the rail head and detection of split defects, deep convolutional networks [20, 21], which in recent years have experienced rapid development. They enable much more effective detection of objects in images. These networks are also an important part of deep networks designed directly for image segmentation [22]. An important feature is the high resistance to both changes in light intensity of the image points and changes in the images subject to segmentation. When developing this algorithm, track images recorded by UAVs on the tracks of the Network Rail (NR) manager were used [23]. The use of UAVs together with artificial intelligence to detect split defects is an important element of novelty presented in this work. After some modification, the proposed method can be used to detect other types of surface defects such as squat, break-outs, skid spots, spalling and head checking [24–26].

The article consists of 6 sections. Section one is the introduction. The second section concerns the research object. Section three discusses the structure of the proposed algorithm. The fourth section presents a deep neural network extracting the rail head, while the fifth section presents a fragment concerning the location of split defects in the rail head. The last section is the conclusion.

2 Object of experimental inspections

Professional UAVs are equipped as standard with devices that allow extremely precise aerial images to be taken, using Ultra 4K cameras with UHD video recording – Ultra High Definition with a resolution of 4096×2160 with differ-

![Figure 1: UAV visualization of the station track inspection: a) image of a track fragment with a turnout, b), c) and d) subsequent enlargements of the image of this track fragment with split rail defects [27].](image)
ent frame rates and a variable viewing system – vertically down or horizontally, depending on the inspected object.

At such high resolution, it is possible to identify defects in the rolling surface of the rail head, markings on the sleepers, the assembly quality of fixings and rail bonds, or damage to the structure and surface of viaducts and bridges [3]. This is a key function when checking a railway track and accompanying infrastructure.

Images from an UAV of the track of a small freight railroad station on the UK network are shown as an example of an object of experimental inspection. The rails have defects in the rail heads in many places, i.e. split defects, both vertical and horizontal [24–26].

Figure 1 shows the result of an inspection on these tracks, conducted by Plowman Craven using a Vogel R3D UAV with 100 MPx resolution optics, flying at a height of 25 m above the tracks [27]. The individual photos in Figure 1a to 1d are successive enlargements of the same part of the station along the tracks with standing rolling stock. It is not difficult to notice that, despite the complete ballast, the condition of the rails in many places practically limits the standard movement of rail vehicles.

3 The Split Defect Detection Algorithm Structure in Rails

The algorithm for the detection of split defects in rails presented in the article, the structure of which is shown in Figure 2, consists of two main parts responsible for:

- extraction of the rolling surface of the rail head from the track image using FCN-8 (Fully Convolutional Network), and
- defect detection in the rail head.

The rail head extraction was performed based on the railroad track image segmentation process into the fragment with the rail head and the remaining part constituting its background. This was implemented by the FCN-8 Fully Convolutional Network. The second algorithm block performs rail head defect detection. This goal is achieved through a heuristic algorithm.

4 Extraction of the Rolling Surface of the Rail Head – FCN-8 Network

4.1 FCN-8 network structure

Extraction of the rail head is performed by the image segmentation process into a fragment with a head and background. The FCN-8 Fully Convolutional Network is used for this task [22]. During the segmentation process, the FCN-8 network assigns a label to each image point. The value of this label depends on the type of object to which the analysed point belongs. In the case of head extraction, there are two types of objects (head and background). Figure 3 shows the FCN-8 network structure.

The FCN-8 network uses a significant fragment of a CNN (Convolutional Neural Network). There are currently three forms of this network: AlexNet [28], VGGNet [29], and GoogLeNet [30]. According to [22], VGGNet [29] is the most accurate segmentation process, which is why it was chosen by the authors. VGGNet is also called VGG 16 and consists of serially connected convolutional layers ReLu and pooling. The last layers in the network are three layers with full connections between neurons. However, FCN-8 only uses one layer with full connections. This network classifies each point in the image on the basis of certain characteristics that describe the objects in the segmented image. These features are extracted through the use of a number of convolutional layers interspersed with the layers ReLu and pooling. The extraction of the features from the image by a single convolutional layer is implemented by means of the following operation:

\[
B(i, j, z) = \left( \sum_{k=-\frac{K-1}{2}}^{\frac{K-1}{2}} \sum_{m=-\frac{K-1}{2}}^{\frac{K-1}{2}} \sum_{n=0}^{N} A(i+k, j+m, n) \right) * W\left( k + \frac{K-1}{2}, m + \frac{K-1}{2}, n, z \right) + b(z)
\]
where: $i \in (0, \ldots, P - 1)$, $j \in (0, R - 1)$, $z \ldots (0, \ldots, Z - 1)$ are the coordinates of the output matrix $B$ generated by the convolutional layer. $A$ is the input matrix fed to its input with the size $P \times R \times N$, $W$ is the weights matrix with the size $K \times K \times N \times Z$. $K$ determines the filter width, $Z$ corresponds to the number of filters used in this layer, while $b$ is a bias vector of the length $Z$.

The submatrix with the size $K \times K \times N$ of the weights matrix $W$ corresponds to a single neuron connected to a limited number of elements (inputs) of the matrix $A$. As can be seen from formula (1), this layer performs the mapping $B = f(A, W)$ mapping the matrix $A$ of the size $P \times R \times N$ into the matrix $B$ with the size $P \times R \times Z$. When the convolutional layer is the first layer of the FCN-8 network, then the matrix $A$ corresponds to the analysed image fed to the network input and $N = 3$ (R, G and B image components). The output matrix $B$ of each convolutional layer is given to the non-linear activation function represented in Figure 2 as a layer ReLu, which performs the following operation $C = \text{max}(0, B)$. Additionally, after several convolutional layers, the matrix $C$ with the size $P \times R \times Z$ is reduced to $P/2 \times R/2 \times Z$ as a result of subsampling or the operation pooling.

The last three layers are layers with full connections. The last one consists of two neurons, one responsible for detecting the rail head and the other for detecting the background. Each of these neurons is connected to all outputs (output matrix elements $C$) of the previous layer. According to [22], the full-connection layer can be treated as a convolutional layer in which the filter width covers the entire area of the input image. As a result, a rough form of the input image is created with the size $H/2 \times G/2 \times 2$, where $H$ indicates height and $G$ the input image width fed to the network, while $2$ corresponds to the number of object classes into which the analysed image is divided (head and background). The return from the rough image form to its original dimensions $(H, G)$ is performed in the following steps (Figure 2):

1. summing $2 \times$ upsampling of the conv7 convolutional layer and the pool4 layer
2. summing $2 \times$ upsampling of the result obtained in point 1 and the pool3 layer
3. $8 \times$ upsampling of the result obtained in point 2.

All network parameters (weights) are defined in the learning process. This process minimizes the error function $L$ in the form of cross-entropy:

$$L = -\frac{1}{S} \sum_{i=1}^{S} P(x_i) \log Q(x_i)$$

where: $P(x_i)$ is the probability to which object class the point $x_i$ belongs. $Q(x_i)$ determines the probability to which feature class the point $x_i$ belongs, which is generated by FCN-8 in the learning process. $S$ is the number of learning datasets fed to the network during learning.

The learning process ends when the error function $L$ is close to 0. This means that both probability distributions are close to each other $(p(x) = q(x))$. 

4.2 FCN-8 network implementation

The FCN-8 network implementation can be divided into the following stages: the learning dataset development, the network construction stage and the network learning and testing stage. The learning dataset includes a pair of 1024×3008 Pixel photos. Figure 4 shows an example of a learning pair fed to the network during the learning process. This pair consists of an image of a railroad track fragment fed to the network input and of the corresponding image after segmentation, which should be generated by the net-
work. In order to increase the learning speed, each of the images was subjected to the standardization process. As a result, the value of each point in the image is included in a distribution with a mean value of zero and a variance of 1. A set of 1500 learning pairs was obtained. This was randomly divided into three disjoint sets: data_train with 1000 learning pairs, data_valid with 250 learning pairs and data_test with 250 learning pairs.

The FCN-8 network was implemented in the Tensorflow environment [31, 32]. The VGG16 network [29], developed and made available by Oxford Visual Geometry Group was used as the main backbone of the FCN-8 network. In order to increase the effectiveness of the network and to speed up the learning process, the transfer learning method was used. It is based on the assumption that a network which has learnt to solve one problem can be used as a starting point for learning to solve another problem [33]. The VGG16 network was used as a boot network, able to recognize objects in images belonging to 1000 different classes, and its model is available in [31]. The network learning was conducted on a computer equipped with an NVIDIA GeForce GTX 750 Ti graphics card. Two sets were used in the learning process: data_train for learning and data_valid for testing.

In the learning process, the AdamOptimizer package was used to minimize the L error function, in which the learning constant was 0.0001. The learning process covered 200 epochs and lasted 11 days.

Figure 4: Example of a learning pair a) An image fed to the network input, b) A corresponding image after segmentation, which should be generated by the network.

Figure 5: Error dependence L during the learning process a) for the data_train set, b) for the data_valid set.
Figure 6 shows the error dependence $L$ during the learning process for the $data\_train$ set (Figure 5a) and the $data\_valid$ set (Figure 5b). The image segmentation process effectiveness was checked on data from the $data\_test$ set. It should be noted that the data from this set did not participate in the network learning process.

According to [34], for the assessment of the image segmentation effectiveness, the measure $IoU$ Intersection Over Union, also called the Jaccard index, can be used. For the extraction of the rail head, this indicator is defined as:

$$ IoU = \frac{T_M \cap P_P}{T_M \cup P_P} $$

where: $T_M$ means the set of points belonging to the actual rail head area in a segmented image, $P_P$ is the set of points belonging to the rail head area generated by the FCN-8 network, while $\cap$ and $\cup$ mean the common part and the sum of the sets $T_M$ and $P_P$ respectively.

The mean value $IoU$ for data from the $data\_test$ set was 0.92. Figure 6 shows examples of track images from the $data\_test$ set after segmentation with the FCN-8 network. Figure 6 a) and b) show the effect of segmentation for the rail with the split defect of relatively small width and length. Figure 6 c) and d) show the effect of segmentation for the rail with the split defect of relatively small width and large length. Figure 6 e) and f) show the effect of segmentation for the rail with the split defect of large width (covering most of the rail head). Additionally, Figure 6 g) and h) show images before and after segmentation for the rail without any defects.

5 Split Defect Detection in the Rail Head

A heuristic algorithm was used to detect split defects in the rail head. It was implemented in the Python language using the OpenCV library. The pseudocode of the proposed algorithm is as follows:

1. Contour generation ($cont$) for the rail head area obtained by the FCN-8 network,
2. Rectangle generation ($rect$) circumscribed around the contour ($cont$) obtained in point 1,
3. for each $x_{rect}$ on the perimeter $rect$, 

4. If \( \text{thres}_{\text{low}} < |y_{\text{rect}} - y_{\text{cont}}| < \text{thres}_{\text{high}} \),

5. draw a line between the points \((x_{\text{rect}}, y_{\text{rect}})\) and \((x_{\text{rect}}, y_{\text{cont}})\).

where: \(x_{\text{rect}}\) is the x coordinate on the rectangle perimeter circumscribed around the contour \(y_{\text{rect}}\) is the coordinate \(y\) of the point \((x_{\text{rect}}, y_{\text{rect}})\) on the rectangle perimeter, \((\text{rect}), y_{\text{cont}}\) is the y coordinate \((x_{\text{rect}}, y_{\text{cont}})\) of the contour \((\text{cont})\). \(\text{Thres}_{\text{low}}\) is the threshold defining the minimum height of the defect, while \(\text{thres}_{\text{high}}\) is the threshold defining the maximum height of the defect. It is assumed that \(\text{thres}_{\text{high}}\) is equal to the rail head width. Figure 7 illustrates the consecutive pseudocode fragments.

For the operation of the split defect detection algorithm, the parameter \(\text{thres}_{\text{low}}\) is decisive. Its task is to eliminate inaccuracies in the segmentation process (small irregularities appearing on the circumference of the rail head after segmentation). The high value of this parameter removes these irregularities while increasing the probability of removing the real defect. On the other hand, its low value increases the probability of segmentation inaccuracies being treated as defects. The Receiver Operating Characteristic (ROC) curve was determined to check the effect of the parameter \(\text{thres}_{\text{low}}\) on the quality of the detection process.

This is a curve showing the dependence of \(\text{TPR} \) (True Positive Rate) on \(\text{FPR} \) (False Positive Rate) at different values of the parameter \(\text{thres}_{\text{low}}\). In the case of the algorithm for split defect detection, \(\text{TPR} \) is defined as:

\[
\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} 
\]

where: \(\text{TP} \) (True Positive) is the number of correctly detected split defects, and \(\text{FN} \) (False Negative) is the number of undetected split defects (treated as minor irregularities in the segmentation process).

However, \(\text{FPR} \) is expressed by the following rule:

\[
\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} 
\]

where: \(\text{FP} \) (False Positive) is the number of small irregularities detected as a split defect, and \(\text{TN} \) (True Negative) is the number of small irregularities not detected as split defects.

Both \(\text{TPR} \) and \(\text{FPR} \) were determined based on data from data_test. As already mentioned, this set included 250 track images which contained 371 split defects and 43 segmentation irregularities. Figure 8 shows the ROC curve for this algorithm. The point with the coordinates \((0, 0)\) corresponds to the value \(\text{thres}_{\text{low}} = \text{thres}_{\text{high}} = \text{width of}\)
Table 1: Confusion matrix for the split rail defect detection algorithm.

<table>
<thead>
<tr>
<th></th>
<th>Detected as Split defect</th>
<th>Irregularities from segmentation by FCN-8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Split defect</td>
<td>305</td>
<td>66</td>
</tr>
<tr>
<td>Irregularities</td>
<td>9</td>
<td>34</td>
</tr>
</tbody>
</table>

Based on Figure 8, the optimum value is \( \text{thres}_{low} = 16 \) for which \( TPR = 0.82 \) and \( FPR = 0.2 \) were obtained. Table 1 shows the confusion matrix for the split rail defect detection algorithm obtained for \( \text{data}_{test} \) and \( \text{thres}_{low} = 16 \). In the row “split defect”, the first cell contains the number of split defects detected as split defects, while the second cell contains the number of split defects detected as irregularities from segmentation by FCN-8. In next row, the first cell contains the number of irregularities from segmentation by FCN-8 detected as split defects and the second cell contains the number of irregularities from segmentation by FCN-8 detected as irregularities. According to this confusion matrix, the system correctly classified \( (305 + 34) \) of \( (305 + 34 + 9 + 66) \) examples, which entails a 81.8% detection (classification) rate.

Figure 9 shows the algorithm effect for sample track images. Figure 9 b), f), h) present the algorithm effect for split defects of large width and length, while Figure 9 d) for the defect of small size.

The authors also verified the invariance of the performance of the algorithm to changes in brightness of the track image. For this purpose, the brightness of each image belonging to the \( \text{data}_{test} \) set has been artificially changed. It was achieved through conversion of RGB image model to HSV (Hue, Saturation, Value) model in which V component is responsible for the brightness of the image. The
brightness of each pixel in the modified image is changed based on a simple relationship, \( I = S \times I_{\text{nom}} \).

Where \( S \) is a multiplier in the range of (0.1, 0.2, \ldots 1.0) and \( I_{\text{nom}} \) is the brightness of each pixel (V component) in the original (unmodified) image. Figure 10 presents the dependence of the detection rate on a \( S \) multiplier. As shown in Figure 10, detection rate is almost constant for a \( S \) multiplier higher than 0.4.

![Figure 10](image)

The detection rate versus changes in brightness of images (represented by a \( S \) multiplier).

### 6 Conclusion

In the study presented in the article, the authors relied on image samples of damaged rails recorded during the railroad track diagnostics by UAVs. This was limited to split rail defects due to the fact that their presence limits train traffic on the route.

The proposed algorithm enabled their detection with an efficiency of approx. 81%. It uses the FCN-8 deep-learning network, implemented in the TensorFlow environment, to extract the rail head by image segmentation. Using this type of network for segmentation increases the resistance of the algorithm to changes in the recorded track image brightness. This is crucial in the case of images recorded by UAVs, where it is not possible to ensure their constant brightness. The detection of these defects in the rail heads was performed using an algorithm implemented in the Python language using the OpenCV library.

To locate a defect, it uses a contour of a separate rail head together with a rectangle circumscribed around it. Errors in split defect detection are caused by the segmentation process inaccuracy, which results from the transition from a rough segmentation image (conv7, Figure 3) to an image corresponding to the size of an image subjected to the segmentation process. Therefore, the image resolution and, in particular, the rail head width have a significant impact on the algorithm accuracy.

In the presented solution, the rail head width was 232 points with 1024x3008 image resolution and the \( \text{thresh}_\text{low} \) parameter obtained from Figure 8 was 16. This means that the presented algorithm can detect split defects with a width of not less than \((16/232)*100\% = 6.9\%\) of the rail width, which is a result that fully meets the operating criteria.

Split defects examined in the work, differ significantly in terms of geometry seen from the top of the rail, compared to other classic surface defects of the rail head such as squats, spalling, break – outs, etc. Article [13] concerns the detection of the above mentioned defects. However, it is difficult to make clear comparisons here, because both the shape of the defects themselves and the algorithms are completely different from those used by the authors.

In subsequent studies using deep-learning networks, based on images from UAVs, the authors intend to select other rail surface defects, as well as damage to the railway infrastructure with a significant impact on railway traffic safety. These include cracks and chipping of railway sleepers, defects in rail fixings, the condition of the track ballast, railway turnouts [8] and structural deformation of electric tractions, bridges and viaducts, railway automation elements and tunnels [9].

**Acknowledgement:** The authors thank Plowman Craven Ltd for a consent for the use of the images of the railroad track recorded by its UAVs systems in this paper.

### References


