Article

Institution of MECHANICAL ENGINEERS
Journal of RAIL AND RAPID TRANSIT

Proc IMechE Part F:
J Rail and Rapid Transit
2023, Vol. 0(0) 1–12

DOI: 10.1177/09544097231156312
journals.sagepub.com/home/pif
SAGE

# Artificial intelligence-aided intelligent obstacle and trespasser detection based on locomotive-mounted forward-facing camera data

**Huixiong Qin, Asim Zaman and Xiang Liu**

**Abstract**

Most fatalities in the railroad industry are trespasser deaths. Additional obstacles such as fallen trees, construction materials, and other debris also pose a danger to railroad operations. Understanding where, when, and how these events occur can help railroads develop trespasser mitigation strategies and improve rail safety. However, obtaining information from forward-facing footage requires immense manual effort. We developed a forward-facing trespassing and obstacle detection system that utilizes artificial intelligence to recognize and understand unsafe events and logs them for later review and analysis. The system dynamically identifies regions of interest, trespassers, and obstacles with a customized detection methodology which uses a semantic segmentation model called DeepLab and an object detection model called YOLOv5. These models were trained by a dataset containing over 10,000 images from various sources, including open-source datasets and videos provided by industry collaborators. The novelties of this research are threefold: the development of an algorithm capable of detecting the railroad's restricted area around the track area, optimizing the trade-off between accuracy and latency to achieve real-time performance, and proposing a universal obstacle detection algorithm. The final system was able to analyze 400-pixel x 400-pixel videos at a rate of 14.5 frames per second (FPS) in an edge-computing device which identified trespassers and obstacles with more than 92% accuracy. The system can adaptively detect both obstacles and trespassers in the train's path, as well as the railroad property area surrounding the tracks. This research offers the railroad industry tools which can provide precise trespasser and obstacle information to improve railroad safety and operations.

**Keywords**

Railway, forward-facing footage, obstacle detection, trespasser detection, artificial intelligence

## Introduction

### Background

"Trespasser deaths on railway rights-of-way and other railway property are the leading cause of fatalities attributable to railway operations in the United States. To address this serious issue, the railway industry, governments (Federal, State, and local), and other interested parties must know more about the individuals who trespass."[1] This statement by the Federal Railway Administration (FRA) on the challenge of trespassing highlights a critical need in the railway industry. According to the database of the FRA's office of safety analysis, between 2010 and 2020, 95 percent of railway fatalities occurred at highway rail grade crossings or involved trespassers.[2] Obstacles like fallen trees, construction materials, and other debris pose an additional danger to railroad operations. From 2012 to 2019, approximately 1800 accidents occurred due to such obstacles.[2] These accidents accounted for approximately 15% of all accidents on railway property and resulted in property damage, injuries, and fatalities.

Trespassing and obstacles are only identified when an accident occurs or when a train operator reports them. However, there are many more unrecorded events, as shown in research by Zaman et al.,[3] which indicate dangerous situations that could result in damage. Therefore, recording, reviewing, and understanding trespassing and obstacle events is necessary to address key challenges in the railway industry.

As these pervasive challenges continue in the rail industry, so too does widespread deployment of cameras in the United States, following the Fixing America's Surface Transportation Act. [4] The FRA requires railroads to install inward and outward-facing cameras in all controlling locomotives of passenger trains. A survey conducted by Jones et al.[5] showed that out of 5450 transit vehicles in service, approximately 68% have forward-facing cameras installed, while 100% of light rail vehicles and 98% of street cars are

Department of Civil and Environmental Engineering, Rutgers University, New Brunswick, NJ, USA

**Corresponding author:**
Xiang Liu, Department of Civil and Environmental Engineering, Rutgers University, 96 Frelinghuysen Road Piscataway, New Brunswick, NJ 08854-8018, USA.
Email: xiang.liu@rutgers.edu

equipped with forward-facing cameras. Railway video systems are a massive source of video data. Therefore, automating the information extraction from this data source has become an immediate need.

As this video dataset grows, there is a parallel rapid development in artificial intelligence (AI) algorithms and computer vision techniques. State-of-the-art open-source AI models with pre-trained weights, such as YOLO[6–9] and large-scale datasets such as RailSem19[10] are being utilized to solve challenges in many industries, especially roadway transportation. While AI has not been widely adopted by the railway industry, there are many applications to which this technology could be applied. This research aims to combine the challenges of trespassing and obstacles, big railway video data, and rapid developments in computer vision AI to improve railway safety. Specifically, this research aims to develop a railway forward-facing trespasser and obstacle detection system based on deep learning (DL) pattern recognition models, trained with images from real world operations.

### *Objectives of research*

This research aims to accomplish the following goals.

- To detect trespassers in the restricted area.
- To detect both obstacles and trespassers in the structure gauge area.
- To customize the model to tradeoff the latency and accuracy for deploying in edge computing devices.
- To allow other studies to be compared with ours, we provide performance benchmarks tested with public annotated datasets and open-source videos.

## Abbreviations & definitions

The following is a list of abbreviations for performance metrics and definitions used in this research.

- TP: True Positive
- TN: True Negative
- FP: False Positive
- FN: False Negative
- P: Precision is the ability of a model to identify only relevant objects. $P = \frac{TP}{TP+FP}$.
- R: Recall is the ability of a model to find all relevant cases. .
- U: Intersection over union measures the overlapping area between the predicted bounding box, $B_p$, and the ground-truth bounding box, $B_{g_t}$. $tIoU = \frac{B_p \cap B_{g_t}}{B_p \cup B_{g_t}}$.
- mAP: The mean average precision (mAP) is a metric used to measure the accuracy of object detectors for all classes. Let c, r, l be decimal numbers and s be an integer, mAP@c represents the mAP for detection whose IoU is greater than c, and mAP@l:s:r is the mean of mAps of $\frac{r-l}{s}$ IoUs varying from l to r with steps of s.
- mIoU: Mean intersection-over-union is a common evaluation metric for semantic image segmentation which first computes the IoU for each semantic class and then computes the average for all classes.
- Track Area: Area between the two rails on which a train travels.
- Structure Gauge Area: The area with potential risk of hitting obstacles or trespassers. It is an extension of the track area. The structure gauge can encompass rails, railway ties, sub-grades, and any other user-defined region of interest.
- Restricted Area: Property owned by transit companies or government transportation departments with prohibited public access.
- Trespasser: Unauthorized person or vehicle in the restricted area, including the structure gauge area.
- Obstacle: Fallen trees, construction materials, and other debris that pose an additional danger to railroad operations.

## Literature Review

To ensure that this research is novel, a literature review, focused on AI-based obstacle detection using forward facing footage, was performed, as summarized in Table 1.

Ristić-Durrant et al.[11] gave a comprehensive review of vision-based on-board obstacle detection research, covering both AI-based and conventional CV methods. In contrast, our review covers mainly AI-based methods since recent studies [11–13] have shown that AI-based methods outperform conventional CV methods. Moreover, we focus on presenting a performance comparison by listing metadata of datasets, accuracies, and hardware specifications of related works.

## Contributions

The novelty of this study is to provide an implementable and generalized solution for three challenges. First, since internet access which allows streaming video data is not always available on locomotives, it is necessary to perform AI-based detections on the edge. However, popular semantic segmentation is computationally expensive, which makes it difficult to utilize AI on computationally lean edge-computing devices with an acceptable frame rate. To achieve a good tradeoff between latency and accuracy, we reduce the resolution of inputs to the semantics segmentation model and then restore the resolution of outputs using a trained light-weight CNN.

Second, approaches from previous studies can only detect trespassers in the track area. Both vehicles and pedestrians in the restricted area (for example, people standing on the ballast area) are also trespassers. Therefore, we combine the traffic object detection and the restricted area prediction to search for trespassing in the restricted area.

Third, existing methodologies for obstacle detection rely on a trained object detector to recognize objects of previously annotated types in training datasets. It is currently unfeasible to gather images on all the types of obstacles that a locomotive may encounter with enough variety to train a reliable obstacle detector. We developed a novel universal obstacle detection algorithm based on the difference between consecutive semantic predictions, aiming at detecting obstacles when they do not belong to any predefined classes.

**Table 1.** Summary of literature review.

| Paper | Dataset | Methodology | Result |
|---|---|---|---|
| [12] | 3000 1080 × 1920 images recorded on the two metro lines including sub-ground tunnel footage, where training set: Validation set: Test set is 25 : 2: 3 | Modified ResNet50 to extract deep features efficiently and used several layers of convolution and deconvolution to upsample the feature for generating pixel-wise track area prediction | 89% of MIoU and the inferencing speed is 20 FPS in a PC with GTX 1080 as the GPU. |
| [14] | The dataset consisted of forward-facing thermal Sequences | Use a modification of AlexNet to detect key points along the rails<br>Put the largest weight on the area furthest away from the train | — |
| [15] | 20,000 of outdoor railway scenes, where training set: Test set is 3 : 1. | Retrain faster RCNN for object (potential obstacle) detection | 94% of object detection precision |
| [13] | Videos were recorded on a locomotive running on a line with the length of 120 km | Gather bounding boxes via YOLO. Then feed them to a customized neural network to estimate distances | Their reported average error of distance measurement was around 6% |
| [16] | 7342 images of railway shunting scenes recorded in different lighting and weather conditions, where training set: Test set is 7 : 3 | Modifies RON by adding feature transfer blocks and receptive field enhancement modules to merge the adjacent feature maps, which helps improve the accuracy in detecting small objects | 89% precision and the inferencing speed is 38 FPS in a PC with GTX 1080 Ti as the GPU. |
| [17] | 1277 720 × 1280 images recorded on three metro lines | Extract deep features using ResNet. They are further fed to a RCNN to gather confidence score and bounding boxes for possible obstacles | 91% of mAP@0.5 and the inferencing speed is 26 FPS in a PC with GTX 1080 Ti as the GPU. |
| [18] | 3000 1333 × 800 images with 30,278 objects annotated. The ratio of training set: Test set is 9 : 1 | Adopt the context extraction module and the content-aware reassembly of features module to enhance, accelerate the deep feature extraction | 90.6% of mAP@0.5 and the inferencing speed is 11 FPS in a PC with RTX 2080 super as the GPU. |
| [19] | 5617 images from 10 km of railway forward-facing footage. The ratio of training set: Test set is 4 : 1 | Modify SegNet by adding dilated cascade connection and cascade sampling to achieve a trade-off between accuracy and latency. | 98% MIoU |

## Methodologies

The AI-based solution to detect obstacles and trespassers in forward-facing railroad footage presented in this research is a three-step process:

1. Object Detection: Recognize vehicles, people, and meaningful railway components, in the video frame and differentiate between pedestrians and legal occupiers.
2. Dynamic ROI: Determine what is railroad property in the video using semantic segmentation.
3. Trespasser and Obstacle Detection: Determine if people, vehicles, or other non-predefined objects, such as trees and rocks, are in the region of interest to identify obstacles and trespassers.

### *Object Detection*

Object detection is a type of computer vision and pattern recognition technique that detects instances of certain classes according to their semantics (the visual meaning), pose, and other features. Object detection models were adopted in this research to find vehicles and pedestrians, and to detect meaningful static railway components, such as switches and signal lights. Specifically, YOLOv5 was adapted because of its superior accuracy and performance.[9]

The YOLO models are special CNNs that infer localization and classification simultaneously. They divide the input image into an $S \times S$ grid and predict $B$ bounding-box candidates for each grid cell. Each bounding box candidate consists of six predictions: $x, y, w, h$, confidence $c$, and the classification $cls$. The $(x, y)$ coordinates represent the center; w and hour represent the width and height. If the center of a ground-truth object falls into a grid cell, that grid cell is responsible for detecting that object. The loss function evaluates the distance between predicted bounding box candidates and their corresponding ground-truths. As the loss drops during the optimization, predicted bounding box candidates approach the ground-truths. Non-maximum suppression is used to select the best bounding box from overlapped candidates, resulting in one object of interest per bounding box. More detail can be found in.[6–9]

### *Dynamic region of interest*

To detect trespassers and obstacles, an algorithm must know which areas in the video frame are designated as the track area, the structure gauge area, and the restricted area. The structure gauge area varies from location to location. For example, light rail track in the city does not usually have railroad ties while rural freight track does. So does the restricted area, which is normally bounded by fences or vegetation but can often be unmarked. We utilize DL-based semantic segmentation models as described below.

Image semantic segmentation is the process of classifying each pixel in an image as a certain class. Several state-of-the-art semantic segmentation models were tested based on the Railsem19 dataset. The best performing model was a modified Deeplab[20] which was modified to create a lightweight semantic segmentation model. DeepLab is broadly composed of two steps:

(1) Encoding: the aim of encoding is to extract essential information from the image. This is done using a pre-trained CNN whose convolutional layers look for different features in an image. It moves through essential information layer by layer and ultimately forms a feature map.
(2) Decoding: the decoding phase takes care of combining extracted features to predict local semantics and reconstructing the output as the required dimension.

Based on our experiments, the resolution of inputs and the selection of the backbone model play important roles in balancing performance and accuracy. DeepLab was customized to optimally balance these two metrics.

MobileNet was selected as the backbone model and is shown in Figure 1. In this model, inputs are resized to a low-resolution image before being fed into DeepLab. The output semantic predictions are restored to the original resolution using a lightweight CNN. In practice, the inference can be further enhanced by cropping the upper part of the images (those areas are mostly sky in forward-facing footage and therefore are not related to the semantics of interest).

## Obstacle and trespassing detection

Finally, the YOLOv5 model and the customized DeepLab algorithm were combined to detect objects in the track area and the restricted area. This combined system initially checks if vehicles and pedestrians are in the track area or the restricted area. It then finds the train's path among the detected track areas and analyzes it to see if there are any sudden changes that would indicate obstacles. We extend the train's own path area on both sides by the structure gauge distance, that can be set by railway agencies. For example, the New York Department of Transportation sets 4 feet as the structure gauge distance.[21] Finally, based on YOLO detections of switch positions, it determines which track the train will follow when a turnout is present.

*Trespasser detection framework.* Figure 2 shows the system framework for detecting trespassers in forward-facing video. First, video frames are fed to the object detector. These objects are then separated into trespassers and legal occupiers, including staff with permission to enter the restricted area such as track workers. Legal occupiers are differentiated from trespassers based on an OpenCV-based template matching algorithm to check if the person is wearing personal protective equipment (PPE), namely a retroreflective green or orange safety vest. Note that, practically, it is very rare for normal people to wear railroad PPE. For vehicles and pedestrians, the algorithm will check if they are within the restricted or track area.

*Obstacle detection framework.* Figure 5 shows the obstacle detection framework. This framework functions like the trespasser detection framework but with several key changes. Non-predefined objects such as rocks, trees, and other debris are not inherently recognized by the object detection model. However, this can be resolved by evaluating occlusions to the predicted semantics of the track area.

Firstly, track areas are recognized via semantic segmentation and the train's current path is identified with the addition of switch position recognition. Let I be the $W \times H$ semantic prediction matrix, C be a proper constant, and T be a threshold defined experimentally. $I_{i,j}$ represents the label of the pixel in the i-th row and the j-th column starting from the upper left corner, whose value is one if and only if the label is track area; otherwise it is 0. Along the $y$-axis, if there are more pixels predicted as track area in the upper zoom than in the lower zoom, that is if $\exists j_0$ s.t. $\sum_{i=0}^{W} I_{i,j_0} - \sum_{i=0}^{W} I_{i,j_0+C} > T$, then there is a splitting track.

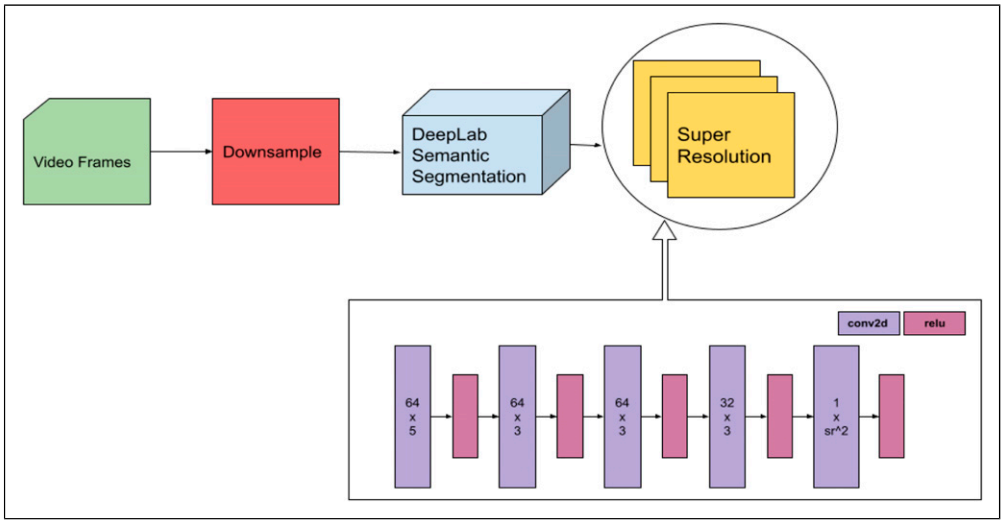


**Figure 1.** Semantics segmentation flow chart.

Secondly, the switch's position is recognized to determine which branched track a train will follow. The switch's position and train's path can be determined visually; therefore, we were able to detect switches and recognize their positions via YOLO. An example of the train's path and switch detection is shown in Figure 3.

Thirdly, to algorithmically determine the structure gauge area in our computer vision model, we extend the train's own path along the *Y*-axis given location-specific parameters — the structure distance $d_I$ in image and the slope of the train's own rails in images $k_I$ — we can extend the train's own path range from $[left_y, right_y]$ to $[left_y - (d_I + k_I \cdot y), right_y + (d_I + k_I \cdot y)]$, where y is the current index along the *Y*-axis.

Fourthly, a contour describing the track area is used to identify non-predefined obstacles. Pixels of the obstacles are less likely to be recognized as the track area. This will cause sudden breaks and sharp changes in the detected tracks and rails. If the reduction of pixels of the train's own path exceeds a certain threshold, then an obstacle is detected.

Below, Figure 4 shows the pseudo code describing the non-predefined obstacle detection.

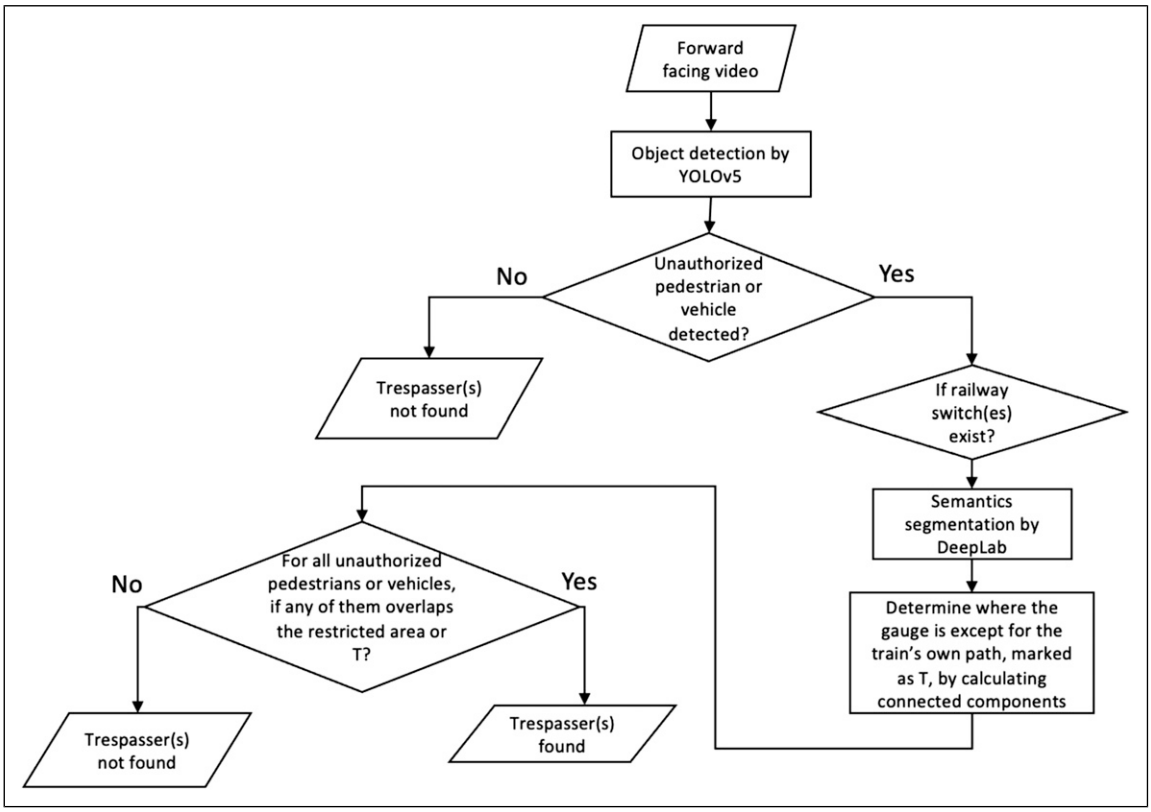


**Figure 2.** Trespasser detection framework.

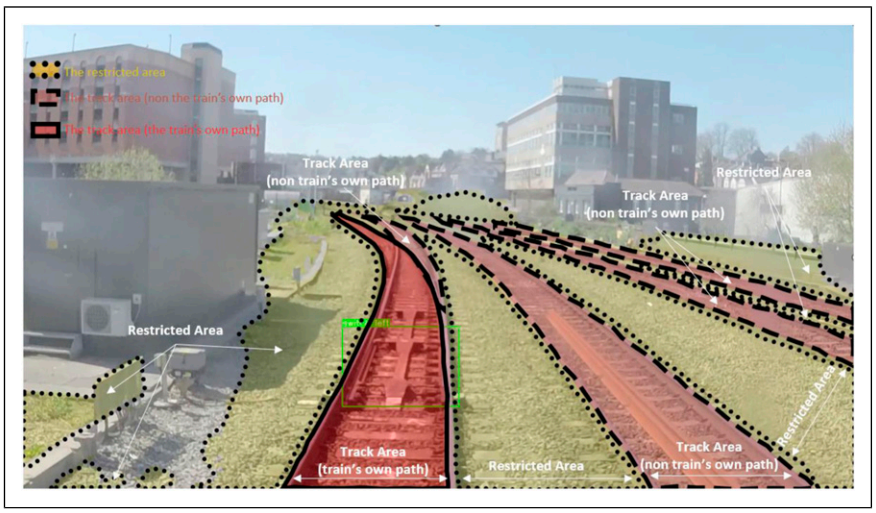


**Figure 3.** Segmented track, restricted area, and train path with switch identification

```
---------------------------------------------------------------------
Input:
   Semantic matrix: S
   contour smoothness threshold: c
   preview contour area queue: Q (fixed size w)
Output:
   Whether there is unknown obstacle
Algorithm:
S' <- a matrix with the same size as S
S'_{i,j} = 1 if S_{i,j} is labeled as "the train's own path",
otherwise S'_{i,j} = 0
   C <- FindContour(S')
   Q.enqueue(|C|)
   if min(Q[w/2:w]) - max(Q[0:w/2]) > c:
      return true
return false
---------------------------------------------------------------------
```

**Figure 4.** Non-predefined obstacle detection pseudo code

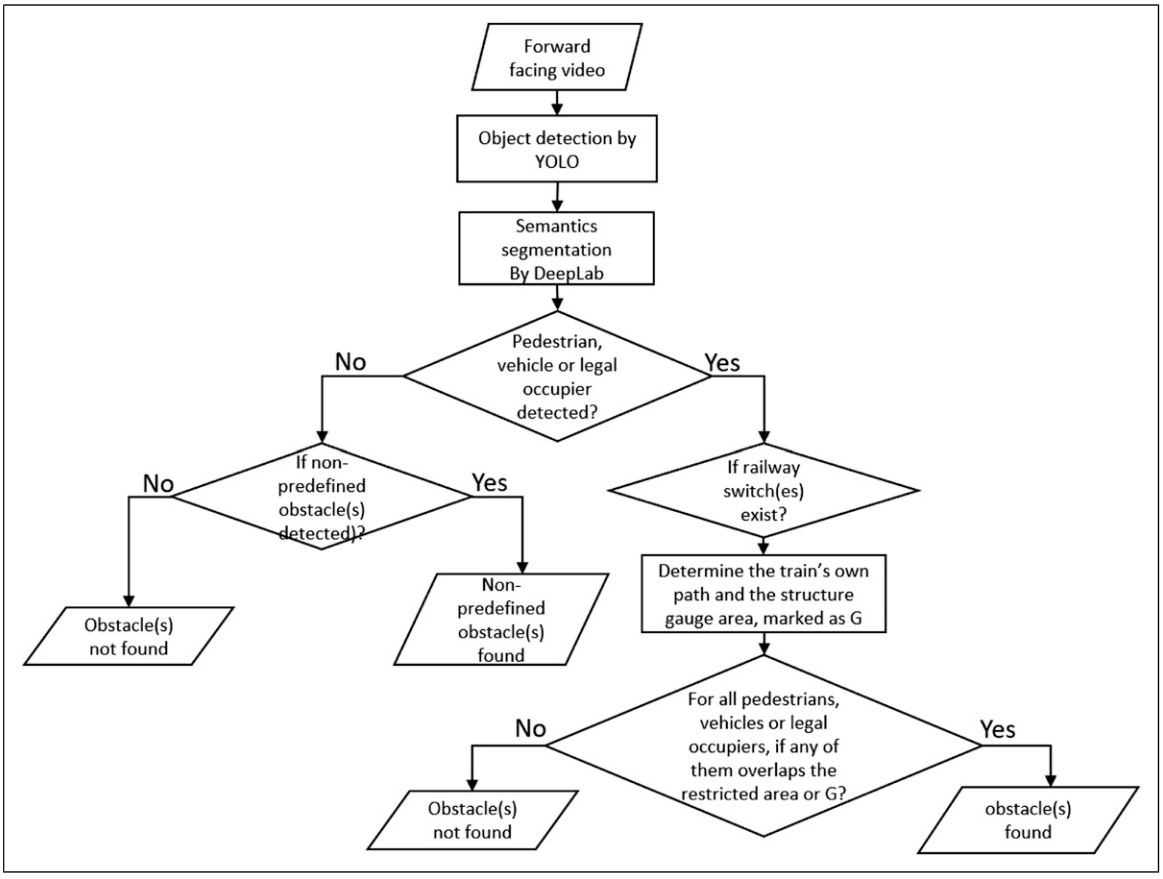


**Figure 5.** Obstacle detection framework

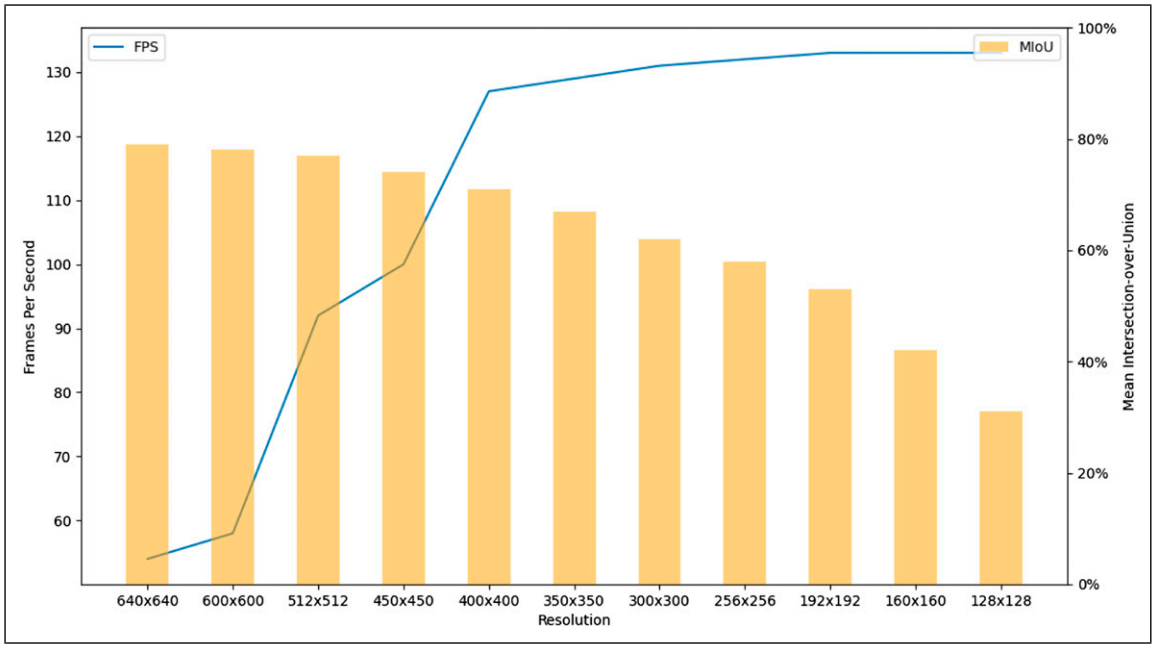


**Figure 6.** Trade-off between performance and accuracy as a function of the input resolution

## Results and discussion

YOLO and the customized DeepLab were trained separately before being combined. A few processed images are attached (Figure 7, Figure 8) to demonstrate highlighted features of the system. All experiments are executed on a Windows 10 operating system with a Nvidia GeForce GTX 1080 graphics card and an Ubuntu 16.04 operating system on a Jetson TX2 (an edge computing device).

### *Model training data*

The four models mentioned in the above section were retrained separately using different datasets because of their varying objectives. First, the object detection training dataset consists of 2782 images from Chicago Transit Authority (CTA) forward-facing footage available on YouTube and 8500 images from RailSem19. Objects of interest were defined as pedestrian, vehicle, track sign, signal light, and railway switch.

Secondly, the RailSem19 dataset was also used to train the semantics segmentation model with pixel-wise semantic annotation and revised classes of semantics. The RailSem19 semantic masks were downsampled to train the super-resolution module while taking original masks as ground truths.

### *Results*

With the obtained data, we retrained the YOLOv5 model and the customized DeepLab model to gather optimized weights. The logs of training processes, performances, and experiment results to tradeoff between latency and accuracy are presented. Some processed results were visualized for demonstration and discussion in Figure 7 andFigure 8.

*Object detection model training results.* The object detection model was trained through transfer learning. This was accomplished by initiating the model with a pretrained weight, freezing initial weights of the backbone, and only adjusting the rest of the weights to minimize the loss. This transfer learning approach was used for two reasons. First, although thousands of images were collected from the subject dataset, this is insufficient to train an accurate YOLO model from scratch. Hundreds of thousands of images are commonly used in training datasets to achieve sufficient accuracy. Second, training a model from scratch takes time and is computationally expensive. The transfer-learning approach requires fewer resources than normal training and accelerates training. The downside to this approach is that it may be less accurate than a model trained with sufficient data without freezing any layers.

The object detection retraining ran for 230 epochs until the loss of the training set converged. The training was completed for YOLOv5 after 230 epochs and reached 98% of mAP@0.5.

*Semantic model training results.* The DeepLab semantic segmentation model was trained on the Railsem19 dataset. This was completed without freezing the backbone due to fewer parameters as compared to YOLOv5 and the diversity of the Railsem19 dataset. The model was optimized with pre-trained weights and the super-resolution model was trained from scratch.

Different combinations of input sizes and backbone models were tested to achieve a balance between performance and accuracy. MobileNetv3, proposed by Howard et al.,[22] is the backbone model with the best tradeoff between the inferencing FPS and accuracy under a fixed input size based on cross validation. Meanwhile, Figure 6 shows how the input size influences the inferencing FPS and accuracy given MobileNetv3 as the backbone model. The results indicate that the input size of 400 pixels x 400 pixels is the resolution which best balances the inferencing FPS and accuracy, given the backbone model.

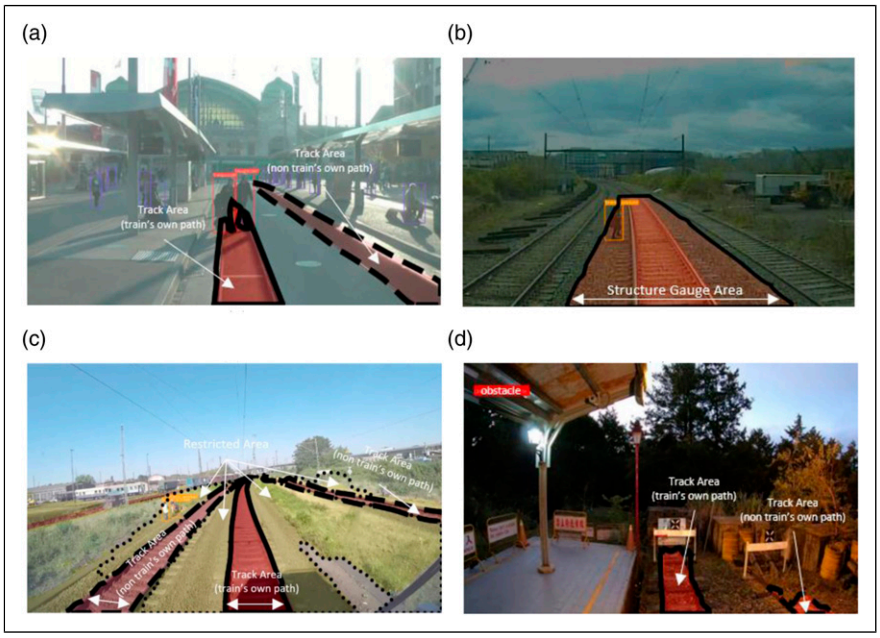


**Figure 7.** Demonstration for the main use cases: (a) detecting trespassing in the train's own path, (b) *detecting trespasser in the structure gauge area*, (c) detecting legal occupiers, (d) *a track termination was detected as obstacle.* Solid lines bound the trains' own paths, dash lines bound the track areas, and dotted lines bound the restricted areas.

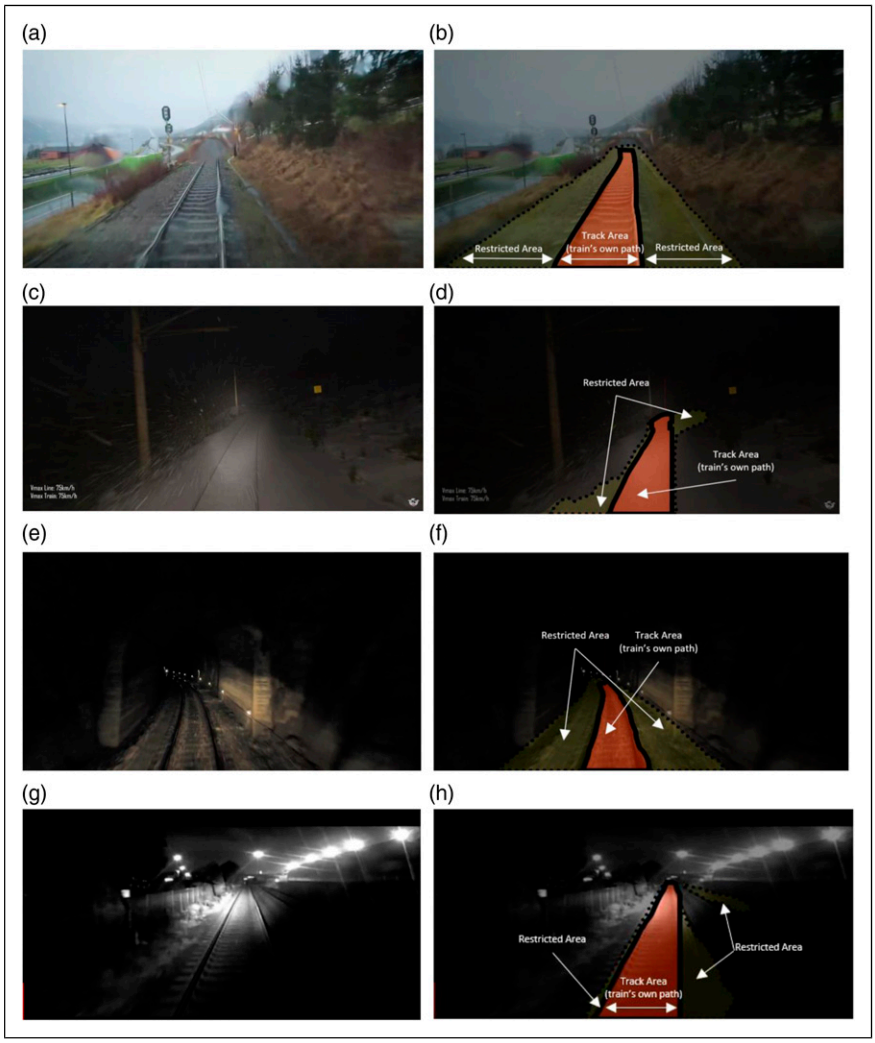


**Figure 8.** Demonstrations in edge cases: (a) rainy daytime (original) ref, (b) rainy daytime (processed), (c) snowy nighttime (original) ref, (d) snowy nighttime (processed), (e) tunnel (original) ref, (f) tunnel (processed), (g) night-vision (original) ref, (h) night-vision (processed). Solid lines bound the trains' own paths and dotted lines bound the restricted areas.

**Table 2.** Video and image data for system validation.

| | | Trespasser | | | Obstacle | | | Legal occupiers | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Dataset | Metadata | Actual | AI | Recall | Actual | AI | Recall | Actual | AI | Recall |
| Railsem19[10] | 8417 images 1920 × 1080 | 34 | 32 | 94% | 141 | 131 | 93% | — | — | — |
| The Illinois Central Mainline | 2h15min 1920 × 1080 30fps | 16 | 15 | 93% | 2 | 1 | 50% | 3 | 3 | 100% |
| CTA Brown Line | 1h20min 1280 × 720 30fps | — | — | — | 2 | 2 | 100% | 50 | 46 | 92% |
| CTA Purple Line | 1h55min 1280 × 720 30fps | — | — | — | 7 | 6 | 86% | 4 | 4 | 100% |
| CTA Orange Line | 1h5min 1280 × 720 30fps | — | — | — | 1 | 1 | 100% | 6 | 6 | 100% |
| CTA Blue Line | 1h30min 1280 × 720 30fps | — | — | — | 1 | 1 | 100% | 1 | 1 | 100% |
| CTA Pink Line | 1h16min 1280 × 720 30fps | — | — | — | 2 | 1 | 50% | 1 | 1 | 100% |
| CTA Green Line | 1h7min 1280 × 720 30fps | — | — | — | 1 | 1 | 100% | 2 | 2 | 100% |
| Total | 10h28min of videos and 8417 images | 50 | 47 | 94% | 157 | 144 | 92% | 67 | 63 | 94% |

*Trespassing and obstacle detection system.* A software tool was developed in this research to wrap the previously described models and detect trespassers and obstacles. To evaluate the system's performance, all images of the RailSem19 dataset were processed by this system. In these images, the track area is identified and segmented into different colors, and hyperparameters are tuned to ensure maximum performance. Few instances of trespassers were present in the image datasets. However, there were many instances of track workers, and the detection of these people was used as a benchmark for the system's performance. Table 2 is a summary of trespassing/obstacle detection accuracy in the RailSem19 dataset. (Table 3 in the appendix shows the annotations.) For videos recorded in the same setup, we evaluate the range of detection by taking the average of distances between the camera and the objects once they were detected. The distances are measured over Google Maps by referring to landmarks. The results show that the anticipated ranges of detecting objects are approximately 30 m for CTA videos and 40m for Illinois Central videos. Some obstacles and trespassers were missed because they were too small in the footage, highly overlapped with other objects, or were unclear because of insufficient illumination. The recall in some videos is disproportionately low due to the small total sample size.

In addition to customizing the model, we also built YOLOv5 with TensorRT, which optimizes AI models for running on Nvidia GPUs, and built Deeplab with TorchScript, which can serialize AI models implemented with the PyTorch framework model to accelerate inferencing speed. After optimizing, the overall system reaches 54 FPS in a windows 10 operating system with a Nvidia GTX 1080 graphics card and 14.5 FPS in Jetson TX2 (a computationally lean edge-computing device).

Figure 7(a) and (b) show examples of pedestrians who are identified in the track and in the foundling area, where they are marked as trespassers. Figure 7(c) shows an example of a case when track workers were in the restricted area instead of the track area. Trespassers in the restricted area could be detected in the same way. Figure 7(d) shows the termination of a railroad track with the system identifying the discontinuation as an obstacle. Such detections can be sent as an alert to locomotive engineers. This further demonstrates the system's ability to detect obstacles when there are no predetermined objects present in the video frame. Original images are from the RailSem19 dataset.

### *Discussion The performance of our proposed system is satisfactory based on a literature review*

Compared to related studies in the prior literature using the same metric, the mAP@0.5 of our object detector is 98% is higher than others,[17,18] which is reasonably good. The performance of our semantic model is in compared to that was reported in,[12] is lower than,[19] but has much higher inferencing speed than both of them. In addition, we tested the system on public videos and released the annotations when trespassing events or obstacles occurred in those videos, which could be of use to other researchers in the future.

## *Edge cases*

Results shown in the above subsections were mainly from sunny daytime footage. We conducted tests under various railway operating conditions, including inclement weather, night/low lighting conditions, and operations inside tunnels to determine the robustness of the system.

When it is rainy, as shown in Figure 8(a), the rail area (the boundary of the track area) in the image can be distorted by rain droplets on the windshield. Heavy snow can obscure the track area, leaving only the rails visible, as shown in Figure 8(c). Although these conditions cause inaccurate detection of the restricted area, the system achieves more than 90% MIoU in the track area detection with only slightly inaccurate expansions or contractions, as shown in Figure 8(b) and (d). In practice, obfuscations due to weather conditions are cleared by windshield wipers and would not persist or cause false positives.

When the train is operating in an environment without sufficient illumination, headlights or night-vision cameras are required by the system to detect obstacles and trespassers. Figure 8(g) is an image gathered by an infrared camera wherein the bottom-right part of the rail is not visible. The system can still precisely detect the visible rail and makes an approximation of the non-visible part, as shown in Figure 8(h), (d) and (f) show the track area detection in low light scenarios.

During inclement weather or low lighting conditions, obstacle detection in the track area is robust if the rails are still visible. On the contrary, it is more challenging to detect restricted areas. The reason for this discrepancy is that the track area is always bounded by clearly identifiable rails, while the boundaries of restricted areas vary widely. Therefore, trespassing detection within the restricted area is

only available under well-lit conditions. To improve the accuracy in segmenting complicated semantics, expanding the training set is needed.

## Conclusions

This research presents a system to detect trespassers and obstacles in forward-facing railroad video in real time using DL-based computer vision techniques. These techniques include the object detection model YOLOv5 and the semantic segmentation model DeepLab. Compared to prior research, we detect trespassing in the restricted area and propose a novel universal obstacle detection algorithm in the track area. We also customized a lightweight semantic segmentation model by conducting a parameters grid search and optimized the semantic segmentation model through input resolution reduction/restoration before/after the semantic segmentation inference. With the above improvements, the system detects trespassers and obstacles with more than 92% accuracy and 54 FPS in a PC with Nvidia GTX 1080 and 14.5 FPS in Jetson TX2. Furthermore, obstacle detection in the track area is robust during bad weather, at nighttime, or inside tunnels with fair illumination due to the clearly identifiable boundaries of the track area.

In the future, we hope to deploy the system to combine recognition with operator warnings. The main reasons for detection failures are that the objects are too small or far away, and/or that the restricted area is inaccurately detected due to illumination. Expanding the annotation of images with small objects, under bad weather, and in night-vision images would further improve accuracy, especially for detecting the restricted area. In addition, the lack of data of actual trespassers and obstacles prevents comprehensive testing of the system. This could be mitigated by conducting a large-scale railway visual data collection and by simulating and recording trespassers in the field or AI generative models such as Generative Adversarial Networks.

### Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

### Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

### ORCID iD

Xiang Liu https://orcid.org/0000-0002-4348-7432

### References

1. Railroad Administration Federal. *Federal Register*. Vol. 86, https://www.railwayage.com/wp-content/uploads/2021/03/2021-05309.pdf (accessed 2 December 2022).
2. *Ten year accident/incident overview*, https://safetydata.fra.dot.gov/officeofsafety/publicsite/Query/TenYearAccidentIncidentOverview.aspx (accessed 2 December 2022).
3. Zaman A, Liu X and Zhang Z. *Video Analytics for Railroad Safety Research: An Artificial Intelligence Approach*, 97. Transp Res Board.
4. *Fixing America's Surface transportation Act*, https://www.fhwa.dot.gov/fastact/(accessed 2 December 2022).
5. Jones M and Gorham T. Transportation technology center inc. inward- and outward-facing audio and video recordings for transit rail vehicles. *FTA Report No 0200*. Online ahead of print. DOI: 10.21949/1520686.
6. Redmon J, Divvala S, Girshick R, et al. *You only look once: unified, real-time object detection*. ArXiv150602640 Cs, http://arxiv.org/abs/1506.02640 (accessed 29 October 2021).
7. Redmon J and Farhadi A. *YOLOv3: an incremental improvement*. ArXiv180402767 Cs, http://arxiv.org/abs/1804.02767 (accessed 9 November 2021).
8. Bochkovskiy A, Wang C-Y and Liao H-YM. *YOLOv4: optimal speed and accuracy of object detection*. ArXiv200410934 Cs Eess, http://arxiv.org/abs/2004.10934 (accessed 29 October 2021).
9. Jocher G. *YOLOv5 by Ultralytics*. https://github.com/ultralytics/yolov5 (accessed 21 October 2022).
10. Zendel O, Murschitz M, Zeilinger M, et al. RailSem19: A Dataset for Semantic Rail Scene Understanding. In IEEE/CVF conference on computer vision and pattern recognition workshops (CVPRW). Long Beach, CA, USA, 2019, pp. 1221–1229.
11. Ristić-Durrant D, Franke M and Michels K. A Review of Vision-Based On-Board Obstacle Detection and Distance Estimation in Railways. *Sensors* 2021; 21: 3452.
12. Wang Y, Wang L, Hu YH, et al. RailNet: A Segmentation Network for Railroad Detection. *IEEE Access* 2019; 7: 143772–143779.
13. Ristić-Durrant D, Haseeb MA, Franke M, et al. Artificial Intelligence for Obstacle Detection in Railways: Project SMART and Beyond. In: Bernardi S, Vittorini V and Flammini F, (eds). *Dependable Computing - EDCC 2020 Workshops*. Cham: Springer International Publishing, pp. 44–55.
14. Wedberg M. *Detecting rails in images from a train-mounted thermal camera using a convolutional neural network. Master thesis*. Linköping University, 2007.
15. Yu M, Yang P and Wei S. *Railway obstacle detection algorithm using neural network*. South Korea: Busan.
16. Ye T, Zhang Z, Zhang X, et al. Autonomous Railway Traffic Object Detection Using Feature-Enhanced Single-Shot Detector. *IEEE Access* 2020; 8: 145182–145193.
17. Xu Y, Gao C, Yuan L, et al. Real-time obstacle detection over rails using deep convolutional neural network. In IEEE intelligent Transportation Systems Conference. 2019, pp. 1007–1012.
18. He D, Ren R, Li K, et al. Urban rail transit obstacle detection based on Improved R-CNN. *Measurement* 2022; 196: 111277.
19. Wang Z, Wu X, Yu G, et al. Efficient Rail Area Detection Using Convolutional Neural Network. *IEEE Access* 2018; 6: 77656–77664.
20. Weber M, Wang H, Qiao S, et al. *DeepLab2: a TensorFlow library for deep labeling*, http://arxiv.org/abs/2106.09748 (accessed 3 June 2022).
21. Gibney B. *Safety Bulletin*, https://www.dot.ny.gov/divisions/operating/employee-healthsafety/repository/RailroadSafety.pdf (accessed 10 October 2022).
22. Howard A, Sandler M, Chu G, et al. *Searching for MobileNetV3*, http://arxiv.org/abs/1905.02244 (accessed 14 May 2022).

# Appendix

**Table 3.** Ground-truths of testing video (open-source).

| Url | Start time (hh:mm:ss) | duration(s) | Label |
|---|---|---|---|
| CTA Green Line | 07:05 | 5 | Legal occupier |
| CTA Green Line | 01:07:25 | 9 | Obstacle |
| CTA Pink Line | 01:16:05 | 4 | Legal occupier |
| CTA Pink Line | 46:15:00 | 6 | Obstacle |
| CTA Blue Line | 05:20 | 5 | Legal occupier |
| CTA Blue Line | 01:26:00 | 20 | Obstacle |
| CTA Orange Line | 24:21 | 3 | Legal occupier |
| CTA Orange Line | 29.58 | 12 | Obstacle |
| CTA Orange Line | 30.37 | 23 | Obstacle |
| CTA Orange Line | 34:42 | 10 | Legal occupier |
| CTA Orange Line | 51:03 | 2 | Legal occupier |
| CTA Orange Line | 54:14 | 5 | Legal occupier |
| CTA Purple Line | 15:20 | 8 | Legal occupier |
| CTA Purple Line | 15:44 | 4 | Legal occupier |
| CTA Purple Line | 28:24 | 12 | Obstacle |
| CTA Purple Line | 48:08 | 14 | Obstacle |
| CTA Purple Line | 50:20 | 30 | Obstacle |
| CTA Purple Line | 57:56 | 14 | Obstacle |
| CTA Purple Line | 01:04:47 | 37 | Obstacle |
| CTA Purple Line | 01:06:30 | 20 | Obstacle |
| CTA Purple Line | 01:55:30 | 15 | Obstacle |
| CTA Brown Line | 11:09 | 2 | Legal occupier |
| CTA Brown Line | 12:00 | 9 | Legal occupier |
| CTA Brown Line | 12:26 | 7 | Legal occupier |
| CTA Brown Line | 13:00 | 7 | Legal occupier |
| CTA Brown Line | 13:30 | 4 | Legal occupier |
| CTA Brown Line | 16:22 | 5 | Legal occupier |
| CTA Brown Line | 16:40 | 9 | Legal occupier |
| CTA Brown Line | 16:55 | 4 | Legal occupier |
| CTA Brown Line | 17:52 | 8 | Legal occupier |
| CTA Brown Line | 18:08 | 5 | Legal occupier |
| CTA Brown Line | 23:45 | 5 | Legal occupier |
| CTA Brown Line | 25:55 | 20 | Legal occupier |
| CTA Brown Line | 31:06 | 4 | Legal occupier |
| CTA Brown Line | 31:22 | 2 | Legal occupier |
| CTA Brown Line | 46:09 | 5 | Obstacle |
| CTA Brown Line | 52:03 | 3 | Legal occupier |
| CTA Brown Line | 56:18 | 5 | Legal occupier |
| CTA Brown Line | 56:42 | 5 | Legal occupier |
| CTA Brown Line | 58:56 | 2 | Legal occupier |
| CTA Brown Line | 01:04:41 | 4 | Legal occupier |
| CTA Brown Line | 01:06:17 | 7 | Legal occupier |
| CTA Brown Line | 01:06:30 | 4 | Legal occupier |
| CTA Brown Line | 01:08:28 | 22 | Legal occupier |
| CTA Brown Line | 01:08:52 | 5 | Legal occupier |
| CTA Brown Line | 01:16:15 | 4 | Legal occupier |
| CTA Brown Line | 01:20:09 | 21 | Obstacle |
| The Illinois Central Mainline | 01:03 | 5 | Trespasser |
| The Illinois Central Mainline | 01:57 | 3 | Legal occupier |
| The Illinois Central Mainline | 17:47 | 3 | Trespasser |
| The Illinois Central Mainline | 18:05 | 3 | Trespasser |
| The Illinois Central Mainline | 32:47 | 4 | Trespasser |
| The Illinois Central Mainline | 33:45 | 4 | Trespasser |
| The Illinois Central Mainline | 46:58 | 9 | Obstacle |

*(continued)*

**Table 3.** (continued)

| Url | Start time (hh:mm:ss) | duration(s) | Label |
|---|---|---|---|
| The Illinois Central Mainline | 57:22 | 3 | Trespasser |
| The Illinois Central Mainline | 01:03:47 | 3 | Trespasser |
| The Illinois Central Mainline | 01:04:21 | 3 | Trespasser |
| The Illinois Central Mainline | 01:10:00 | 2 | Trespasser |
| The Illinois Central Mainline | 01:12:40 | 3 | Trespasser |
| The Illinois Central Mainline | 01:12:47 | 3 | Trespasser |
| The Illinois Central Mainline | 01:13:14 | 5 | Trespasser |
| The Illinois Central Mainline | 01:15:51 | 4 | Legal occupier |
| The Illinois Central Mainline | 01:30:05 | 4 | Trespasser |
| The Illinois Central Mainline | 01:43:36 | 7 | Trespasser |
| The Illinois Central Mainline | 01:47:59 | 4 | Trespasser |
| The Illinois Central Mainline | 01:59:38 | 5 | Obstacle |
| The Illinois Central Mainline | 02:12:47 | 5 | Trespasser |