Railway Asset Detection and Geolocation

by

Georgios Karagiannis

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science
University of Copenhagen

Summary

The increasingly complex modern railways require advanced management systems to automate costly and time consuming maintenance processes. Such systems incorporate detailed maps of the railway assets such as signs, signals, control boxes etc. In addition, railway maps are the foundation of railway simulators used for training locomotive operators. Currently, development and update of these maps is carried out by trained personnel through manual inspection of multi-sensor data. Recent advances in machine learning and computer vision have paved the way for designing deep learning detection models able to achieve very high accuracies in challenging tasks. The performance of these methods has attracted the interest of the industry providing reliable automatic solutions to complex and large scale problems such as railway mapping. This dissertation proposes a pipeline that automates two main tasks of mapping development:(i) Image based object detection and (ii) 3D object localisation. For the detection task, we apply state-of-the-art deep learning detection algorithms in panoramic images. Then, we combine the image based detections with known 3D camera positions to estimate the 3D positions of the objects.

Resumé (Danish)

Den voksende kompleksitet af moderne jernbaner kræver avancerede styringssystemer for at automatisere dyre og tidskrævende vedligeholdelsesprocesser. Sådanne systemer indeholder detaljerede kort over f.eks. skilte, signaler, kontrolbokse osv. Kortene kan desuden udgøre grundlaget for jernbanesimulatorer, som bruges til uddannelse af lokomotivoperatører. I øjeblikket udføres og opdateres sådanne kort manuelt af uddannet personale påbasis af multisensordata. Den seneste udvikling af maskinlæring og datamatsyn har banet vejen for at designe modeller til dyb læring (eng. Deep Learning) til billedbaseret detektering. Sådanne metoder har vist sig i stand til at opnåen meget høj nøjagtighed. Dette har tiltrukket industriel interesse for at levere pålidelige automatiske løsninger til store komplekse opgaver som f.eks. kortlægning af jernbaner. I denne afhandling foreslås en pipeline, der automatiserer to hovedopgaver i kortlægning: (i) Billedbaseret objektdetektering og (ii) 3D-objektlokalisering. Til detekteringsopgaven anvendes avancerede algoritmer til "Deep Learning" baseret påpanoramabilleder. Derefter kombineres de billedbaserede detekteringer med kendte 3D-kamerapositioner for at estimere 3D-positionerne for objekterne.

Preface

This thesis was prepared at the Image Analysis, Computational Modelling and Geometry (IMAGE) section of the department of Computer Science at the University of Copenhagen. It constitutes a partial fulfilment of the requirements for the degree of Doctor of Philosophy (PhD) at the University of Copenhagen.

The PhD research has been funded by COWI A/S and the Innovation Fund Denmark under the industrial Ph.D. program, grant number 5189-00160B. The research was performed under the main supervision of Associate Professor Søren Ingvor Olsen (University of Copenhagen), Søren Andersen (COWI A/S) and the co-supervision of Kim Steenstrup Pedersen (University of Copenhagen) and Søren Findsen (COWI A/S).

The work has been carried out at COWI A/S in Lyngby and at the University of Copenhagen, Computer Science department, omputational Modelling and Geometry section from January 2017 to February 2020.

Acknowledgements

First and foremost, I would like to thank my supervisors. Many thanks to Søren Olsen, the principal supervisor from the university. Søren took the load of supervision and with a lot of patience, he supported and guided me the last three years. He was always alarmed to keep the project on track and always respectful to my unconventional working practices. I would never be able to do this without his guidance and patience. Thanks to Kim Pedersen, the secondary supervisor from the university. Despite his extremely busy schedule, he was always available and ready to solve so many of my problems. I also want to thank Søren Findsen, my secondary supervisor from COWI. He has been very supportive from the beginning. Whenever I asked for help or data, he did his magic and came back to me in a few hours. Finally, a deep thank you to Søren Andersen, my principal supervisor from COWI. He is the main reason that this PhD created in the first place and the reason I have been part of it. Søren, in a very natural way, made always sure that all parties are satisfied with the progress and provided me every tool available to ensure that I will complete this project as planned. He has always been a friend and mentor to me and I am deeply grateful to him.

Many thanks to COWI and the Innovation Fund Denmark for funding this research. Special thank you to all the people in COWI Mapping and Geoservices for their continuous support and ideas. It is great to feel that every person in the department is interested in the progress and success of the PhD.

Thanks to my beloved friends here in Denmark. Alessandro, Alex, Antoine, Danae, Gaia, Javier, Jesus, Lefteris thank you very much for your support and your

understanding all these years. It is an honour to be a your friend.

Thanks to my girlfriend, Nadine. Her smile, her patience and her love have been invaluable during the last three years. Thank you for doing this with me and thank you for being you.

Finally, a special thanks to my family back in Greece who has consistently and unconditionally supported me in anything I have decided to do. Μαμά, μπαμπά και Νίκο σας ευχαριστώ πολύ για όλα.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Railways have been continuously renewed and improved, optimising and consolidating their operations. Examples of major complex design activities taking place today, in Denmark as well as abroad, include large scale modernisation of signal systems and electrification of major railway lines [1]. After completion, these increasingly complex systems are handed over to operational organisations subject to sharp demands for rationalisation and efficiency, which translates into more tasks to be handled by a constant or decreasing number of staff. The response to both challenges are advanced management systems, which incorporate a detailed representation of the network, its assets and surroundings. Such systems in particular facilitate automation of processes and improved decision support, minimising the requirement for expensive and inefficient on-site activities [1]. A fundamental requirement of such management systems is detailed and current maps and databases of railway assets, such as poles, signs, wires, switches, cabinets, signalling equipment, as well as the surrounding environment including trees, buildings and adjacent infrastructure [1]. The detailed maps may also form the basis for a simulation of the railway view as seen from the locomotive driver viewpoint. Such simulations/videos are used in the training of locomotive operators and support personnel. Ideally, the maps should be constantly updated to ensure currency of the databases as well as to facilitate detailed documentation and support

of maintenance and construction processes in the networks.

1.2 Objectives

Currently, mapping of railway assets is largely a manual and highly labour intensive process, limiting the possible levels of detail and revisit times [1]. By automating the asset mapping process, the cost may be reduced significantly and the mapping can be applied more regularly with improved quality and consistency. This project aims to employ advanced algorithms from the fields of computer vision and machine learning to automate major parts of the asset map production process. Recent advances in the fields of computer vision and machine learning [2–4] offer compelling prospects that it is possible to overcome existing limitations and realize considerable automation of the mapping process. We aim to develop, test and demonstrate a prototype of a system for railway asset mapping with a high accuracy in detection, recognition, and localization on available data with manually labelled ground truth. We endeavour to automate a major fraction of the manual work, leaving only final verification to human specialists, while improving the quality and production speed of the output. To ease manual post processing, high detection rate is required whereas a somewhat high false detection rate is accepted.

To our knowledge, no such system exists and only little research has been carried out in the specific field. Railway asset detection is an application almost ignored in the literature. Different from e.g. the Google Street-view 360 degree image capture and modelling, the interesting objects for railways are not (large) buildings, but tiny structures like signs, signals,poles and wires. Detection and accurate localization of such objects is challenging but only sporadically investigated in the literature. While asset maps are relevant in many contexts, railways in their complexity and importance to society are a particularly significant case. A number of recent industrial projects in COWI (the company who partially funded this PhD), with highly resolved input datasets and requirements for extreme detail, provided the conditions

for exploring and evaluating new technological solutions to overcome the challenges. The state-of-art in image-based object detection and classification is very advanced and competitive [5–7].

While the focus of this thesis is on railways, mapping of railway assets is not very different in concept to mapping of assets related to other major infrastructure, such as roads and power lines where similar requirements for improved management systems exist. We expect this project to provide a significant knowledge gain about the procedures for automation of manual work. One may therefore regard this PhD as an initial step towards a general unlocking of opportunities and benefits in a far wider context for industrial applications in COWI.

1.3 Pipeline of industrial railway projects

Here we will describe briefly how a railway mapping project is currently conducted in COWI. This is important because the focus of this PhD is to fully or partially automate parts of these processes. The first step of such projects is data acquisition. A set of panoramic cameras mounted on a train wagon travel along a railway capturing images continuously. These images are then stitched and afterwards, trained operators mark objects of interest on them. Simultaneously, the operators find the location of each detected object on an aerial image. It is important to note that the main purpose of the panoramic images is to help operators identify the objects on the aerial images. Detection on the aerial images are preferred because they are georeferenced with high accuracy (below 10 centimetres) [1]. This is why for these projects there is no need for a very high GPS accuracy on the panoramic cameras. Here, for the dataset on which we performed our experiments on geolocation, the camera positions are known with two metres accuracy. The next step of the process is the classification of the objects according to strict specifications provided by the client. The high similarity of railway objects makes this process time consuming. The full manual process from the detection on panoramic images, the identification on aerial imagery to the final

classification can take up to 7 minutes per image depending on the amount of objects present [1]. Given that an image is captured every 5 metres, a railway segment might consist of hundreds of thousand images [1]. Taking into account processing time per image we can see that the cost of such projects is extremely high. The main reason behind the amount of time spent per image is the demand on high accuracy for the final result. The requirements on completeness is 99% and on 3D position it is 10 centimetres [1]. These numbers make clear why such projects are so labour intensive and therefore expensive.

The partial or full automation of some of the steps will lead to a crucial reduction in cost and delivery times. The full automation of the process is the ideal scenario but given the requirements in accuracy it is not yet feasible to the best of our knowledge. However, a partial automation remains important. A system that can detect automatically the objects on the panoramic images, even with lower than 99% accuracy can save thousand man-hours. The reason is that the operators will spend less time on each image only correcting inaccuracies. In addition, if an automatic object detector can be combined with an accurate geolocation algorithm, the man-hours will be minimised.

Finally, depending on the purpose of each railway mapping project there might be additional steps in the process. One common type of project is the creation of a railway simulator whose purpose is to train locomotive operators. The railway mapping consists a part of such projects. The next steps include Computer Generated Imagery (CGI) and are out of scope for this thesis. Figure 1.1 shows a view of how the final simulation looks like and figure 1.2 a panoramic image based on which the simulator was created.

1.4 Thesis Outline

In the following chapters, we will first present previous and current, relevant work on the fields of object detection, classification and localisation. Next, we will present the

Figure 1.1: Simulation of a railway in Brisbane, Australia [8]

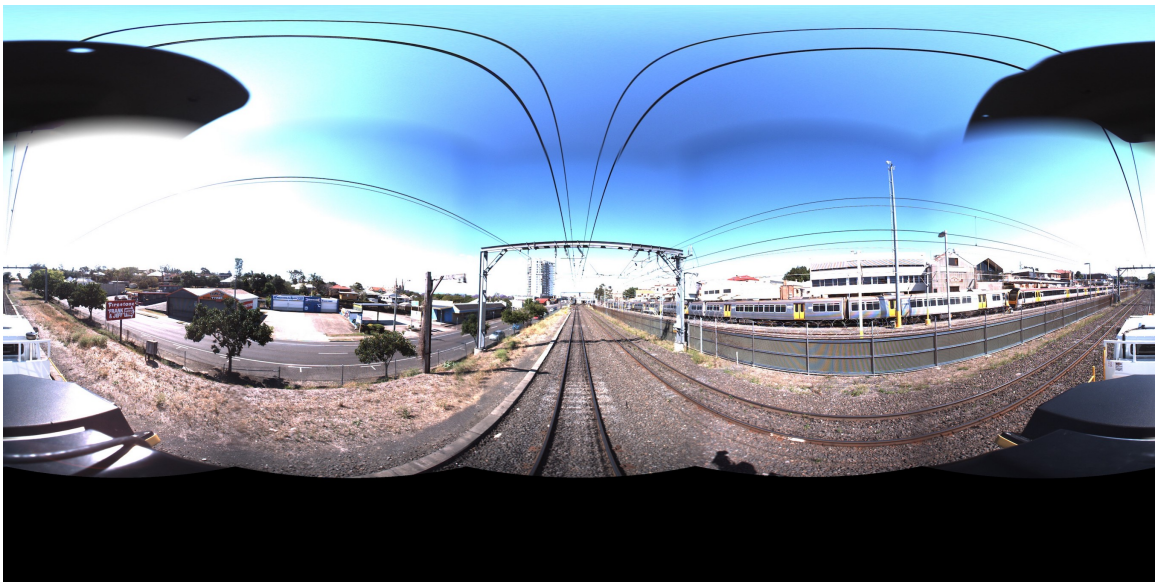


Figure 1.2: Example of panormamic image of a railway in Brisbane, Australia used for the development the railway simulator shown in figure 1.1

two datasets on which we performed our experiments. The data consists of panoramic images obtained from trains for projects carried out by COWI in the past years. . In the following chapter we introduce our work on detection of signs and signals based on state-of-the-art detection algorithms. Afterwards, we will present our suggested

solutions to the problem of 3D localisation of detected objects based on the camera 3D positions. The chapters of the object detection and 3D localisation consist of our scientific contributions in the original layout of the journals in which they have been published/submitted. Finally, we will discuss our conclusions, the challenges of the tasks and suggestions for future work.

Chapter 2

Background

In this chapter, we will present relevant scientific work on the two main topics that concern this thesis in general and in association to railway asset mapping: automatic object detection and object geolocation. Both areas are of very high scientific interest resulting in a high publishing rate. This high rate turns state of the art methods of today to be outdated in a time span of a few months or even a few weeks. For this reason, we will focus primarily on scientific methods available to us at the time we worked on each task and in addition we will discuss the most recent findings on these topics.

2.1 Object Detection

Automatic object detection in images is one of the oldest topics of interest in computer vision [9]. The last decade this field was revolutionised by the development of deep learning methods [2, 3, 10–26] . These methods have outperformed significantly older traditional approaches [27] in competitions such as ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [5]. For this reason, on our experiments in this thesis, we used only deep learning methods and thus, we will discuss only such approaches.

A milestone in the deep learning era on object detection algorithms has been the Regional-based Convolutional Neural Network (R-CNN) presented by Girshick *et al.*

in 2014 [10]. The authors suggest a *two-stage* detection method. First, they generate 2000 *region proposals* on the original image using selective search [28] where it is most likely to find an object of interest. Then, these regions are fed to a Convolutiona Neural Network (CNN) which acts as a feature extractor and performs object classification and detection. This approach was the base for a number of significant methods including its descendants Fast R-CNN [11], Faster R-CNN [2] and Mask R-CNN [29]. These methods have shown very high accuracy in competitive challenges such as ILSVRC [5], COCO[6] and PascalVOC [7]. Two-stage detectors show high accuracy compared to other methods, however, they are computationally expensive. Faster R-CNN, which is among the fastest two-stage detectors, needs one second to process an image of 500×500 image [16].

In parallel, researchers have focused on improving the speed of object detectors towards real-time processing by introducing *one-stage* detectors. One-stage detectors treat object detection as a simple regression problem by taking an input image and learning the class probabilities and bounding box coordinates. The most prominent one-stage detectors are YOLO [14] and SSD [12] with their descendants DSSD [13], YOLO9000 [15] and YOLOv3 [16]. These approaches can process from about 40 [12] up to 60 [16] frames per second which makes them significantly faster than the two-stage detectors. However, the improvement in speed comes at a cost in detection accuracy, especially in small sized objects [3]. In COCO dataset, Faster R-CNN achieves 36.2 % mean Average Precision (mAP) while YOLO9000 only 21.6 % and SSD 31.2 %. In large objects Faster R-CNN achieves 52.1%, YOLO9000 35.5 % and SSD 49.8 %. However, in small objects, Faster R-CNN achieves 16.2% compared to 5.0% and 10.2% of YOLO9000 and SSD respectively [3]. One-stage detectors experience this drop in accuracy mainly due to extreme class imbalance encountered during training between "easy" and "hard" foreground and background samples. In general, the majority of the pixels correspond to background and only few to the objects we would like to detect. Also, the background objects are more easily identifiable (sky,

grass, sea etc.) compared to more complex objects that we may be interested to detect (pedestrians, cars, signs). The loss functions used in most one-stage detectors treat all training samples with the same weight. Therefore, during training, they are overwhelmed by the vast majority of "easy" background or foreground samples defying the more "hard" objects. This problem was tackled in RetinaNet [3] by introducing the *Focal Loss*. The loss function used in RetinaNet introduces a balancing parameter to the *cross-entropy* loss function which multiplies the loss with the inverse class frequency. This way high populated classes do not dominate training. Moreover, the authors introduced a focusing parameter to the cross-entropy loss which adds high weight to hard foreground samples and little weight to the easy ones. RetinaNet, a one-stage detector, based on Focal Loss has achieved higher accuracy than two-stage detectors in COCO (40.8 % overall mAP and 24.1 % on small objects) at a higher processing rate (5 frames per second).

All the aforementioned approaches achieve detection based on the concept of region proposals. Faster R-CNN [2] introduced the concept of *anchors* as an alternative to region proposals towards higher efficiency. Anchors are a set of bounding boxes of different yet predefined scales and aspect ratios which are selected according to statistical characteristics of the dataset. They represent the scales and aspect ratios that the objects of interest have. So, instead of looking for any aspect ratio at any scale for an object, based on the training dataset, we roughly predefine the shapes that the objects are expected to have. In Faster R-CNN the authors use three different scales at three different aspect ratios each, resulting in nine anchors in total. The concept of anchors was later adapted by most detectors (eg. YOLO9000, SSD and RetinaNet). Recently, anchor-free detectors have shown promising results both in terms of accuracy and speed [25, 30–39]. These approaches are focused on the detection of *anchor-points* [31, 32, 35–37, 39] or *key-points* [25, 33, 34] rather than anchor boxes. Anchor-points are pixels on the feature maps of the CNN associated with the features at their locations at each level of the feature pyramid. Based on

the distance of the anchor points from the boundaries of the bounding boxes, the model can encode or decode the bounding boxes accordingly[30]. On the other hand, key-point detectors predict different key-points of the bounding boxes such as corners [25], the centre [33] or extreme points [34]. These key-points are then used to create a bounding box. These methods are all one-stage detectors (except [37]) their speed rate between 15 frames per second [30] to 5 frames per second [33] in COCO. Also, they achieve very high accuracy rates ranging from 38.2% [34] to 41.0% [30] mAP.

For this thesis, we performed experiments in our datasets based on some [2, 3, 11–15, 17, 20] of the aforementioned approaches. Others [19, 22–24, 30, 33, 34, 36–39] were presented at a time after we had concluded our experiments for the detection task and therefore we did not perform any experiments based on them. More details for Faster-RCNN, YOLO, SSD and RetinaNet follow in chapter 4, where we present thoroughly the methods and our experiments.

2.2 Object Geolocation

The estimation of the 3-dimensional (3D) world coordinates of objects combining known camera positions and image coordinates of objects is a concept that traces back to the 1980's [40]. It spans across multiple scientific fields such as Photogrammetry, Remote Sensing and Computer Vision. Essentially, we would like to estimate projection matrices and 3D points which project to image points with a minimum reprojection error. Hence, it is a multi-view *3D reconstruction* problem [41]. A very popular method to tackle this problem is called *bundle adjustment* [41], an iterative optimisation algorithm who aims to adjust the bundle of rays connecting the 3D objects and the centres of the cameras. It is usually the last step for most feature based 3D reconstruction approaches [41–51]. The minimisation problem posed in bundle adjustment is solved by applying optimisation algorithms such as Levenberg–Marquardt (LM) [52, 53] or Powell's Dog Leg (DL) [54, 55]. Depending on the amount of cameras and objects involved the optimisation can become extremely costly and in order to

address this drawback, a sparse version of LM is applied [41].

A specific case of a 3D reconstruction problem is when a controlled and calibrated set of cameras is not available. This is known as a *Structure from Motion* (SfM) problem and the goal is to estimate 3D structures from 2D views by finding correspondences among the different views. There are multiple techniques to find these correspondences with the most prominent being feature detector algorithms such as Sclae-Invariant Feature Transform (SIFT) [56] and Speeded-Up Robust Features (SURF) [57]. Another form of 3D reconstruction worth mentioning is *Simultaneous Localization and Mapping* (SLAM) [58]. This approach is applied when neither the environment nor the location of the camera is known, however it is out of the scope of this thesis.

More details and our experiments regarding bundle adjustment and 3D reconstruction are provided in chapter 5.

Chapter 3

Data

3.1 General Information

Two datasets were used in all our implementations. The images have been acquired and stitched by COWI on behalf of their customers. We have obtained the rights to use these datasets from COWI and the their particular customers. However, the datasets are not publicly available. Both datasets are from Australia, they show different railways and were acquired from different sets of cameras. They share some similarities but have important differences. The first dataset, which is from Brisbane, Australia was obtained in 2013 using a *ladybug 3* set of cameras [59] (Figure 3.1). The second shows a part of the railway from Melbourne to Sydney, Australia. Here, the images were acquired using a custom made set of cameras (Figure 3.2). The first dataset was used to train and validate the object detection and classification models while the second was used mainly for the geolocation task of this thesis.

An important note related to both datasets is that they were acquired and processed for the purpose of industrial projects in COWI and not for our project. As we mentioned in section 1.3, the main purpose of these datasets is to ease the identification of objects on aerial images, there was no intention to automatically detect objects on them or perform a 3D reconstruction. The simplicity of this purpose brings several inconveniences to our project related to stitching, annotations and camera positioning accuracy.

Stitching: First, the objective of stitching was to have good looking images. Thus, there was no interest in possible geometric inaccuracies. The only information available to us regarding stitching is the software used. For the first dataset the ladybug 3 SDK [61] was used and for the second, the open source software *Hugin* [62]. Both software have numerous user-defined parameters and we were unable to retrieve any information on the selected values in both cases. Thus, we cannot present any specific information regarding the stitching process on any of the datasets here. In sections 3.2 and 3.3 we will point to the specific inaccuracies that we discovered on images from visual inspection.

Figure 3.1: Ladybug3 omnidirectional camera system [59].

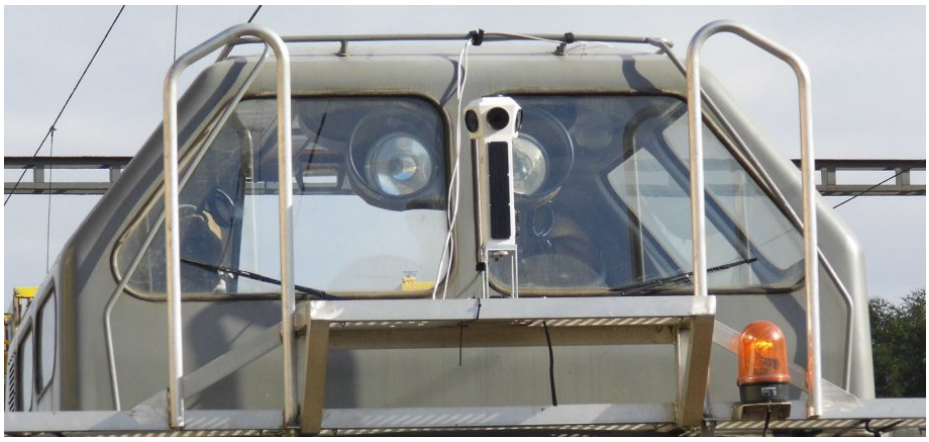


Figure 3.2: Custom made omnidirectional camera system mounted on the train locomotive[60].

Annotations: The annotations on both datasets have been carried out manually by trained operators in COWI India Private Ltd. (CIL). The operators annotated only the panoramic images eliminating the possibility of using the raw images on our experiments (the images before stitching). Also, each dataset annotated by multiple operators resulting in inconsistencies in class labelling and the borders of the bounding boxes. We manually corrected most of these inconsistencies.

Finally, in the next two sections we will present the two datasets. They are also presented in our publications (sections 4.2, 4.3 and 5.2). Therefore, the information in this chapter overlaps at some extent with the data sections on each publication.

3.2 First Dataset

This is the main dataset used in this project. It consists of 47,912 360° images of 5400×2700 pixels and contains in total 121,528 samples of signs and signals combined. Figure 3.3 shows an image form this dataset at full resolution. The position of the objects in this figure is representative of the whole dataset. The majority of the objects (76 %) are located at the centre or at the two sides along the horizontal axis of the image. In total, 42% of the objects are located at the edges of the image and 34% around the centre. Moreover, the objects located at the edges of the image appear smaller in size compared to the ones at the centre because they are more distant from the camera (see small yellow bounding box in figure 3.3 at the right edge compared to yellow bounding boxes at the centre). A heat map of all the available samples shows that the position of the objects along the vertical axis of the image range from just above the centre and below. This allowed us to remove parts of the image where no objects are expected to be and reduce the size of the image on our experiments to 5400×1957 pixels.

The objects are separated in 25 classes which belong into three main categories: signs, signals and position lights (green, yellow and red bounding boxes in figure 3.3). Ten classes represent signs, ten signals and five position lights. In total, 53,689

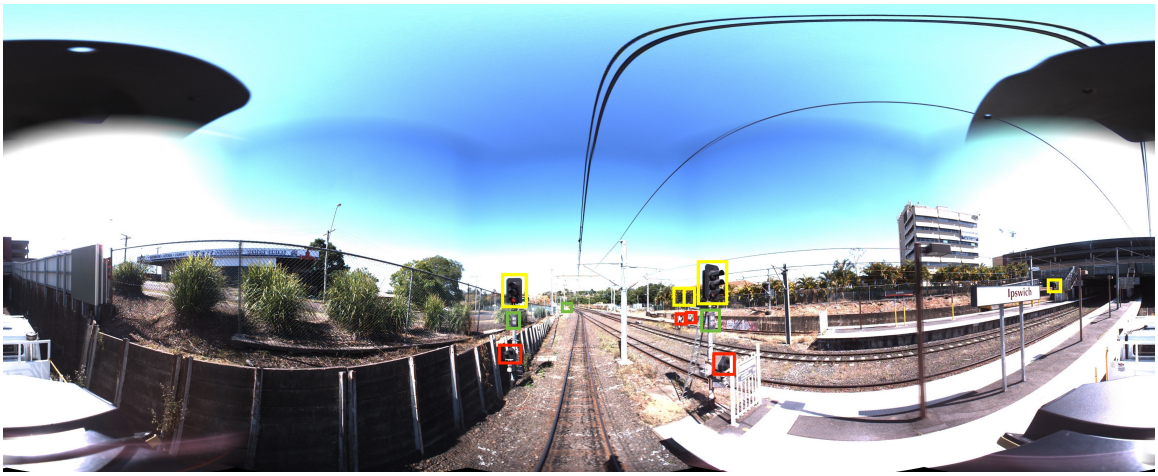


Figure 3.3: Example of image of the main dataset. Bounding boxes with different colours represent objects of different main class categories: green-signs, yellow-signals and red-position lights.

Figure 3.4: Samples of each sign class. From left to right in first row: speed sign, diverging right speed sign, diverging left speed sign, diverging left and right speed sign. From left to right in second row: other type of signs, utility sign, back view of diamond shaped sign, back view of circular shaped sign, back view of rectangular shaped sign, signal identification sign.

samples correspond to signs, 59,427 to signals and 8,412 to position lights. Figures 3.4, 3.5 and 3.6 show examples of each different class. For clarity we used large samples on these examples. The average size of samples is significantly smaller.

The interclass similarity is apparent on the examples shown in figures 3.4, 3.5 and 3.6, resulting in a challenging classification task. Specifically, the signs on the first row in figure 3.4 differ only by a small extension (attached arrows). This extension can be barely distinguished in small samples. Similarly, the position lights shown in

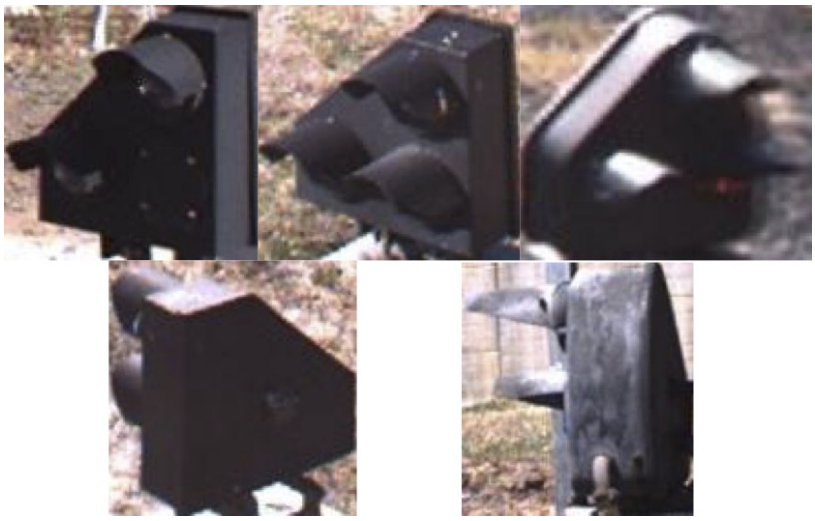


Figure 3.5: Samples of each position light class. From left to right in first row: position light with two elements, position light with three elements, position light with two elements and letter. From left to right in second row: back view of any position light, side view of any position light.

Figure 3.6: Samples of each signal class. From left to right in pairs for both rows: single element signal front view (top) and side view (bottom), double element signal front view (top) and side view (bottom), triple element signal front view (top) and side view (bottom), quadruple element signal front view (top) and side view (bottom), other signals (top) and back view of any signal (bottom).

figure 3.5 have all the same shape and differ only by a small element at the bottom right side where on the first type there is nothing, on the second there is a light and on the third a letter. The interclass similarity on signals classes is akin to position lights. From a distance, the differences among the two, three and four element signals are almost indiscernible.

An unusual characteristic of this dataset is that there are classes representing the back views and side views of objects. A back view class may include samples of objects corresponding to different front views. For instance, there are three different front view classes of position lights but one side view class and one back view class for all of them. The reason is that from a side or back view it is only possible to identify the object as position light but cannot specify which one exactly. The usefulness of these classes is to improve the possibilities of detection of all the objects of interest even with classification inaccuracies. The reason is that in industrial railway projects like this handled in COWI, completeness is unconditionally mandatory for the client. Thus, detecting the objects of interest is significantly more important than accurate classification. This convention also alleviates with the effect of interclass similarities discussed previously. For the same reason, the toll of increased amount of false positives is acceptable when it comes with increased detection rate. Though, this statement is treated as a guideline and we cannot specify a threshold on acceptable *precision* (eg. the number of correct positive predictions divided by the total number of positive predictions). This dataset is presented thoroughly in sections 4.2 and 4.3.

3.3 Second Dataset

The second dataset consists of panoramic images along a railway in Melbourne, Australia acquired in 2017. In the dataset there are 1,755 unique objects each of them observed on average in 4 images. In total, there are 8,210 images where at least one object of interest is present. The size of the images is significantly larger compared to the previous dataset (8192×4096). The higher resolution of the images on this dataset resulted in significantly larger samples compared to the ones of the first dataset (110^2 and 30^2 pixels size on average respectively). However, the samples of this size are still considered small compared to samples in competitive datasets [6]. Figure 3.7 shows an image example of this dataset. A significant difference compared to the first dataset (figure 3.3) is the camera set-up. In this case the camera is

Figure 3.7: A sample image of the second dataset. The main difference with figure 3.3 is the higher resolution. We can also see that the blending of the individual images is smoother in this case.

mounted in front of the locomotive and not on top of it. This reduces the amount of instances seen for each object because the train blocks the objects that appear at the centre of the image.

This dataset was used for the geolocation task of this project. The reason is that for the first dataset there was no GPS information available. It is significantly smaller compared to the first, however, it was possible to perform the full pipeline of this thesis (detection - classification - geolocation) only on this dataset. On the other hand, the ground truth annotations were created based on a different class guide which has no one-to-one correspondence to the first dataset. Thus, we could not validate the performance of the models trained on one dataset to the other. In figure 3.8, we show an example of the different classification standards between the two datasets. The classification on the first dataset is object oriented while on the second it is structure oriented. This difference between the two datasets shows that depending on the project and the client specifications, the classification standards may vary.

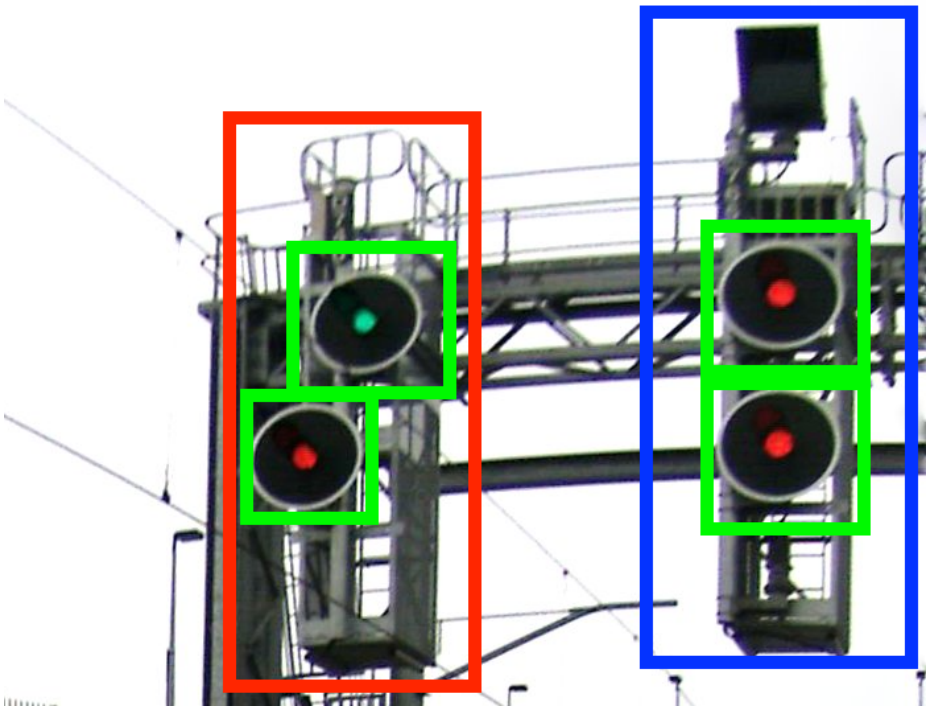


Figure 3.8: Difference in classification standards between our two datasets. For the standards of the first dataset, we see four single-element signals in this image (green bounding boxes). For the standards second dataset, we see two different structures (red and blue bounding boxes).

To overcome this problem we manually converted the classes to match some of the classes on the first dataset and we fine-tuned the trained models on this dataset. Specifically, this dataset consists of only front views of objects (no side or back view classes). Also, because of the smaller size of the dataset (9,317 compared to 121,528), we had no samples available for the class *diverging left and right speed sign*. For these two reasons, the classes on this dataset are 13 instead of 25: single, double, triple and quadruple element signal front view (4 classes for signals); position light with two elements, position light with three elements, position light with two elements and letter (3 classes for position lights); speed sign, diverging right speed sign, diverging left speed sign, other type of signs, utility sign, signal identification sign (6 classes for signs). Figure 3.9 shows the amount of samples available for each class. We can see that there is significant imbalance among the different classes. To achieve

a better distribution, we augmented the samples of the least populated classes (all three position light classes, diverging right speed sign, diverging left speed sign, other type of signs and utility sign). Specifically, on each sample of these classes we applied two random crops and two rotations. Also, this dataset was mainly used for the geolocation task and the only object detector we applied on this dataset (RetinaNet [3]) is robust against class imbalance (more details follow in chapter 4).

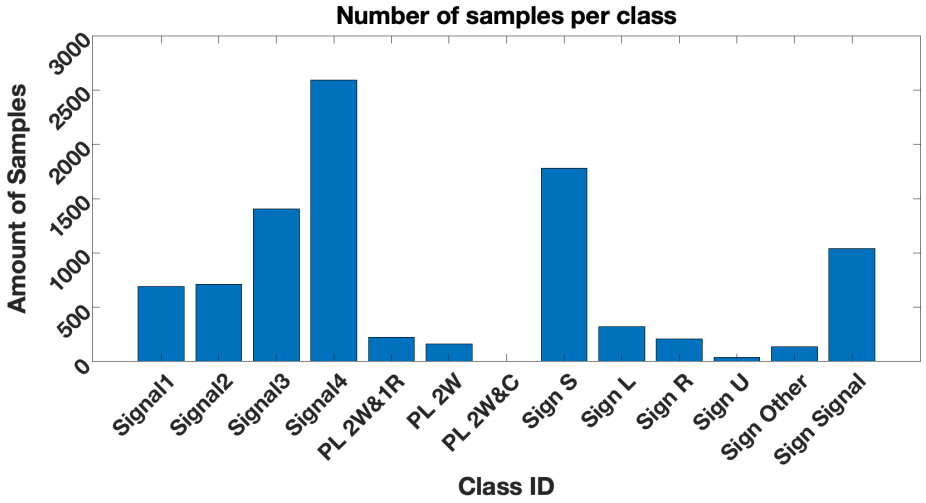


Figure 3.9: Availability of samples per class for the second dataset.

The GPS accuracy of the camera positions on this dataset is two metres on average. However, very often the GPS is higher than two metres. High positioning error is observed when objects such as trees, buildings or bridges interfere with the signal, especially in areas around train stations. In such cases, the GPS error is above ten metres and sometimes above thirty metres. Fortunately, the accurate positions of the track centrelines were available from an independent source. The accuracy of the centrelines is in the order of centimetres and thus, they were used in order to further improve the camera positions. When the train travels in the open, away from stations and urban areas, there are only two tracks and most often the camera positions match the track centreline. Since the trip of the camera was performed on a single track, we force all camera positions to be on this specific track. This provides a significant improvement on the camera position accuracy, especially in places where

multiple tracks are present like the example shown in Figure 3.10. More details on this dataset follow in section 5.2.

Figure 3.10: Camera positions (green dots) along with the track centrelines (red lines). The blue line is the track along which the camera was moving. We can see that the green dots are not on top of the blue line. The positions shown have an average perpendicular distance of 8 metres from the track. On the left side of the figure we mark some extreme positions with a black circle. We ignored such positions on our computations.

Chapter 4

Object Detection

4.1 Introduction

Detection of the railway signs and signals has been the main focus of this thesis. To solve this task, we focused solely on deep learning state of the art methods based on the achievements of methods on competitive detection challenges such as COCO [6] and ILSVRC [63]. For this thesis the achievement of highest possible accuracy has been significantly more important than optimum efficiency. Given this principle, our first approach was the implementation of Faster R-CNN, a method which at the time outperformed other state of the art methods in terms of accuracy with a high toll on efficiency. Afterwards, we implemented three more methods with varying accuracy/-efficiency ratios in order to determine the most appropriate method or combination of methods suitable for our task. In total, we implemented on the first of the two datasets presented in chapter 3 the following methods: YOLO, SSD, Faster R-CNN and RetinaNet. Our experiments resulted in two scientific publications (one still under peer review). In the following two sections (4.2 and 4.3) we present these two publications. In chapter 6, section 6.1 we discuss our conclusions on sign and signal detection.

4.2 Deep Learning for Detection of Railway Signs and Signals

This paper has been presented in Computer Vision Conference in Las Vegas, USA on April 2019. Later it was published in the Springer magazine *Advances in Computer Vision*:

G. Karagiannis, S. Olsen, and K. Pedersen, "Deep learning for detection of railway signs and signals", in *Advances in Computer Vision*, K. Arai and S. Kapoor, Eds., Cham: Springer International Publishing, 2020, pp. 1–15

Deep Learning for Detection of Railway Signs and Signals

Georgios Karagiannis[1,2], Søren Olsen[1], and Kim Pedersen[1]

[1] University of Copenhagen,
Department of Computer Science,
2200, Copenhagen, Denmark,
geka@di.ku.dk, ingvor@di.ku.dk, kimstp@di.ku.dk
[2] COWI A/S, Parallelvej 2, 2800, Lyngby, Denmark,
geks@cowi.com

Abstract. Major railway lines need advanced management systems based on accurate maps of their infrastructure. Asset detection is an important tool towards automation of processes and improved decision support on such systems. Due to lack of available data, limited research exists investigating railway asset detection, despite the rise of Artificial Neural Networks and the numerous investigations on autonomous driving. Here, we present a novel dataset used in real world projects for mapping railway assets. Also, we implement Faster R-CNN, a state of the art deep learning object detection method, for detection of signs and signals on this dataset. We achieved 79.36% on detection and a 70.9% mAP. The results were compromised by the small size of the objects, the low resolution of the images and the high similarity across classes.

Keywords: Railway, object detection, object recognition, deep learning, Faster R-CNN

1 Introduction

The ever increasing modernisation of signal systems and electrification of major railway lines lead to increasingly complex railway environments. These environments require advanced management systems which incorporate a detailed representation of the network, its assets and surroundings. The aim of such systems is to facilitate automation of processes and improved decision support, minimising the requirement for expensive and inefficient on-site activities. Fundamental requirements are detailed maps and databases of railway assets, such as poles, signs, wires, switches, cabinets, signalling equipment, as well as the surrounding environment including trees, buildings and adjacent infrastructure. The detailed maps may also form the basis for a simulation of the railway as seen from the train operator's viewpoint. Such simulations/videos are used in the training of train operators and support personnel. Ideally, the maps should be constantly updated to ensure currency of the databases as well as to facilitate detailed documentation and support of maintenance and construction processes in the

networks. However, with currently available methods, mapping of railway assets is largely a manual and highly labour intensive process, limiting the possible levels of detail and revisit times. The response to this challenge is to automate railway asset mapping based on different sensor modalities (2D images or 3D point clouds) acquired from ground or air.

Despite the high demand for automatic asset detection along railways, there is very little research on this field[1]. Here, we present an approach on detection of signs and signals as a first step towards automatic generation and update of maps of railway environments. We implent an object detection model based on Faster R-CNN (Region-based Convolutional Neural Network) presented by Ren *et al.* in[2] on a dataset used to map a railway of 1,700 kilometres in 2015. The mapping was carried out manually from a private company[3] with a lot of experience in such projects. Currently, many such projects exist around the world and to the best of our knowledge are still carried out manually (people going through all images and mark objects of interest). Our approach aims to show the performance of an advanced object detection algorithm, such as Faster R-CNN, on a novel dataset used in a real world project.

2 Literature Review

2.1 Previous approaches

The research on automatic object detection along railways is sparse, compared to the analogous, popular field of road furniture detection, mainly due to the lack of available railway traffic data[1]. Most of the research is focused on passenger detection[3, 4] or track detection[5–9] for different purposes. The limited research existing focused on detection of only a single type of object (sign recognition[10], sign detection[11] or wire detection[12]).

Marmo *et al.* [10] presented a classical approach for railway signal detection. It is focused on detecting a specific type of signals (single element) in video frames and classify it according to the colour of the light (green-pass, red-no pass). The implementation is based on simple image processing techniques such as histogram analysis, template matching and shape feature extraction. The method resulted in 96% detection accuracy and 97% classification accuracy in a total of 955 images which is impressing for this type of approaches. The advantage of this method is efficiency, however it is focused on a very specific type of signals and the examples presented are scenes of low complexity.

Arastounia[12] presented an approach for detection of railway infrastructure using 3D LiDAR data. The approach is focused on detection of cables related to the railway (i.e. catenary, contact or return current cables), track bed, rail tracks and masts. The data covers about 550 meters of Austrian railways. The approach is based mainly on the topology of the objects and their spatial properties. Points on the track bed are first detected from a spatially local statistical analysis. All the other objects of interest are recognised depending on their spatial relation

[3] Second Affiliation.

with the track bed. The overall average detection achieved by this method is 96.4%. The main drawback of this approach is that it depends on a sophisticated type of data that needs special equipment to capture and it is more complicated to process.

Agudo *et al.* in[1] presented a real-time railway speed limit and warning signs recognition method on videos. After noise removal, Canny edge detection is applied and an optimised Hough voting scheme detects the sign region of interest on the edge image. Based on gradient directions and distances of points on the edge of a shape, candidate central points of signs occur. Then, recognition is achieved by applying shape criteria since the signs are either circular, square or rectangular. The method scored 95.83% overall accuracy on classification. However, even though the dataset had more than 300,000 video frames, only 382 ground truth signs existed and the authors do not provide any score for the detection accuracy.

2.2 Convolutional Neural Networks

All of the above approaches are using traditional image analysis methods to solve object detection problems. To the best of our knowledge, there are no published methods that attempt to solve object detection in railways based on Convolutional Neural Networks (CNNs). CNNs represent the state of the art concept in Computer Vision for classification, detection and semantic segmentation. Regarding object detection, we can divide most CNN-based methods into two categories: region-based and single shot methods. The most characteristic representative of the first is Region-based CNN (R-CNN) and its descendants Fast, Faster and the recent Mask R-CNNs[13],[14],[2],[15]. From the second category, most representative methods are You Only Look Once (YOLO)[16] and Single Shot multibox Detector (SSD)[17]. In general, region based methods are considerably slower but more effective. Also, region based methods show better performance on smaller objects[16, 2]. Given the performance shown in competitive challenges and the fact that our dataset consists mainly of very small objects, we Faster R-CNN[2] consider more suitable for our problem.

3 Data Analysis

3.1 Dataset

In our case, the dataset consists of 47,912 images acquired in 2013 and show the railway from Brisbane to Melbourne in Australia. The images were acquired in three different days in the morning with similar sunlight conditions. The camera used is a *Ladybug3* spherical camera system and the images are panoramic views of size 5400×2700 pixels. The images were processed manually by the production team of the company[4] for annotations and resulted in 121,528 instances of railway signs and signals.

[4] Second Affiliation.

Fig. 1. Instances of signals and signs. First row from left to right (class name in parentheses when it is different from the description): *speed sign* (Sign S), *unit sign* (Sign U), *speed standard letter* (Sign Other), *position light main* (PL 2W), *position light separately* (PL 2W& 1R), *signal with two elements* (Signal2 F). Second row: *diverging left-right speed sign* (Sign LR), *diverging left speed sign* (Sign L), *signal number* (Sign Signal), *signal with one element* (Signal1 F), *signal with three elements* (Signal3 F), *signal with four elements* (Signal4 F).

Fig. 2. Instances of speed signs at different lighting conditions, viewpoints and scales.

The samples are originally separated into twenty five classes. Each class is in fact a subclass of two parent classes, signs and signals. Fifteen classes correspond to signals: three different types of position lights with their back and side view, signals with one, two, three or four elements (lights), their side and back view and other type of signals. Also, ten classes correspond to different types of signs: speed signs, diverging left speed signs, diverging right speed signs, diverging left-right speed signs, unit signs, signal number signs, other signs, back views of circular signs, back views of diamond signs and back views of rectangular signs. From the total amount of samples, 67,839 correspond to signals and 53,689 to signs. Figure 1 shows some instances of signs and signals. Each of these instances correspond to a different class. We can see the high similarity among the classes both for signs and signals. Specifically, diverging left, diverging right, diverging left-right and regular speed signs are very similar and especially when they are smaller than the examples shown in figure 1. Similarly, even for humans it is often hard to distinguish between signals with three or four elements when they are small. All examples shown here are larger than their average size on the dataset for clarity.

Figure 2 shows examples of regular speed signs. with different viewpoint, size and illumination. These examples illustrate the scale, viewpoint and illumination variation of the objects but at the same time the high similarity among the classes.

Fig. 3. Instances of signals with one element (Signal1). All four examples belong to the same class even though they have different characteristics. From left to right: Some have long (first and second), no top cover at all (third) or short cover(last). Also, some have no back cover (first), other have circular (second and third) or semicircular (last)

Figure 3 shows examples of the same class of signals (Signal1). It is important to note that the class is one of the least represented in the dataset with only a few hundred samples available. Though, despite the low availability we can see that there is significant intra-class variability. The same is observed in all the other classes of signals except the ones corresponding to position lights.

Figure 4 shows the amount of available samples for each class. These quantities vary widely for the different classes. For instance, there are about 23,000 samples available for the front view of 4-lamp signals but only a few hundreds for position lights with two lamps or for diverging left-right speed signs. Our dataset reflects the real distribution of objects along a railway, which means that in a railway there exist very few position lights with two lamps and diverging left-right speed signs. Therefore, this level of imbalance among the classes is unavoidable in real applications. However, in deep learning, large amounts of samples are necessary to train a robust model.

A common workaround to ensure adequate balance among classes is to apply some data augmentation techniques (e.g. apply random crops, rotations, scaling, illumination changes etc. on the existing samples). However,such techniques cannot solve the problem, without causing bias to the dataset, in our case because the difference in available samples is too high. Past observations[18], have found that non-iconic samples may be included during training only if the overall amount of samples is large enough to capture such variability. In any other case, these samples may act as noise and pollute the model. Thus, it is necessary for some classes with similar characteristics to be merged. By merging all signs except speed and signal number signs to a general class *signs other*, we get a class with about 30,000 samples. Also, we merged all position lights to a single class, resulting in about 10,000 samples for this class. Finally, the front and side views of each signal class were merged into a single class. The back views of all signals remained a separate class because there was no specific information available on which type of signal each back view belonged to. After these operations, we end up with ten classes of at least a few thousand samples each (figure 5). This way, the misrepresentation problem is softened, however we introduce high intra-class variability. The least represented class is the single element signals

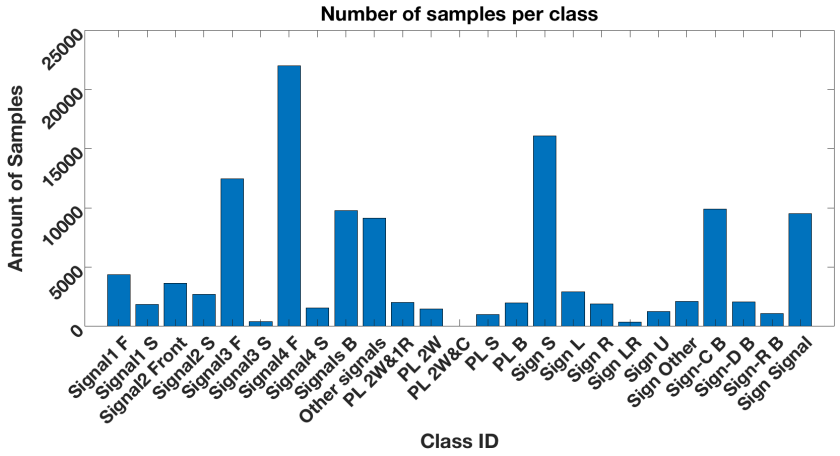


Fig. 4. Amount of sample instances per class before merging some classes. The imbalance among the classes is too high.

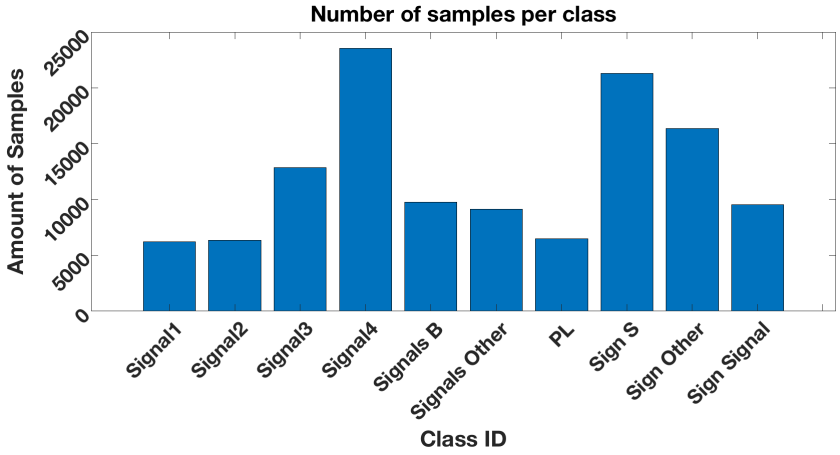


Fig. 5. Amount of sample instances per class. After merging some classes with similar characteristics we end up with a more balanced dataset.

(Signal 1) with about 6,000 samples which is still about four times less than the most dominant class, but more manageable.

Another important aspect of the samples is their size. Figure 6 is a histogram of the size of the samples in square pixels. About 65% of the samples have area less than 1000 pixels ($\approx 32^2$) and 89% less than 2500 pixels (50^2). Given the size of the panoramic images, a 50^2 sample corresponds to 0.018% of the whole image. In COCO[18], one of the most challenging datasets in Computer Vision, the smallest objects correspond to 4% of the entire image. The winners for the 2017 *COCO Detection: Bounding Box* challenge achieved less than 55% accuracy.

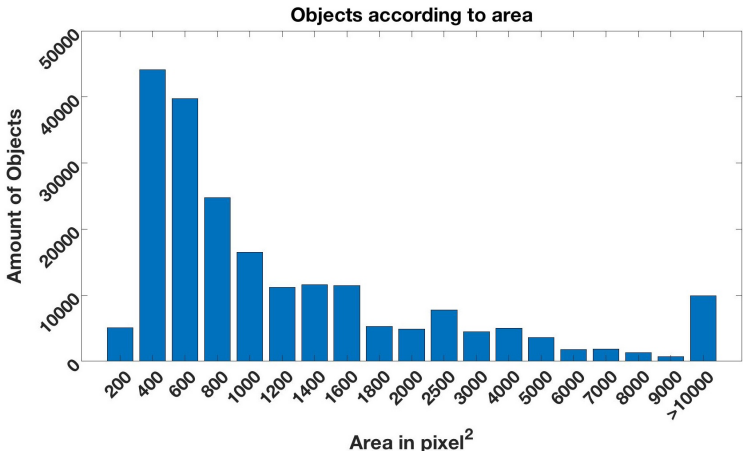


Fig. 6. Amount of samples according to their size in pixel^2. Most samples (89%) are smaller than 50^2 pixels.

This is an indication of the difficulty of our problem, in terms of relative size of objects.

A reason behind the high amount of very small objects in our dataset is that the data was acquired by driving on a single track. However, in many sectors along the railway there are multiple parallel tracks and the signs and signals corresponding to these tracks appear in the dataset only in small sizes since the camera never passed close to them. One way to limit the small object size problem in our dataset is to split each panoramic image into smaller patches with high overlap, small enough to achieve a less challenging relative size between objects and image. Specifically, in our approach, each image is split into 74 patches of size 600^2 pixels with 200 pixels overlap on each side. Even at this level of fragmentation, a 50^2 object corresponds to 0.69% of a patch size. A consequence of splitting the images into smaller patches is that the same object may now appear in more than one patches due to overlap. In fact, while on the panoramic images there exist 121,528 object instances, on the patches that were extracted, there exist 203,322 instances. The numbers shown in Figure 6 correspond to the objects existing on the patches.

3.2 Railway vs road signs and signals

Here, it is important to highlight the difference between the problem described in this paper and the more popular topic of the detection of road signs and signals. The most important difference is the size of the objects. The height of a road sign varies from 60 to 150 centimetres depending on the type of road[19] while in railways it is usually less than 40 centimetres high[20]. Given also that most of the times the signs are located only a few centimetres from the ground supported by a very short pole, it much harder to detect. Also, in railways, signs are often

very similar but have different meaning like the first two examples of the second row in figure 1. At the same time, it is very common along the same railway objects of the same class to look different as shown in figure 3. Thus, a detector of railway signs and signals needs to be able to distinguish objects based on fine details. Finally, in railways the signs and the signals are often combined creating more complex structures posing an extra challenge on a detection algorithm (eg. the detected signals shown in figure 10). Given the above differences, we consider railway signs a more challenging detection problem.

4 Methodology

4.1 Faster R-CNN

For the detection of signs and signals, we applied the Faster R-CNN presented by Ren *et al.* in[2] using ResNet-101[21] as feature extractor. We decided to implement this approach mainly motivated by its high performance on competitive datasets such as Pascal VOC 2007 (85.6% mAP), Pascal VOC 2012 (83.8% mAP) and COCO (59% mAP). The main drawback of this approach compared to other acknowledged object detection methods such as YOLO[16]) or SSD is its high processing time (three to nine times slower depending on the implementation[17]). However, the sacrifice in time pays off in accuracy, especially in this dataset since this method performs better on small objects[2].

Here we will present some key points of Faster R-CNN. First, Faster R-CNN is the descendant of Fast R-CNN[2] which in turn is the descendant of R-CNN[13]. As their names imply, Fast and Faster R-CNNs are more efficient implementations of the original concept in[13], R-CNN. The main elements of Faster R-CNN are: (1) the *base network*, (2) the *anchors*, (3) the *Region Proposal Network (RPN)* and (4) the *Region based Convolutional Neural Network (R-CNN)*. The last element is actually Fast R-CNN, so with a slight simplification we can state that Faster R-CNN = RPN + Fast R-CNN.

The base network is a, usually deep, CNN. This network consists of multiple convolutional layers that perform feature extraction by applying filters at different levels. A common practice[14],[2],[16] is to initialize training using a pre-trained network as a base network. This helps the network to have a more realistic starting point compared to random initialization. Here we use ResNet[21]. The second key point of this method is the anchors, a set of predefined possible bounding boxes at different sizes and aspect ratios. The goal of using the anchors is to catch the variability of scales and sizes of objects in the images. Here we used nine anchors consisting of three different sizes (15^2, 30^2 and 60^2 pixels) and three different aspect ratios $1:1$, $1:2$ and $2:1$.

Next, we use RPN, a network trained to separate the anchors into foreground and background given the Intersection over Union (IoU) ratio between the anchors and a ground-truth bounding box (foreground if $IoU > 0.7$ and background if $IoU < 0.1$). Thus, only the most relevant anchors for our dataset are used. It accepts as input the feature map output of the base model and creates two

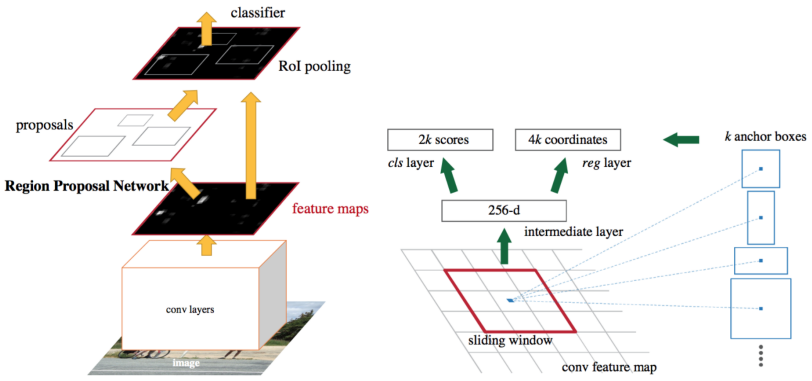


Fig. 7. Right: The architecture of Faster R-CNN. **Left:** The Region Proposal Network (RPN). Source: Figures 2 and 3 of [2]

outputs: a 29 box-classification layer containing the foreground and background probability for each of the nine different anchors and a 4 9 box-regression layer containing the offset values on x and y axis of the anchor bounding box compared to the ground-truth bounding boxes. To reduce redundancy, due to overlapping bounding boxes, non-maximum suppression is used on the proposed bounding boxes based on their score on the box-classification output. A threshold of 0.7 on the IoU is used resulting in about 1,800 proposal regions per image in our case (about 2,000 in the original paper).

Afterwards, for every region proposal we apply max pooling on the features extracted from the last layer of the base network. Finally, the *Fast* RCNN is implemented, mainly by two fully-connected layers as described originally in[14]. This network outputs a $1 \times (N + 1)$ vector (a probability for each of the N number of classes plus one for the background class) and a $4 \times N$ matrix (where 4 corresponds to the bounding box offsets across x and y axis and N on the number of classes). Figure 7 shows the structure of Faster R-CNN and the RPN.

The RPN and R-CNN are trained according to the 4-step alternate training to learn shared features. At first, the RPN is initialized with ResNet and fine tuned on our data. Then, the region proposals are used to train the R-CNN separately, again initialized by the pre-trained ResNet. Afterwards, the RPN is initialized by the trained R-CNN with the shared convolutional layers fixed and the non-shared layers of RPN are fine tuned. Finally, with the shared layers fixed, the non-shared layers of R-CNN are fine tuned. Thus, the two networks are unified.

4.2 Evaluation method

For the evaluation of detection, an overlap criterion between the ground truth and predicted bounding box is defined. If the *Intersection over Union* (IoU) of

these two boxes is greater than 0.5, the prediction is considered as True Positive (TP). Multiple detections of the same ground truth objects are not considered true positives, each predicted box is either True-Positive or False-Positive (FP). Predictions with IoU smaller than 0.5 are ignored and count as False Negatives (FN). Precision is defined as the fraction of the correct detections over the total detections ($\frac{TP}{TP+FP}$). Recall is the fraction of correct detections over the total amount of ground truth objects ($\frac{TP}{TP+FN}$)[22].

For the evaluation of classification and overall accuracy of our approach, we adopted mean Average Precision (mAP)[23] as a widely accepted metric[22]. For each class, the predictions satisfying the overlap criterion are assigned to ground truth objects in descending order by the confidence output. The precision/recall curve is computed and the average precision(AP) is the mean value of interpolated precision at eleven equally spaced levels of recall[22]:

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1\}} p_{interp}(r) \tag{1}$$

where

$$p_{interp}(r) = max_{\tilde{r}:\tilde{r} \geq r} p(\tilde{r}) \tag{2}$$

Then, the mean of all APs across the classes is the mAP metric. This metric was used as evaluation method for the Pascal VOC 2007 detection challenge and from that time is the most common evaluation metric on object detection challenges[18].

5 Results

The network was trained on a single Titan-X GPU implementation for approximately two days. We trained it for 200k iterations with a learning rate of 0.003 and for 60k iterations with learning rate 0.0003. The model achieved an overall precision of 79.36% on the detection task and a 70.9% mAP.

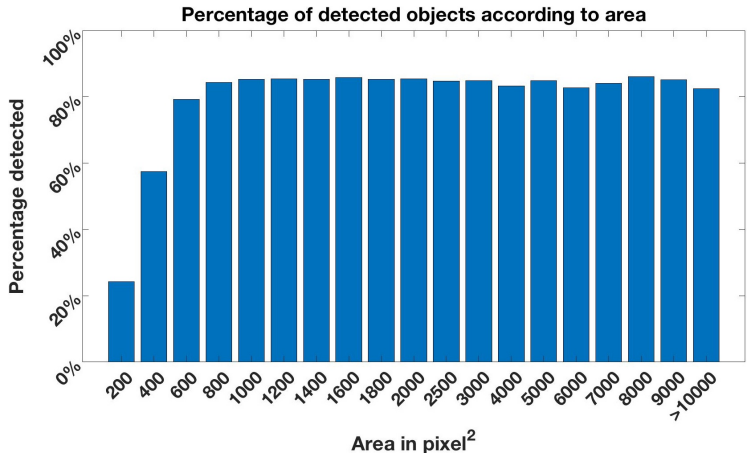


Fig. 8. Percentage of objects detected according to their size in pixels^2. The algorithm performed very well on detecting objects larger than 400 pixels (more than 79% in all size groups). On the other hand, it performed poorly on very small objects (less than 200 pixels) detecting only 24% of them. The performance was two times better on objects with size 200-400 pixels (57%), which is the most dominant size group with about 45,000 samples (see figure 6).

The precision level of detection is considered high given the challenges imposed by the dataset as they are presented in section 3. Figure 8 shows the detection performance of the algorithm with respect to the size of the samples. We can see that the algorithm fails to detect very small objects. About 76% of objects smaller than 200 pixels are not detected. A significantly better, but still low, performance is observed for objects of 200-400 pixels size (57% detection rate). On the other hand, the algorithm performs uniformly well for objects larger than 400 pixels (79% for sizes 400-600 and more than 83% for larger than 600 pixels). These results show that, in terms of object size, there is a threshold above which the performance of Faster-RCNN is stable and it is unaffected by the size of the objects. Therefore, if there are enough instances and the objects are large enough, object size is not a crucial factor. In our case, this threshold is about 500 pixels.

An interesting metric for our problem would be to measure the amount of unique physical objects detected. The goal of object detection in computer vision, usually, is to detect all the objects that appear in an image. In mapping, most of the times, it is important to detect the locations of the physical objects. If we have multiple images showing the same physical object from different point of views and at different scales, in mapping, it would be sufficient to detect it at least in one image and not necessarily in all images. We expect that our approach will perform better in this metric, however, information about unique physical objects is not available for this dataset.

Fig. 9. Example of detection. Two successful detections and one false positive. We can see the illumination variance even in a single image with areas in sunlight and in shadow. The sign is detected successfully despite its small size and the partial occlusion from the bush. The entrance of the tunnel was falsely detected as a signal with three elements.

Figures 9 and 10 show two representative examples of the results. We can see in these figures the variance in illumination conditions and the small size of the objects. The speed signs that appear on these images are very small but they belong to the most common object size interval for this dataset (200 - 400 pixels). By looking at these figures, we can also realise the difficulty of the classification task of this dataset. Even for a human, it is hard to decide on which class the objects on these images belong to.

Figure 11 is a confusion matrix that summarises the performance of the method on classification. The diagonal represents the Average Precision (as de-

Fig. 10. Example of detection. A successful detection of more complex object structures. Different signals mounted on top of each other did not confuse the algorithm.

scribed in section 4.2) for each class and the bottom right cell is the mean of these precisions. A first observation is that there is significant variation in accuracy for the different classes. As expected, the accuracy is decreased for the classes trained with few samples (*position lights*) and for those with high similarities with other classes (*back view of signals*). An interesting exception to this observation is the class *signals with one element* (S1). Despite the low amount of training samples and the fact that all classes of signals show high resemblance among each other, the classification accuracy is above the overall and among the highest.

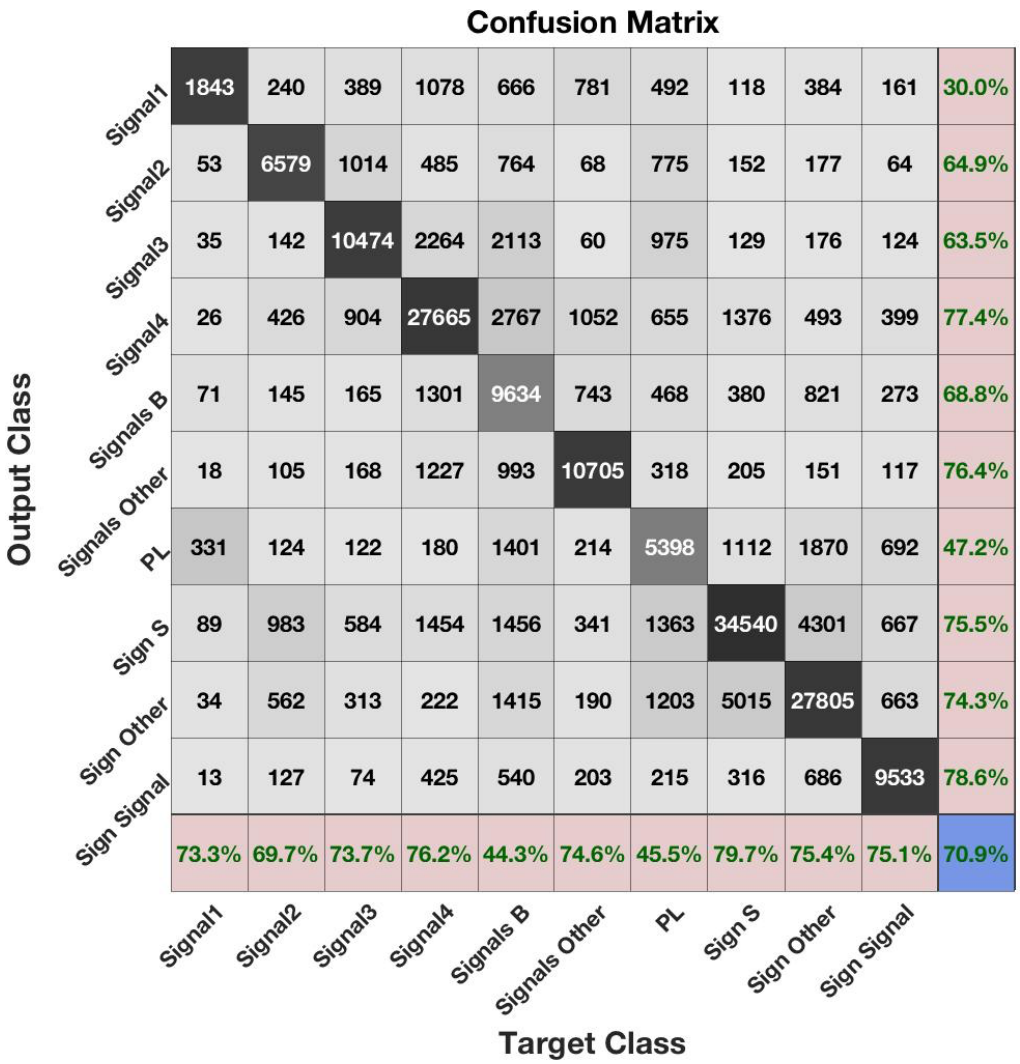


Fig. 11. Confusion matrix showing the classification AP of the detected objects. Rows represent the predictions and columns the ground truth objects. Last row and last column (red colour) summarise the performance per class. Bottom right cell (blue colour) is the mAP of the method. See Figure 2 for class name explanation.

6 Conclusions

In this paper, we presented a novel dataset used in real world projects for mapping of railway infrastructure. The dataset is very challenging mainly because the objects are very small relatively to the size of the entire images but also absolutely in terms of pixels^2. We implemented a state of the art deep learn-

ing method for detection and classification of signs and signals on this dataset scoring 79.36% on the detection task and 70.9% on the classification task. We believe that the accuracy of the model will be higher in a similar dataset with better resolution images. Also, given that the object positions along a railway follow specific regulations, taking into account the spatial relationships among the objects can improve accuracy. Moreover, it would be interesting to apply other state of the art methods on this dataset and analyse the strengths and weaknesses of each method on the same tasks. In addition, for mapping tasks such as the current, a more appropriate metric would be to evaluate detections based on the amount of physical objects and not on objects existing in the images. In conclusion, our results are promising and suggest further investigation into the use of deep learning for railway asset detection and mapping.

References

1. D. Agudo, . Snchez, J. F. Vlez, and A. B. Moreno, "Real-time railway speed limit sign recognition from video sequences," in *2016 International Conference on Systems, Signals and Image Processing (IWSSIP)*, May 2016, pp. 1–4.
2. S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in Neural Information Processing Systems*, vol. 2015-, pp. 91–99, 2015.
3. D. Zheng and Y. Wang, "Application of an artificial neural network on railway passenger flow prediction," in *Proceedings of 2011 International Conference on Electronic Mechanical Engineering and Information Technology*, vol. 1, Aug 2011, pp. 149–152.
4. T.-H. Tsai, C.-K. Lee, and C.-H. Wei, "Neural network based temporal feature models for short-term railway passenger demand forecasting," *Expert Systems with Applications*, vol. 36, no. 2, Part 2, pp. 3728 – 3736, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417408001516
5. N. S.Suresh, R.Prabhu, "Implementation of railway track crack detection and protection," *International Journal Of Engineering And Computer Science*, vol. 6, no. 5, May 2017. [Online]. Available: http://ijecs.in/index.php/ijecs/article/view/3535
6. J. Sadeghi and H. Askarinejad, "Application of neural networks in evaluation of railway track quality condition," *Journal of Mechanical Science and Technology*, vol. 26, no. 1, pp. 113–122, Jan 2012. [Online]. Available: https://doi.org/10.1007/s12206-011-1016-5
7. X. Gibert, V. M. Patel, and R. Chellappa, "Robust fastener detection for autonomous visual railway track inspection," in *2015 IEEE Winter Conference on Applications of Computer Vision*, Jan 2015, pp. 694–701.
8. D. Sinha and F. Feroz, "Obstacle detection on railway tracks using vibration sensors and signal filtering using bayesian analysis," *IEEE Sensors Journal*, vol. 16, no. 3, pp. 642–649, Feb 2016.
9. S. Faghih-Roohi, S. Hajizadeh, A. Nez, R. Babuska, and B. D. Schutter, "Deep convolutional neural networks for detection of rail surface defects," in *2016 International Joint Conference on Neural Networks (IJCNN)*, July 2016, pp. 2584–2589.
10. R. Marmo, L. Lombardi, and N. Gagliardi, "Railway sign detection and classification," in *2006 IEEE Intelligent Transportation Systems Conference*, Sept 2006, pp. 1358–1363.

11. M. Melander and I. Halme, "Computer vision based solution for sign detection," *European Railway Review*, Oct 2016.
12. M. Arastounia, "Automated recognition of railroad infrastructure in rural areas from lidar data," *Remote Sensing*, vol. 11, Nov 2015.
13. R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *Ieee Conference on Computer Vision and Pattern Recognition (cvpr)*, pp. 580–587, 2014.
14. R. Girshick, "Fast r-cnn," *Proceedings (ieee International Conference on Computer Vision)*, vol. 2015, pp. 1440–1448, 2015.
15. K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.
16. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *Ieee Conference on Computer Vision and Pattern Recognition (cvpr)*, vol. 2016-, pp. 779–788, 2016.
17. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," *Lecture Notes in Computer Science*, vol. 9905, pp. 21–37, 2016.
18. T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick, "Microsoft coco: Common objects in context," *Lecture Notes in Computer Science*, vol. 8693, pp. 740–755, 2014.
19. G. B. D. for Transport, *Traffic Signs Manual*. London, United Kingdom: The Stationary Office, 2013.
20. R. Safety and S. Board, *Lineside Operational Safety Signs*. London, United Kingdom: Railway Group Standard, 2009.
21. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Ieee Conference on Computer Vision and Pattern Recognition (cvpr)*, vol. 2016-, pp. 770–778, 2016.
22. M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *Int. J. Comput. Vision*, vol. 88, no. 2, pp. 303–338, June 2010. [Online]. Available: http://dx.doi.org/10.1007/s11263-009-0275-4
23. G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*. New York, NY, USA: McGraw-Hill, Inc., 1986.

4.3 Detection of Railway Signs and Signals

This paper is currently under peer review for publishing in Springer magazine *Machine vision and applications*:

G. Karagiannis, S. Olsen, and K. Pedersen, "Detection of railway signs and signals", *Under peer review in Machine Vision and Applications*, 2019

Noname manuscript No.
(will be inserted by the editor)

Detection of Railway Signs and Signals

Georgios Karagiannis · Søren Ingvor Olsen · Kim Steenstrup Pedersen

Received: date / Accepted: date

Abstract Asset mapping in railways is highly important for railway management systems. Automation is necessary to substitute the current largely manual and highly labour intensive asset detection. Despite the high amount of commercial projects and a multimillion market, research on this topic is limited due to the lack of publicly available data. Here, we apply four state of the art deep learning methods for object detection on a dataset used in a railway mapping commercial project. The challenges of the dataset are the small size of objects, the interclass imbalance on sample availability and the high interclass similarity. We train and test and compare YOLO, SSD, Faster R-CNN and RetinaNet. We achieve results ranging from 55.41% to 86.56% mAP for the different models and computational efficiency ranging from 72 to 174 milliseconds per frame.

Keywords Railway · object detection · YOLO · SSD · Faster R-CNN · RetinaNet

G. Karagiannis
COWI A/S
Parallelvej 2, 2830, Lyngby, Denmark
Tel.: +45-50297619
E-mail: geks@cowi.com

S. Olsen
University of Copenhagen, Department of Computer Science
Universitetsparken 5, 2100, Copenhagen, Denmark
Tel: +45-93 56 57 22
E-mail: ingvor@di.ku.dk

K. Pedersen
University of Copenhagen, Department of Computer Science
Universitetsparken 5, 2100, Copenhagen, Denmark Tel: +45-35 32 14 55
E-mail: kimstp@di.ku.dk

1 Introduction

The importance of railway asset mapping is ever increasing due to increased modernisation and complexity of railways. Advanced management systems are developed to automate maintenance processes and minimise on-site activities which are inefficient and costly. Such systems are dependent on detailed and accurate maps of the railway assets such as signs, signals, poles, control boxes etc. Detailed maps are also used to create simulations of railways from the train operator's viewpoint. Such simulators are used for training train operators therefore, it is important to update them frequently. These two different uses of railway asset maps form a multi million USD market around the world in which COWI A/S is actively involved. Currently, creation and update of detailed railway maps is largely manual thus, labour intensive and very costly. The high cost limits the level of detail and the update frequency of the maps. The solution to this problem is to automate as many processes as possible, starting with detection and geolocation of assets using images.

Despite the high demand for automation of the mapping task and the large market, there is very little research on this field[1]. In this paper, we apply state of the art object detection algorithms to detect railway signs and signals on a dataset used to map the railway that connects Brisbane and Melbourne in Australia. The dataset consists of 47,912 360° panoramic images of size 5400×2700 pixels acquired in 2013 using a Ladybug 3 camera and corresponds to 1,700 kilometres railway. We train, test and compare four different Convolutional Neural Network (CNN) architectures which are acclaimed methods for object detection in the field of computer vision: Faster R-CNN[2], Single Shot multibox Detector (SSD) [3], You Only Look Once [4] and

RetinaNet [5]. The four approaches vary in efficiency and accuracy performances and all of them have won or were ranked high in international object detection challenges such as ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [6] and Common Objects in Context (COCO) [7].

In contrary to railway asset detection, road asset detection is a field in which high amount of research has been conducted for the past decades [8–14]. Thus, it is important to underline the differences between these two similar tasks. First, road signs are much larger in size compared to railway signs. A road sign can have a height from 60 to 150 centimetres [15] while a railway sign can rarely have a height of more than 40 centimetres [16]. Also, a railway sign detector needs to be able to distinguish objects of different classes (which often have significantly different meaning) based on fine details because there is high level of resemblance across different types of railway signs. At the same time, in most classes, there is high intra-class variability. Moreover, railway signs are often supported by very short poles challenging the detection process because they are very close to the ground. Furthermore, along railways, it is very common to see many different objects combined on complex structures sometimes without space among them. In these cases, an automatic detector (even the human eye sometimes) struggles to distinguish the different objects. Examples of the above challenges are shown in Figures 1, 3 and 6 in section 3. These differences combined with the low amount of available data make railway asset detection an arduous task.

This paper aims to show the performance of advanced object detection algorithms on a field with low amount of research conducted, on a novel dataset used in a real world commercial project. We compare the strengths and the weaknesses of these methods for such problems and we propose when it is more suitable to choose one of them.

2 Literature Review

2.1 Previous approaches

Despite the high demand for automated railway sign detection, the research is sparse mainly because at the best of our knowledge there are no annotated publicly available datasets for this field[1]. Computer vision research in railways is mostly focused on passenger detection [17,18], track detection [19–23] or obstacle detection [24] for different purposes, mainly safety. The limited number of existing work is focused on only a specific type of object (sign detection [25,26], sign recognition [27] or wire detection [28]).

Also, the vast majority of the research was conducted before the outburst of CNNs in Computer Vision and the approaches used are classical. Marmo *et al.* [27] detected a specific type of signals (single element) in video frames using template matching, histogram analysis and shape feature extraction. The dataset used consisted of 955 images and they achieved 96% and 97% accuracy on detection and classification respectively. The achieved accuracy is impressive considering the methods used and very efficient. However, the images used were of low complexity and they focused on a very specific type of signal.

Arastounia [28] presented a solution that detects multiple types of objects along railways namely the track bed, the tracks, masts and different types of cables (i.e. catenary, contact or return current cables). The method uses 3D LiDAR data and is tested on half kilometre long railway in Austria. The approach is based on the spatial relationship among these objects and mainly in relation to the track bed. The track bed is detected by applying local statistical analysis and from that each different object is detected depending on their distance in all spatial dimensions from the trackbed and their shape. The method achieved 96.4% accuracy. The main disadvantage of this method is that it is based on a sophisticated type of dataset which is inefficient to acquire for long railways. Apart from the expensive special type of equipment, the acquisition of this type of data needs more time and the railway needs to be inactive during this process. This is the reason that RGB images are preferred over LiDAR on projects of railway asset mapping.

Agudo *et al.* in [1] performed recognition of speed limit and warning signs on videos in real time. Their approach is based on Canny edge detection and an optimised Hough voting scheme to detect the ROI of signs on the edge image. Then, according to the directions of gradients and the distances of points on the edges of the detected shapes they define the centres of the signs. Finally, given that the sign shapes are either square, circular or rectangular, shape criteria are used to recognise the signs. This approach was tested on video with 300,000 frames but only 382 instances of signs and achieved 95.83% on classification. The authors do not provide any accuracy scores on detection.

Finally, even more recent approaches [29,26] that use deep learning methods are focused on very specific objects. Mikrut *et al.* [29] use a CNN to detect a specific type of sign along a railway in Poland and Ritika *et al.* use a CNN to detect a specific type of signal. Both approaches have high detection scores (90% and 94.7% respectively), however, neither of them provide details on the models and architectures used.

3 Data Analysis

3.1 Dataset

The dataset used here consists of 47,912 panoramic images of size 5400×2700 pixels. The images acquired in 2013 using a *Ladybug3* spherical camera system. The reason that the images are panoramic in such projects is to be able to see objects belonging to multiple track lines (opposing or parallel) by a single pass avoiding multiple trips. The annotation process was performed manually by experts resulting into 121,528 instances of railway signs and signals.

The samples are separated in twenty five classes which belong in two main categories: signs (10 classes) and signals (15 classes). The different classes of signs are (in parentheses the way we refer to them in figures and tables in this paper): speed signs (Sign S), diverging left speed signs (Sign L), diverging right speed signs (Sign R), diverging left-right speed signs (Sign LR), unit signs (Sign U), signal number signs (Sign Signal), other signs (Sign Other), back views of circular shaped signs (Sign-C B), back views of diamond shaped signs (Sign-D B) and back views of rectangular shaped signs (Sign-R B). Also, the 15 classes corresponding to signals are: front views of three different types of position lights (PL 2W&1R, PL 2W, PL 2W&C), side views of all position lights (PL S), back views of all position lights (PL B), front and side views of signals with one (Siganl1 F, Signal1 S), two (Siganl2 F, Signal2 S), three (Siganl3 F, Signal3 S) or four elements (Siganl4 F, Signal4 S), side view of any signal (Signals S) and finally, one class for any other type of signals (Signals B). In total, 53,689 samples correspond to signs and 67,839 correspond to signals. In figure 1 examples of different signs and signals are shown. Each of the examples shown belong to a different class and, for clarity, they are larger than the average size of the training samples in the dataset. We can observe the high similarity among classes of the same main category (signs and signals). The three different types of speed signs (speed sign-top row first, diverging left-right speed sign-bottom row first and diverging left sign-bottom row second) shown in figure 1 differ only on the sides where the "diverging" signs have an extension. Even more challenging is the distinction of signals with two, three or four elements (top row-last, bottom row-fifth and bottom row-last), especially, when the intensity values of the background is dark.

Figure 2 illustrates the intraclass diversity of samples for the class of speed signs depending on the different viewpoint, scale and illumination. Similarly, in figure 3 we can see the variation of the single element signals class. In this case, apart from the viewpoint, illumination and scale, there is variation on the overall look of the actual objects due to inconsistent selection of manufacturer. Also, for this class (single element signals) there are only a few hundred samples available thus, the difficulty of grasping the variability is increased. The intra-class variability is a characteristic of all classes of signals.

Fig. 1: Instances of signals and signs. First row from left to right (class name in parentheses when it is different from the description): *speed sign* (Sign S), *unit sign* (Sign U), *speed standard letter* (Sign Other), *position light main* (Position Light 2W), *position light separately* (Position Lights 2W&1R), *signal with two elements* (Signal2 Front). Second row: *diverging left-right speed sign* (Sign LR), *diverging left speed sign* (Sign L), *signal number* (Sign Signal), *signal with one element* (Signal1 Front), *signal with three elements* (Signal3 Front), *signal with four elements* (Signal4 Front).

Fig. 2: Instances of speed signs at different lighting conditions, viewpoints and scales.

Fig. 3: Variability of signals with one element (Signal1). From left to right: Some have long (first and second), no top cover at all (third) or short cover(last). Also, some have no back cover (first), other have circular (second and third) or semicircular (last)

The availability of samples for all 25 classes is shown in figure 4. We can see that the availability of samples varies significantly. For some classes there are only a few hundred samples available while for others more than 10,000. This distribution is a reflection of a real railway where several types of objects appear rarely along it, while others appear very often. However, in order to train a robust machine learning model, it is necessary

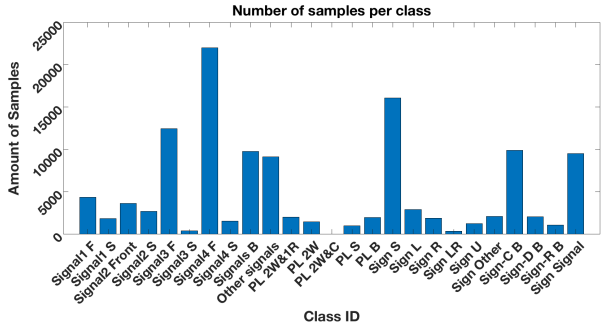


Fig. 4: Amount of sample instances per class showing the high imbalance in quantity among different classes.

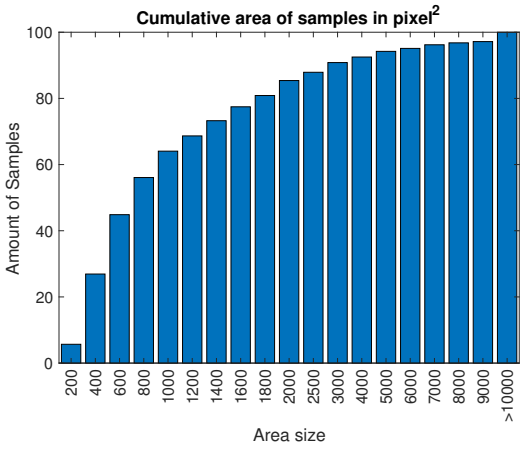


Fig. 5: Cumulative amount of samples according to their size in pixel^2. About 89% of the data samples are smaller than 50^2 pixels.

to have a large amount of samples per class and avoid extreme imbalances in the amount of samples among classes.

In order to increase the availability of data for the classes with the fewest amount of samples, we implemented some data augmentation processes. We applied random horizontal (and vertical for the signs) shear, random horizontal and vertical translations, scaling by down-sampling (for larger objects we used more scales and for smaller less or even no scaling) and brightness variations (in HSV colorspace). After training and evaluating some models, we discovered that the data augmentation was adequate for the models to converge and thus we decided to perform the comparisons without applying any further class balancing techniques, like, for instance, merging some of the classes. This decision comes with the side effect that the intraclass variation is limited to the level of the original dataset.

The most significant characteristic of the samples of our dataset is their size. Figure 5 shows the amount of samples in relation to their area in pixels^2. Most samples in this dataset have size smaller than 32^2 pixels (approximately 65%) and the vast majority (89%) are smaller than 50^2 pixels. The reasons behind the high amount of small samples are mainly two. First, railway signs, that represent almost half of the data, are small objects (less than 40 centimetres high, see Section **??**). Secondly, the data acquisition is carried out from a train, thus, the distance between the camera and the objects is always at least a few meters. This distance is longer for signs given that they exist only a few centimetres above the ground and therefore, they are not visible for the camera when the train is close to them. In addition, the data was acquired by a train along a single track. However, along the railway there are at least two parallel tracks for the different directions and very often more than two parallel tracks, with signs and signals dedicated for each of them. These signs and signals will produce small training samples considering the longer distance between them and the camera.

Small objects in a image are considered objects with small absolute area in pixels^2. However, the authors of the COCO dataset [7] (the most competitive dataset for general purpose detection tasks) claim that it is not only the absolute size of the samples that affects performance but also the relative size of the objects to the total size of the image. Given the size of the images in our dataset (5400×2700), a sample of size 50^2 corresponds to only 0.018% of the entire image. In the COCO dataset, the smallest objects correspond to 4% of the entire image. This indicates that in terms of absolute and relative size, the objects in our dataset are very small.

In order to increase the relative size of the objects, we decided to split the panoramic images in smaller patches (600^2 pixels) with overlap of 200^2 pixels on each direction. In cases where the same object appeared in multiple patches due to the overlap, we kept only one for training to avoid overfitting. Also, in model evaluation, a detection of the same object in multiple patches counted for only one true positive and the rest were ignored (we did not count them as false positives). Now, using this patch size, an object of 50^2 pixels corresponds to 0.69% of the image size. This is circa 14 times more than originally but still it is approximately 6 times smaller than the smallest objects in COCO.

4 Methodology

In this section , we will present the keypoints and our implementations of the four different models, namely

Faster R-CNN, YOLO, SSD and RetinaNet, used in chronological order they were originally published. We have chosen to test these four methods as they are the state of the art in terms of accuracy and efficiency.

4.1 Faster R-CNN

Faster R-CNN [2] is the most representative of two-stage object detection methods. Two-stage detectors are considered those which perform an intermediate task (a region proposal step in this case) in order to produce an output. Faster R-CNN has achieved high performance in competitions such as Pascal VOC 2007 (85.6% mAP), Pascal VOC 2012 (83.8% mAP) and COCO (36% mAP). In comparison with one-stage detectors, such as YOLO, SSD and RetinaNet, the two-stage detectors lack in efficiency. Depending on the implementation Faster R-CNN can be three to nine times slower [3]. The toll in efficiency is balanced with higher accuracy and especially in detection of small objects [2] where YOLO and SSD struggle.

Faster R-CNN is the evolution of its predecessors R-CNN [30] and Fast R-CNN [31]. Fast and Faster R-CNN are more efficient and more accurate versions of the original R-CNN concept. The key elements of Faster R-CNN are: (1) the *base network*, (2) the *anchors*, (3) the *Region Proposal Network (RPN)* and (4) the *Region based Convolutional Neural Network (R-CNN)*. The last element is in fact Fast R-CNN and in a simplified way we can state that Faster R-CNN = RPN + Fast R-CNN.

In [2] Ren *et al.* used as a base network the VGG-16 [32] to produce a feature map. Here we use Resenet-101 which is more accurate and more tolerant to overfitting [33]. Next, we selected anchors appropriate our detection task. Anchors are predefined boxes of different scales and aspect ratios which describe as best as possible the objects we want to detect. The anchors are ranked from the RPN and classified in foreground and background according to their score. In our implementation we use three different sizes of anchors (15^2, 30^2 and 60^2 pixels) and three different aspect ratios ($1:1$, $1:2$ and $2:1$) resulting in nine anchors in total.

The RPN takes as input the feature map output of the base network and classifies the anchors into foreground and background based on their Intersection over Union (IoU) between them and the ground-truth bounding boxes (we use an IoU threshold of 0.7). The outputs of RPN are a 2×9 classification layer and a 4×9 regression layer. The classification layer contains the foreground and background probability for each of the nine anchors and the regression layer the x and y axis offsets between each anchor and the ground truth boxes. To compute the probabilities, the RPN minimizes the objective function shown in Equation 1. In addition, non-maximum suppression is implemented based on the classification output to reduce multiple overlapping bounding boxes.

$$L\left(\{p_i\},\{t_i\}\right) = \frac{1}{N_{cls}} \sum L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum p_i^* L_{reg}(t_i, t_i^*) \quad (1)$$

where p_i is the predicted probability of an anchor in a mini-batch with index i to be foreground, p_i^* is the ground truth label which is equal to 1 if p_i is foreground and 0 otherwise. Similarly, t_i is a vector containing the predicted bounding box coordinates and t_i^* the ground truth coordinates corresponding to a successfully predicted foreground anchor. The classification loss L_{cls} is log loss over two classes (foreground and background and the regression loss ($Lreg$) is $L_{reg}(t_i, t_i^*) = R(t_i, t_i^*)$ where R is the loss function $smooth_{L_1}$ presented in [31] and shown in Equation 2. Classification and regression losses are normalised by N_{cls} and N_{reg}, the mini-batch size and the number of anchor locations respectively. Finally, λ is a parameter that ensures equal weight on classification and regression losses. The reason is that N_{cls} is 32 in our case while N_{reg} is roughly 2400. Therefore, we use a λ equal to 80.

$$R = smooth_{L_1}(x) = \begin{cases} 0.5x^2, & if\ |x| < 1 \\ |x| - 0.5, & otherwise. \end{cases} \quad (2)$$

4.2 You Only Look Once approach

YOLO suggested a new approach for object detection. Redmon *et al.* reframe object detection as a *single regression problem* which is simpler compared to the more complex pipeline suggested on Faster R-CNN. It is a single stage object detection method that computes the bounding boxes and the classification in one pass of the single image. The main strengths and weaknesses of YOLO are [4]:

+ Approaches detection as a regression problem.
+ It is very fast (45 frames per second and 155 fps for the smaller version *Fast YOLO*).
+ Processes the whole image at once (accounts for contextual information of the objects).
+ Shows good generalization performance.
− Struggles with small objects.
− Lacks on accuracy.

The authors of YOLO divide the images in a $S \times S$ grid and for each grid cell they predict B bounding

boxes. For each bounding box, a confidence score is predicted reflecting the confidence that the box contains an object of interest and how accurately the box describes the object. Ideally, when there is no object in the grid cell, the score should be zero, otherwise, it should be equal to the IoU between the bounding box and the ground truth. Thus, each bounding box has 5 values: x and y coordinates of the center of the bounding box, width, height and confidence score. In addition, for each grid cell, a conditional class probability is predicted for each class. At test time, the conditional class probabilities are multiplied by the box predictions producing confidence scores per class for each predicted bounding box:

$$\begin{aligned} Pr(Class_i|Object) \times Pr(Object) \times IoU_{pred}^{truth} \\ = Pr(Class_i) \times IoU_{pred}^{truth} \end{aligned} \tag{3}$$

During training, from the predicted bounding boxes per grid cell, only one is responsible for each object, the one that has higher IoU with the ground truth. This results to *specialised bounding box predicors* because eventually each of them becomes better at predicting specific sizes, aspect ratios or even classes.

This process is implemented by a CNN inspired by GoogLeNet architecture [**?**]. The network consists of 24 convolutional layers and two fully connected layers. The first twenty convolutional layers are used for feature extraction. They are pre-trained on ImageNet [**?**] using half resolution images followed by an average pooling and a fully connected layer. The last four convolutional and the two fully connected are used for detection. They are trained using randomly initialized weights on full resolution images. Finally, the inception modules of GoogleNet are substituted by 1×1 reduction layers followed by 3×3 convolutional layers.

Also, the authors use a *leaky* rectified linear activation function:

$$\phi(x) = \begin{cases} x, & \text{if } x > 0 \\ 0.1 \cdot x, & \text{otherwise} \end{cases} \tag{4}$$

Also, for efficiency they choose to optimise for sum-squared error. This optimisation weighs equally the errors for classification and localisation, but also independently of the bounding box size. To partially address the latter, they compute the square root of the width and height of the bounding box. Moreover, this optimisation approach favours gradients from cells containing objects since the ones that do not contain have confidence close to zero. To avoid this, they use two parameters ($\lambda_coord = 5$ and $\lambda_noobj = 0.5$) to increase the loss for predictions containing an object and decrease predictions without an object. Finally, the following multi-part loss function is optimised during training:

$$\begin{aligned} \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[\left(x_i - \hat{x}_i\right)^2 + \left(y_i - \hat{y}_i\right)^2 \right] \\ + \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[\left(\sqrt{w}_i - \sqrt{\hat{w}_i}\right)^2 + \left(\sqrt{h}_i - \sqrt{\hat{h}_i}\right)^2 \right] \\ + \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left(c_i - \hat{c}_i\right)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} \left(c_i - \hat{c}_i\right)^2 \\ + \sum_{i=0}^{S^2} \mathbb{1}_{i}^{noobj} \sum_{c \in classes} \left(p_i(c) - \hat{p}_i(c)\right)^2 \end{aligned} \tag{5}$$

where $\mathbb{1}_i^{obj}$ denotes if an object of interest appears in grid cell i and $\mathbb{1}_{ij}^{obj}$ represents that the jth bounding box of grid cell i is responsible to that prediction.

In our implementation, we use similar training parameters that the authors of YOLO suggested in [4]. We use a 12×12 grid size (instead of 7×7) and 2 bounding box predictors B per grid cell (same as original YOLO). We choose a different grid size because our image patches are slightly larger than the ones in Pascal VOC (600^2 compared to approximately 500^2) and the objects of interest in our dataset are smaller. The reason is to avoid a known drawback of YOLO. Due to the cell grid approach and the single "responsible" predictor per cell, YOLO struggles with overlapping small objects. By selecting a more dense grid we minimise the amount of multi-class grid cells. Given the number of classes C (10 in our case), we end up with a tensor: $S \times S \times (B \cdot 5 + C) = 12 \times 12 \times 20$. The number of B bounding box predictors is multiplied by 5 because they consist of 5 predictions as we described in the previous section (x, y, width, height and confidence).

Moreover, we use a batch size of 16, a momentum of 0.9 and a decay of 0.0005 as suggested in [4].

4.3 Single Shot MultiBox Detector approach

SSD is asingle-stage approach inspired by some key components of Faster R-CNN. On COCO and Pascal VOC challenges, SSD has shown lower but comparable accuracy with Faster R-CNN while it is significantly faster. SSD uses the concept of predefined anchors presented in Faster R-CNN (they are called *default boxes* in [3]) and a similar loss function but it performs detection and classification simultaneously in a single pass of the image. The most significant characteristic of SSD is that it computes classification scores and bounding box offsets based on multi-scale predictions on multi-scale feature maps. This is achieved by adding convolutional layers of different size to the end of the base network (VGG-16 in the original paper).

The default boxes in SSD are different for feature maps with different resolutions which leads to more accurate prediction bounding boxes. For the m_{th} feature map, the scale of the k_{th} default box is calculated by:

$$s_k = s_{min} + \frac{s_{max} - s_{min}}{m-1}(k-1), \quad k \in [1, m] \tag{6}$$

where s_{min} and s_{max} are the scales of the lowest and the highest layers (0.2 and 0.9 respectively in both our and the original implementations).

In our implementation we use the same aspect ratios as in the original paper (1:1, 1:2, 1:3 , 2:1, 3:1) and one extra default box for the 1:1 aspect ratio whose scale is $s_k = \sqrt{s_k s_k + 1}$, resulting in 6 default boxes for each location in each feature map.

During training and similarly to Faster and Fast R-CNN, SSD minimises the following weighted sum of localisation and confidence losses:

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + L_{loc}(x, l, g)) \tag{7}$$

where N is the amount of correctly matched default boxes (when $N = 0$, the loss is set to 0). The confidence loss ($L_{conf}(x, c)$) between the i_{th} default box and the j_{th} ground truth box is the softmax loss over confidences (c) of multiple classes (p):

$$L_{conf}(x, c) = -\sum_{i \in Pos}^{N} x_{ij}^{p} log(\hat{c}_i^p) - \sum_{i \in Neg} log(\hat{c}_i^0) \tag{8}$$

where

$$\hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)} \tag{9}$$

The localisation loss (L_{loc}) is the loss between the predicted (l) and ground truth (g) boxes. It is the same $smooth_{L_1}$ loss function used in Fast and Faster R-CNN. Regression is performed to offsets for the center (cx, cy), the width (w) and the height (h) of the i_{th} default box d to the j_{th} ground truth box g:

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \sum_{m \in \{cx, cy, w, h\}} x_{ij}^{k} smooth_{L1}(l_i^m - \hat{g}_j^m) \tag{10}$$

$$\hat{g}_j^{cx} = (g_j^{cx} - d_i^{cx})/d_i^w \quad \hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy})/d_i^w \tag{11}$$

$$\hat{g}_j^w = log\frac{g_j^w}{d_i^w} \quad \hat{g}_j^h = log\frac{g_j^h}{d_i^h} \tag{12}$$

A significant problem of this method is the high imbalance between the amount of default boxes corresponding to negative examples and corresponding amount of positive examples because of the background in the image is more dominant than the foreground. To limit this imbalance, the authors suggest a hard negative mining technique which limits the selection of negative boxes to be based on the highest confidence loss. This tactic results in an imbalance of at most 3:1.

4.4 RetinaNet

RetinaNet is signle-stage approach that aims to perform as fast as single-stage detectors such as SSD and as accurately as two-stage detectors like Faster R-CNN. The concept of this model is based on the Feature Pyramid Network (FPN) [**?**] and a new loss function presented in [5], called *Focal Loss* (FL).

The Focal Loss function is based on the Cross Entropy (CE) function and by adding a modulating factor which reduces the contribution of easy examples (examples that have very high probability scores and are easily identified as foreground or background) to the total loss. This twist to the CE function solves a major problem of single-stage detectors, the foreground-background high imbalance. The Cross Entropy function is $CE(p, y) = CE(p_t) = -log(p_t)$ where $y \in \{\pm 1\}$ is the ground truth class, $p \in [0, 1]$ is the probability computed by the model for the class $y = 1$ and p_t equals to:

$$p_t = \begin{cases} p, & \text{if } y = 1 \\ 1 - p, & \text{otherwise} \end{cases} \tag{13}$$

Based on CE, the Focal Loss is then defined as:

$$FL(p_t) = -(1 - p_t)^{\gamma} log(p_t) \tag{14}$$

where $\gamma \geq 0$ is a focusing parameter. The authors in [5] show that the effect of the γ factor has minor impact on the distribution of the loss for positive examples while it has massive impact on the loss distribution of the negative examples, forcing the loss to focus on the hard negative examples. The authors achieved best results with a γ equal to 2. We performed our tests using different γ values (0-3) and have achieved best results with $\gamma = 2.3$. In general, higher γ value means that samples with very high probability have significantly lower contribution to the loss. Also, the authors suggest to use an α-balanced FL where for $y = 1$ the FL is multiplied by a balancing factor α and by $1 - \alpha$ otherwise. In our implementaion the factor α is set as the inverse class frequency in order to limit the class representation imbalance problem of our dataset.

The architecture of RetinaNet is fairly simple. First, a Feature Pyramid Network built on top of a Residual Network (ResNet101 in our case) is used as backbone for feature extraction and two sub-networks are attached to the FPN for classification and bounding

box regression. The FPN creates a multi-scale feature pyramid from the input image using a top-down pathway. Top-down pathway in this particular set-up means that moving down through the convolutional layers C5 to C2 of ResNet, each time we up-sample the previous layer by a factor of 2 using 4 nearest neighbours and apply a 1×1 convolutional filter to them. Afterwards, a 1×1 convolutional filter is applied to the corresponding feature map. Then, we add the convolved convolutional layer and feature map element-wise and apply a 3×3 convolutional filter. This technique improves multi-scale detection and minimizes the aliasing effect.

In FPN, the concept of anchors presented in Faster-RCNN (described in section 4.1) is used. In our implementation at each level of the FPN we use the same set of anchors each time rescaled accordingly (aspect ratios remain the same). Also, at each FPN level two parallel Fully Convolutional sub-Networks (FCN) are attached, one for classification and one for regression. Both sub-networks have four 3×3 convolutional layers followed by ReLU (Rectified Linear Unit) activation functions. The classification sub-network computes for each class at each anchor the probability containing an object and the regression sub-network computes the offset of the predicted bounding box with the ground truth box.

4.5 Summary

Our implementations of the four different methods are performed on the basis of fair comparison and maintaining intact as much as possible the components and parameters suggested by the authors at the original papers. The main change to ensure fair comparison is that for all methods we used ResNet-101 as base network. Besides, Residual Networks were presented after most of these methods and given their advantages over other architectures [33] we believe it is more suitable to use one of them. All the other changes, that we made and described in each subsection, were made to address our specific task and dataset.

All the networks were trained on a single NVIDIA Tesla P100 GPU. All models were trained for 200k iterations with a learning rate of 0.003 and for 60k iterations with learning rate 0.0003. As expected, YOLO was the most efficient among all spending 64 milliseconds to process a frame while SSD needs 72 ms, RetinaNet 121 ms and F-RCNN 174 ms respectively. The frame processing rate of SSD is slightly contradicting to the claims of its authors when compared to SSD. In [3], the authors report that SSD is as fast as fast as YOLO but in all our experiments it was always slightly slower. This difference can be justified from the differences in our implementation since we did not use the source code provided by the authors.

4.6 Evaluation method

The evaluation of the predicted detections for each model is based on overlap between the ground truth and the predicted bounding boxes. A predicted box is considered True Positive (TP) if the *Intersection over Union* (IoU) is greater than 0.5 and the assigned class fits the ground truth class. In case of multiple detections of the same ground truth object, only one prediction is accepted as TP, the rest are considered as False Positives (FP). Predicted boxes with IoU lower than 0.5 are considered False Negatives (FN). The fraction of correct detections over the total amount of detections ($\frac{TP}{TP+FP}$) is the definition of *Presicion* and the fraction of the number correct detections over the number of ground truth objects ($\frac{TP}{TP+FN}$) is the definition of *Recall* [**?**].

As evaluation metric of the overall accuracy of the models we use the mean Average Presicion metric [**?**]. For this metric, class-wise predictions with IoU greater than 0.5 are sorted in descending order according their confidence. Then, for each class, on the precision/recall curve, at eleven equally spaced levels of recall, the interpolated precision is precision is computed (this is the class-wise average precision-AP):

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1\}} p_{interp}(r) \tag{15}$$

where

$$p_{interp}(r) = max_{\tilde{r}:\tilde{r} \geq r} p(\tilde{r}) \tag{16}$$

The mean value of all APs is the mAP expressed as percentage. This is the most common metric used in computer vision detection competitions [**?**,7].

5 Results

Table 1 contains the average precisions per class for the four different models and on the last row of this table, the mAP for each model is shown. We can see that RetinaNet has performed significantly better compared to the rest. A first significant observation is that the models that perform worse (YOLO and SSD) show lower performance on every class, showing that at least class-wise they have no other advantage against F-RCNN and RetinaNet apart from their computational efficiency. In general, the level of detection accuracy is considered high, especially for Faster R-CNN and RetinaNet, given

the challenges imposed by the dataset as they are presented in section 3.

As an indication we compare the relative overall mAP of the four models here to the relative overall mAP of these models on the COCO dataset as shown in [5]. In COCO, YOLO achieved 21.6% mAP, SSD 33.2 %, Faster R-CNN 36.2% and RetinaNet 40.8%. As a general indication, we can say that in COCO, RetinaNet performed about 1.9 times better than YOLO, 1.23 times better than SSD and 1.13 times better than Faster R-CNN. On our dataset, RetinaNet performed 1.56, 1.42 and 1.15 times better than YOLO, SSD and Faster R-CNN respectively. We can say that Faster R-CNN had a similar relative performance, however, YOLO performed significantly better in our dataset and SSD significantly worse. The low performance of SSD is not surprising because our dataset is dominated by small objects which is a drawback for this method according to [3]. For the same reason, the performance of YOLO is surprising since, according to the authors of this method, YOLO struggles to detect small objects [4]. The explanation of the drop in performance of SSD is given in Figure 7 where we can see the percentage of objects that were not detected by each model according to their area. In this figure we can see that YOLO and SSD failed to detect about the same amount of objects smaller than 600^2 pixels. Though, the amount of objects smaller than 600^2 pixels represent almost 50% of the total amount of samples (see Figure 5 in Section 3). Therefore, this size of objects is very small for both YOLO and SSD resulting in a similar relative performance by these two compared to COCO. On the other hand, the relatively higher performance of YOLO can be explained by the lower complexity of our dataset compared to COCO.

Furthermore, in Table 1 we can see that YOLO performed better on signals than signs, SSD shows similar performance on both while Faster R-CNN and RetinaNet perform slightly better on signs. In general, signals are significantly larger objects than signs but the different types of signals differ little from each other. If we combine this information with the findings in Figure 7, we can deduce that YOLO is more affected than Faster R-CNN and RetinaNet by the size of the objects and thus performs worse on signs. At the same time, RetinaNet and Faster R-CNN less affected by the size of the objects show similar performance on the two categories. In summary, the performance of all models in signs is compromised because some of them are too small to detect (RetinaNet and Faster R-CNN are less affected). At the same time, the performance of the models on signals is compromised mainly because of misclassification and not lack of detection (see also below analysis of Table 2).

Also, all models performed significantly worse on position lights. One reason is the resemblance of position lights with signals as we shown in Figure 1 in Section 3. Given the resemblance of these objects we believe that the main reason is the effect of the imbalance in the total amount of samples for each class. The dataset contained ten times more samples of signals than position lights. We believe that this lead to some level of overfit for the signal classes against the position lights. Attempting to solve this problem, we adjusted the balance among classes by removing samples from the most populated classes keeping maximum 5,000 samples per class. However, the models that we trained on the balanced dataset performed worse overall and particularly on the most populated classes with not any significant gain in accuracy on the less populated classes. For this reason, we decided to keep the models presented here despite this drawback.

By looking at the results of specific classes, the most notable scores for all models belong to the class of position lights 2W C. For this class the available samples were less than 100 and the partition of the dataset used for testing contained only eight samples. The results on this class are not informative but they are included for completeness. Another interesting result is the low accuracy of all models on front view of signals with three elements (Signals3 Ft). This class is among the most populated and these signals among the largest objects in size in our dataset. However, it was highly confused with the front views of signals with two and four elements (see Table 2 for distribution of misclassifications). On the other hand, the front views of signals with two and four elements were among the most successfully detected objects. Regarding signs, all models performed well on speed signs but struggled to detect the signal signs despite the fact that this is the third most populated class of signs. We believe that the main reason is because these signs are located on the pole of a signal which is often thick and dark in colour (like this specific type of signs) and makes the signs indistinguishable.

In addition, by observing the class-wise average precisions for each model, we can see that RetinaNet and Faster R-CNN are more stable and have a uniform accuracy across the classes in contrast to YOLO and SSD. The standard deviation of the Average Precisions for each model, excluding the outlier class PL 2W C, is 0.15, 0.14, 0.08 and 0.05 for YOLO, SSD, Faster R-CNN and RetinaNet, respectively. The low values of the standard deviations (especially for Faster R-CNN

Class Name	YOLO	SSD	F-RCNN	RetinaNet
Signals1 Ft	48%	44%	56%	81%
Signals1 Side	30%	31%	72%	84%
Signals2 Ft	75%	74%	83%	93%
Signals2 Side	86%	85%	89%	95%
Signals3 Ft	67%	66%	76%	81%
Signals3 Side	56%	60%	71%	85%
Signals4 Ft	87%	86%	91%	94%
Signals4 Side	62%	66%	77%	85%
Signals Back	64%	67%	80%	87%
Other Signals	39%	76%	89%	92%
mAP Signals	**61.39%**	**65.52%**	**78.41%**	**87.74%**
PL 2W 1R	59%	60%	75%	85%
PL 2W	45%	48%	72%	89%
PL 2W C	12%	12%	12%	50%
PL Side	29%	36%	66%	82%
PL Back	58%	60%	78%	86%
mAP PL	**40.62%**	**43.24%**	**60.62%**	**78.38%**
Speed Sign	77%	77%	85%	92%
Sign L	58%	67%	82%	91%
Sign R	60%	66%	83%	94%
Sign LR	52%	70%	81%	91%
Sign U	42%	53%	72%	88%
Sign Other	55%	61%	76%	87%
Circular B	68%	69%	81%	88%
Diamond B	54%	79%	89%	94%
Rectangle B	50%	59%	74%	89%
Signal Sign	53%	53%	69%	81%
mAP Signs	**56.93%**	**65.43%**	**79.19%**	**89.56%**
mAP	**55.41%**	**61.04%**	**75.16%**	**86.56%**

Table 1: Average precision per class for the four different methods.

and RetinaNet) is very important because they prove that these models are very robust.

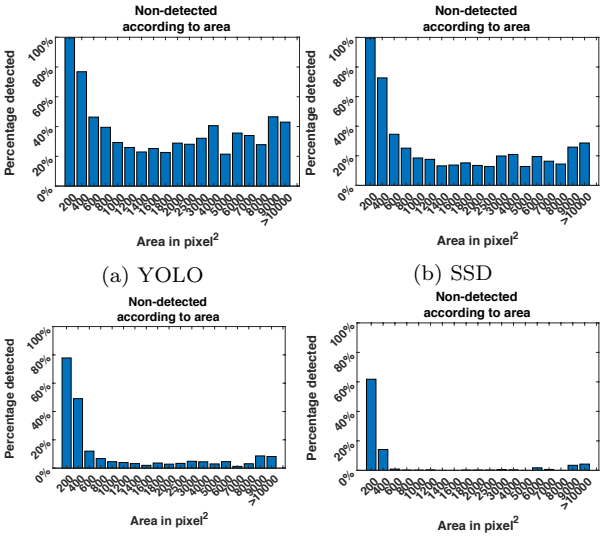
Table 2 is the distribution of the misclassified false positives per class of the RetinaNet model. The table shows, for each class, the three most popular classes at which the false positives were assigned to. The percentage in the parentheses refers to the total amount of the false positives for this specific class. For this table, we considered only the model predictions that accurately detected an existing object with IoU higher than 0.5, but classified it incorrectly. Based on the statistics of this table, we can claim that, in most cases, the model falsely assigned, to a successfully detected object, a class with high resemblance with the correct one. For all the front and side views of signals, the model assigned the front view of another signal. The level of resemblance among the different types of signals, especially when observed from long distance, is so high that it is hard even for a human to classify them successfully. However, in a railway mapping project, these objects belong to different classes and despite the level of similarity it is important to classify them appropriately otherwise, the produced map will be inaccurate.

Another interesting observation is that for the back views of the signals there is not a class that dominates the false positives. The three most dominant classes range from 12-15%. Interestingly, the two most dominant of the three are not other signal classes but the back views of circular and rectangular shaped signs. The explanation for this is that even though the front views of signals are painted black, their back views are often painted gray as the back sides of the signs. Moreover, in Table 2 we can see that the classes representing position lights are mostly misclassified as signals and not as other position lights. As we described before, we believe that this is a consequence of the imbalance in the amount of samples per class causing overfitting of signals over position lights. The false positive distributions of the sign classes show similar behaviour with the signals. Classes with high similarity like speed signs, diverging left, diverging right and diverging left-right speed signs dominate the false positives distributions of each others. The same is observed for the back views of the different shapes of signs.

Figure 7 shows how many objects were not detected as percentages according to their area in pixels^2. We can see that all models struggled to find small objects. YOLO and SSD did not detect any object smaller than 400^2 pixels and about 20% of the ones of area from 400^2 to 600^2 pixels. The RetinaNet implementation is performing significantly better than any of the other three, even from Faster R-CNN. This result is surprising because according to [5] and [7], Faster R-CNN is performing better on smaller objects in general object detection datasets like COCO. Also, for all models, we can see that their performance is unrelated to the size of the objects from a size and above. For YOLO, this size is about 1000^2 pixels, for SSD around 800^2 pixels and Faster R-CNN and RetinaNet about 600^2 pixels.

Moreover, an interesting metric for our problem is to measure the amount of unique physical objects detected. The goal of object detection in computer vision, usually, is to detect all the objects that appear in an

Fig. 6: Example of detection. Red boxes represent a successful detection with a false classification. The green box is an object that was detected and classified successfully. We can see that it is hard even for a human to detect the objects on the red boxes. The true classes of the red boxes are: sign other and diverging-right speed sign for the left and right red box respectively.



(a) YOLO (b) SSD

(c) F-RCNN (d) RetinaNet

Fig. 7: Amount of the objects that each model did not detect in terms of area size in pixels^2.

image. In mapping, most of the times, it is important to detect the locations of the physical objects. If we have multiple images showing the same physical object from different point of views and at different scales, in mapping, it is sufficient to detect it at least in one image and not necessarily in all images. For this reason, in a semi-automatic way we estimated the corresponding physical object ID for each sample on this dataset. Based on these IDs, we counted how many unique physical objects were detected at least once from each model. Finally, the recall of each model based on the physical objects are 74.13%, 77.59%, 87.34% and 95.22% for YOLO, SSD, Faster R-CNN and RetinaNet, respectively. The score of RetinaNet on this metric is very high, however, in a real world project, it is not sufficient to fully automate the process (a result is considered satisfactory when it is above 99%). However, a detector with this high level of accuracy can be used to suggest positions of objects leaving only the quality control to human operators. Overall, the integration of the RetinaNet model on a project saves some thousands man hours.

Figure 6 shows a detection example of RetinaNet of a scene with multiple objects present. In the image, there are three different signs which were detected successfully but the two of them were misclassified (red boxes). In this example, we can see the high difficulty of the problem. The two misclassified objects are very small that we can barely see, however, RetinaNet managed to detect but misclassify them. These signs belong to the most common object size interval for this dataset (200 - 400 pixels).

Class Name	1st	2nd	3rd
Signal1 F	Other signals(47%)	Sign-C B(37%)	Sign Signal(2%)
Signal1 S	Signal1 F(45%)	Signal2 F(20%)	Sign-C B(15%)
Signal2 F	Signal1 F(73%)	Other signals(14%)	Sign-C B(4%)
Signal2 S	Signal1 F(10%)	Signal3 S(10%)	PL S(10%)
Signal3 F	Signal4F(44%)	Signal2 F(40%)	Other signals(10%)
Signal3 S	Signal4 F(11%)	Signal4 S(11%)	Signal2 F(10%)
Signal4 F	Signal3 F(20%)	Other signals(13%)	Signal4 S(11%)
Signal4 S	Signal1 F(40%)	Other signals(33%)	Signal3 F(7%)
Signals B	Sign-C B(15%)	Sign-R B(13%)	Signal4 F(12%)
Other signals	Signals B(74%)	Sign-C B(4%)	Sign Signal(4%)
PL 2W&1R	Sign Signal(15%)	Other signals(11%)	PL 2W(9%)
PL 2W	Signal1 F(23%)	Signal2 F(17%)	Signal1 S(16%)
PL 2W&C	Signal1 F(14%)	Signal2 F(14%)	Signal3 F(14%)
PL S	PL 2W(21%)	Signal1 S(20%)	Sign Signal(12%)
PL B	Sign-R B(19%)	Signals B(16%)	Sign Signal(14%)
Sign S	Sign R(33%)	Sign Other(24%)	Sign L(15%)
Sign L	Sign S(33%)	Sign R(25%)	Sign Other(20%)
Sign R	Sign S(25%)	Sign L(23%)	Sign Other(12%)
Sign LR	Sign S(24%)	Sign L(23%)	Sign-C B(18%)
Sign U	Sign S(13%)	Sign LR(13%)	Sign Other(13%)
Sign Other	Signal1 F(44%)	Other signals(32%)	Sign-C B(17%)
Sign-C B	Sign S(27%)	Sign-D B(12%)	Sign L(9%)
Sign-D B	Sign-C B(40%)	Sign S(18%)	Sign L(11%)
Sign-R B	Sign Other(36%)	Sign-D B(33%)	Sign-C B(17%)
Sign Signal	Sign Other(44%)	Other signals(22%)	Signal4 S(6%)

Table 2: Distribution of misclassified false positives per class of RetinaNet model. The three columns show, for each class, in descending popularity order, to which classes the model erroneously classified the false positives. The percentages in parentheses refer to the proportion of the total amount of false positives for that specific class.

6 Conclusions

In this paper, we applied four state of the art detection algorithms in a novel dataset used in real world projects. The four models showed high performance despite the challenges imposed by the size of the objects, the imbalance in the amount of samples per class, the intraclass variability and high resemblance of objects belonging to different classes. The order in performance of the models was the same as in the COCO dataset [5] but with different relative performances. The Retina network showed the best overall performance and the best combination of efficiency and effectiveness. YOLO showed the lowest performance but best efficiency. Also, in our dataset YOLO performed relatively better than on COCO. SSD showed the third overall performance and efficiency. Finally, Faster R-CNN was the least efficient model and second most accurate. We conclude that Faster R-CNN does not have any advantage over the Retina network in the detection task that we tested them.

The main challenge for the all four models proved to be the small size of the objects. All of them showed low recall values on objects smaller than 200^2 pixels which are mainly signs. We discovered that for each model there is a different threshold in size that above which size is no more a factor that affects performance. For objects above this size, signals prove to be more challenging to classify than signs due to the fact that different types of signals differ little. We believe that the Feature Pyramid Network and the α-balanced Focal Loss used in RetinaNet where the main reasons for the high performance of this method because they handle better the two main challenges of our dataset: small size of objects and class representation imbalance.

All models showed the lowest performance on position lights. We believe that the low performance is caused by the imbalance in amount of samples in these classes. To tackle this issue, we modified the dataset to achieve better balance among the classes. The performance of the models in these classes were improved when we performed tests on the balanced dataset however, with a toll on the overall performance. For this reason, we presented here the models with higher performance. As an evaluation metric we used the widely used mean Average Precision (mAP) metric. In addition, we measured the accuracy of the models in a metric more relevant to a railway mapping project, which measures the amount of physical objects detected instead of instances in images. In this metric, the best model, RetinaNet, showed 95.22% recall. Finally, we believe that the performance of the models can be improved by taking into account the spatial relationships

among objects of different classes along railways based on the railway specifications that they follow.

For future work, we believe that a dataset with higher resolution images will improve results. Also, we believe that, as a post-processing step, a statistical analysis on the multiple detections of the same physical objects will lead in a more accurate classification and therefore in higher mAP. RetinaNet could detect almost every object but the accuracy was affected mostly from misclassifications (objects of interest were detected but misclassified). Finally, along a railway, there are topological relationships among the objects. The presence of a type of object foretells the presence of other types of objects. The railway authorities possess this knowledge but in our case was not available.

References

1. D. Agudo, . Snchez, J. F. Vlez, and A. B. Moreno, "Real-time railway speed limit sign recognition from video sequences," in *2016 International Conference on Systems, Signals and Image Processing (IWSSIP)*, May 2016, pp. 1–4.
2. S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in Neural Information Processing Systems*, vol. 2015-, pp. 91–99, 2015.
3. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," *Lecture Notes in Computer Science*, vol. 9905, pp. 21–37, 2016.
4. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *Ieee Conference on Computer Vision and Pattern Recognition (cvpr)*, vol. 2016-, pp. 779–788, 2016.
5. T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *CoRR*, 2017.
6. J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *CoRR*, 2017.
7. T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick, "Microsoft coco: Common objects in context," *Lecture Notes in Computer Science*, vol. 8693, pp. 740–755, 2014.
8. S. Maldonado-Bascon, S. Lafuente-Arroyo, P. Gil-Jimenez, H. Gomez-Moreno, and F. Lopez-Ferreras, "Road-sign detection and recognition based on support vector machines," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 2, pp. 264–278, June 2007.
9. C.-Y. Fang, S.-W. Chen, and C.-S. Fuh, "Road-sign detection and tracking," *IEEE Transactions on Vehicular Technology*, vol. 52, no. 5, pp. 1329–1341, Sep. 2003.
10. G. Loy and N. Barnes, "Fast shape-based road sign detection for a driver assistance system," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 1, Sep. 2004, pp. 70–75 vol.1.
11. G. Piccioli, E. D. Micheli, P. Parodi, and M. Campani, "Robust method for road sign detection and recognition," *Image Vision Comput.*, vol. 14, pp. 209–223, 1996.
12. Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, "Traffic-sign detection and classification in the wild," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
13. Y. Zhu, C. Zhang, D. Zhou, X. Wang, X. Bai, and W. Liu, "Traffic sign detection and recognition using fully convolutional network guided proposals," *Neurocomputing*, vol. 214, pp. 758 – 766, 2016.
14. S. K. Berkaya, H. Gunduz, O. Ozsen, C. Akinlar, and S. Gunal, "On circular traffic sign detection and recognition," *Expert Systems with Applications*, vol. 48, pp. 67 – 75, 2016.
15. G. B. D. for Transport, *Traffic Signs Manual*. London, United Kingdom: The Stationary Office, 2013.
16. R. Safety and S. Board, *Lineside Operational Safety Signs*. London, United Kingdom: Railway Group Standard, 2009.
17. D. Zheng and Y. Wang, "Application of an artificial neural network on railway passenger flow prediction," in *Proceedings of 2011 International Conference on Electronic Mechanical Engineering and Information Technology*, vol. 1, Aug 2011, pp. 149–152.
18. T.-H. Tsai, C.-K. Lee, and C.-H. Wei, "Neural network based temporal feature models for short-term railway passenger demand forecasting," *Expert Systems with Applications*, vol. 36, no. 2, Part 2, pp. 3728 – 3736, 2009.
19. N. S.Suresh, R.Prabhu, "Implementation of railway track crack detection and protection," *International Journal Of Engineering And Computer Science*, vol. 6, no. 5, May 2017.
20. J. Sadeghi and H. Askarinejad, "Application of neural networks in evaluation of railway track quality condition," *Journal of Mechanical Science and Technology*, vol. 26, no. 1, pp. 113–122, Jan 2012.
21. X. Gibert, V. M. Patel, and R. Chellappa, "Robust fastener detection for autonomous visual railway track inspection," in *2015 IEEE Winter Conference on Applications of Computer Vision*, Jan 2015, pp. 694–701.
22. D. Sinha and F. Feroz, "Obstacle detection on railway tracks using vibration sensors and signal filtering using bayesian analysis," *IEEE Sensors Journal*, vol. 16, no. 3, pp. 642–649, Feb 2016.
23. S. Faghih-Roohi, S. Hajizadeh, A. Nez, R. Babuska, and B. D. Schutter, "Deep convolutional neural networks for detection of rail surface defects," in *2016 International Joint Conference on Neural Networks (IJCNN)*, July 2016, pp. 2584–2589.
24. S. Mockel, F. Scherer, and P. F. Schuster, "Multi-sensor obstacle detection on railway tracks," in *IEEE IV2003 Intelligent Vehicles Symposium. Proceedings (Cat. No.03TH8683)*, June 2003, pp. 42–46.
25. M. Melander and I. Halme, "Computer vision based solution for sign detection," *European Railway Review*, Oct 2016.
26. S. Ritika, S. Mittal, and D. Rao, "Railway track specific traffic signal selection using deep learning," *CoRR*, 2017.
27. R. Marmo, L. Lombardi, and N. Gagliardi, "Railway sign detection and classification," in *2006 IEEE Intelligent Transportation Systems Conference*, Sept 2006, pp. 1358–1363.
28. M. Arastounia, "Automated recognition of railroad infrastructure in rural areas from lidar data," *Remote Sensing*, vol. 11, Nov 2015.
29. S. Mikrut, Z. Mikrut, A. Moskal, and E. Pastucha, "Detection and recognition of selected class railway signs," *Image Processing and Communications*, vol. 19, no. 2-3, pp. 83 – 96, 2014.

30. R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *Ieee Conference on Computer Vision and Pattern Recognition (cvpr)*, pp. 580–587, 2014.
31. R. Girshick, "Fast r-cnn," *Proceedings (ieee International Conference on Computer Vision)*, vol. 2015, pp. 1440–1448, 2015.
32. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, 2014.
33. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Ieee Conference on Computer Vision and Pattern Recognition (cvpr)*, vol. 2016-, pp. 770–778, 2016.

Chapter 5

Object Geolocation

5.1 Introduction

The geolocation of railway objects based on detections is the secondary focus of this thesis. A detailed map requires knowledge of object positions with high accuracy (below 10 centimetres). For this task, we could perform experiments only on the second dataset presented in chapter 3. For the first dataset we did not have camera positions at our disposal and for the second this information was available with an average accuracy of 2 metres. For the estimation of the object positions we applied a *bundle adjustment*. In bundle adjustment, we aim to estimate the projection matrices and the 3D points who project exactly their corresponding image points by minimising the reprojection error. Bundle adjustment is usually the final step of 3D reconstruction algorithms [41]. For the minimisation of the reprojection error we applied the Levenberg-Marquard (LM) algorithm [52, 53]. We performed our experiments based on the 2D positions of the objects on the images and the known camera positions. We used two different sets of 2D positions. The first set was the ground truth bounding boxes annotated manually by experts. The second set consisted of the detections after applying the RetinaNet as presented in section 4.3. By using automatically detected objects we create an automatic pipeline that receives as input a set of panoramic images and outputs the 3D positions of the objects that are present in the images. This pipeline is the goal of this thesis and can reduce significantly the time and cost

of the current production processes on industrial railway mapping projects. In the following section we present our experiments which led to a scientific paper which is currently under peer review. In chapter 6, section 6.1 we discuss our findings on 3D position estimation along with our findings in automatic object detection.

5.2 Geolocation of Railway Signs and Signals

This paper is currently under peer review for publishing in *ISPRS Journal of Photogrammetry and Remote Sensing*:

G. Karagiannis, S. Olsen, and K. Pedersen, "Geolocation of railway signs and signals", *Under peer review in ISPRS Journal of Photogrammetry and Remote Sensing*, 2020

Geolocation of Railway Signs and Signals

Georgios Karagiannis[a,b,*], Søren Ingvor Olsen[a], Kim Steenstrup Pedersen[a]

[a]*Department of Computer Science, University of Copenhagen: Universitetsparken 5, 2100, Copenhagen, Denmark*
[b]*COWI A/S: Parallelvej 2, 2800, Lyngby, Denmark*

Abstract

Detection and geolocalisation of railway objects is an important aspect of railway maintenance. These tasks require high accuracy and in real world projects are performed manually using panoramic images. It is a very time consuming and expensive procedure, therefore, automating partially or fully these tasks will result in a significant improvement on cost and delivery times. Here, we address the automation of geolocalisation of railway signs and signals based on detections on panoramic images. We apply bundle adjustment on initial object position estimations of objects and achieve accuracy lower than 2 meters which is the accuracy of the camera GPS. In addition, we propose a pipeline towards full automation from data acquisition to object UTM coordinates. We fine tune a pre-trained RetinaNet detector and apply the geolocation algorithm on the detected bounding boxes. The detector achieved 94.5% recall and the UTM coordinates were estimated with a 1.82 meters error.

Keywords: Railway Mapping, Bundle Adjustment, Geolocation, Object Detection, Deep Learning

1. Introduction

The accurate knowledge of geographic locations of railway objects is a key component for maintenance and safety. Currently, railway object local-

*Corresponding author
Email addresses: geka@di.ku.dk, geks@cowi.com, giorkarag@cowi.com (Georgios Karagiannis), ingvor@di.ku.dk (Søren Ingvor Olsen), kimstp@di.ku.dk (Kim Steenstrup Pedersen)

isation is carried out with traditional surveying methods and manual image processing which include manual detection of objects in panoramic images and aerial photos simultaneously. These methods are time demanding, require expensive equipment and trained personnel. The process of object detection and geolocalisation based on known camera positions can be automated by applying modern and traditional methods of computer vision and photogrammetry.

Here we propose a cheap and efficient pipeline for geolocating railway objects based on panoramic images. First, a camera system mounted on a wagon acquires panoramic images along the railway. The camera system is also equipped with a GPS receiver to provide camera position in a global coordinate system. Next, an object detector is applied on the images [1] to obtain the image coordinates of objects of interest. Based on the camera positions and the camera model we can derive an initial estimation of the global coordinates of the objects. Finally, the estimated positions are optimized by bundle adjustment. In this paper, we focus mainly on the 3D position estimations given bounding boxes on images and less on the object detection part. However, we will present results of the full pipeline. For more details on detection of railway objects in panoramic images, we refer the reader to [1].

The final steps of the aforementioned pipeline, namely the computation of three dimensional positions or structures (3D) from images (2D) is called Structure-from-Motion (SfM) or 3D reconstruction in the fields of Photogrammetry and Computer Vision. It is a problem that numerous tasks in these fields face and has enabled high amount of conducted research over the past decades. For a feature-based 3D reconstruction, such as our case, Bundle Adjustment (BA) [2, 3, 4, 5, 6, 7] and Simultaneous Localization and Mapping (SLAM) [8, 9, 10, 11, 12] are the most popular approaches.

SLAM until recently was used mostly for sensors such as sonars or laser scanners and not cameras [13]. When SLAM is applied on images is commonly referred as visual-SLAM (vSLAM). This is a probabilistic method that maps the environment around the sensor and then localizes the sensor in this environment. This is achieved by triangulating the 3D positions of a set of points that are tracked in a sequence of frames and simultaneously, using these positions, estimating the camera pose. These estimations occur in a probabilistic way and therefore, the disadvantage of SLAM in regard to our problem is that it needs a high amount of observations in each view since, in a railway, only a handful of objects are visible in an image.

Bundle Adjustment is an optimization algorithm originally conceived in 1958 in [14] that minimises the reprojection error between the observed and predicted image points. This is expressed as a sum of squares of non-linear functions and is computed by non-linear least squares algorithms [7]. A very popular and efficient algorithm for Bundle Adjustment [2, 7] is Levenberg-Marquardt (LM) [15, 16]. This algorithm has a relatively simple implementation and an efficient convergence strategy [17]. This method suits best on our problem due to the limited amount of observations on each view and the inaccurate knowledge of the camera position and intrinsics matrix.

In the following sections, we will describe the dataset used, analyse the localisation method and present the results based on a) manual annotations of objects on images and b) output of an object detector.

2. Methodology

In this section, we will present the dataset, the problem and the steps we propose to obtain 3D positions of objects observed in the images.

2.1. Data

The dataset used in our experiments consists of panoramic images along a railway located in Melbourne, Australia and was acquired in 2017. In the dataset there are 1755 unique objects each of them observed on average in 4 images. In total, there are 8210 images where at least one object of interest is present. The objects belong to 13 different classes (4 classes for signals, 3 for position lights and 6 for signs). The detector we used [1] was trained in a similar dataset for 25 classes. Our classes do not correspond one to one with the ones used in [1], thus, we fine-tuned the detector on this dataset. We will not present fully the dataset on object detection terms here because our main focus is the 3D positions estimations and not the detection. Figure 1 shows an example of a panoramic image from this dataset. The two red ellipses show two errors in stitching of the top view camera (directed to see the sky) with the side view cameras. These errors appear only in areas where no objects are present and therefore are insignificant for our purpose. Finally, the blue bounding boxes show examples of the objects of interest. On the right side and closer to the camera we see 2 single-element signals. On the left side we see 2 double-element signals located at ground level and on their right 2 single-element signals. Note the difference in size among the samples. Small objects are common in this dataset. The detection of such small objects is

Figure 1: Full panoramic image of the dataset used. We can see the sinusoidal behaviour of distortion by observing the cables on top of the image. The two red ellipses at the middle top and top left area of the image show two visible stitching errors (there is a small cable discontinuity). The blue bounding boxes show the objects of interest on the image. On the right side there are 2 single-element signals while on the left side we see 2 double-element signals (placed at ground level) and 2 single-element signals.

crucial to our method because they provide more observations per object and hence, more degrees of freedom during optimisation.

In order to estimate the initial 3D positions of the objects, at least two observations of an object (different images) are necessary. Though, to optimize the initial estimations we need at more than two (see section 2.2) observations of each object. Moreover, along a railway there are areas where no objects of interest exist followed by areas in which some objects are observed. This lack in continuity in observations means that the dataset consists of smaller independent sequences of images where objects are continuously observed following and followed by areas with no objects. This characteristic prevents us from applying the optimization in the whole dataset at once and thus we can only optimize the positions of the objects only in the independent sequences. There are 428 such sequences in the dataset averaging 4.7 images per sequence. From these sequences 363 consist of two or more images (there are 65 isolated images showing at least an object). In total, we can compute initial positions for 1371 unique objects (objects visible in more than one image) and only 232 out of 428 image sequences are qualified for optimization given the restrictions defined previously.

The accuracy of the camera GPS sets the limit on the computed object

positions, since the they are derived based on the camera position. In this dataset, the camera positions are known with two meters accuracy on average and occasionally is above ten meters which is considered poor in a real world project. However, a map of the centrelines of the railway were available to us with high accuracy. Based on these centrelines and given that the data acquired by driving on a single rail track, we adjusted the camera positions forcing them (by orthogonal projection) to be on top of this specific track centreline. Note that this improves the accuracy of the camera positions only across the track but not along it and also, we cannot compute the exact level of improvement achieved. Figure 2 shows an example where the camera positions are many meters off the track. The red line represents the track along which the camera travelled during image acquisition and the green dots are the camera GPS positions. Indicatively, we measured the perpendicular distances of two camera to the red track (short blue lines at the centre and right side of the figure). These distances are 8.9 and 18.1 meters respectively. Note that this error is only along the vertical axis. The misplacement along the horizontal axis (roughly along the track) cannot be specified. This is a representative example of errors in camera GPS positions around train stations. The replacement of the initial camera positions with their orthogonal projection to the train track softens at some extent the magnitude of the error but not entirely (unknown error along track). Finally, we evaluate our final position estimations with ground truth object positions available. Ground truth positions are known with 10 centimetres accuracy.

2.2. Initial position estimation

We geolocate the detected objects on images based on the recorded GPS location of the camera and the geometry of the images. The images are panoramic meaning that the horizontal axis corresponds to a full circle. Also, the panoramic images occur from six cameras and hence, there is no principal point. We assume a virtual principal point at the middle of the horizontal axis of the image initially estimated at $c_x = 4096$ in pixels (the size of the images is 8192×4096 pixels). The angular projection α of an object detected at position x along the horizontal axis of the image is:

$$x = \left(\frac{\alpha}{\pi}\right) c_x = \frac{x - c_x}{c_x} \tag{1}$$

An object O detected at x will be located at position (X, Z). Let X be the perpendicular distance of the object to the virtual optical axis defined by

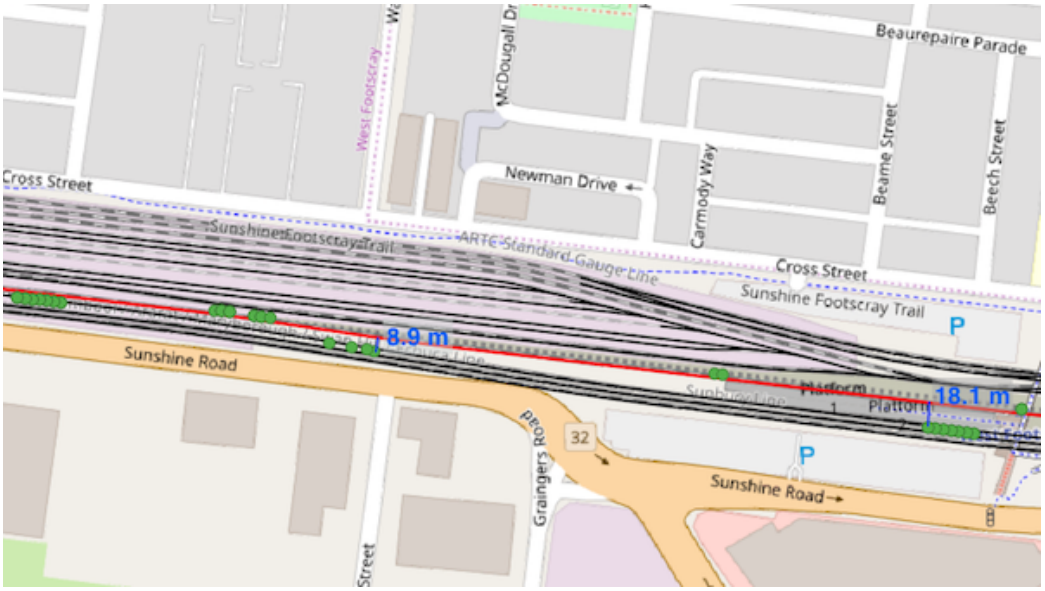


Figure 2: Map of part of the railway in Melbourne, Australia. Black lines represent all track centrelines while the red line represents the track on which the camera moved during data acquisition. Green dots are camera positions. We can see that many camera points are completely off the red line. Indicatively, we measured we measured the perpendicular distance to the red line for two camera positions (short blue lines at centre and right side of the image). The two examples are 8.9 and 18.1 meters off only in perpendicular distance (no information of error along horizontal axis). This is a common error, especially in areas close to train stations or when the signal is obstructed.

the virtual principal point c_x and Z the distance from the optical center along the optical axis to the projection. Then, the horizontal projection through equation 1 is given by:

$$X = Ztan(\alpha) \tag{2}$$

In addition, each object is detected in multiple consecutive images which allows the computation of object locations through triangulation. We assume that the object $O = (X, Z)$ is observed in two images with known camera positions P_1, P_2. Then, the two camera positions are connected through equation 3.

$$P_2 = P_1 + \Delta(cos\gamma, sin\gamma) = (v_1, v_2) \tag{3}$$

where P_i is the camera position in a global Cartesian coordinate system (UTM), Δ_i is the distance between consecutive camera positions and γ_i the

orientation change of the camera axis between consecutive camera positions. On our experiments, we consider $\gamma = 0$ as this angle is minor compared to the two meters error of GPS (see section 2.4).

Also, we assume that the optical axis of the camera is oriented horizontally to make an angle u with the train. We consider that the train orientation at P_2 is $(cos\delta, sin\delta)$. Then, the position of an object in UTM coordinates can be computed by:

$$X = v_1 + b_1 \frac{\beta_2 v_1 - beta_1 v_2}{beta_1 b_2 - beta_2 b_1} \tag{4}$$

$$Y = v_2 + b_2 \frac{beta_2 v_1 - beta_1 v_2}{beta_1 b_2 - beta_2 b_1} \tag{5}$$

where $beta_1 = cos(u + a_1)$, $beta_2 = sin(u + a_1)$, $b_1 = cos(\delta + a_2)$, $b_2 = cos(\delta + a_1)$, $v_1 = \Delta_1 cos(\gamma)$ and $v_2 = \Delta_1 sin(\gamma)$. This is a system of two equations with two unknowns.

We may now expand the computation of object UTM coordinates to multiple objects observed in multiple different images. The position x of the i_{th} object in j_{th} image is given by:

$$X = P_j + [cos(\delta_i + a_i), sin(\delta_i + a_j)]k_j \tag{6}$$

where k_j is an unknown scaling parameter. For n physical objects observed in m images there are $2n + m$ unknowns $x = (X, Y, k_1, ...k_m)^\top$. Let $p = (P_1^x, P_1^y, P_2^x, P_2^y, ...P_m^x, P_m^y,)$ be the $2m$ vector of camera positions for m observations. Also, $c_i = cos(\delta_i + a_i)$ and $s_i = sin(\delta_i + a_i)$. Then we can construct a $2m \times (2n + m)$ matrix:

$$A = \begin{bmatrix} 1 & 0 & -c_1 & 0 & & \\ 0 & 1 & -s_1 & 0 & & \\ 1 & 0 & 0 & -c_2 & & \\ 0 & 1 & 0 & -s_2 & & \\ \vdots & \vdots & \vdots & \vdots & \ddots & \\ 1 & 0 & 0 & \cdots & 0 & -c_n \\ 0 & 1 & 0 & \cdots & 0 & -s_n \end{bmatrix} \tag{7}$$

By solving $Ax = p$ can compute the position of the objects in UTM coordinates. Figure 3 shows a drawing of the camera positions, an object along with the distances and angles used for the estimation of the object

position. In summary, in order to estimate the positions of the objects we need at least $2n+m$ equations for n number of objects observed in m images. Each image gives two equations per object, therefore, we can estimate the position of an object from two images. However, in order to minimise the error in estimated positions we need to apply an optimisation algorithm which requires more than two observations per object. The more observations of an object, the more accurate the estimated positions will be.

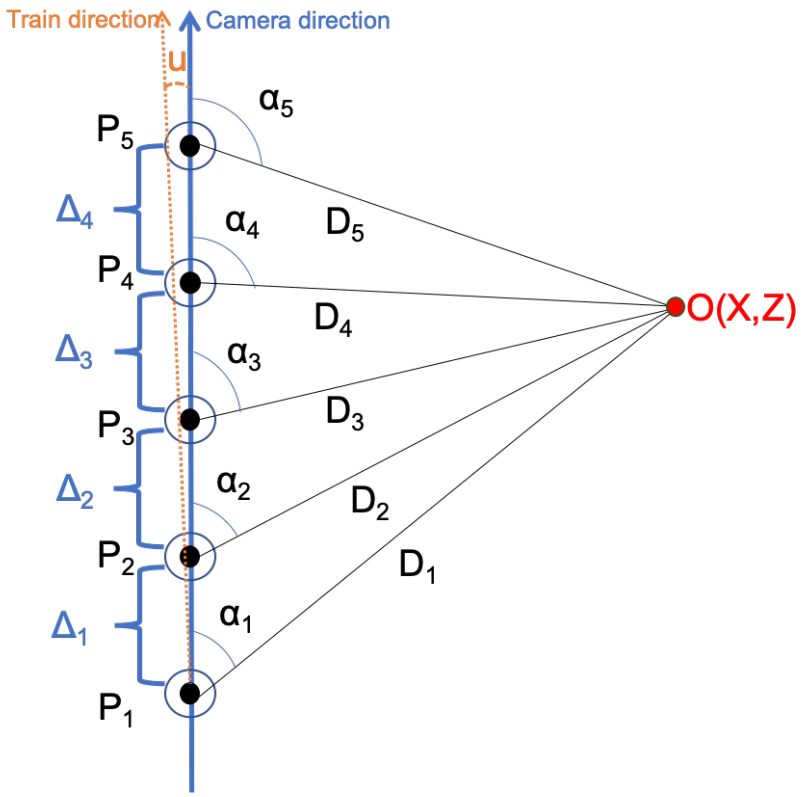


Figure 3: Drawing of camera and object positions. P_i represents camera positions in UTM coordinates, Δ_i the distance between camera positions in meters, D_i the distance between the cameras and the object in meters, O the position of an object in UTM coordinates. The blue line shows the direction of the camera while the dashed orange line the direction of the train. The angle between these two lines is u.

2.3. Bundle Adjustment

In practice, when reconstructing from multiple views, a final bundle adjustment, i.e. a non-linear optimisation involving all the parameters, and

using the reprojection error as loss, is usually applied. In the present case, using a batch approach where we search for all poses and all object positions in one optimisation process is not be tractable due to discontinuities in object presence along the railway. Instead an incremental approach is appropriate.

Assume that for each new images we first apply the procedure sketched in section 2.2 to estimate initial positions for objects that have been seen just a single time before. Then we apply bundle adjustment on all positions (X_i, Z_i) of all visible objects and on the pose (P_j, v_j) local to the current position. Since only a few objects may be simultaneously visible the number of unknowns is small allowing a fast update.

Here we will formulate the basic minimisation expression. Let (X_i^0, Z_i^0) be the initially derived position of object i visible from the actual location P_j and let ${}_ij$ be the angular observed position of object i at position j. Let, as before P_j^0 and ${}_j^0$ be the initial estimate of the pose related to image j. For all i and j (where object i is visible in frame j) we may compute the squared reprojection error:

$$r_{i,j} = \|a_{ij} - \delta_j - arctan\left(\frac{X_i - P_j}{Z_i - Z_j}\right)\|^2 \tag{8}$$

The combined model error E for a group of images is then the sum of all $r_{i,j}$ over all indices where it is (locally) defined. To this error we also add the deviation of the camera GPS since it is noisy:

$$E_{GPS} = \epsilon_1\|P_j - GPS_j\|^2 + \epsilon_2\|\delta_j - GPS_{or}\|^2 \tag{9}$$

where GPS_j is the GPS camera position in j^th image and GPS_{or} the corresponding orientation.

Then the total model error forms to:

$$\begin{aligned} E_m = \sum \|a_{ij} - \delta_j - arctan\left(\frac{X_i - P_j}{Y_i - Y_j}\right)\|^2 \\ +\epsilon_1\|P_j - GPS_j\|^2 + \epsilon_2\|\delta_j - GPS_{or}\|^2 \end{aligned} \tag{10}$$

In our case, the optimisation method used is Levenberg-Marquard (LM). The LM is an iterative estimation algorithm. It behaves as a *Gauss-Newton* (GN) method when the solution on an iteration is close to a local minimum, causing rapid convergence. On the other hand, when the current solution is far from a local minimum it behaves as a *gradient descent* (GD), taking a

short step in the steepest descent direction. Here, we will present the basic principles of LM since an extensive analysis of the algorithm is beyond the scope of this paper. For a detailed analysis of LM algorithm we refer the reader to [18, 19, 17, 20].

2.3.1. Gauss-Newton

Let $X = f(P)$ be a non-linear, functional relation a measurement vector X and a parameter vector P. Assume that we measure X_0, an approximation of the true value $\bar{X}$. Then, we want to compute $\hat{P}$ which satisfies $X = f(\hat{P}) - \epsilon$ where $\|\epsilon\|$ is minimised. We assume that the function f is locally linearly represented by the Jacobian matrix $J = \frac{\partial f}{\partial P}$ evaluated at P_i. Let P_0 be an estimated initial estimation of P, $\epsilon_0 = f(P_0) - X$ and the approximation of f at P_0 be $f(P_0 + \Delta) = f(P_0) + J\Delta$. The next step is to find a set of parameters $P_1 = P_0 + \Delta$ for which $f(P_1) - X = f(P_0) + J\Delta - X = \epsilon_0 + J\Delta$ is minimum [18]. It is then a matter of minimising the term $\|\epsilon_0 + J\Delta$ over Δ. We can estimate the vector Δ by solving the *normal equations* [18]:

$$J^\top J\Delta = -J^\top \epsilon_0 \tag{11}$$

In summary, based on the initial estimation P_0 we iteratively approximate the solution $\hat{P}$ following the formula $P_{i+1} = P_i + \Delta_i$ where Δ_i is the solution[18]:

$$J\Delta_i = -J\epsilon_i \tag{12}$$

2.3.2. Levenberg-Marquard

In LM, the normal equations 11 become the *augmented normal equations* [18]:

$$(J^\top J + \lambda I)\Delta = -J^\top \epsilon \tag{13}$$

where I is the Identity matrix and the *damping* term λ has multiple effects:

- When $\lambda > 0$, the coefficient matrix is positive definitive. This guarantees that Δ moves in along a descent direction (*gradient descent*) [19].
- For large λ, through equation 13 the solution Δ is [19]:

$$\Delta = -\frac{J^\top \epsilon}{(J^\top J + \lambda I)} \simeq -\frac{1}{\lambda} J^\top \epsilon \tag{14}$$

which means that the algorithm takes a small step in the steepest direction [19].

- When λ is small, the method essentially becomes the Gauss-Newton (section 2.3.1) converging quickly to the final solution

In summary, the Levenberg-Marquard algorithm switches between Gauss-Newton and gradient descent depending on how close the current solution is to the minimum. When the solution is far from the minimum, GD is applied to direct the solution slowly but surely towards the minimum. On the other hand, when the solution is close to the minimum, the GN is applied to achieve fast convergence [18]. The initial value of λ is defined as $\lambda_0 = \tau max\{J\top Jii\}$ where τ is a user defined value [17]. If X_0 is a good approximation of $\bar{X}$, then a small τ is suggested (10^{-6}) [19]. In our case we used $\tau = 10^{-3}$. Each time we solve the augmented equations 13, the value of λ is updated. If the solution obtained leads to smaller error, then $\lambda_{i+1} = \lambda_i/10$, otherwise $\lambda_{i+1} = \lambda_i \cdot 10$

As we mentioned in previous sections, the nature of our problem dictates that an LM optimization cannot be applied in all observations in all images because of lack of continuity. Thus, we apply the optimization for sequences of images where there is continued presence of observed objects. This reduces the amount of the parameters involved allowing us to use the bare LM algorithm and not its sparse version.

2.4. Error Sources

In the model described in the previous section, we have addressed some sources of errors but ignored others. Firstly, we assumed that the direction of the camera axis for consecutive frames does not change (angle *gamma* in equation 3). In reality, this is not true since there is a slight change in the direction of the camera at each position. However, we have tested this assumption using synthetic data. We have set the camera positions with an average of two meters error and the change in camera direction between two consecutive frames varying by 1%. On average, the difference on estimated positions with versus without adding the angular error ranged below one centimetre which is insignificant compared to the two meters error provided from the GPS. We selected a low error in this angle because the railways are designed with high precision to follow a straight line.

Also, in previous section we described that the angle between the camera position and the object is obtained by the horizontal image coordinate. This

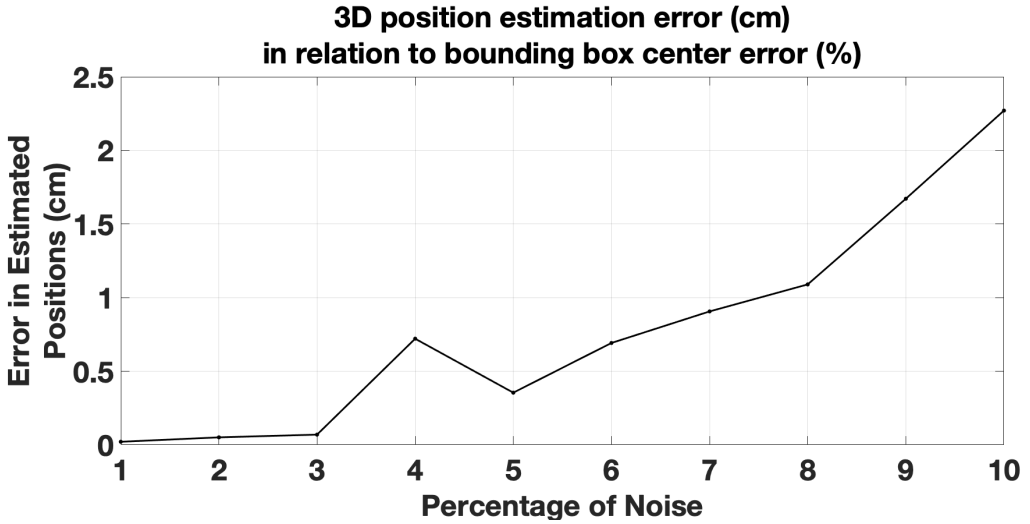


Figure 4: Impact of error in annotations. We applied 1-10% of each bounding box dimension error on the retrieved centres. The impact on the 3D position estimation ranges from a few millimetres to 2.3 centimetres for 10% error in annotations.

coordinate is the center of the bounding box of each detected object. In our dataset the bounding boxes are carefully annotated by an expert, however inaccuracies cannot be avoided. Similar inaccuracies are expected also by automatic detection of bounding boxes, if not higher. The impact of this inaccuracy can be tested in synthetic data. Figure 4 shows the error in centimetres when we add noise on the image coordinates of the bounding box centres. We choose to add percentage based noise and not absolute values in pixels. In cases where the object is large or closer to the camera, the bounding box is large and thus the center is obtained with more pixels error compared to a small bounding box. In addition, every pixel of a small bounding box, that shows an object further away, corresponds to higher error in distance. For these two reasons, we believe that percentage based noise is more appropriate here. The noise levels range from 1-10% of the bounding box dimensions and the corresponding error ranges from a few millimetres to 2.3 centimetres. Again, this error is a small fraction of the two meters error from the GPS.

Moreover, the stitching process to create the panoramas introduces errors leading to position inaccuracies. Unfortunately, there was no specific information available to us regarding the parameters and the model used during stitching which made our task more challenging. We would like to men-

tion here that the manual annotations have been implemented only on the panoramas. Therefore, we can perform our experiments only on the panoramas and accept any errors introduced from stitching. The only problem that we could identify by visual inspection were some discontinuities at the top of the image 1. These errors exist because of the poor stitching of the vertical camera (camera that looks in the sky) with the rest. The errors exist only at the stitching borders of this camera where no objects exist. Also,, the levels of distortion on the images vary on different locations in a sinusoidal way (see figure 1). This means that observations to the center and the edges of the image are more accurate compared to the ones in the space between.

In addition, the distance of a specific object (observed on multiple views) from each camera position may vary by several meters. We have verified by testing in multiple objects on multiple sequences that the accuracy of the computed object positions is related to the distance of the object to the camera. The observations from cameras closer to the object lead to more accurate computations. Specifically, for this test we considered all objects that are visible in more than six images (570 total objects) and estimated the positions of these objects in UTM by taking into account different amounts of observations. We estimated the position of each object four times: using all available observations, using only the five closest to the camera, the four closest and the three closest. Figure 5 shows the accuracy in meters for each different case. We can see that there is a significant loss in accuracy when cameras far away from the objects are used in the computations of initial positions.

However, for the optimisation process the amount of observations per object has the opposite effect. Figure 6 shows the resulted error in estimated positions depending on the amount of observations per object used in optimisation. We can see that even though the five closest objects gives estimated positions with 3.8 meters error on average, in optimisation results in best accuracy with 1.41 meters error on average. This is because more observations allow more degrees of freedom and also the optimisation takes into account all objects in a sequence of views where continuity is preserved. Therefore, we only use the three closest observations for the initial position estimation and five in optimisation.

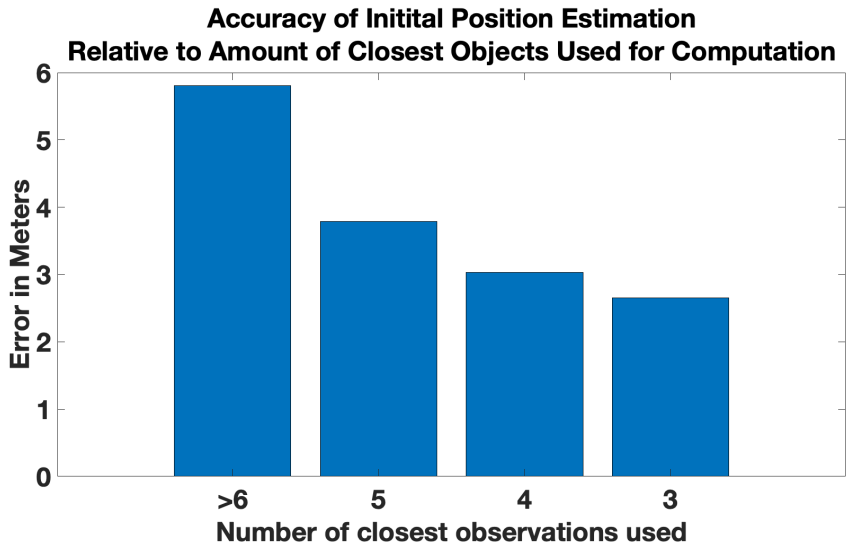


Figure 5: Error in estimated initial positions of objects depending on the amount of observations used in computations. When the available observations are more than 6 for an object, we see that we obtain on average up to two times more accurate estimation by using only the observations closest to the camera.

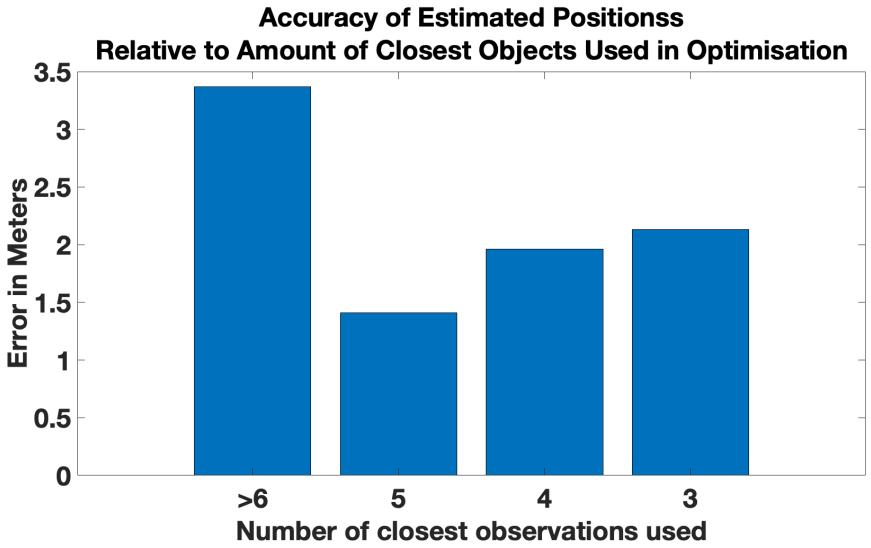


Figure 6: Error in estimated initial positions of objects depending on the amount of observations used in computations. When the available observations are more than 6 for an object, we see that we obtain on average up to two times more accurate estimation by using only the observations closest to the camera.

3. Results

We applied the methodology described in the previous section in two ways. First, we use the ground truth image coordinates of the objects and validate the accuracy of geolocation on the ground truth locations. Secondly, we apply an object detection CNN trained in a different, larger dataset and fine tuned in this dataset. Then, we apply the geolocation process on the detections of the CNN in order to evaluate a fully automated pipeline from data acquisition to object geographical coordinates. For both experiments we tested the original camera locations and the adjusted to the centrelines. Here we present only the results using the adjusted camera locations as they are more meaningful.

3.1. Ground Truth bounding boxes

In this scenario, we used the image coordinates of the objects as provided by the manual annotation of the images (we refer to these as ground truth). Based on the centres of the ground truth bounding boxes and the camera positions we computed the UTM coordinates of the objects according to the methodology described in section 2. To evaluate the accuracy of the computations, we measured the Euclidean distance between the estimated positions and the known accurate UTM coordinates of the objects. Figure 7 is a histogram showing this distance for each of the 1371 objects. The average of all distances is 1.41 meters which is below the 2 meters accuracy of the camera positions. We consider these estimations acceptable based only on the camera GPS accuracy and the assumptions made (camera model, stitching etc.). Though, for a railway map estimated positions should be known with no more than 10 centimetres accuracy. As we see from our experiments, the optimisation process can improve at some extent the final positions, however it is strictly limited to the scale of error of the GPS.

3.2. Detected bounding boxes

The image coordinates of the objects in this experiment were acquired by applying a pre-trained CNN based on RetinaNet as presented in [1]. The detected objects from the CNN were filtered according to the amount of images they appeared. Only objects detected in at least two images were used in order to, at least, be able to estimate the initial positions. Thus, even though RetinaNet resulted in around 89.0% *accuracy* ($\frac{True\ Positives}{Number\ of\ Objects}$), this number dropped to 83.9% due to the filtering (removed true positives where

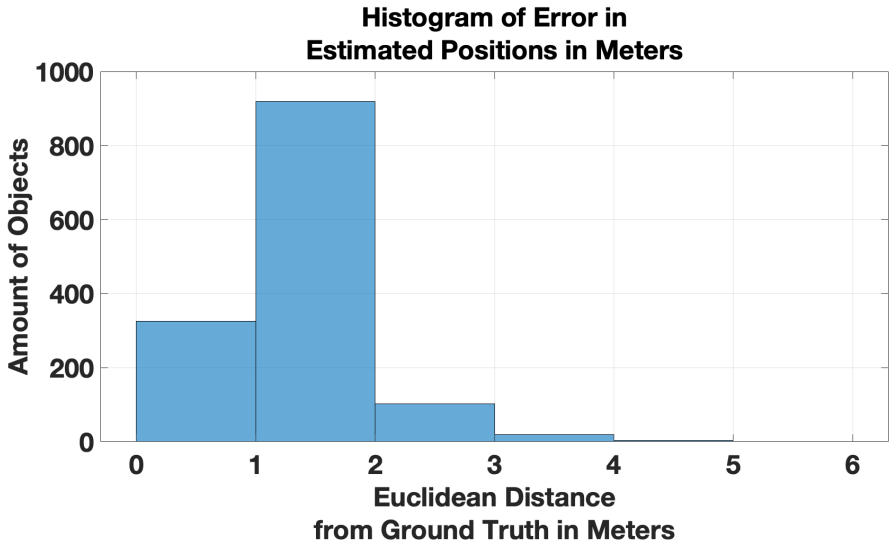


Figure 7: Error in computed positions of objects detected from RetinaNet in meters (Euclidean distance from ground truth)

only one detection is available for an object). In total, before filtering, the algorithm detected, at least once, 1669 objects out of 1755 along with 129 false positives (95.1% accuracy in terms of physical objects). After filtering, 1296 unique objects were detected (out of the 1371 possible). Also, the filtering had a positive side effect of reducing the false positive rate from 7.6% to only 1.1% (14 false positives among 1310 detections). The drop in accuracy is significant, however, this is expected since the same behaviour is observed in the ground truth as we discussed in section 2.1 (out of 1755 total objects only 1371 existed in at least two images). In summary, if we assume that the available objects to geolocate in this dataset are 1371, RetinaNet detected 1296 which translates into 94.5% accuracy.

Afterwards, the positions of the 1296 objects were estimated according to the methodology describe in section 2. Then we computed the Euclidean distance between each estimated position and the corresponding ground truth. The average distance in this scenario is 1.82 meters which is again below the camera GPS accuracy. It seems like a paradox that the estimated positions are more accurate compared to the camera GPS positions . Even if the camera model, the stitching processing and all the assumptions made were error-less, we should expect an accuracy equal to the camera GPS at best. The improvement in accuracy is justified by the adjustment of the camera

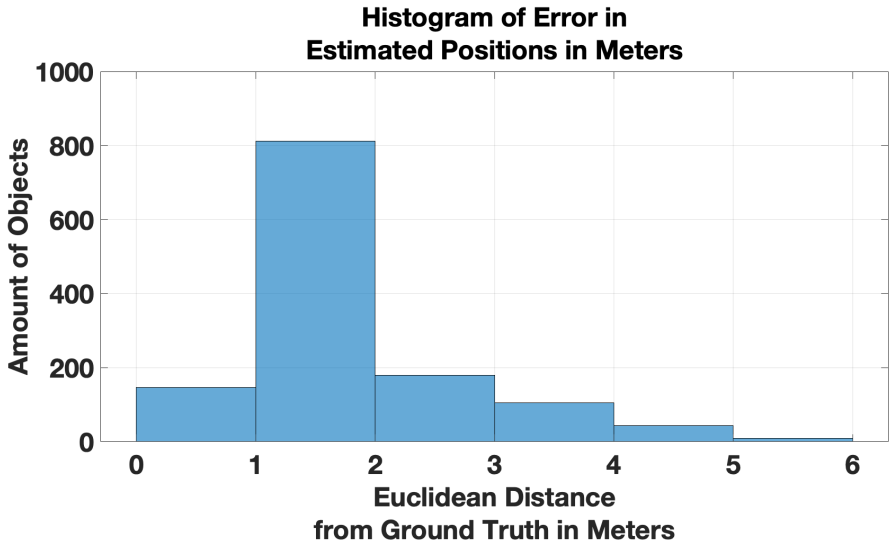


Figure 8: Error in computed positions of objects in meters (Euclidean distance from ground truth)

positions to the railway centrelines. Without this adjustment, the computed positions vary from two to tens of meters depending on the sequence of images. In cases where the GPS signal is obstructed, the error is low but, for instance, when the train is passing through a station this error is higher than twenty meters. Figure 8 shows the Euclidean distance between the computed position and its corresponding ground truth in meters for each object detected from RetinaNet. For better quality of the graph we removed 17 outlier positions which were exceeding 20 meters error in distance. Compared to the first scenario, the positions are on average 0.41 meters less accurate. We believe that this difference exists due to two reasons. First, the bounding boxes of the ground truth data are very tight to the object leading to an accurate value of the bounding box center. In the contrary, the bounding boxes of RetinaNet are larger and contain also some pixels of background. The second reason is that sometimes the bounding box created from the detection algorithm is misplaced. The detection algorithm has an Intersection over Union (IoU) criteria of 50% to accept a detection. This means that the detected object is not required to match perfectly the ground truth bounding box in order to be accepted. Therefore, a detected bounding box of 51% IoU will be accepted but will possibly give a center several pixels away from the ground truth. However, this is a favourable trade-off because an increase of

the IoU criteria would result in fewer correct detections.

4. Conclusions

In this paper, we presented a pipeline to automatically estimate sign and signal positions along a railway from panoramic images. We estimated the positions based on automatic detections of objects on images and known camera positions. The detection of objects based on RetinaNet achieved a 94.5% accuracy. This number reflects the amount of objects detected multiple times over the amount of objects that appear multiple times in images. Our method estimated object positions with an average error of 1.82 meters on automatic detections and 1.42 on ground truth bounding boxes, given a GPS error of 2 meters. The method improved the positions by almost 30% when applied in ground truth bounding boxes and almost 10% based on automatic detections. We believe that the assumptions made here have minor impact to the final accuracy compared to the GPS error. Therefore, we firmly believe that if the camera positions are known with an error of the order of a few centimetres, the methodology presented here can match the same accuracy. We verified this claim on synthetic data experiments, although we could not perform experiments on the real dataset due to unavailability of more accurate camera locations.

However, in an industrial project, an accuracy at the order of meters is considered low. From our experience in such industrial projects, we know that the requirements for completeness in objects is above 99% and in position accuracy below 10 centimetres. However, such numbers are unseen by fully automated methods in such complex projects. The achieved accuracies here, in both detection and geolocation, are significant. Although this pipeline cannot fully automate an industrial project, it can reduce or eliminate some expensive, labour intensive and time consuming processes. The reason is that by applying this pipeline, the manual processes are limited to only correct the inaccuracies. At the moment we do not have specific numbers on the cost reduction impact of our method on an industrial project because it has not fully integrated yet. In addition, the pipeline can be of use in tasks where the requirements in accuracy are less strict.

In summary, the accuracy of GPS proves to be crucial for this task. More accurate knowledge of camera positions will allow further investigation on the impact of other parameters involved in position estimation. Such parameters are the stitching process, the camera model, the mounting of the camera on

the vehicle, the bounding box accuracy etc. We partially investigated some of these aspects using synthetic data but clear conclusions can be made by experimenting in real data.

Acknowledgement

Our research has been funded partially by Innovations Fund Denmark and COWI A/S as part of an industrial PhD.

References

[1] G. Karagiannis, S. Olsen, K. Pedersen, Detection of railway signs and signals, Under peer review.

[2] B. Triggs, P. F. McLauchlan, R. I. Hartley, A. W. Fitzgibbon, Bundle adjustment - a modern synthesis, in: Proceedings of the International Workshop on Vision Algorithms: Theory and Practice, ICCV '99, Springer-Verlag, London, UK, UK, 2000, pp. 298–372.
URL http://dl.acm.org/citation.cfm?id=646271.685629

[3] P. Beardsley, P. Torr, A. Zisserman, 3d model acquisition from extended image sequences, Computer Vision - Eccv '96. 4th Eurpean Conference on Computer Proceedings (1996) 683–95 vol.2.

[4] A. W. Fitzgibbon, A. Zisserman, Automatic camera recovery for closed or open image sequences, in: European Conference on Computer Vision, Springer-Verlag, 1998, pp. 311–326.

[5] R. I. Hartley, Euclidean reconstruction from uncalibrated views, in: Proceedings of the Second Joint European - US Workshop on Applications of Invariance in Computer Vision, Springer-Verlag, Berlin, Heidelberg, 1994, pp. 237–256.
URL http://dl.acm.org/citation.cfm?id=647302.760233

[6] M. I. A. Lourakis, A. A. Argyros, Efficient, causal camera tracking in unprepared environments, Comput. Vis. Image Underst. 99 (2) (2005) 259–290. doi:10.1016/j.cviu.2005.02.001.
URL https://doi.org/10.1016/j.cviu.2005.02.001

[7] M. Lourakis, A. Argyros, Sba: A software package for generic sparse bundle adjustment, ACM Trans. Math. Softw. 36 (1) (2009) 2:1–2:30. doi:10.1145/1486525.1486527.
URL http://doi.acm.org/10.1145/1486525.1486527

[8] H. Durrant-Whyte, T. Bailey, Simultaneous localization and mapping: part i, IEEE Robotics Automation Magazine 13 (2) (2006) 99–110. doi:10.1109/MRA.2006.1638022.

[9] A. J. Davison, I. D. Reid, N. D. Molton, O. Stasse, Monoslam: Real-time single camera slam, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (6) (2007) 1052–1067. doi:10.1109/TPAMI.2007.1049.

[10] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, M. Csorba, A solution to the simultaneous localization and map building (slam) problem, IEEE Transactions on Robotics and Automation 17 (3) (2001) 229–241. doi:10.1109/70.938381.

[11] R. Mur-Artal, J. M. M. Montiel, J. D. Tards, Orb-slam: A versatile and accurate monocular slam system, IEEE Transactions on Robotics 31 (5) (2015) 1147–1163. doi:10.1109/TRO.2015.2463671.

[12] Davison, Real-time simultaneous localisation and mapping with a single camera, in: Proceedings Ninth IEEE International Conference on Computer Vision, 2003, pp. 1403–1410 vol.2. doi:10.1109/ICCV.2003.1238654.

[13] A. J. Davison, I. D. Reid, N. D. Molton, O. Stasse, Monoslam: Real-time single camera slam, IEEE Trans. Pattern Analysis and Machine Intelligence 29 (2007) 2007.

[14] D. Brown, A Solution to the General Problem of Multiple Station Analytical Stereotriangulation, RCA Data reducation technical report, D. Brown Associates, Incorporated, 1958.
URL https://books.google.gr/books?id=FikPPwAACAAJ

[15] K. LEVENBERG, A method for the solution of certain non-linear problems in least squares, Quarterly of Applied Mathematics 2 (2) (1944) 164–168.
URL http://www.jstor.org/stable/43633451

[16] D. W. Marquardt, An algorithm for least-squares estimation of nonlinear parameters, SIAM Journal on Applied Mathematics 11 (2) (1963) 431–441. doi:10.1137/0111030.
URL http://dx.doi.org/10.1137/0111030

[17] M. L. A. Lourakis, A. A. Argyros, Is levenberg-marquardt the most efficient optimization algorithm for implementing bundle adjustment?, in: Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, Vol. 2, 2005, pp. 1526–1531 Vol. 2. doi:10.1109/ICCV.2005.128.

[18] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, 2nd Edition, Cambridge University Press, USA, 2003.

[19] K. Madsen, H. Nielsen, O. Tingleff, Methods for non-linear least squares problems (April 2004).
URL http://www2.imm.dtu.dk/pubdb/views/edoc_download.php/3215/pdf/imm3215.pdf

[20] C. T. Kelley, Iterative methods for optimization, SIAM, 1999.

Chapter 6

Discussion & Future Work

In this chapter, we will discuss our findings on automatic object detection and 3D position estimation presented in previous chapters. In addition, we will present our thoughts on how this pipeline can be further improved in the future.

6.1 Discussion

On the detection task we achieved 86.56% mAP and 95.22% recall on physical objects on all classes with RetinaNet on the first dataset. On the second dataset, we achieved 89.0% mAP and 94.5% recall on objects that appear more than once in the dataset. Mean average precision is a strict evaluation method for our purpose. Many objects have been detected successfully but were misclassified. Completeness in detection is the number one priority on an industrial railway mapping project, even if misclassified because the manual work is limited only to the correction of the class.

In both papers on object detection, we claim that a more fair evaluation method would be based on physical objects. Also, we claim that a single detection of a physical object is sufficient for purpose. However, this statement is not entirely true. First of all, as we described in the geolocation paper (section 5.2), we need at least two detections per object in order to estimate the 3D position of an object and more than two to apply an optimisation algorithm. However, since the *human in the loop* (i.e. the full pipeline is not fully automated) cannot be avoided until we

reach the specifications described in chapter 1 (99% mAP on automatic detection and 10 centimetres on 3D position estimation), a single detection per object is very valuable. The reason is that an operator will spend less time to mark the same object in the consequent frames, based on the single detection, than search the whole image without any prior knowledge.

Moreover, the major hindrance of the detection task has been the small size of objects. Small sized objects has been the weak spot for all state of the art detection algorithms [3, 13, 15, 30]. Our dataset is dominated by very small samples. In literature there are several suggested methods to address this weakness including attentive feature selection [67], a Super Resolution technique based on Generative Adversarial Networks (GAN) [68] or Multiple Receptive Fields (MRF) [69]. All these methods show great potential but the improvements on detection of small objects are limited. We believe that the simplest way out in our specific task is higher resolution images. Ultra high resolution (8K) panoramic sets of cameras (8K) are now available at a cost of a few thousand dollars [70]. This claim is supported by the improved mAP of Retina between the two datasets (86.56 % vs 89.0%). The model showed improved accuracy despite the fact that it was only fine tuned on the second dataset while it was fully trained on the first. The main difference between the two datasets has been the resolution of the images which resulted in significantly larger objects in the second dataset (110^2 pixels average object size compared to 30^2 pixels on the first dataset). In chapter 4 we analysed the impact of object sizes in pixels on the detection accuracy. Thus, we believe that the detection accuracy on the second dataset increased thanks to the higher resolution images. An obvious problem with very high resolution imagery (such as 8K) is their size which will increase significantly the processing time. However, railway objects appear in very specific areas of the image. By applying detection only in these areas, we can achieve reasonable processing rates. After all, high detection accuracy is the ultimate goal on railway mapping and not real time processing.

In addition, the improved accuracy of RetinaNet on the second dataset shows that generalisation is possible for different datasets with minimum additional training. However, in order to verify the level of transferability, it is necessary to perform more tests in different datasets of Australian or other national railways.

Finally, on the geolocation task, our method achieved high accuracy (1.82 metres), bearing in mind the 2-metres average accuracy of the camera positions. We were able to test the impact of other minor sources of errors only in synthetic data. Therefore, if certain conditions are met (more accurate camera positions and known stitching process) we believe that the method presented in chapter 5 can estimate object positions with an even better accuracy. However, we were unable to define the exact potential of the method due to lack of more accurate camera positions. Moreover, we evaluated our estimated positions on ground truth positions of the objects. Nevertheless, the ground truth positions are not free of errors. In fact, in our case the ground truth positions are known with a 10 centimetre accuracy (the accuracy of the aerial imagery) thus, even if we minimise the effects from all error sources, we will not be able to verify our estimations with an accuracy below 10 centimetres.

6.2 Future Work

In the introduction of this dissertation (section 1.1) we noted that the panoramic image datasets are auxiliary to the specialists for the identification of objects on aerial imagery. Their role is served even with less accurate stitching, GPS positions or resolution. In addition, the camera set-up was very different between the two datasets. In the first dataset, the camera had a clear front view while in the second the front view was blocked by the train. Despite the better detection accuracy achieved on the second dataset, we believe that the camera set-up on the first dataset is more appropriate. The reason is that the objects that appear at the centre of the image are clear front views and are larger in size. In the second dataset, the camera set-up did not come at a cost in accuracy because of the high resolution of the images. In

general, we believe that if the data acquisition is carried out with the intention for use in our proposed pipeline, the results will improve both the automatic object detection and 3D position estimation tasks.

Specifically, it is important to back the camera with a more reliable and accurate GPS receiver. Currently there are commercially available devices claiming accuracies below 10 centimetres [71] at a cost of a few hundred US dollars. They can achieve such accuracies by combining multiple Global Navigation Satellite Systems (GNSS) like GPS, Glonass, Galilleo and Beidou along with their differential counterparts (DGPS) [72]. In addition, a number of tie points along the railway will be of assistance for improved positions in post processing.

Similarly, it is important to perform stitching with all parameters available. In our experiments, the errors originated from stitching were insignificant because at best, we could only outmatch the two metres accuracy of the camera positions. However, in cases where an accuracy close to 10 centimetres is feasible, stitching errors will affect the final position error to a greater extent and therefore need to be addressed.

Regarding the detection on images, we believe that improvements can be made by acquisition of higher resolution imagery and a consistent training set. Most challenges on the detection task were raised by the small size of the objects. A training set that contains more and larger objects can improve the detection accuracy and simultaneously, minimise the false positive rate. This can be achieved in two ways. First, a set of cameras able to capture higher resolution images. Second, the creation of a database containing carefully annotated samples from multiple projects. We can continuously enrich this database with future similar projects from around the globe. This will help also towards generalisation of the detector for use in different datasets in multiple countries. To achieve this, we can introduce a step to this pipeline where the models are constantly improving through addition of new samples and elimination of incorrect samples. In this step, depending on the available data, we can perform either full retrain of the model or on-line training based on stochastic gradient descent

(in contrast to batch gradient descent in our batch-based training). The advantage of on-line training is that it does not make any assumption about the data distribution. This means that the data can adapt on the fly to changes in the dataset [73] caused by new samples.

Furthermore, the detection accuracy can be improved by applying optical flow and tracking to improve detections. The outcome of our limited experiments with optical flow was not satisfactory, mainly due to the misclassification rate of the detectors. However, we believe optical flow and tracking algorithms have the potential to improve detections. The improvement can be achieved based on object detections with high confidence. Then, a tracking algorithm [74–82] can predict the position of the same object on previous and consequent frames. The knowledge provided by the tracking algorithm can then be combined with the automatic detection results to improve classification accuracy (by correcting the class on detections with low confidence). Some tracking algorithms [78, 79, 81, 82] are robust to complete occlusion as well. This means that in cases where in a frame an object is not visible (eg. a tree covers it), the detector may completely fail to detect an object but the tracker will predict its position. Similarly, we can accept detections with a lower confidence in case they are located in areas where the tracker expects an object. Finally, tracking algorithms are computationally cheaper than deep learning detectors (even [82], a deep learning tracker, reports a 100 fps processing rate.). This allows us to substitute the object detector with a tracker after a single high-confidence detection of an object. In these cases, a single detection of a physical object will be even more valuable.

In addition, the equally high detection accuracy on the second dataset shows a potential in generalisation and transferability of CNNs in different datasets. However, the two datasets we used here are both from the same country (Australia) and therefore the railway objects follow the same rules. In general, railway signs and signals in different countries may vary a lot in sizes colours and configurations. We could not test the performance of our detector on a railway of a different country due to data

unavailability. We believe that it would be interesting to test the detector in railways of multiple different countries in order to define the level of generalisation.

Finally, we believe that the pipeline presented here for railway signs and signals can be applied on other types of objects with few or no adjustments. Such objects can be control boxes, poles, catenaries or even cables. Of course, in order to extend the pipeline to more objects, data for training the models would be necessary.

Chapter 7

Conclusion

In this thesis we developed, tested and demonstrated a full pipeline for the estimation of 3D positions of railway signs and signals from panoramic images. The pipeline consists of two main parts: automatic object detection and 3D object localisation.

For the first task, we applied multiple state-of-the-art object detection algorithms in two different datasets of railway objects. Our best implementation (RetinaNet) achieved 86.56% and 89.0% mAP on the first and second dataset respectively. The models used in both datasets were trained on the first dataset while for the second, we only fine-tuned the model. The high mAP on the second dataset shows the potential of transferability and generalisation of the method with minimum extra training. We believe that the accuracy can be improved by continuous training of the models with supplementary samples and by using higher resolution cameras.

For the second task, we applied bundle adjustment on object 3D positions based on known camera 3D positions. We achieved a 1.82 metres accuracy on average on objects of the second dataset. This accuracy is a slight improvement to the 2 metre average accuracy of the available camera positions. The accuracy in estimated positions is highly dependent to the accuracy of the known camera positions. The 2-metre accuracy of the camera positions did not allow us to investigate in depth the impact of other sources of error on the final position estimations such as the panorama stitching, bounding box misplacement on detection, camera orientation

change between consecutive frames. We believe that the approach we presented can match the accuracy of the camera positions even when they are known with less than two metres accuracy on average.

In summary, the accuracies achieved here can highly improve the existing manual, expensive and time consuming processes on railway map development both in terms of cost reduction and product delivery times. Finally, we presented our view on how we can improve this pipeline in the future.

Bibliography

[1] COWI, *Personal communication with people from mapping and geoservices*, 2020. [Online]. Available: https://www.cowi.com/tags/mapping-and-geo-services.

[2] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks", eng, *Advances in Neural Information Processing Systems*, vol. 2015-, pp. 91–99, 2015, ISSN: 10495258.

[3] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection", *CoRR*, 2017. eprint: 1708.02002.

[4] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks", 2017, pp. 5987–5995. DOI: 10.1109/CVPR.2017.634.

[5] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks", *CoRR*, 2017. eprint: 1709.01507.

[6] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick, "Microsoft coco: Common objects in context", eng, *Lecture Notes in Computer Science*, vol. 8693, pp. 740–755, 2014, ISSN: 16113349, 03029743.

[7] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge", *Int. J. Comput. Vision*, vol. 88, no. 2, pp. 303–338, 2010, ISSN: 0920-5691. DOI: 10.1007/s11263-009-0275-4.

[8] Corys, *Corys train simulators*, 2020. [Online]. Available: https://www.youtube.com/watch?v=t4cSsNuudXI.

[9] J. M. S. Prewitt, "Object enhancement and extraction", in *Picture Processing and Psychopictorics*, New York: Academic, 1970, pp. 75–149.

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation", eng, *Ieee Conference on Computer Vision and Pattern Recognition (cvpr)*, pp. 580–587, 2014, ISSN: 2332564x, 10636919.

[11] R. Girshick, "Fast r-cnn", eng, *Proceedings (ieee International Conference on Computer Vision)*, vol. 2015, pp. 1440–1448, 2015, ISSN: 15505499, 23807504. DOI: 10.1109/ICCV.2015.169.

[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector", eng, *Lecture Notes in Computer Science*, vol. 9905, pp. 21–37, 2016, ISSN: 16113349, 03029743. DOI: 10.1007/978-3-319-46448-0_2.

[13] C. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "DSSD : Deconvolutional single shot detector", *CoRR*, vol. abs/1701.06659, 2017. arXiv: 1701.06659. [Online]. Available: http://arxiv.org/abs/1701.06659.

[14] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection", eng, *Ieee Conference on Computer Vision and Pattern Recognition (cvpr)*, vol. 2016-, pp. 779–788, 2016, ISSN: 2332564x, 10636919. DOI: 10.1109/CVPR.2016.91.

[15] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger", *CoRR*, vol. abs/1612.08242, 2016. arXiv: 1612.08242. [Online]. Available: http://arxiv.org/abs/1612.08242.

[16] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement", *CoRR*, vol. abs/1804.02767, 2018. arXiv: 1804.02767. [Online]. Available: http://arxiv.org/abs/1804.02767.

[17] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks", in *Advances in neural information processing systems*, 2016, pp. 379–387.

[18] Z. Shen, Z. Liu, J. Li, Y.-G. Jiang, Y. Chen, and X. Xue, "Dsod: Learning deeply supervised object detectors from scratch", in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1919–1927.

[19] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6154–6162.

[20] T. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection", *CoRR*, 2016.

[21] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Single-shot refinement neural network for object detection", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4203–4212.

[22] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, "Libra r-cnn: Towards balanced learning for object detection", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 821–830.

[23] X. Zhang, F. Wan, C. Liu, R. Ji, and Q. Ye, "Freeanchor: Learning to match anchors for visual object detection", in *Advances in neural information processing systems*, 2019.

[24] Y. Li, Y. Chen, N. Wang, and Z. Zhang, "Scale-aware trident networks for object detection", in *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[25] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints", in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 734–750.

[26] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks", in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 764–773.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., Curran Associates, Inc., 2012, pp. 1097–1105.

[28] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders, "Selective search for object recognition", *International Journal of Computer Vision*, 2013. DOI: 10.1007/s11263-013-0620-5. [Online]. Available: http://www.huppelen.nl/publications/selectiveSearchDraft.pdf.

[29] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN", in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[30] C. Zhu, F. Chen, Z. Shen, and M. Savvides, *Soft anchor-point object detection*, 2019. arXiv: 1911.12448 [cs.CV].

[31] L. Huang, Y. Yang, Y. Deng, and Y. Yu, "Densebox: Unifying landmark localization with end to end object detection", *arXiv preprint arXiv:1509.04874*, 2015.

[32] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, "Unitbox: An advanced object detection network", in *Proceedings of the 24th ACM international conference on Multimedia*, ACM, 2016, pp. 516–520.

[33] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Object detection with keypoint triplets", in *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[34] X. Zhou, J. Zhuo, and P. Krahenbuhl, "Bottom-up object detection by grouping extreme and center points", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 850–859.

[35] T. Kong, F. Sun, H. Liu, Y. Jiang, and J. Shi, "Foveabox: Beyond anchor-based object detector", *arXiv preprint arXiv:1904.03797*, 2019.

[36] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection", in *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[37] J. Wang, K. Chen, S. Yang, C. C. Loy, and D. Lin, "Region proposal by guided anchoring", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2965–2974.

[38] Z. Yang, S. Liu, H. Hu, L. Wang, and S. Lin, "Reppoints: Point set representation for object detection", in *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[39] C. Zhu, Y. He, and M. Savvides, "Feature selective anchor-free module for single-shot object detection", in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[40] P. Maillard and F. Cavayas, "Automatic map-guided extraction of roads from spot imagery for cartographic database updating", *International Journal of Remote Sensing - INT J REMOTE SENS*, vol. 10, pp. 1775–1787, 1989. DOI: 10.1080/01431168908904007.

[41] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. USA: Cambridge University Press, 2003, ISBN: 0521540518.

[42] M. Lourakis and A. Argyros, "Sba: A software package for generic sparse bundle adjustment", *ACM Trans. Math. Softw.*, vol. 36, 2009.

[43] R. I. Hartley, "Euclidean reconstruction from uncalibrated views", in *Applications of Invariance in Computer Vision*, Springer-Verlag, 1993, pp. 237–256.

[44] R. Szeliski and S. B. Kang, "Recovering 3d shape and motion from image streams using nonlinear least squares", eng, *Journal of Visual Communication and Image Representation*, vol. 5, no. 1, pp. 10–28, 10–28, 1994, ISSN: 10959076, 10473203. DOI: 10.1006/jvci.1994.1002.

[45] P. Beardsley, P. Tort, and A. Zisserman, "3d model acquisition from extended image sequences", eng, *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 1065, pp. 684–695, 1996, ISSN: 16113349, 03029743. DOI: 10.1007/3-540-61123-1_181.

[46] A. W. Fitzgibbon and A. Zisserman, "Automatic camera recovery for closed or open image sequences", eng, *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 1406, pp. 311–326, 1998, ISSN: 16113349, 03029743. DOI: 10.1007/bfb0055675.

[47] H. Y. Shum, Q. Ke, and Z. Zhang, "Efficient bundle adjustment with virtual key frames: A hierarchical approach to multi-frame structure from motion", eng, *Proceedings of the Ieee Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 538–543, 1999, ISSN: 2332564x, 10636919.

[48] Z. Zhang and Y. Shan, "Incremental motion estimation through modified bundle adjustment", eng, *Proceedings 2003 International Conference on Image Processing (cat. No.03ch37429)*, vol. 2, II–343–6 vol.3, 2003.

[49] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch, "Visual modeling with a hand-held camera", eng, *International Journal of Computer Vision*, vol. 59, no. 3, pp. 207–232, 2004, ISSN: 15731405, 09205691.

[50] M. I. Lourakis and A. A. Argyros, "Efficient, causal camera tracking in unprepared environments", eng, *Computer Vision and Image Understanding*, vol. 99, no. 2, pp. 259–290, 2005, ISSN: 1090235x, 10773142. DOI: 10.1016/j.cviu.2005.02.001.

[51] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: Exploring photo collections in 3d", eng, 2010.

[52] K. Levenberg, "A method for the solution of certain non-linear problems in least squares", und, *Quarterly of Applied Mathematics*, vol. 2, no. 2, pp. 164–168, 164–168, 1944, ISSN: 15524485, 0033569x. DOI: 10.1090/qam/10666.

[53] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters", *SIAM Journal on Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963. DOI: 10.1137/0111030. [Online]. Available: http://dx.doi.org/10.1137/0111030.

[54] P. Rabinowitz, *Numerical methods for nonlinear algebraic equations*, eng. Gordon and Breach, 1970, 199 s.

[55] M. L. A. Lourakis and A. A. Argyros, "Is levenberg-marquardt the most efficient optimization algorithm for implementing bundle adjustment?", in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 2, 2005, 1526–1531 Vol. 2. DOI: 10.1109/ICCV.2005.128.

[56] D. G. Lowe, "Distinctive image features from scale-invariant keypoints", *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[57] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)", *Comput. Vis. Image Underst.*, vol. 110, no. 3, 346–359, 2008, ISSN: 1077-3142. DOI: 10.1016/j.cviu.2007.09.014. [Online]. Available: https://doi.org/10.1016/j.cviu.2007.09.014.

[58] H. Durrant-Whyte and T. Bailey, "Simultaneous localisation and mapping (slam): Part i the essential algorithms", *IEEE ROBOTICS AND AUTOMATION MAGAZINE*, vol. 2, p. 2006, 2006.

[59] https://www.flir.com. (2019). Ladybug3 firewire, [Online]. Available: https://www.flir.com/products/ladybug3-firewire/.

[60] A. S. View, *Tailored camera system*, 2017. [Online]. Available: http://www.applied-streetview.com/products/recording/cameras/.

[61] FLIR, *Overview of the ladybug image stitching process*, 2013. [Online]. Available: https://www.flir.com/support-center/iis/machine-vision/application-note/overview-of-the-ladybug-image-stitching-process/.

[62] Hugin, *Hugin - panorama photo stitcher*, 2016. [Online]. Available: http://hugin.sourceforge.net.

[63] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge", *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015. DOI: 10.1007/s11263-015-0816-y.

[64] G. Karagiannis, S. Olsen, and K. Pedersen, "Deep learning for detection of railway signs and signals", in *Advances in Computer Vision*, K. Arai and S. Kapoor, Eds., Cham: Springer International Publishing, 2020, pp. 1–15.

[65] G. Karagiannis, S. Olsen, and K. Pedersen, "Detection of railway signs and signals", *Under peer review in Machine Vision and Applications*, 2019.

[66] G. Karagiannis, S. Olsen, and K. Pedersen, "Geolocation of railway signs and signals", *Under peer review in ISPRS Journal of Photogrammetry and Remote Sensing*, 2020.

[67] K. Yi, Z. Jian, S. Chen, and N. Zheng, "Feature selective small object detection via knowledge-based recurrent attentive neural network", 2019.

[68] J. Noh, W. Bae, W. Lee, J. Seo, and G. Kim, "Better to follow, follow to be better: Towards precise supervision of feature super-resolution for small object detection", in *The IEEE International Conference on Computer Vision (ICCV)*, 2019.

[69] S. Sun, "Multiple receptive fields and small-object-focusing weakly-supervised segmentation network for fast object detection", *CoRR*, vol. abs/1904.12619, 2019. arXiv: 1904.12619. [Online]. Available: http://arxiv.org/abs/1904.12619.

[70] TECHE, *Te720 pro*, 2019. [Online]. Available: https://www.teche720.com/en/te720.html.

[71] hemisphere, *C321+*, 2020. [Online]. Available: https://insidegnss.com/hemisphere-gnss-releases-next-generation-s321-and-c321-gnss-smart-antennas/.

[72] G. P. S. (GPS), *Nationwide differential gps system (ndgps)*, 2016. [Online]. Available: https://www.gps.gov/systems/augmentations/.

[73] J. Duchi and Y. Singer, "Efficient online and batch learning using forward backward splitting", *Journal of Machine Learning Research*, vol. 10, pp. 2899–2934, 2009. DOI: 10.1145/1577069.1755882.

[74] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features", eng, *Proceedings of the 2001 Ieee Computer Society Conference on Computer Vision and Pattern Recognition. Cvpr 2001*, vol. 1, I–511–18 vol.1, 2001, ISSN: 2332564x, 10636919.

[75] P. Viola and M. Jones, "Robust real-time face detection", eng, *Proceedings of the Ieee International Conference on Computer Vision*, vol. 2, p. 747, 2001. DOI: 10.1109/iccv.2001.937709.

[76] B. D. Lucas and T. Kanade, "Iterative image registration technique with an application to stereo vision", eng, *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, vol. 2, pp. 674–679, 1981.

[77] C. Tomasi and T. Kanade, "Detection and tracking of point features", eng, 2010.

[78] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters", eng, *Ieee Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 6870486, 583–596, 2015, ISSN: 19393539, 01628828, 21609292. DOI: 10.1109/tpami.2014.2345390.

[79] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, 2012, ISSN: 1939-3539. DOI: 10.1109/TPAMI.2011.239.

[80] B. Babenko, M. Yang, and S. Belongie, "Visual tracking with online multiple instance learning", in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 983–990. DOI: 10.1109/CVPR.2009.5206737.

[81] A. Lukezic, T. Vojir, L. Cehovin Zajc, J. Matas, and M. Kristan, "Discriminative correlation filter with channel and spatial reliability", in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[82] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks", eng, *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9905, pp. 749–765, 2016, ISSN: 16113349, 03029743. DOI: 10.1007/978-3-319-46448-0_45.