Master thesis

Track trespasser detection from a moving train with a multimodal camera setup

Maarten Vangeneugden

August 19, 2021

Abstract

This work researches the possibility of a viable safety system for trains, specifically: a system that's able to detect track trespassers in a timely manner, to allow the train driver to take adequate action. In order to accomplish this, a multimodal camera setup was used in an attempt to combine the properties of different camera types. In collaboration with the NMBS/SNCB/NGBE, the footage required for testing the written software could be obtained, resulting in an hour of footage of a normal RGB camera, an RGB camera with telescopic lens, and a short-wave infrared camera. The footage of the telescopic camera was a bit vague and using the full potential of optical zooming was not possible, but the quality was sufficient to demonstrate the feasability of the proposed system. To extract the region of the recorded footage on which to perform motion analysis, four different rail detection algorithms were programmed and evaluated, both in terms of accuracy and efficiency. One of these (rail contour matching) delivered very good results. Information from this was used to apply motion detection on the relevant region in the recorded footage. When this region falls outside of the area covered by the telescopic camera, super-resolution was used to mitigate the loss in detail as much as possible. Different models were compared for this purpose, of which the Lanczos technique gave the best results in terms of quality and required time. The available time did not allow for a lot of detailed improvements or expansions to the created software, but various possibilities have been listed that would solve the main problems present in the current implementation.

Keywords: Motion detection, image processing, trains, AI, safety.

Foreword

When the subject for this thesis was published at Ghent University, the original title was "Early detection of moving objects with multi-base stereo cameras from a moving vehicle". I was introduced to the work of my supervisor Drs. Ing. Gianni Allebosch and my promotors: A patented algorithm that exploited the parallax effect to efficiently discern between moving and stationary objects [1].

Amongst the different subjects to choose from, it stood out as a very interesting subject; given its very general-formulated title, there was a possibility to give it a unique spin: Combining a safety-oriented project with my personal interest in trains. This also offered some interesting advantages over urban applications.

Since then, the focus has shifted in a significant way. So much so that the aforementioned algorithm couldn't be used in this thesis anymore. Instead, it became a very communication-intensive project; finding the right people in a big government company on its own would be difficult enough, but the COVID-19 pandemic made it very resource-intensive work, since unnecessary direct contact was forbidden by Belgian law. Since a lot of people have been working from home, most communication went through e-mail.

My promotors, supervisors and I had sincere doubts whether finding all the right people in time would be possible. In that regard, I have been very fortunate to receive a lot of help with finding the right people. Several entities that could be of help have been contacted (Lineas, Infrabel, DVIS[1]), but only one returned a positive answer; the employees of NMBS/SNCB/NGBE[2] introduced us to the practicalities of recording videos from a train driver's perspective, including the problems that go along with it.

Solving this problem is not solely out of personal interest, but also in the interest of increasing the safety of daily railway operations: Track trespassers cause a lot of delays and accidents on the Belgian railways. This thesis might not have an effect on the delays, but it could be a help with avoiding accidents.

Although the use of multiple cameras for motion analysis is a mature research topic, No academic paper could be found that applied this for trains.

As with a lot of projects, they are not the sole work of a single person. I have had the support of a lot of people in the process of writing this thesis. So before continuing, I'd like to express my gratitude for their help to the following people:

- My promotors Prof. Dr. Ir. Peter Veelaert and Dr. Ing. David Van Hamme, for allowing me to take on this subject, but also to let me give a personal spin on the project.
- My supervisor(s) Drs. Ing. Gianni Allebosch and Drs. Zuhaib Ahmed. Both have been a huge help in providing theoretical base knowledge and technical assistance, such as cameras to use for recordings. I have had a lot of contact with them throughout this (particulary contactless) year, and their insights have aided me a lot as well.
- The NMBS/SNCB/NGBE and their (former) employees, in particular Geert Thys, Hugo Raddoux and Roger Ceunen, who were instrumental in making it possible to record the footage on a train in operation, a necessity for finishing this thesis.

[1] Dienst voor Veiligheid en Interoperabiliteit der Spoorwegen

[2] Nationale Maatschappij der Belgische Spoorwegen (NL), Société Nationale des Chemins de fer belges (FR), Nationale Gesellschaft der Belgischen Eisenbahnen (DE)

- Jos Verboven, mijn grootvader die mij in contact bracht met de juiste mensen bij de NMBS, en van wie ik de liefde voor treinen heb overgekregen. Hij stond ook altijd klaar als ik dringend iets op kot nodig had, en heeft zonder verplichting een belangrijke taak in mijn leven op zich genomen, en dat verdient op z'n minst een vermelding in deze thesis.
- Jonathan Driessen, the person that's literally been by my side almost every day, and who has been (and still is) a big support during difficult times.
- My friends and family, for their social support whenever I needed it.
- Tom Dries from ThermalFocus. Through his company, he was able to provide me with a short-wave infrared camera on very short notice and trusted me with his (very expensive) equipment, which was really helpful in collecting that footage.
- The technical side of this thesis could never be completed in time without the work of many other people. Developing software like OpenCV, Python, GNU, just to name a few, is a meticulous work of multiple years, if not decades. In informatics, we stand on the shoulders of giants, and only thrive when we share our software in freedom.

I hope this work will be a valuable addition to the academic field of informatics.

Extended abstract

This thesis proposes a possible addition to the current safety systems used by railway companies for timely detecting track trespassers; people that trespass the prohibited areas around railways without proper clearance to do so. This is an important problem to solve, because of the delays and accidents caused by these people. Current countermeasures to trespassers have mostly been passive techniques such as static cameras and fencing. However, to the best of our knowledge, no paper exists that explores what the possibilities are of a multi-base camera setup for solving this problem.

What became very clear during the literature study for this thesis was the low amount of relevant and useful papers in the field of railways; there is not a lot of reseach being done, and it's unlikely that this will change in the (near) future.

In order to evaluate the software that would be used for this project, there was a need to collect video footage from the perspective of a train driver. This footage had to be from a normal RGB camera, plus a telescopic RGB camera. Different types of cameras (thermal, infrared, ultraviolet) have been evaluated, to see if (and how) their properties could be used to augment the final image to be used for processing. A short-wave infrared camera seemed like the best choice for this, but unfortunately, the results of this were underwhelming, making the use of so-called fusion algorithms impossible. Luckily, in hindsight, this didn't seem to be a very big issue.

Figure 1 shows a condensed diagram of the entire system in normal operation.

./afbeeldingen/diagram.png

Figure 1: General diagram of how the program functions.

To that end, a rail detector mechanism is required to extract the region of interest. That is; the area in the recorded footage where the train will be heading towards. Rail detection is a problem that doesn't have a generally agreed on solution, so part of this thesis revolves around creating and evaluating different techniques to detect rails in a given frame.

Eventually, the best method (both in terms of speed and accuracy) consisted of a combination of the Canny edge detector, together with a contour matcher to extract the edges that belong to the rails. The system uses information from the previous iterations, and uses a homography matrix to perform error-correction when one of the rails can't be detected properly. An example of this rail detection is given in figure 2

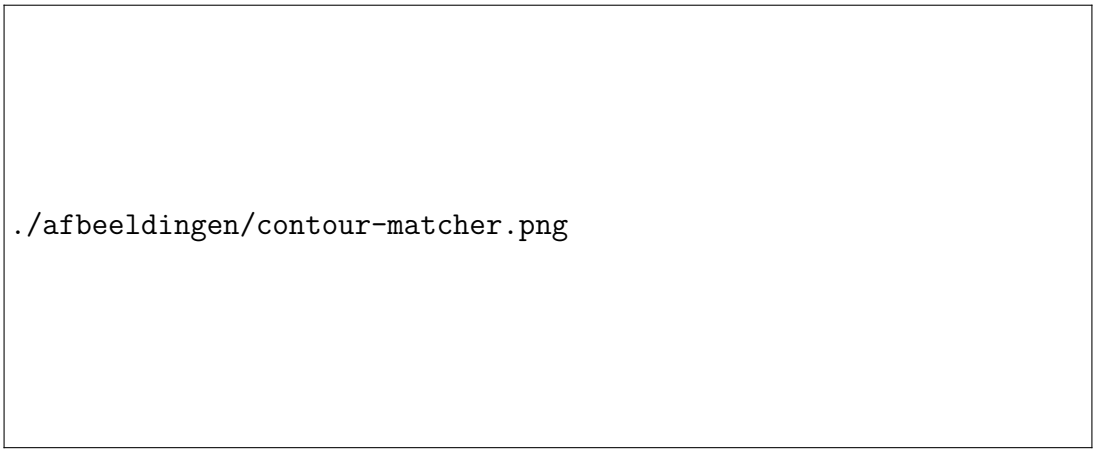

Figure 2: Rail contour matching in action. The colour does not indicate which rail is it, but how that rail could be detected. Green indicates The system could detect the rail without any problem. Orange Means the rail was not detected, but was estimated based on the green rail, using a homography matrix transformation and offset based of the width of the railway. It's clear this estimation is very accurate.

In some sharp curves (in the collected dataset only on some corners on the North-South connection in Brussels), it's possible that the region to be extracted doesn't fall within the area covered by the telescopic camera, in which case the normal camera has to provide a fallback region, which of course can't provide the same amount of detail as the telescopic camera. To try and increase the clarity of the image, seven different models for super-resolution were evaluated, consisting of both neural network models and "simple" algorithms. The results are listed in tables 1 and 2, respectively for 2× and 4× upscaling. Values in bold indicate the best score. When looking at the best results for the least amount of time, Lanczos comes out as the best technique. The fact that its results are comparable to neural network super-resolution models could be attributed to the neural networks undergoing "general" training; they're not trained on railway data specifically, but on general image sets. If a railway-specific image dataset would be available, it's very likely that the neural networks would perform much better than Lanczos.

Table 1: Comparison with 2× upscaling

Model	Required time (s)	MSE	PSNR (dB)	SSIM
EDSR	486.4067	**25.5789**	**34.0207**	**0.9615**
ESPCN	0.7349	34.6671	32.6645	0.9488
FSRCNN	0.8755	31.0755	33.1262	0.9536
LapSRN	6.9317	37.9758	32.2725	0.9457
Bicubic	0.0198	41.7501	31.9043	0.9447
Nearest neighbour	**0.0053**	91.0413	28.5286	0.9061
Lanczos	0.1181	35.8823	32.5594	0.9513

Table 2: Comparison with 4× upscaling

Model	Required time (s)	MSE	PSNR (dB)	SSIM
EDSR	140.03269	**172.1500**	**25.7540**	**0.7526**
ESPCN	0.2448	200.5467	25.0830	0.7181
FSRCNN	0.3972	200.3464	25.0896	0.7163
LapSRN	9.09537	193.1753	25.2479	0.7235
Bicubic	0.0148	208.8358	24.9160	0.7257
Nearest neighbour	**0.0057**	270.4777	23.8019	0.7000
Lanczos	0.0987	205.8743	24.9765	0.7253

Motion analysis is performed using a relatively simple motion detection algorithm: By subtracting the (estimated) background from a set of consecutive frames, motion that takes place far away in front of the train can be detected. The simplicity of the implementation allows fast execution of the motion detector, which is a requirement for this application. The detected motion is presented to the train driver in the form of a heatmap, an example of which can be seen in figure 3.

Figure 3: Example of the motion detected in the frame, presented to the train driver in the form of a heatmap.

The complete system as proposed gives a good indication that a specialised solution for this problem is feasible, but a lot of work remains. Further improvements should ideally focus on fixing some of the main problems currently in the rail and motion detector, after which expansions could be added to further improve the accuracy and possibilities of the entire system.

Contents

1 Introduction

1.1 What does this thesis research?

Three research problems are being addressed in this thesis:

- Examining the possibilities of railway detection and comparing different techniques for doing so
- Developing a practically useable basis for further research in this field
- Trying to see if creating a working motion detection system is viable for this particular application field

Railway detection is an important requirement for a safety system for trains. It makes it possible to determine where the train is heading for, to discern between railways and motorways [3], to calculate speed, ... In short, it provides a lot of useful information that can be exploited to remove unwanted processing.

The visual acuity of humans is limited in distance. We can only see details in our close environment. While binoculars allow us to see details from objects that are far away, they're not a suitable solution for train drivers, since they must be able to operate the train. Still, being able to see over great lengths would definitely help in timely detecting track trespassers, a major societal problem that will be discussed in detail momentarily.
This problem is (read: seems) closely related to detecting pedestrians in automotive applications. What used to be a safety assistance module in newer cars is starting to become a vital requirement with the advent of self-driving vehicles. The technology for this is already available.
And yet, in railway environments, applications of this technology have been notably absent throughout the years. In fact, almost all railway companies rely on their train drivers to see trespassers before it's too late, despite their natural inability to do so.[4]
Clearly, research into interpreting railway environments can serve as a solution to a big problem that hasn't had the attention it deserves. Even if this thesis did nothing but kickstart further innovation in this field, that in itself would be a big leap forward in making the backbone of Europe's public transport infrastructure safer and more reliable.

Without it, detecting track trespassers would be a lot more difficult. For one thing, it would require complex motion detection analysis, which implies heavy post-processing that would need expensive hardware to finish in time.

The common goal of answering these research problems is to design a system that can interpret the railway situation far away (and up close) in an efficient manner. This means: A software solution that relies on understandable, easy to compute algorithms, that delivers results in real-time. The solution also tries to take financial cost into account: The most performant options also come with a hefty pricetag, for which no money may be available.

This thesis also takes a look at existing research in this field, particulary with applications to railway environments. Admittedly, compared to road vehicle applications, there are a lot less academic resources to explore.

[3] For example, the E19 between Brussels and Antwerp and the E40 between Brussels and Liege.
[4] This is also being supplemented by passive measures, such as fencing in high-risk areas.

The few resources that do exist differ in quality, but some of them provide interesting findings that will help with implementing the software for the detector, and will be mentioned in chapter 4.

1.2 Why railways?

1.2.1 Less variables

The general goal of this thesis (i.e. when it was published) was to research the possibilities of motion detection with stereo cameras. This is currently a hot topic with car manufacturers, where driver assistance technologies are being researched as well.

However, from a computer's (and oftentimes a human's) perspective, roads are relatively chaotic environments. There are pedestrians, trams, cars, lorries, ... all with their own speed, traffic rules, destinations, ... The amount of variables one has to account for is gigantic.
This also means that motion analysis becomes a lot more difficult; not only should it be able to make a distinction between real and false motion (how is a tree 'moving' past you different motion from a pedestrian walking past you?), but it should also determine the type of motion, and if it can pose a danger (A pedestrian crossing the street is potentially dangerous, but an opposing car making a right turn isn't).

Railways on the other hand, are very structured environments, with strict guidelines that all railway personnel have to abide to. The rigid structure of railways can be translated to assumptions that help with simplifying the solution.

I also foresee that this thesis can be a part of future applications with automatic train operation, as being able to interpret the railway environment (correctly) is a critical requirement before we can even begin to think about letting a train depart without a human operator.

1.2.2 A direct practical use

Although the general purpose of this thesis is investigating the possibilities and properties of images from multiple cameras, I also opted for a practical application of this research, one that heavily affects Belgium as a whole: The problem of track trespassers. In spite of all the campaigns, the implemented safety measures, and the obvious danger trains pose, they're responsible for a lot of delays in Belgium. It goes without saying that those who are unfortunate enough to get hit by an oncoming train often cause emotional suffering with both train drivers and relatives that are left behind.

Delays Infrabel have done extensive research on both the causes and effects of track trespassers, in order to develop a strategy to tackle this problem. For 2020, Infrabel determine that track trespassers are responsible for 95.943 minutes of delay, which rougly translates to over four hours per day. This was caused by just 614 trespassers. This happens because a delay of one train propagates throughout the rest of the network, causing delays for other trains as well, either because the route is blocked, or because passengers need to make an interchange, delaying waiting trains. These delays eventually also have an (indirect) economic impact, which is

hard to determine exactly, but surely not negligible. Furthermore, they also decrease the trust of the population in trains as a reliable option for transportation.

A chronic problem In spite of all the safety measures and campaigns, people die every year because of this [2]. Alas, trespassers are here to stay: Infrabel think that more than half of all trespassers are convinced that they're capable of correclty estimating the dangers. 32% are "Rebelling risk takers"; people that know their actions are dangerous, but decide to undertake them anyway. Only direct repression (i.e. prosecution) can have an effect on this group, but this does not happen a lot.

1.3 Structure

Since this thesis requires obtaining footage from a train during normal operation, examining what types of footage to use will be the first step, and starts with chapter 2, where camera technologies will be discussed. The findings of that will be used in chapter 3, where the set-up used will be discussed, including an evaluation of the recordings themselves. Chapter 4 starts with a quick introduction on homography and why it's useful in this thesis, followed by a description and evaluation of four different rail detection algorithms. Chapter 5 discusses super-resolution techniques, which could be used as a fallback for when the telescopic camera can't deliver enough information. Chapters 6 and 7 focus on the safety solutions for the train driver: Respectively, calculating the stopping distance and detecting suspicious motion in the vicinity of the railways. Finally, chapter 8 makes the closing remarks of this thesis, with a short conclusion, accompagnied by a list of possible improvements for this project.

1.4 Terminology and translations

Even though this thesis is in English, this is not an official language in Belgium. Since a significant part of the text is related to the national railway company, there are no official translations available for so-called railway jargon, as these only exist in Dutch and French. While most terminology can (and will) be translated (for example "stuurpost" and "konvooi"), there are also some terms that are more difficult to translate correctly (for example "miszending"). These terms (and their abbreviations) are translated as well as possible, but might not always be clear. In that case, further information is provided in the glossary on page 55.

2 Camera technologies

Of course, research that uses cameras for detection requires a thorough comparison of different camera technologies that are available. Knowledge of the advantages and disadvantages of different cameras allows to make an informed decision about the final camera setup that will be used for the field recordings.
Only the properties that are related to this thesis will be discussed; explaining all differences and characteristics would not be relevant to the subject at hand, putting it beyond the scope of the thesis.

The cameras that are being explained in detail here are listed in ascending order of the wavelengths they operate on. A diagram of the electromagnetic spectrum can be found in appendix A.

2.1 Ultraviolet camera (UVC)

Ultraviolet cameras (UVCs) capture light rays whose wavelengths lie between ionizing radiation and the visible spectrum. Ultraviolet (UV) rays are most commonly known for causing sunburn, but also have interesting properties in fields like medicine and security.

However, with regards to what is important for this thesis, UVCs only have disadvantages. The cameras themselves are a lot more **expensive**, and recording it delivers no information that can't be obtained with a standard RGB camera.

Just like RGB cameras, UVCs **can't see during nighttime**. In addition to that, with the increased usage of Light-emitting diodes (LEDs), the recordings from these cameras would even be completely **dark in illuminated environments without sunlight**, like tunnels.[5] Because of all these problems, UVCs will not be considered further.

2.2 Standard RGB cameras

These cameras don't require a lot of explanation: Standard RGB cameras (henceforth referred to as "common cameras" or "CCs") capture the electromagnetic waves that our eyes are able to see as well, rendering images that closely resemble reality as perceived by humans.

The primary advantage is that **CCs are affordable**; they are the most ubiquitous, and don't cost much when compared to the quality they offer. That same ubiquity also means that CCs typically offer higher resolutions than cameras for other wavelengths.

The technology for these cameras has progressed enough that even high-quality models come in (very) **small sizes**, making them ideal for placement inside the driver's compartment, because the camera's mustn't obstruct the view in a way that could pose a danger.

Now, the most important feature to use this camera, is the possiblity to use a telescopic lens. While all these types can perform a digital zoom, a telescope is required to perform an optical zoom, allowing for sharp, detailed footage on a long distance.

However, they also have an important disadvantage, because **they have trouble seeing during the night.** Luckily, technological advancements still make it possible to record clear footage in dark environments, if we're willing to accept noise and glaring effects in the recordings.

2.3 Near infrared cameras

NIRCs operate on infrared signals that lie just beyond the spectrum visible to humans.

NIRCs **can see through smoke and fog**, unlike common cameras. This is a major advantage for Belgian railways: Not only is fog a common weather occurrence, but a lot of railroads go through rural areas, where morning mist can severely impede the view from afar. NIRCs do not suffer from this phenomenon at all, but since they can't see colour, the recordings are only in greyscale values.

[5] The reason LEDs are known for their efficiency, is that they only emit light rays with visible wavelengths, unlike other light sources like halogen lamps. The energy that would otherwise be wasted on invisible light (like UV light) can then be saved.

Alas, NIRCs have a hard time operating in darkness. This can be somewhat mitigated with Infrared (IR) LEDs, since these emit light invisible to common camera's, avoiding flare problems. But the range of these LEDs is limited to 500 metres [3], the absolute minimum distance required for detection.

2.4 Short-wave infrared cameras

These cameras combine the advantages of both thermal and near infrared cameras: They can see through glass, smoke, and fog, while safely placed inside the driver's compartment. On top of that, they can generate a clear view during nighttime[6], so no external illumination is necessary.

Unfortunately, these advantages come at a **very high cost**, costing tens of thousands of euros for a high quality model. For that high cost, the quality of the recordings isn't very great.

On its own, Short-wave infrared (SWIR) cameras would not be very useful. However, combining the footage with that of a normal RGB camera could very well pose very interesting results in enhancing the footage on which the software can operate. CITE FUSION

2.5 Thermal cameras / Long-wave infrared cameras

The thermal camera (TC), while technically also an infrared camera, does have some notable differences from the other infrared cameras.

The most notable difference is that TCs rely on radiated signals, whereas other IR cameras use reflected signals.

TCs are especially useful at night, because they don't rely on visible light, only on the warmth emitted by bodies.

TCs operate on mid-wave and long-wave infrared light, which is mostly blocked by the earth's atmosphere[7].

Unfortunately, TCs have a major disadvantage: The infrared signals they are sensitive to are blocked by normal[8] types of glass. In fact, normal glass acts as a mirror for these waves, so recording from a driver's compartment would result in an image of the driver, not the railway.

Thermal camera issues On technical aspects alone, the thermal camera would probably be the most useful choice to complement the RGB camera, because detecting heat signals would make it possible to detect trespassers in a relatively straightforward way.

However, this does require that thermal sequences are also available, and there are no datasets available that feature thermal railway footage. The remaining solution is thus attaching a TC to the outside of a train. In consideration with all the people involved, it was decided not to make thermal vision recordings.
The camera would be exposed to very extreme weather conditions because of the train's speed. It would also be necessary to find a safe way to attach the thermal camera to the front of the train, and it was not possible to

[6] SWIRCs can operate at night because of night sky radiance, the result of light from the ground being reflected in the sky. However, in absolute darkness, these cameras can't see anything, contrary to thermal cameras.

[7] This is why on thermal images, the sky is always black; the radiation it can detect is naturally filtered.

[8] There are some types of glass that do not block these infrared waves, but this glass is not used in Belgian trains.

foresee which type of locomotive would be used on the day of recording.
Even assuming it's possible to do this, other students and researchers at Ghent University rely on this thermal camera for their research, and long-wave infrared sensors don't come cheap. The camera could fall from the train, or become defective some other way. All these arguments together meant it was not possible to use a thermal camera for the recordings.

One might argue that the thermal recordings don't necessarily need to be from railway environments, and urban road footage would suffice, which could either be recorded manually, if not available in any publicly available dataset. But even then, some important problems remain which makes transferring findings to a railway environment not feasible:

- The camera that was available at Ghent University Image Processing and Interpretation department (IPI) features a recording resolution of 368×288, which is very low and only useful for movement that's very close to the camera already. There is no optical zoom hardware available for this camera, so there was no guarantee that the footage would be usable, let alone helpful.

- Urban environments feature a lot more movement from a lot more objects, all with their own heat, due to the intrinsic 'controlled chaos' of the setting. This makes it hard to discern what should be considered important, and what can safely be ignored as noise.

2.6 Closing remarks

Using different camera types to exploit the advantages of different technologies seems an interesting approach, where the normal RGB camera footage would serve as the basis from which to build a decent basic system.

It should be noted that most of the aforementioned problems would not pose a problem during clear environmental conditions, but that can't be guaranteed in Belgium. Fog, mist, rain, darkness... All could have a negative effect on the software, resulting in a more complex solution (to deal with it during processing) or less useful results.

None of the camera types examined offer a perfect solution. However, it is possible to use a so-called fusion algorithm that combines the information from IR and Red/Green/Blue (RGB) images into a single image [4]. There are numerous different fusion models available [5], so it would be worthwile to obtain footage from different camera types and see if that is a viable solution for this project.

That's why the eventual setup will consist of a normal RGB camera, an RGB camera with a telescopic lens, and a SWIR camera for the IR footage. Out of all the different IR camera types, the SWIR camera has the most complementary properties to the RGB camera, and should give the best results when using a fusion algorithm.

These cameras will be used in the next chapter, where the recordings themselves will be discussed.

3 Recordings

Applying the written software and evaluating it requires acquiring video footage of a realistic train journey. In the previous chapter, the different camera types for doing so were examined. After looking at the relevant properties,

the choice was made for using a fusion algorithm to combine the advantages of IR and RGB recordings. In this chapter, the expected problems prior to recording are listed, followed by a brief discussion of the day of recording. Afterwards, the recordings themselves are analysed.

3.1 Possible recording issues

Knowing all problems beforehand is not possible, but there are some general ones that we can expect. These are explained in detail here, but also how we expect to mitigate them by the proposed set-up.

3.1.1 Distant detection

Belgian trains can often reach speeds of 160km/h. Some popular services like the Ostend-Eupen IC sometimes go about 200km/h. With even half that speed, a train can easily need half a kilometre before coming to a full stop with an emergency brake.

Needless to say: A train driver needs to be able to detect suspicious movement as far away as possible, in order to intervene before it's too late. A detector that can only detect what the train driver can already see on his own is effectively useless.

We try to solve this problem by utilizing telescopic lenses: This would provide a zoomed view of what can be expected, long before the train driver could take notice.

Using a telescopic lens also provides an additional benefit: If the area of the telescopic footage is cropped so it only contains the area we're interested in (specifically, the area around the railway), the difference in subsequent frames becomes a lot more negligible, which could remove the need for a complex (i.e. computationally expensive) motion detection algorithm, further reducing the time needed to process the images.

3.1.2 Night time

Train drivers can't see in the darkness. Especially in long tunnels, even the headlights are of little use in illuminating the railway. Instead, train drivers have to follow a safety protocol when entering tunnels.

A possible help for this is a thermal camera, which can detect heat signals, independent of weather conditions. However, as was pointed out in the previous chapter, thermal cameras can't see through normal glass, and were ruled out for use in this project. The proposed fusion of SWIR and RGB footage should act as an alternative to the thermal camera.

3.1.3 Sun glare

Especially during sunset, the camera images may be exposed to a strong light source that can obscure the resulting video.

It's possible that other camera types don't suffer from glare as much as regular cameras do. This would make it possible to fuse the footage of different recordings to remove the glare.

3.2 Recording setup

3.2.1 Cameras

After taking all the different properties from the camera technologies into account, the following camera setup was used:

- One RGB camera
- One RGB camera with telescopic lens
- One SWIR-camera

How these cameras fit into the complete system is shown in figure 4. Additional sensors to record other types of information (speedometers, GPS, ...) were not used during recording; some information could be deduced from the footage itself (such as the speed), or it was not clear how this additional information could be an added benefit.

Figure 4: General diagram of how the system processes the recorded footage into Useful information for the train driver. Note that the SWIR camera is not used in any way, because the footage could not be used in a fusion algorithm because of numerous recording problems. Fortunately, the normal RGB footage was clear enough in dark environments as well, so this was not too big of an issue.

3.2.2 Driver's compartment

The normal RGB camera is mounted above the telescopic camera, and are vertically on the same line. The SWIR camera is mounted next to the other cameras, so as to not obstruct the view of the train driver too much, and

because there was no possibility to mount it in another position. The construction in the driver's compartment is shown in figure 5.

Earlier field visits to the Schaarbeek marshalling yard[9] were made to collect information about difficulties related to making railway recordings.

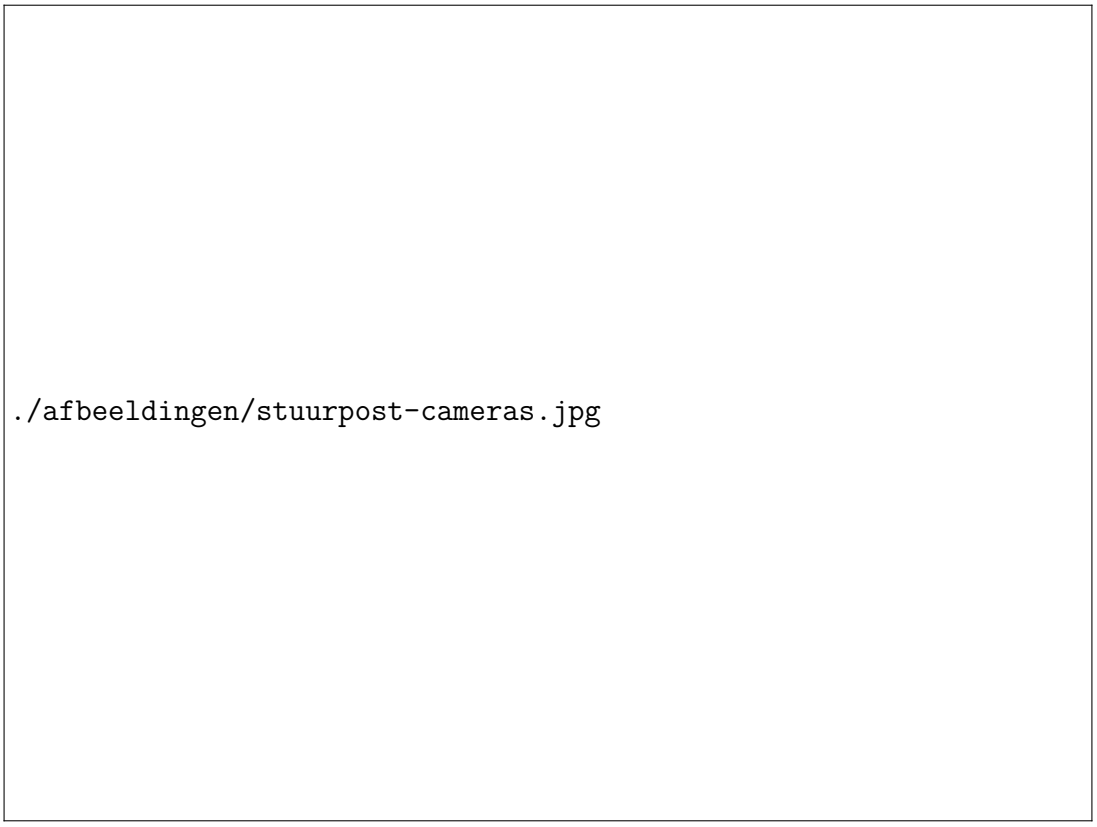

Figure 5: Set-up of the different cameras in the driver's compartment. Picture made at the Forest marshalling yard [11] prior to departure to Brussels-South.

3.2.3 Route

The recordings were made at Wednesday, May 12th 2021, starting at 09:33 from Brussels-South [12] and ending around 10:35, minutes before entering the Namur train station [13].

This route (line 161) was chosen for its interesting properties; it features a lot of hills and tunnels, wide and tight curves. The line cuts through both rural and urban regions and the line was being upgraded to four tracks during the time of recording, so there would be a high probability of movement around the railroad environment. To test how the SWIR camera (and by extension the other cameras) would behave in darkness, the lights of the tunnel between Brussels-North[14] and Brussels-Schuman were manually dimmed for the duration of the recordings.

[9] Schaarbeek-Vorming (NL), Schaerbeek-Formation (FR/DE)
[12] Brussel-Zuid (NL), Bruxelles-Midi (FR), Brüssel-Süd (DE)
[13] Namen (NL), Namur (FR), Namür (DE)
[14] Brussel-Noord (NL), Bruxelles-Nord (FR), Brüssel-Nord (DE)

3.3 Recording results

3.3.1 Normal camera

The footage of the normal camera was very good, delivering clear images with a frame rate of 50Hz. The equipment also offered several additional techniques such as image stabilisation, which were set to automatically find the best configuration.

In darkness, these cameras perform way better than expected because the film speed was automatically configured, which did cause a noise effect in dark areas. That's not a really big problem, except for when exiting a tunnel, resulting in a flare-like moment until the ISO value is lowered. This glare was an expected issue, and could also happen with a low-hanging sun.

3.3.2 Telescopic camera

There were some issues with this footage, although none of the kind that poses a critical problem for using it in the thesis itself to test the software. Figure 6 shows the main properties of the footage.

Figure 6: Example of the telescopic footage, with normal footage at the same timestamp to give an idea of the optical zoom. The image is not as sharp as it should be, and the high ISO setting in dark areas causes the railway signs to flare up the recording. However, the most important property of this footage is preserved; an optically zoomed recording during standard train operation.

The camera turned itself off after every 20 minutes of recording. We were not aware of this at the time of recording, and since we're not allowed in the driver's compartment all the time, we were only able to record 40 minutes of footage for a one-hour trip.

An issue with the quality is that the focal point was not correctly set; even though the resolution is the same as the one from the normal camera, the footage is notably vague. Since the point of this camera was to have a clear image on a distance beyond the train driver's vision, this was a bit disappointing. However, this is not

inherently because of the telescope, and proper configuration in a real-life deployment setting would solve this problem.

3.3.3 SWIR-camera

In spite of the expected results, the supposed benefits of using a SWIR-camera could not be achieved, which was a bit unfortunate. Figure 7 shows a representative frame.

Figure 7: Image from the SWIR-camera footage. A bit interesting to note is that some railway signs seem to be enabled and some aren't. However, they're all enabled, but some are equipped with LEDs, which don't emit light in the infrared spectrum.

First, configuring this camera was not an easy task; the technology is very different from normal RGB cameras, and requires using special software by FLIR, the company that also made the SWIR-camera used. The software leaves all configuration to the user, without providing a way to automatically search the best settings, even though some configurations are effectively useless compared to others.

In normal circumstances, the camera was able to capture the environment well enough, given the limited resolution and recording frequency. Of course, having high-resolution images wasn't the goal of using this camera, so this wasn't that big of an issue.

An actual issue was the nearly useless footage in tunnels, shown in figure 8. Even with additional lighting sources, the SWIR-camera would require manual tuning during recording to see anything. This was particulary disappointing, because enhancing the RGB camera footage using the SWIR-footage was the primary reason to use that camera in the first place, which would make its high cost worthwile.

Figure 8: Leaving Brussels-Central on the SWIR-camera. One can make out some specific elements, but definitely not enough to use it for any kind of fusion with the RGB footage.

4 Rail detection

When building a system that's supposed to improve the safety of railroads, rail detection becomes a vital part of the solution. One could argue that there are easier methods to detect track trespassers, like static cameras on high-risk areas. However, a system that travels with the train is much more flexible, as it can be deployed in every railway environment. Sending a camera with the train during operation means that the camera is constantly pointed at the area that needs to be guarded, i.e. the area right in front of an oncoming train, which is technically the most riskful area. Also, detecting where the rails are makes it possible to decide what area of the video should be checked, and which areas are safe.

While performing a literature study into the current state of the art for this problem, it became very clear that there is very little published research available. While there definitely are some proposals (of which two will be discussed in this chapter), most related research focuses on detecting rail defects, for which multiple proposals have been made [6] [7] [8] [15]. So much so, that Zhang, Yu, Yang, et al. developed Rail-5k, an entire dataset dedicated to benchmarking rail defect detection algorithms.

If a paper actually does focus on rail detection, there are oftentimes complicating factors when trying to use its findings in this project; the detection is being done from an aerial perspective[12], the setup is not suited for deployment in regular operation [13], or some other specific problem.

[15] Most of these proposals are the result of applying a general-purpose Convolutional neural network (CNN) on labeled images, with YOLO [9] being a frequently used model. Very rarely, a paper that's more of an insult to research than an actual proposal also pops up [10].

In short, rail detection from the train driver's perspective is a very niche research field, for which no standard technique exists. This offers ample opportunity for making a considerable contribution to solving this problem.

First, a key component called the homography will be explained. After that, the two operational modes (called 'phases') of the program are discussed, followed by a detailed description and comparison of four different rail detection techniques.

4.1 Homography

This chapter makes a lot of use of so-called homographies, so explaining how it works and how it's used in advance can help in understanding the importance of it.

4.1.1 On geometric spaces

When recording a video, we record the view on the world from the perspective of the camera. The world is "projected" on the camera, if you will. With this projection, a 3D scene gets transformed into a 2D image, since a camera can only record in two dimensions. The obvious result of this is that information about that same world gets lost or altered in a significant way:

- Far away objects can be obscured by closer objects
- Distance between objects can't easily be calculated
- Parallel lines become concurrent lines
- All points in the world (seem to) converge in a vanishing point
- ...

However, it's also clear that there must exist a correlation between the 3D world and 2D image. In fact, this correlation happens to be a linear one, allowing us to relate the points in both spaces using linear algebra.

4.1.2 On coordinate systems

The concept of a coordinate is well known: a set of numbers that describe the position of a point in a space. Depending on the application at hand, different types of coordinates can be used, and some examples are given in figure 9.

Most people are familiar with the Cartesian coordinates, used in an Euclidean space. In this system, all coordinate axes are perpendicular to each other, with the same unit of length. This system is very straightforward to understand, because of its intuitive properties and applications in everyday tasks.

However, recall that different types of coordinates can be used to describe the same point. Depending on the problem at hand, using Cartesian coordinates could make the solution vastly more difficult. This is the case when we're working with projections of reality, so another type of coordinates is used for this purpose.

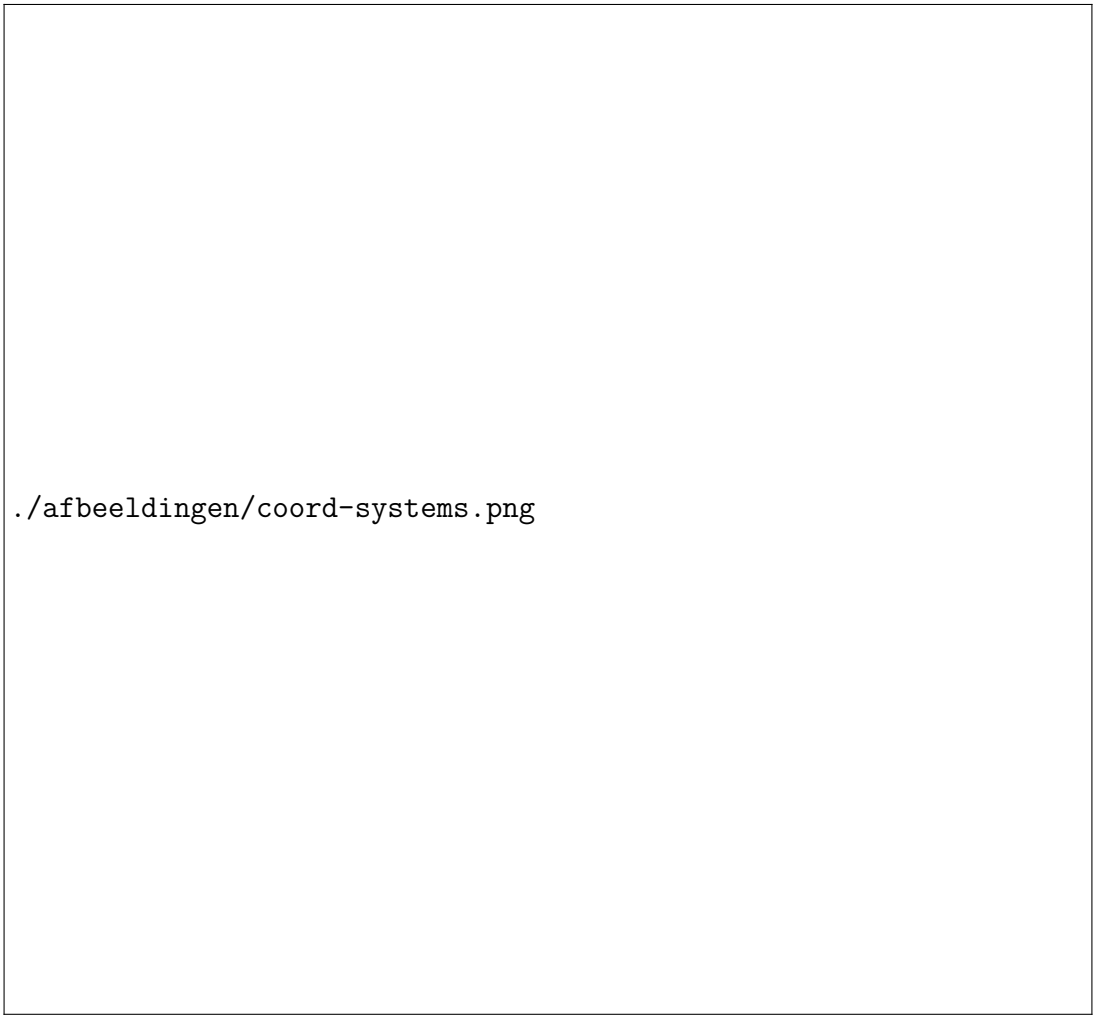

Figure 9: Examples of three different coordinate systems: Cilindrical (red), spherical (green), and Cartesian (blue). [14] Depending on the application, One coordinate system is vastly easier to use than the other, even though it's possible to convert between each type of vector; the same point can be described in all of these systems, so an equation can be formulated to do so. From respectively cilindrical and spherical to Cartesian: $(r, \phi, z) \mapsto (x, y, z) = (r \cos \phi, r \sin \phi, z)$ and $(r, \theta, \phi) \mapsto (x, y, z) = (r \cos \phi \sin \theta, r \sin \phi \sin \theta, r \cos \theta)$.

We introduce the concept of **projective coordinates**. Projective coordinates are related to Cartesian coordinates with the following formula:

$$(x_p, y_p, z) \mapsto (x_c, y_c) = \left(\frac{x_p}{z}, \frac{y_p}{z}\right), z \neq 0$$

With x_p, y_p and x_c, y_c respectively the projective and Cartesian coordinates.

One might immediately notice that, unline the coordinate systems listed in figure 9, projective coordinates have more numbers than the Cartesian coordinates they map to. This is because projective coordinates are a type of **homogenous coordinates**, meaning that different coordinates represent the same point if multiplied by a non-zero scalar, hence:

$$(x_p, y_p) = (\lambda x_p, \lambda y_p), \lambda \neq 0$$

The consequence of this is that coordinates that are on the same line in 3 dimensions, get represented by the same coordinate in 2 dimensions. This projection can be seen in figure 10.

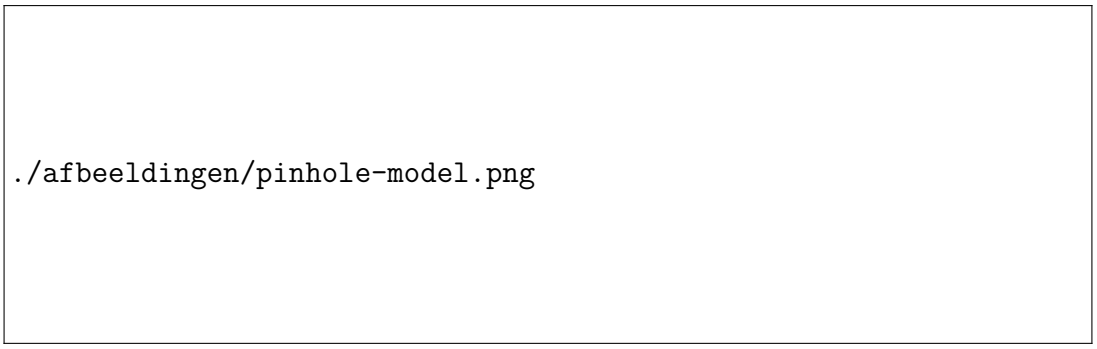

Figure 10: Visual explanation of the relation between projective and Cartesian coordinates. From the camera centre, we obtain a projection of reality that's being mapped on the image plane. This shows what is meant with how coordinates that only differ by a scalar represent the same coordinate. [15]

When we use the same names for variables as used in figure 10, we note that the relation between both coordinates is represented by:

$$x = f\frac{X}{Z}, y = f\frac{Y}{Z}$$

This is the same linear transformation between different coordinates when the concept of projective coordinates was introduced. These transformations can be written using homogenous coordinates like this:

$$Z \cdot \overrightarrow{x} = Z \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = K_f \cdot \overrightarrow{X}$$

Using homogenous coordinates, we can get rid of the division by Z, further simplifying calculations. Here, we have a linear transformation in the projective space, and by the properties of linear transformations (the chain rule in particular), we can thus also change the projection using a matrix multiplication, given that we know how that other projection looks like, i.e. how a certain object in the projection looks like in real life.

4.1.3 The homography matrix in practice

So how can this knowledge help with rail environments? In the previous section, it was established that there's a linear relation between different projections of the same thing.

This implies that we can construct a matrix that transforms coordinates from one projection into another projection. The advantage of this, is that this makes it possible to transform a deformed flat object into a rectified object. We call this the homography matrix **H**.

An important requirement for generating a transformation matrix **H** is that our image contains a **planar** object, i.e. an object for which we know it has an almost completely flat characteristic, e.g. roads, walls, railways[16], ...

Since **H** transforms between planes with equal dimensions, the inverse matrix $\mathbf{H}^{-1}$ exists, and the rectified object can be transformed into its deformed 'original' perspective.

Homography matrices make it possible to simplify a lot of algorithms; instead of having to embed a lot of extra code to account for the projection of reality, it's possible to assume Cartesian coordinates, and transform the final results to the "real" plane with a linear transformation. In this chapter, homography matrices will be used for rail detection. In later chapters, they will also be an important tool for distance measuring and speed calculation.

4.2 Normalisation phase

Upon starting the program, the preliminary normalisation phase will be executed first. Normalisation is a crucial step in order to remove variables that complicate the underlying algorithms. Although it's definitely possible (and easier) to manually normalize the frames, that's not a viable option to perform every single time the software is activated, let alone if a regular employee with no direct access to the software has to do it.

While this phase is active, the software will not perform any kind of advanced detection. The time necessary to finish this phase is not that important, since it's only being executed once. In the written implementation, this phase takes around five seconds to complete, which is more than fast enough.

The phase consists of a 60 second repetition while trying to deliver the required variables. If it hasn't been able to do so by then, it will display a warning on the resulting screen, notifying the driver that manual intervention is required.[17]

In normal circumstances, this phase will only be run once. However, it can also act as a "reset button" for the program, if the track detection is completely wrong for example.

The normalisation phase makes but one assumption: The track ahead of the train is straight, without curvature.

[16] One might say a railroad is not an example of a flat object, since the rails are laying on top of the track ballast, which itself is not really flat either. However, it's possible to think of the top of the rails as the floor of an invisible plane, which can't have any noticeable relief. Otherwise, the train would derail. It's on that part that the transformation matrix is being calculated.

[17] This manual intervention would consist of a very simple procedure, where the train driver would just click on four points of the displayed railway track in the shape of a trapezium.

4.2.1 Centralise

When the frames are sent to the other modules (rail detector, motion detector, ...), they must be oriented in a way that both rails are exactly in the middle, i.e. each rail must be the same distance from the centre column of the image. Thus, this step should return what part of the image should be clipped in order to get such an image.

To do this, the image is first transformed to a black-white image with binary thresholding. This helps accentuate the rails due to an interesting phenomenon: Friction of the train wheels on top of the rails cause a slight erosive wear that makes the top very glossy. The result is that, looking at the pixels, the rails are a lot brighter than their surroundings if there's a light source in front of the train (so the light rays are reflected into the camera). Due to this, it's possible to use a high threshold for the black-white filter.

A trivial rail detector will then begin from the bottom of the image, and searches for two rails, starting from the centre, going outwards to the borders.[18] When it detects a sudden change from black to white (i.e. a rail), it will mark it as a part of the rail.[19]

When both starting points for the rails have been discovered, they are iteratively extended to the top, by colouring neighbouring white pixels.
Do note that only possible neighbours are considered, not all adjacent pixels are automatically marked as rail. This extension process is also fault-tolerant, because there can be errors in the individual pixels, or rail pixels can overlap with overilluminated environments. A good example of this is visible in figure 11.

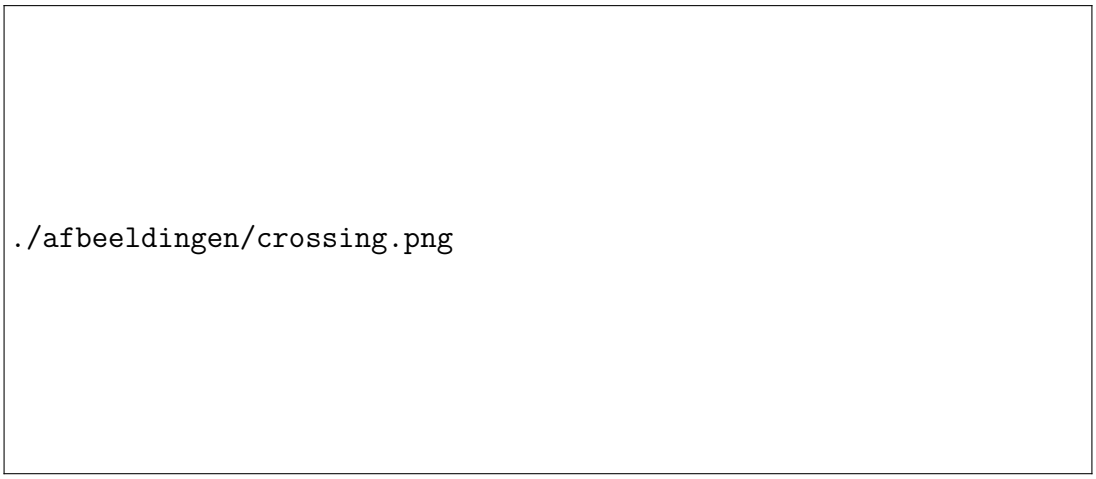

Figure 11: The crossing overlaps with the track, and the extension process will consider the possibility that they're railway pixels, until that is no longer a viable option.

Eventually, due to the limited resolution, the rails will eventually seem to "run out" in the distance, and only black pixels will be visible. At this moment, the algorithm stops iterating, and the resulting rails are evaluated:

[18] It might seem weird to perform centralisation when the algorithm already assumes it can start from the centre to look for the rails. It is still a necessary step, because this is part of the flexibility of the system: It doesn't require manually entering the position of the camera, it just needs to be placed "somewhere close to the middle" of the driver's compartment. Sometimes, this is not possible, for example in figure 5; the windscreen of the train is split by a vertical bar in the middle, so the camera couldn't physically be in the middle. That's why the "software centralisation" is necessary, in addition to the "hardware centralisation". A possible expansion of the system could include finding a way to get rid of this restriction altogether, but this is out of scope for this thesis.

[19] Although this technique is admittedly trivial and did not occur in any other paper discovered during the literature study, it is surprisingly effective while testing it on multiple video files. Also, this is just the preliminary phase where, again, the only goal is to obtain some normalisation parameters, and nothing else.

- If both rails extend beyond the middle of the screen, they are assumed to be the rails.
- If at least one of the rails is missing (or is not long enough), the algorithm repeats with the next frame in the buffer.

This ensures that the white pixels detected by the normalisation phase are the rail's pixels, and not artefacts from the track ballast.

The normalisation phase now has sufficient information to decide how the image should be cropped in order to put the rails in the centre.

4.2.2 Warp perspective (Homography)

Now that the rails have been found and the screen is centralised, all the required normalisation parameters are known. This information can now be used to create a homography matrix. From the camera's perspective, the rails seem to converge to a vanishing point, because they're parallel to each other in reality.

This is why the assumption of a straight track during the normalisation phase is necessary: If the tracks are straight, and the track is centered in the middle, then the vanishing point should also be in the middle of the screen. The difference between the expected vanishing point and the real vanishing point is enough information to warp the perspective. A representation of how that looks like internally can be seen in figures 12 and 13.

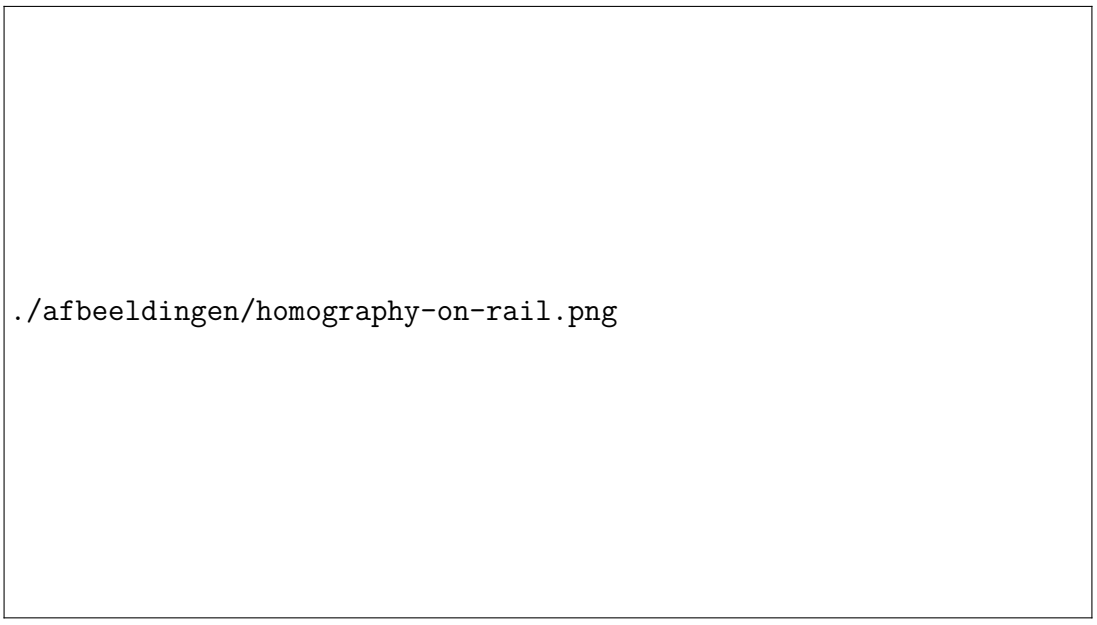

Figure 12: Illustration of warping the perspective using the rails. We know that rails always are the same distance from each other. Combined with the assumption that the railway is not curved, we can be sure that the blue trapezium is actually a projection of the red rectangle. This is enough information to calculate a homography matrix **H**.

Figure 13: Result of applying the homography matrix on the images from the camera. The fact that the rails are now the same distance from each other means distances are the same in both dimensions. Note the black triangles in the bottom corners; the viewpoint of the camera doesn't cover these areas, so no information is available there.

4.3 Detection technologies

As previously mentioned, detecting railroads is a problem that doesn't have a generally accepted solution. After the normalisation phase is complete and the homography matrix has been constructed, the actual rail detection can commence.

This gives way for a section in which four different techniques will be compared; how do they function, what do they require, what are the results?

4.3.1 Rail filling

Rail filling is the most trivial algorithm: Add adjacent rail pixels until no rail pixels can be detected. Then, extract a couple of key points for curve inter- and extrapolation, and draw the railway curve.

One might notice that this is the same technique used to centralise the frame. Indeed, in theory the same technique can be used for the rest of the program.

Evaluation RF provides very accurate results given its trivial implementation, and this accuracy increases with the quality of the camera; the higher the resolution, the further away rail pixels can be detected.

Additionally, RF has a very interesting property that the other algorithms don't: When close enough to a switch, RF can detect the position of that railway switch, and predict the direction the train will be going.

Unfortunately, a disqualifying property of RF for this phase is the long processing time for a frame, which also increases with the quality of the recording, because detecting the rail pixels hinges crucially on the previously detected rail pixels, so it cannot be accelerated using concurrent execution. The other techniques either don't have

to evaluate as many points, or use multithreaded algorithms. For the normalisation phase, this is not a problem, because time is not an important factor to take into accout there.

4.3.2 Bézier curve matcher

One technique with a limited amount of self-correcting properties is a simple Bézier curve matcher (BCM).

BCM exploits the property of railways often having wide curves, without sudden changes in the curvature. If the angle of the railway in the previous frame is given, it's only necessary to generate a handful of templates with only a slight change in curvature.

This technique is highly related to the rail detection algorithm proposed by Ukai, Nassu, Nagamine, et al., in which the authors also employ pre-made rail candidates. Some of the assumptions from that paper are also made in BCM, quote: [16]

> … the rail shape and position do not change radically between consecutive frames, and that curvature changes are smooth: for example, if the rails are making a curve to the right, they do not suddenly turn to the left [immediately after].

Generating the Bézier curves Bézier curves are easy ways to generate smooth curving lines, which serve as the template railways to match against.

The curves are first generated in the Cartesian plane, making sure that the eventual templates map closely on the rails in reality.

Since the two rails are always at the same distance from each other in the transformed frame, two Bézier curves with the same parameters are created, offset by a set amount of pixels from each other. These curves are then transformed using $\mathbf{H}^{-1}$ to the projective plane, at which point they're compared to the other templates.

In addition to continuing the previous curve, BCM generates two additional candidates, going to the left or to the right of the previous curve, with a predifined skew s, as shown in figure 14.

Scoring All generated templates are compared against a frame obtained after Canny edge detection [17], where the edges (and thus the rails) are marked in white pixels.

If a pixel of the template matches with a white pixel, the score is incremented. When the scores for all templates are available, the template with the highest score is chosen as the new rail curve.
To account for detection errors, the score of a generated candidate needs to outperform the current curvature by at least 5%. Without this, BCM is prone to flipflopping between two templates, since there's no deterrent system in place to mitigate that.

Evaluation This approach works well in general railway environments, with slight curves and a lot of straight trajectories.

Implementing the algorithm is a bit more involved than it is for rail filling; it requires working in two different spaces (Euclidean and projective), so the implementation of the algorithm is more complex to write than RF.

The algorithm is fast enough to operate in real time, taking only a couple of seconds per frame to complete.

Figure 14: Technical representation of the BCM technique. The previous best matching template is grey, with two adjacent candidates in red and blue.

Unfortunately, the algorithm begins to hamper when the train enters a non-general region, such as around depots or railways with reduced speed. In these areas, the curvature of the rails is often too large for the algorithm to properly cope with. Because of its assumption that the railway can be described using a cubic Bézier curve, BCM cannot correctly account for multiple different curves, for example in figure 15.

Although BCM works fine on slight curves, there's a couple of problems resulting from its simplicity:

- Sharp curves overstep the maximum difference between subsequent candidates, and since every generated candidate is equally worse, BCM will stop updating the track.
- Small errors in track detection can be corrected, but in practice, BCM is not able to rectify big detection mistakes.
- When two candidates have a very small difference in score, BCM starts "flip-flopping" between the two, instead of staying with the previous curve.
- Switches and crossings are not properly examined for their status. So it's possible that the train suddenly changes track, throwing BCM off route.

4.3.3 Point matching

Point matching (PM) is another algorithm that was evaluated for this project. Instead of working with a continuous line, PM limits itself to a predefined set of point couples.

Figure 15: Situation in which a curve to the left is immediately followed by a curve to the right. This type of curvature can't be properly represented using a Bézier curve defined by only three points.

Each couple can be moved independently from the other couples, and the points that make up a couple are each positioned on one of the rails.

The advantage of this compared to BCM is that the shape of the railway no longer has to resemble a three-point Bézier curve, but is instead able to take on a wide variety of shapes.

Scoring In practice, these points of the couples occupy a certain amount of pixels on the frame. The frame in question is a frame of the train driver's perspective with Canny edge detection applied to it.
Scoring thus happens by evaluating how many 'edge pixels' are covered by 'point pixels'. If more edge pixels overlap with the point pixels, then that means that particular point is a better match to the rail than another point. A visual example of how this works is given in figure 16.

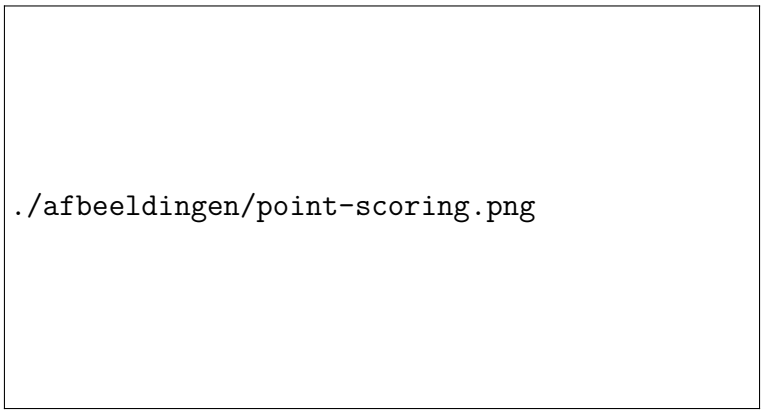

Figure 16: Example of scoring the different candidates. The vertical lines represent the detected rails, with the red and blue circles being candidate points. Both candidates overlap with the rail, but the blue candidate occupies more area, and thus more pixels, which gives it a higher score than the red candidate.

4.3.4 Rail contour matching

Rail contour matcher (RCM) is a technique that combines Canny edge detection [17] and contour matching[18].

When using Canny edge detection, the result is a black frame with the edges denoted by white pixels, but no way of extracting the edges themselves. Locating them manually would require iterating over the entire frame and marking the white pixels, which would make this only marginally better than RF.

Fortunately, the algorithm by Suzuki and Abe offers a highly performant way to extract contours from a frame. These contours are not just pixels on a frame, but are (iterable) lists of polygon coordinates. These allow filtering out irrelevant edges, determining the length of the edges that do matter, and connecting edges that are only disconnected by a single pixel, but clearly belong to the same edge. Figure 17 shows RCM in action.

Scoring There is no scoring system in place for RCM. Either RCM detects the rail in the current frame, or it doesn't. In the latter case, the detection from the previous frame is used; the implemented system saves the last correct detection, so it can be recalled when the current frame can't deliver the position of the rails.

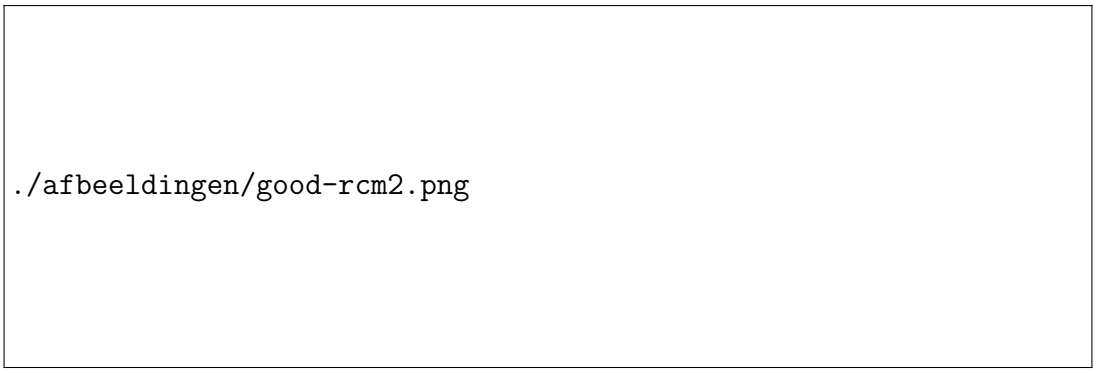

Figure 17: RCM detects the rails in front of the train. The colour of each rail indicates how it was detected. Green means detection happened without Any error. Yellow, orange and red each indicate a higher error level.

Error correction RCM also contains an error correction method. Oftentimes, only one of the two rails is detected.

In such a case, RCM will try to offset the starting point of the non-detected rail based on the starting point of the detected rail. Sometimes, this is enough for CM to detect the other rail as well.
If this is not the case, the detected rail is 'copied' to the non-detected rail, where it will act as a stand-in until the rails are detected in a future frame. Figure 18 shows an example of this.

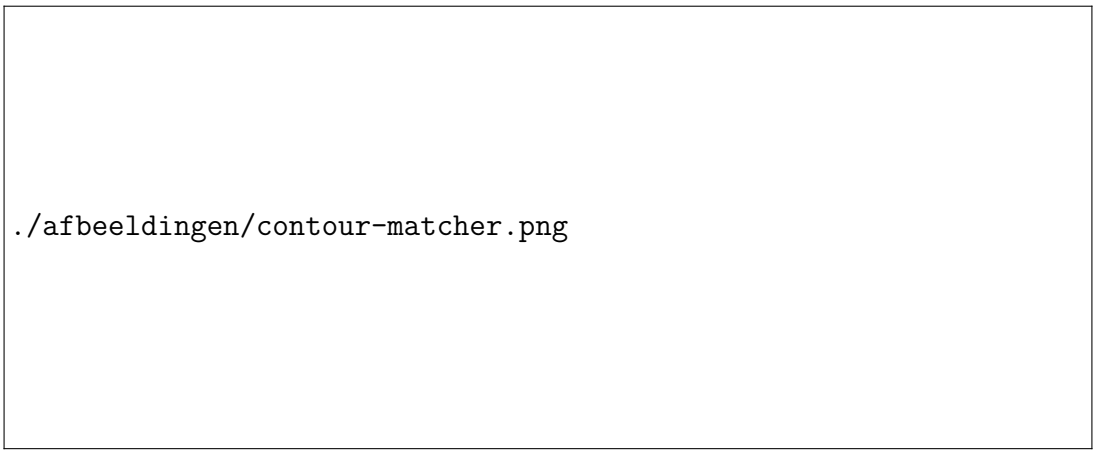

Figure 18: RCM with error correction. The colour does not indicate which rail is it, but how that rail could be detected. Green indicates The system could detect the rail without any problem. Orange Means the rail was not detected, but was estimated based on the green rail, using a homography matrix transformation and offset based of the width of the railway. It's clear this estimation is very accurate.

Evaluation The processing time is that of a non-optimised environment. That is, the entire frame is processed, instead of a clipped region. However, the system is more performant than the three other techniques as-is, because a lot of the complex code is handled by OpenCV itself, rather than being implemented in Python.

4.4 Conclusion

Rail contour matching is the best overall choice in practice for rail detection. It works fast, and the results are very accurate; in the recorded dataset, RCM could figure out the position of the rails in more than 90% of the cases,

when including the error corrections. The problems with faulty detections manifest in complex rail situations like switches, especially double switches; Canny edge detector has a lot of trouble with these rails, and this propagates to the rest of RCM. When improving RCM, priority should probably go to reducing the confusion with switches. The information offered by RCM will be used in the chapter about motion detection to extract the Region of interest (ROI), i.e. the region where we have to look out for track trespassers.

5 Image enhancement

The telescopic camera offers a view that's already beyond what the train driver could possibly see, thanks to its optical zoom. The rail detector from the previous chapter tells what part of the telescopic camera footage should be cropped out, the so-called ROI.

However, the cameras are statically mounted, and as such do not account for curves in the railway. If the curvature is too tight, the ROI that should be shown to the train driver will be outside of the telescopic camera's view.

One could opt for designing a moving platform for the telescope, but this is not a practical solution either, as the interior of the driver's cab can change, making it difficult to create a one-size-fits-all solution.

A more suitable option is crop the ROI from the normal camera instead and to try enhance that area whenever necessary, which is an easier and more practical solution, not in the least because in practice, this "image enhancement" is seldom necessary.

5.1 Testing approach

The frames in figure 19 from the recordings is used as a reference.

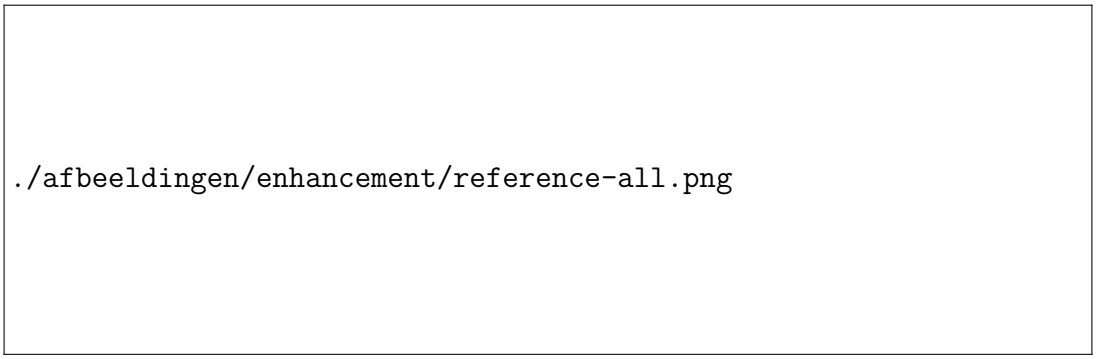

Figure 19: Pictures used for reference.

Although one could use an existing dataset like Div2K, the results for those already exist, and don't necessarily reflect their score "in the field"; that is, on railway frames. [20] The frame is downscaled [21] to 480×270 and 960×540, respectively four and two times as small. These dimensions are close to the dimension used for the telescopic camera, so operating on this size renders results that can be expected in a real deployment setting.

[20] Div2K is comprised of all kinds of different pictures (animals, plants, fields, buildings, ...).

[21] For downscaling, the pixel area relation is used, as is advised by the OpenCV documentation [19].

For each enhancement model, the following values are measured:

- Time to generate the enhanced frame
- Mean squared error (MSE)
- Peak signal to noise ratio (PSNR)
- Structural similarity image metric (SSIM)

The PSNR offers a well understood, objective measurement for the difference between images. However, PSNR is an based on MSE measurement, and these will give undue penalties on impairments (such as noise) that the average viewer wouldn't notice at all [20], as shown in figure 20. SSIM is a better measurement in that regard, so for completeness' sake, all of these metrics will be compared.

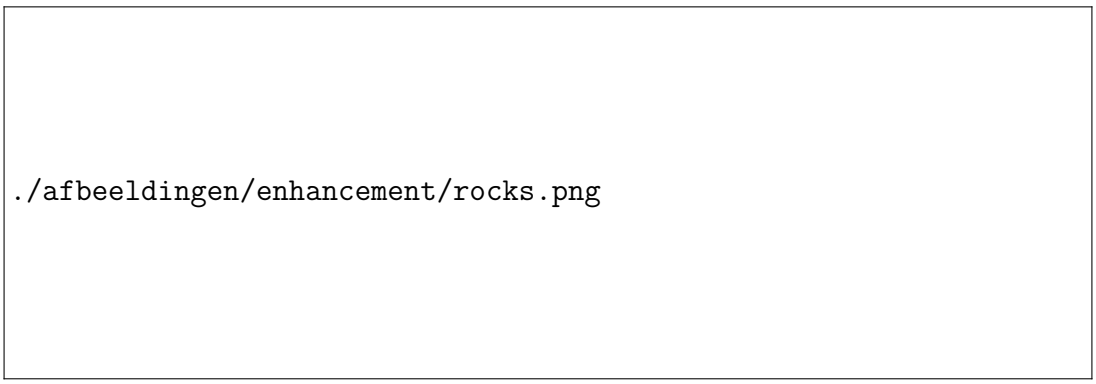

Figure 20: Example of how PSNR can give a lower score because of noise that a human wouldn't even notice. (Left picture is the original). The middle picutre received a noise mask on the rocks, the right picture received an equal amount on the grey area. Both pictures have a PSNR of 28dB, but every person would agree that the picture on the right is affected worse. SSIM would give a score of 0.95 and 0.81 on the middle and right picture respectively. [21]

5.1.1 Enhanced deep super-resolution network (EDSR)

EDSR was developed by Lim, Son, Kim, et al. as an improvement on existing SR models, and its superior performance earned the model a first prize in the NTIRE2017 Super-Resolution Challenge [23]. The authors found that applying batch normalization would remove the flexibility of the network, so omitting those resulted in a better performance.

These results can easily be reproduced with the reference image, with EDSR being the all-round winner with the quality metrics.

Unfortunately, to obtain its outstanding results, EDSR pays a heavy price when it comes to processing time, taking multiple minutes to complete its work, which disqualifies it for real-time applications.

5.1.2 Efficient sub-pixel convolutional neural network (ESPCN)

Shi, Caballero, Huszár, et al. developed a model with the explicit goal of providing a well-working SR network, while still being useful for real-time applications:

> One big advantage of our network is its speed. This makes it an ideal candidate for video SR which allows us to super-resolve the videos frame by frame.

This is visible in the results. Although the performance is only marginally better than LapSRN , it's an improvement nonetheless, while also having the fastest execution speed of all NN's. If the only choices were comprised of neural networks, ESPCN would be the best choice for enhancement.

5.1.3 Fast super-resolution convolutional neural network (FSRCNN)

Unfortunately, this model does not excel in any of the important metrics. The model, constructed by Dong, Loy, and Tang is more focused on improving the existing SRCNN[26]. While it's definitely a well working model, it's outperformed in time by ESPCN, which is practically just as good.

5.1.4 Laplacian pyramid super-resolution network (LapSRN)

In advance, LapSRN looked like a decent candidate for comparison, since Lai, Huang, Ahuja, et al. claimed speed was a differing factor of their model in [27]:

> Our LapSRN accommodates both fast processing speed and high capacity of deep networks. Experimental results demonstrate that our method is faster than several CNN-based super-resolution models, e.g. VDSR, DRCN, and DRRN.

While that may be true for those models, the claim doesn't hold up for 2 of the other NN's that were used in this comparison, and ESPCN is notably absent from the runtime/performance comparison chart in the accompanying paper.

The long runtime required for the marginal improvement on the used images does not make LapSRN a viable candidate for the goal at hand.

5.1.5 Nearest neighbour

Nearest neighbour is a very simple upscaling algorithm; as implied in the name, it interpolates the "new" pixels based on the value of the closest "old" pixel.

The simplicity makes it the fastest algorithm by a large margin, taking but a couple milliseconds to complete. Of course, the speed and simplicity also make it the worst in terms of image quality, resulting in blocking artefacts all over the frame. This is not very surprising, as explained in figure 21.

Figure 21: Visualization of why nearest neighbour results in blocking. The black dots represent the pixels of the original image. The pixels that lie in the square cell surrounding these dots will take the exact same value as the dot, resulting in blocking artefacts. [28]

5.1.6 Bicubic

This interpolation technique takes more pixels into account than just the nearest pixel, by looking at the evolution of adjacent pixel values in the rest of the image. The technique renders slightly better results than mere bilinear interpolation, with comparable performance.

Being only marginally slower than nearest neighbour, the bicubic model provides drastically better results. When ranking the models from worst to best, the time/quality ratio improvement is the highest between these two models. Still, this steep improvement is not enough to call it a good model, especially when compared to 5.1.7.

5.1.7 Lanczos

Just like bicubic and nearest neigbour, Lanczos does not use a neural network for its upsampling, but a kernel for interpolating the 'new' pixels. As shown in the results table, this still keeps it very fast compared to the neural networks.

Surprisingly, for the given example image, the SSIM score of Lanczos matched those of the neural networks. In that regard, that means that Lanczos offers the best quality-efficiency ratio of all the tested models.

5.2 Other techniques

There were two other promising SR techniques that didn't make it to comparison, but should be mentioned for completion.

5.2.1 Enhanced super-resolution generative adversarial networks (ESRGAN)

This model, proposed by Wang, Yu, Wu, et al., is one of the youngest models available for SR, and achieves very interesting results, at least on paper. However, there are some problems not shared by the other models:

- ESRGAN excels mostly when looking at 'thread-like' features of an image. In the images presented, the authors mostly put emphasis on animal fur, leaves, ... But these don't represent how the results are for

this particular application.

- The time required to finish is too long to be of practical use in real-time applications.
- The model was not published under a free license, and prohibits commercial usage of ESRGAN, so it could be argued it can't legally be used for the Belgian railways anyway.

5.2.2 Multi-frame super-resolution

This type of algorithms is unique compared to all the others, since it uses more than one frame to return an enhanced version. Also, when talking about Multiple-frame super-resolution (MFSR), there are also different kinds of SR, like spatial and temporal. For this project, the interest lies with spatial MFSR.

The main problem with using this technique is that it involves multiple complex steps that require solving, like image fusion and motion estimation, as was discussed by Luong. All these steps have indivdually been the subject of numerous research papers, each with their own set of applications in which they're best used.

Alas, since the availability of deep learning techniques, the state of the art for MFSR has started to shift from algorithmic approaches to using neural networks like HighRes-net [31]. Most of these do not have implementations available, and recreating them all from their papers (not to mention training them) would take up too much time for within the scope of this thesis.

5.3 Results

Two modes are compared, one where the image is doubled, and one where it is quadrupled, in both case across both image dimensions. The results are summarized in table 3 and 4. The best scores for each measurement are in bold. The model that would perform best in deployment is also bold, because it offers the best results for the least amount of time.

The results were obtained on a GNU/Linux computer system with an Intel i5-6200U (4 cores at 2.8GHz) processor.

Table 3: Comparison with 2× upscaling

Model	Required time (s)	MSE	PSNR (dB)	SSIM
EDSR	486.4067	**25.5789**	**34.0207**	**0.9615**
ESPCN	0.7349	34.6671	32.6645	0.9488
FSRCNN	0.8755	31.0755	33.1262	0.9536
LapSRN	6.9317	37.9758	32.2725	0.9457
Bicubic	0.0198	41.7501	31.9043	0.9447
Nearest neighbour	**0.0053**	91.0413	28.5286	0.9061
Lanczos	0.1181	35.8823	32.5594	0.9513

5.3.1 Conclusion

When only considering quality, it's not difficult to see that neural networks outperform the 'traditional' upscaling techniques. Especially EDSR scores very favourably.

Table 4: Comparison with 4× upscaling

Model	Required time (s)	MSE	PSNR (dB)	SSIM
EDSR	140.03269	**172.1500**	**25.7540**	**0.7526**
ESPCN	0.2448	200.5467	25.0830	0.7181
FSRCNN	0.3972	200.3464	25.0896	0.7163
LapSRN	9.09537	193.1753	25.2479	0.7235
Bicubic	0.0148	208.8358	24.9160	0.7257
Nearest neighbour	**0.0057**	270.4777	23.8019	0.7000
Lanczos	0.0987	205.8743	24.9765	0.7253

However, when time is also of the essence, Lanczos becomes the most interesting choice. When it comes to choosing a good model in artificial intelligence, it's good practice to choose a baseline model that renders good results without machine learning, in this case the Lanczos method. By applying Occam's Razor to the results, the model of choice for SR in this program is the Lanczos model.

The most probable reason for the barely noticeable improvement with NNs is that the neural networks were all trained for general-purpose applications, with general datasets. The problem with that, is that some features specific to railway environments could not be exploited. Of course, there is no dataset specifically for that, so manually training a network would not be possible either, so with neural networks, we're limited to these models anyway. It's very probable that, should such a dataset be available, the results would be a lot better than Lanczos.

In practice, when looking at the enhanced pictures, it should be remarked that the visual differences are very hard to spot. Of course, this is a personal observation, but in line with the objective metrics.

With all being said and done, it bears repeating that this usage of super-resolution techniques is not necessary in most operating environments. Places like the Noord-Zuidverbinding (with very tight corners) are most likely to trigger usage, but the risk of trespassers there is very low anyway. On the vast majority of the Belgian railway network, the curvatures are wide enough so as to not require post-capture enhancement.

6 Odometry

Odometry is the term used to describe the usage of data to determine the movement of an entity. Everyday examples of this are speedometers and GPS systems. Odometry has a number of useful applications. When it comes to improving railway safety, determining the stopping distance of a train could be a useful addition.

When camera images are used for determining movement, this process is called visual odometry. These techniques are useful when there are no other odometry specific sensors available. Since this project relies almost exclusively on video recordings, this will be the type of odometry that will be used.

6.1 Speed calculation

A train driver must always respect the speed limitations of the railway the train is currently driving on. This is easily seen on the driver's dashboard.

What's not so easy to determine is the distance it takes for the train driver to come to a standstill, should the convoy have to perform an emergency stop.

For this project, there's no connection between the train's instruments and the software itself. Again, the system proposed is supposed to be a base system on which additional software can be deployed, while being easily installed in a variety of different driver's compartments.

Although stereo cameras are often preferred for odometry, it's also possible with a single camera, as shown by Van Hamme et al.[32], but instead of that, a more simplified version was implemented. This was possible because the railway tracks make the environment a lot more constrained in terms of variability. We're also only interested in the displacement in one direction, not in other information like the rotation of the convoy.

Homograpy will be an essential tool in calculating the speed from the recorded video.

6.1.1 Optical flow

Optical flow is the term used for the apparent motion of objects in a video when either the camera or those objects move positions.

Essentially, if we want to calculate our speed, this means we will want to estimate the so-called egomotion in our video.

First, a starting frame is chosen, P_1. On this frame, we apply the Lucas-Kanade optical flow algorithm [33]. This algorithm tries to detect the movement of a set of points throughout subsequent frames.
To avoid tracking features that could be confused for adjacent features, only the features that are most likely to be tracked correctly are used, the other ones are removed.

From this reduced set of feature points, we further limit the set by excluding points that are not on the ground, i.e. we only use the points that are in the immediate vicinity of the front of the train.

Since a train can only move forward, all feature points will have progressed towards the camera. But since this happened in the projective space, their distances vary depending on the location of the feature.
That's where **H** comes in: We apply a linear transformation on the first and last frame where the feature $\overrightarrow{f_i}$ is present:

$$\lambda f_i' = \lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \mathbf{H} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

This results in two transformed frames where the displacement of the features can be correctly inferred (cfr. The homography matrix in practice); without applying the homography matrix, the feature displacements can't be translated to an actual distance. We now do have an actual distance, but in pixels, which can't be directly translated to e.g. metres.

However, when we combine these findings with the rail detector, we can make use of the distance between the rails in the real space, which is always 1.435m, and provides a good way to deduce how much pixels constitute one metre.

If the frame rate of the recording is also known, then we can directly calculate the displacement over time, i.e. the speed of the train.

Disappearing features When moving forwards, it's possible that feature points will disappear from the view of the camera.[22] This problem is easily solved by reversing the order in which the frames are presented to the odometry module. That way, all features move away from the camera, but with the same speed as they would come towards it. In effect, the displacement is the same, that's why the frames are analysed in reverse.

6.1.2 Results

Despite its apparent simplicity, this method is surprisingly accurate. Tests on the recordings show that the estimated speed differs in range of 5km/h from the expected actual speed. However, note that there is no ground truth available; no other sensors have been used to collect additional information. The "actual" speed of the train is based on an informed guess; train drivers try to drive as fast as they're allowed to, so if the dataset shows that the train has passed a railway sign with a 50km/h speed limit, it can be assumed that the actual speed of the train will almost be that particular speed. Such an example is shown in figure 22.

With very high speeds, the feature detection has some problems with detecting enough points on the ground to make an accurate speed calculation, mostly due to motion blur. In such a case, the system will simply keep its previously recorded speed.

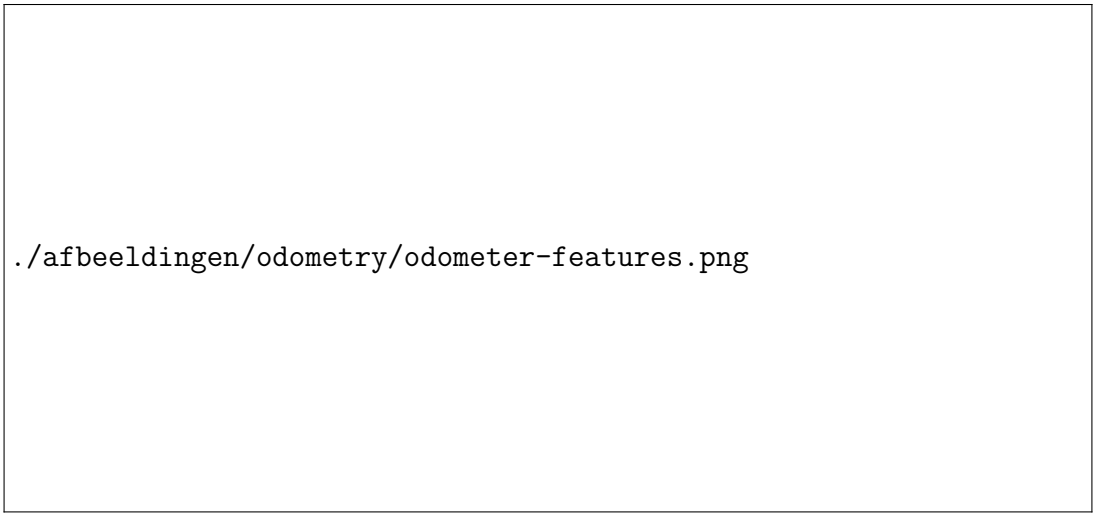

Figure 22: Features detected by Lucas-Kanade's optical flow algorithm. On the left is the first frame where all well trackable points are found. After five frames, their displacement is visible in the right frame. Notice why it's important to limit the displacement calculation to features that are on the planar object used for the homography; most other features have little to no displacement, which would have a distorting effect on averaging out all the displacements. For this particular example, the odometer reports an estimated speed of 47.02km/h, in an area with a speed limit of 50km/h.

[22] Even with reversing the order of the frames, it's possible that some features don't get tracked correctly. This causes a skew in the resulting estimated speed, but this is why an average is taken across all features, which minimizes the effect of these errors.

6.2 Stopping distance

Having calculated the speed within a practical margin of error, an extra application of the odometry is displaying the stopping distance of the train.

The stopping distance can be calculated by the following formula:

$$d_{stop} = v_0 \cdot t_r + \frac{v_0^2}{2 \cdot a_{brake}}$$

With

d_{stop} The stopping distance in metres

v_0 The velocity of the vehicle in metres per second

t_r The reaction time of the driver in seconds

a_{brake} The braking coefficient of the vehicle

For trains, the braking coefficient is set to 1[23], because of the limited friction between the wheels and rails. The low denominator explains the long stopping distance for trains.

This stopping distance calculation is also available in the program, and shown on the display for the train driver. However, displaying the position on the rail where the train will eventually come to a full stop when performing an emergency brake requires some extra programming in order to be properly displayed as well.

7 Motion detection

In the introduction, the importance of railway safety was discussed, while putting emphasis on the cost of track trespassers. Timely detection of these people is key for ensuring accidents are limited to a minimum.

In the previous chapters, the fundamental problems were discussed, along with possible solutions. In its current state, the software is able to automatically determine where the train will move to, and can extract that region from the telescopic camera, so the train driver can see what's happening there.

In this chapter, that extracted region will be a crucial element for motion detection. The motion detection algorithm will be based on background subtraction, which gives way to a couple of interesting advantages, but also some complications specific to working with a telescopic camera in a railway environment.

7.1 Advantages

7.1.1 Trivial algorithms could suffice

The work of my supervisor and promotors shows that accurate and fast motion detection on short distances is a very complicated problem to get right [1]. This is necessary, because for a computer, everything at close distance seems to be in motion, trees and pedestrians alike, because of the parallax effect.

[23] To give an idea of how low this braking coefficient is: For a well-maintained car equipped with ABS, the coefficient is set to 10.

When working on long distances however, this is not nearly as big a problem, since everything that's stationary also appears to be stationary; the parallax effect becomes negligible. This means that objects that are moving fast enough also stand out from their surroundings, making it possible to use relatively simple motion detection algorithms.

7.1.2 Region of interest

Applying a motion detection algorithm on the complete frame can be a time-consuming task, and will definitely result in unnecessary detections. Using the rail detector, it's possible to extract the exact region we're interested in, making the process a lot faster.

7.2 Complications

7.2.1 Faulty rail detection

The rail detector proposed in chapter 4 (RCM) performs very well, and with additional work, a lot of the teething troubles can also be mitigated. A perfect solution however is not practically feasible, so the motion detector ought to compensate for erroneous detections.

7.2.2 Vibration

The motion detector will operate on footage from the telescopic camera. While vibrations from a normal camera don't pose a real problem, vibrations are excacerbated by a linear factor relative to the distance. Should the proposed setup be used in practice, additional dampening techniques (read: external hardware for the camera itself) could be used, if the extra cost and material are acceptable. Whatever the case, for the set-up used for this project, vibration could still be a problem.

7.3 Implementation

The motion detector (MD) runs in parallel with the rail detector, which returns the coordinates of the ROI to be extracted from the telescopic footage. An update to MD is triggered whenever a new telescopic frame is available. Every new telescopic frame $\mathbf{f}_i$, MD stores the clipped frame $\mathbf{r}_i$, together with its respective coordinate $\overrightarrow{c_{\mathbf{r}_i}} = (x_{\mathbf{r}_i}, y_{\mathbf{r}_i})$. This coordinate defines the upper left corner of $\mathbf{r}_i$. The lower right corner is offset by a constant vector, so the rectangle can be defined by only one coordinate.

A faulty rail detection would also return a faulty coordinate, which will most likely be suspiciously far away from the previous coordinate. To account for that, every subsequent ROI will only be displaced by a maximum of 2 pixels in each dimension, regardsless of how far the actual displacement between $c_{\mathbf{r}_i}$ and $c_{\mathbf{r}_i}$ actually is:

$$\|\overrightarrow{c_{\mathbf{r}_i}} - \overrightarrow{c_{\mathbf{r}_{i+1}}}\| \leq \sqrt{8}$$

The collection of subsequent frames continues until n[24] frames are available.

7.3.1 Background estimation

Since it's possible that $\exists i \neq j : \overrightarrow{c_{\mathbf{r}_i}} \neq \overrightarrow{c_{\mathbf{r}_j}}$, the different ROIs might not correctly overlap with each other. This will make background estimation a lot less accurate, which will already be influenced by the (albeit very small) egomotion of the camera. To compensate that, all clipped frames are clipped once more, this time to extract the region covered by all frames, visually explained in figure 23.

Figure 23: Example of extracting the common region for $n = 3$ frames. The stored coordinates are not equal to each other, so their corresponding ROIs all cover a slightly different region. The commonly covered region (marked by the dashed line) is the area to which all frames are clipped, after which their coordinates do overlap exactly.

At this point, the background is estimated over all frames by using the median value over greyscale pixels, resulting in a "baseline frame" that represents the background shared by all the frames. If a pixel value is sufficiently close to that of the baseline frame, it can be assumed it's not motion.

Suspected motion is being extracted by using a thresholding function, and accumulated in a motion frame. Applying this function over the n frames returns a heatmap of change in the frames, as exemplified in figure 24. Compared to just presenting the difference per frame (which would give results that are very hard to notice), this clearly shows the movement over time, so the train driver can make an informed decision on how to react.

After this first motion detection, the oldest clipped frame $\mathbf{r}_0$ is removed from the list, and the newest frame is added to the front, essentially being a First in, first out (FIFO) stack.

[24] $n = 16$ for the implementation.

Figure 24: Example of the heatmap after motion detection. Even though the telescope has a forward-moving egomotion, only the passengers on the platform are marked on the heatmap as a suspicious movement. The system is not flawless; note the tiny faulty detection on the left rail and platform.

Taking the results of the motion detector into account, it's worth noting that the quality of this footage wasn't very good to begin with, as was pointed out when discussing the recording results of the telescopic camera. It's possible that an improved setup could render even better results than can be provided at the time of writing.

7.4 Problems with motion detection

The remarkable efficacy (relative to the complexity of the implementation) doesn't mean that there aren't any noteworthy problems. A visual summary of all the problems can be seen in figure 25. During curves, objects that are very close leave long "stripes" of horizontal erroneous markings. Vibrations cause vertical "motion" markings. Misclassifications seem to happen a lot with strong colour differences, so railway signs often get marked as well. In truth, these are fairly minor, and the train driver is supposed to notice these signs anyway.

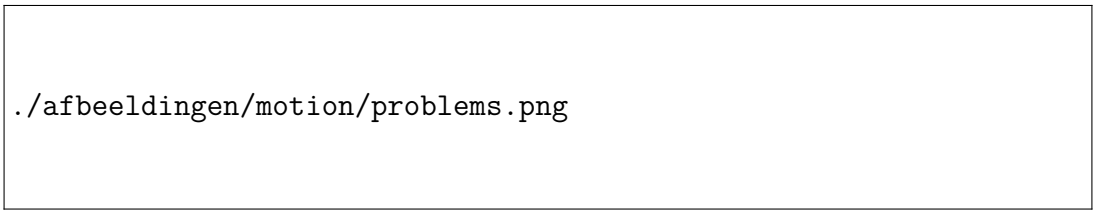

Figure 25: Faulty detections due to curves, vibration and misclassification respectively.

In spite of these problems, there is an upside: The directions of the striping of curves and vibration are not random, but a symptom of their causes. Further improvement could thus take the form of trying to recognise the shapes present in the heatmap. A neural network could be useful here: By learning to detect the shape of the blobs, it could filter these out, leaving only the actually interesting motion blobs for the train driver.

7.4.1 Curves

The sharper a curve, the more faulty motion detection will be displayed. This is partly mitigated by the ROI moving with the curve a while before entering it, but in sharp curvers, even this is not enough, and the MD is not very useful there. Luckily, the sharp curves only occur in areas where the train isn't driving very fast anyway.

7.4.2 Vibration

Should this setup be applied on a road vehicle, the vibrations could definitely have caused a lot of issues; Belgian roads are notoriously bumpy. Luckily, this doesn't seem to be a very big issue on railroads. A trivial reason for this is the rails themselves; all trains travel on the exact same position, and the degradation to the rails could be assumed perfectly random for this purpose. As such, the displacement over the route happens without many noticeable bumps. The vibrations that do happen on normal tracks could be attributed to breathing switches and fishplates, but this problem is a lot less noticeable than it could have been thanks to the usage of jointless track where it's possible, as is clearly visible in figure 26.

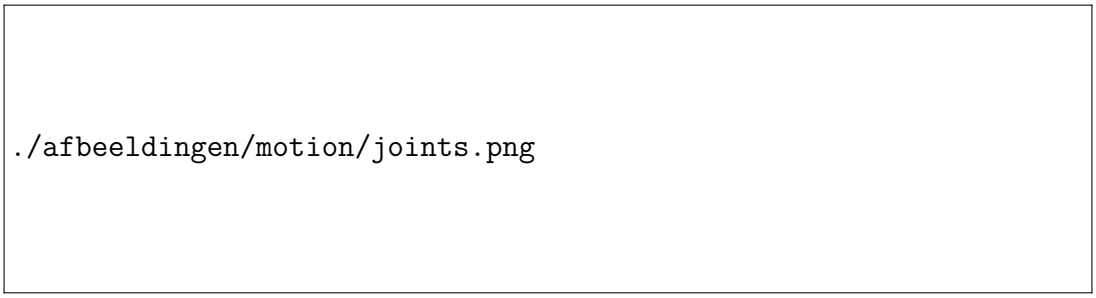

Figure 26: From left to right: A breather switch [34], a fishplate [35], and jointless track [36]. Even though these dents are extremely small, the sheer weight and speed of the trains going over them make them very noticeable and maitenance heavy. Removing the joints solves both these issues, but unfortunately, this technique can't be applied everywhere.

The problem of vibration does become very clear when the train passes switches; by design, these parts of the route feature a large 'dent' in an otherwise very flat rail, as illustrated in figure 27. This is also visible in the recordings themselves.

8 Conclusion and future possibilities

The previous chapters discussed the development of an efficient safety system, but there is still a lot of room left for improvements and possible extensions.

That's not to say that the system as-is doesn't show a lot of potential. Given the limited timespan, the lack of existing research papers, the simplicity of the used set-up, and the fact that the telescopic camera was not properly configured, the software that's been programmed works fast enough while delivering notably good results.

The main conclusion that I'd make from this thesis is that the thing it needs the most is **time**. Time to work out the details, to increase the accuracy of the rail and motion detectors, to record more footage in other weather

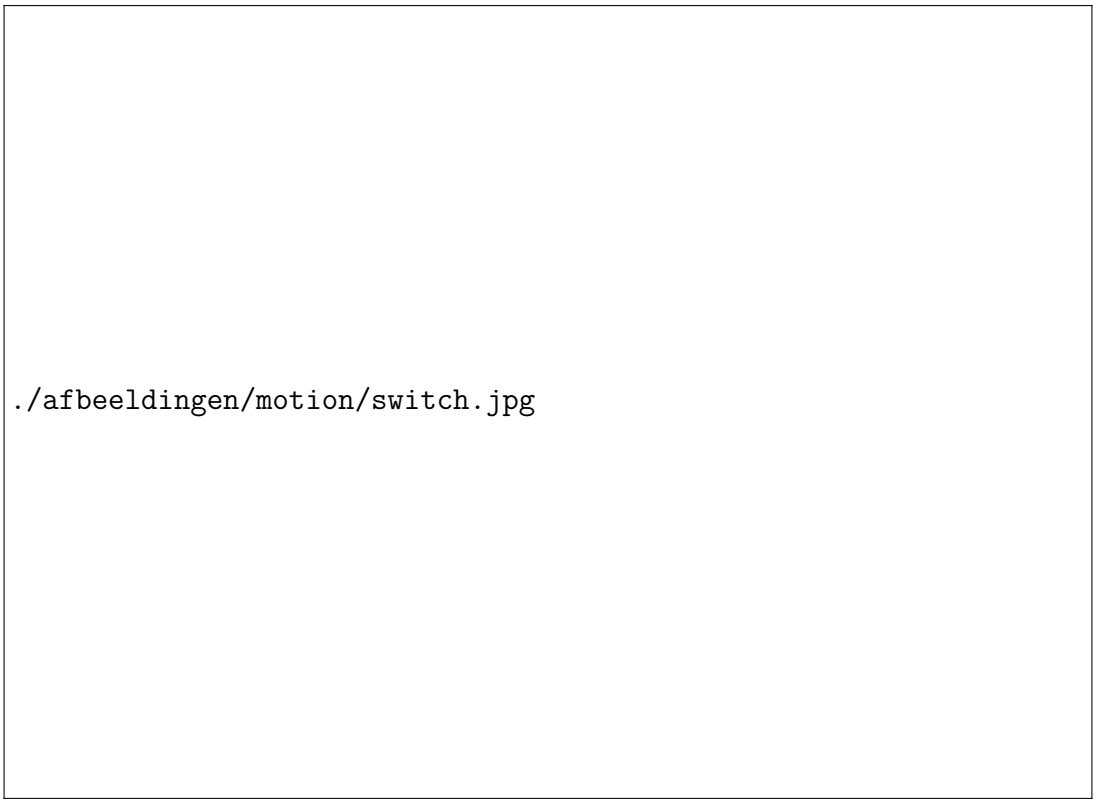

Figure 27: Picture of a switch with the so-called 'frog' on the left and 'guard rail' on the right.

environments, ... Surely, not every research track for this subject will result in success, and even while writing this thesis, not all ideas I had have been winners. Still, I feel this thesis is just scratching the surface of what's possible when trains, artifical intelligence, and image processing meet each other.

To conclude this thesis, this final chapter will take a more detailed look at some of the possibilities below that surface.

8.1 Line study

A train driver degree does not grant its owner access to all Belgian railways Said person must always do a so-called "line study" before he's granted access on that particular line. On this line study, the conductor learns about all the different elements: How the signals behave, where the emergency phones are placed, where to expect switches, ... An example of a line study card is included in appendix B.

All this serves as an extra layer of knowledge about what to expect. Technically, it's also possible to enter this information manually in the software[25], which could make the signal detector a lot more efficient: If it knows the next signal is going to limit the speed, it's not necessary to be on the lookout for it.

Although these documents are available as PDF files, interpreting them would require a completely seperate program, which is less interesting with relation to the scope of this thesis.

[25]The line study fiches all follow a common pattern, so while manually entering them is possible, it's perhaps easier to ask NMBS/S-NCB/NGBE for the digital files themselves, so they can automatically be read by a third program.

8.2 Different testing environments

This project was conducted in collaboration with the Belgian railway companies during the COVID-19 pandemic. As such, it was not feasible to obtain recordings for a lot of different environments, but it's not uncommon to think that a lot of factors could influence the working performance of the software. A couple of parameters that were not taken into account:

Camera types The recordings were created with professional equipment. It would be interesting to see how versatile the software operates with different types of equipment.

Positioning The software currently requires that the cameras be placed somewhere near the middle of the front of the train, but an exact limit to this has not been defined. Knowing which positions in the train can be used for the camera would be valuable information for practical deployment. so testing some promising candidates

Environment Different weather and time conditions could have a severe impact on the recordings and how they are processed.

Testing all possible combinations of these parameters would be too time-consuming, and wouldn't render enough unique information. But if the research of this thesis were to continue, accounting for these differences would definitely become worth the time.

8.3 Neural networks

A lot of papers that discuss motion detection utilize a deep learning approach, and the results are often impressive: The KITTI Vision Benchmark Suite for stereo images [26] shows that almost all the best performing algorithms use a neural network in a primary role (or more specifically, CNNs). One could argue that neural networks are the best choice when doing motion detection, so it may seem weird that this thesis doesn't even discuss it, let alone use it.

However, the decision to forego neural networks for motion detection, as well as rail detection was very much intentional. The main reason being that a neural network essentially functions as a black box; one can't deduce how the individual neurons learn, what neuron is perceptible to what element, ...

Also, using neural networks doesn't excuse one from knowing and understanding the data you're using. There aren't any datasets available for this specific subject, and using "related" datasets will probably only render poor results, because there's no reason to assume that transfer learning[27] can be applied with this problem.

Tweaking a neural network is also limited to "superficial" parameters, like batch size, amount of perceptrons, activation functions, ... which do not map to any problem in specific.[28] It would make it impossible to know why

[26] `https://www.cvlibs.net/datasets/kitti/eval_scene_flow.php?benchmark=stereo`

[27] This is a topic in machine learning where the knowledge learned by training on a particular problem is being transferred to a related problem. Research into what's possible with transfer learning is very much a work in progress, and there are no strict rules known about what is possible and what isn't.

[28] This is one of the reasons why neural networks are used so much in recent years: They don't focus on a specific type of problem, but can be applied to a plethora of different problems. But that also means that the hyperparameters that one can configure are not focused on specific problems either.

something is detected and something else isn't.

In my opinion, I think a good thesis needs more than just taking an existing NN and turning the knobs on it until the results are satisfactory.

I doubt that it would result in an interesting dissertation,[29] when there is so much room for improvement when one takes time to think about how to solve the problem "algorithmically".

Even if we assume that the best setting could be found in a short time span, there is still the problem of required computation time. Complex NNs would only become feasible after performing the operation

on a Graphical processing unit (GPU), but for real-time applications, one would need a state-of-the-art (read: expensive) GPU, and one goal of the system is to be both flexible and affordable.

The issue of time also played a role in foregoing . If time is limited, one always has to make a trade-off between trying to achieve the theoretically most optimal result, and a suboptimal solution that's practically useful, but leaves room for improvement.

groundbreaking results, so trying to improve (or match) the record holders isn't a realistic goal. Instead, the goal was to create a **practical** solution: One that is understandable to use, works fast on regular hardware, is cheap enough for the NMBS/SNCB/NGBE, and reaches acceptable results. The motion detector system from chapter 7 seems to deliver on that goal.

However, further improvements do not need to refrain dogmatically from using NNs. Rather, I think the best application for NNs in this project is in conjunction with an algorithmic solution like the one currently used for motion detection. As mentioned in that chapter, there are some faulty rail detections that can't be measured directly, but do have some noticeable properties (for example, the horizontal "stripes" in curves). Because most of the motion analysis has already been done using a simple and efficient algorithm, a tiny CNN that could learn which motion is real and which isn't could be a valuable addition to the motion detector.

8.4 Code improvement

Because of time constraints, the bulk of the software for this thesis was written in Python, which is a great language for quick imperative programming. However, it's not the best choice when computational time is of the essence. Luckily, library bindings like NumPy and OpenCV can mitigate that for the most computationally expensive parts, but some components that were manually written do slow down the pipeline because of its sequential nature. In effect, the system is only as fast as its slowest module.

Since the main purpose was to show that the proposed detection pipeline is feasible, not all possible optimizations are included in the code. The feasability-oriented focus meant that proper coding standards became a second priority. A lot of operations can be performed concurrently, or even on specialized hardware, which would greatly increase the processing speed.

Given more time, rewriting the code in a more appropriate language could definitely solve some of these problems. Perhaps only the slowest parts need to be rewritten in a more low-level language, which would require less time, while also maintaining the advantages of Python.

[29] This approach of taking an existing neural network and trying to make it a little bit better has caused a policy update for submissions on the KITTI Vision Suite: Only submissions with substantial improvement and academic merit are accepted.

8.5 Multi-frame SR

This thesis explored the use of multi-frame super-resolution, but only briefly due to the lack of time and available implementations. However, the kind of video that was recorded is a prime candidate for this kind of super-resolution.

If the results from the deep learning methods for image enhancement are any indication for speed, then the current state of the art is probably not a good solution.

8.6 Railway signal interpretation

Interpreting railway signals would be a possible expansion on this project, and a necessity when considering Automatic train operation (ATO).

8.6.1 Driving regimes

Belgian trains, when driving in so-called normal regime, drive on the left. This offers an interesting assumption, because it implies that the part of the image that needs to be examined, also needs to be extended to the right in order to cover both railway tracks.

However, this is not always the case. It's also possible that the train will have to drive on the 'opposing' track. In such a case, the previous assumption no longer holds: The frame now needs to be extended to the left.

Of course, driving on the "wrong" track is not a common situation for trivial reasons: Oncoming trains could frontally hit the train, with devastating results. However, it could also be necessary, for example to bypass a defective convoy.

Train conductors are able to differentiate between a miszending and a 'safe' regime change by the so-called chevron on rail signals (see figure 28). When a chevron is visible on a signal, the conductor knows a change of regime is imminent, and that the convoy will soon be driving on the opposing side. Should this happen without the chevron, the conductor must intiate an emergency stop.

Interpreting the presence of the chevron allows the detector to know in what direction the frame should be extended, making detecting (and interpreting) railway signals also useful for improving the obstacle detector.

Of course, a regime change also implies that the next switch will most likely be diverging to the other side of the track.

8.7 Automatic train operation

Multiple countries in Europe have been experimenting with autonomous train operation, because of the increased safety being offered by technological advancements. One notable example is Transport for London's Docklands Light Railway, where all rolling stock utilize ATO. The DLR has had no major accidents caused by proper automatic operation so far.

A detection algorithm as featured in this thesis would be an essential part of such a system, should it be used on the Belgian railway network. And this is not necessarily a hypothetical future: The new M7 rolling stock ordered

Figure 28: Example of railway sign with chevron, indicating a change in regime. Since this is taken on the right track, the train will switch over to the left track. The next switch will thus be in the diverging position. This information could be exploited in future work. [37]

by NMBS/SNCB/NGBE are already equipped with cameras in the control trailer, although it's currently meant for safety purposes during shunting.

References

[1] P. Veelaert, D. Van Hamme, and G. Allebosch, "Motion segmentation in video from non-stationary cameras," English, WO/2019/229075 (PCT filing), 2019. [Online]. Available: https://patentscope.wipo.int/search/en/detail.jsf?docId=WO2019229075.

[2] Infrabel. "Jaarlijks veiligheidsverslag 2020." (2020), [Online]. Available: https://infrabel.be/sites/default/files/generated/files/report/RAS%5C%202020_NL%5C%20I-SCPA.pdf.

[3] I. Electro-Optics. "Nir (near-infrared) imaging cameras." (), [Online]. Available: https://www.infinitioptics.com/technology/nir-near-infrared.

[4] J. Ma, C. Chen, C. Li, and J. Huang, "Infrared and visible image fusion via gradient transfer and total variation minimization," Information Fusion, vol. 31, pp. 100–109, Sep. 2016. DOI: 10.1016/j.inffus.2016.02.001.

[5] X. Zhang, P. Ye, and G. Xiao, "VIFB: A visible and infrared image fusion benchmark," CoRR, vol. abs/2002.03322, 2020. arXiv: 2002.03322. [Online]. Available: https://arxiv.org/abs/2002.03322.

[6] S. Yanan, Z. Hui, L. Li, and Z. Hang, "Rail surface defect detection method based on yolov3 deep learning networks," in 2018 Chinese Automation Congress (CAC), 2018, pp. 1563–1568. DOI: 10.1109/CAC.2018.8623082.

[7] Z. Liang, H. Zhang, L. Liu, Z. He, and K. Zheng, "Defect detection of rail surface with deep convolutional neural networks," 2018 13th World Congress on Intelligent Control and Automation (WCICA), pp. 1317–1322, 2018.

[8] Z. Chen, Q. Wang, K. Yang, T. Yu, J. Yao, Y. Liu, P. Wang, and Q. He, "Deep learning for the detection and recognition of rail defects in ultrasound b-scan images," Transportation Research Record, vol. 0, no. 0, p. 03 611 981 211 021 547, 0. DOI: 10.1177/03611981211021547. eprint: https://doi.org/10.1177/03611981211021547. [Online]. Available: https://doi.org/10.1177/03611981211021547.

[9] 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.

[10] M. Karakose, O. Yaman, M. Baygin, K. Murat, and E. Akin, "A new computer vision based method for rail track detection and fault diagnosis in railways," International Journal of Mechanical Engineering and Robotics Research, vol. 6, pp. 22–27, Jan. 2017. DOI: 10.18178/ijmerr.6.1.22-27.

[11] Z. Zhang, S. Yu, S. Yang, Y. Zhou, and B. Zhao, "Rail-5k: A real-world dataset for rail surface defects detection," CoRR, vol. abs/2106.14366, 2021. arXiv: 2106.14366. [Online]. Available: https://arxiv.org/abs/2106.14366.

[12] S. Sahebdivani, H. Arefi, and M. Maboudi, "Rail track detection and projection-based 3d modeling from uav point cloud," Sensors, vol. 20, no. 18, 2020, ISSN: 1424-8220. DOI: 10.3390/s20185220. [Online]. Available: https://www.mdpi.com/1424-8220/20/18/5220.

[13] M. Adham and A. El-Shazly, "Railway tracks detection of railways based on computer vision technique and gnss data," Nov. 2020. DOI: 10.11159/iccste20.269.

[14] Dmcq, Spherical coordinate system, based on file:3d spherical but with reversed notation for angles as often used in maths, © CC-BY-SA 3.0, Apr. 2012. [Online]. Available: https://commons.wikimedia.org/wiki/File:3D_Spherical_2.svg.

[15] P. Veelaert, Multi-view geometry: Part 1, Sep. 2019.

[16] M. Ukai, B. Nassu, N. Nagamine, M. Watanabe, and T. Inaba, "Obstacle detection on railway track by fusing radar and image sensor," 2011.

[17] J. Canny, "A computational approach to edge detection," IEEE Trans. Pattern Anal. Mach. Intell., vol. 8, no. 6, pp. 679–698, Jun. 1986, ISSN: 0162-8828. DOI: 10.1109/TPAMI.1986.4767851. [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.420.3300&rep=rep1&type=pdf.

[18] S. Suzuki and K. Abe, "Topological structural analysis of digitized binary images by border following," Comput. Vis. Graph. Image Process., vol. 30, pp. 32–46, 1985.

[19] O. developers. "Geometric image transformations." (2021), [Online]. Available: https://docs.opencv.org/master/da/d54/group__imgproc__transform.html.

[20] Z. Wang and A. C. Bovik, "Mean squared error: Love it or leave it? a new look at signal fidelity measures," IEEE Signal Processing Magazine, vol. 26, no. 1, pp. 98–117, 2009. DOI: 10.1109/MSP.2008.930649.

[21] D. Bull, Communicating pictures. 2014.

[22] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," CoRR, vol. abs/1707.02921, 2017. arXiv: 1707.02921. [Online]. Available: http://arxiv.org/abs/1707.02921.

[23] R. Timofte, E. Agustsson, L. V. Gool, M.-H. Yang, L. Zhang, B. Lim, S. Son, H. Kim, S. Nah, K. M. Lee, X. Wang, Y. Tian, K. Yu, Y. Zhang, S. Wu, C. Dong, L. Lin, Y. Qiao, C. C. Loy, W. Bae, J. Yoo, Y. Han, J. C. Ye, J.-S. Choi, M. Kim, Y. Fan, J. Yu, W. Han, D. Liu, H. Yu, Z. Wang, H. Shi, X. Wang, T. S. Huang, Y. Chen, K. Zhang, W. Zuo, Z. Tang, L. Luo, S. Li, M. Fu, L. Cao, W. Heng, G. Bui, T. Le, Y. Duan, D. Tao, R. Wang, X. Lin, J. Pang, J. Xu, Y. Zhao, X. Xu, J. Pan, D. Sun, Y. Zhang, X. Song, Y. Dai, X. Qin, X.-P. Huynh, T. Guo, H. S. Mousavi, T. H. Vu, V. Monga, C. Cruz, K. Egiazarian, V. Katkovnik, R. Mehta, A. K. Jain, A. Agarwalla, C. V. S. Praveen, R. Zhou, H. Wen, C. Zhu, Z. Xia, Z. Wang, and Q. Guo, "Ntire 2017 challenge on single image super-resolution: Methods and results," in 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2017, pp. 1110–1121. DOI: 10.1109/CVPRW.2017.149. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2017_workshops/w12/papers/Timofte_NTIRE_2017_Challenge_CVPR_2017_paper.pdf.

[24] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," CoRR, vol. abs/1609.05158, 2016. arXiv: 1609.05158. [Online]. Available: http://arxiv.org/abs/1609.05158.

[25] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," CoRR, vol. abs/1608.00367, 2016. arXiv: 1608.00367. [Online]. Available: http://arxiv.org/abs/1608.00367.

[26] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," CoRR, vol. abs/1501.00092, 2015. arXiv: 1501.00092. [Online]. Available: http://arxiv.org/abs/1501.00092.

[27] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Fast and accurate image super-resolution with deep laplacian pyramid networks," CoRR, vol. abs/1710.01992, 2017. arXiv: 1710.01992. [Online]. Available: http://arxiv.org/abs/1710.01992.

[28] Zykure, Illustration of nearest neighbor interpolation on a random dataset, © CC-BY-SA 4.0, 2016. [Online]. Available: https://commons.wikimedia.org/wiki/File:Interpolation-nearest.svg.

[29] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, C. C. Loy, Y. Qiao, and X. Tang, "ESRGAN: enhanced super-resolution generative adversarial networks," CoRR, vol. abs/1809.00219, 2018. arXiv: 1809.00219. [Online]. Available: http://arxiv.org/abs/1809.00219.

[30] H. Luong, "Advanced image and video resolution enhancement techniques.," 2009.

[31] M. Deudon, A. Kalaitzis, I. Goytom, M. R. Arefin, Z. Lin, K. Sankaran, V. Michalski, S. E. Kahou, J. Cornebise, and Y. Bengio, "Highres-net: Recursive fusion for multi-frame super-resolution of satellite imagery," CoRR, vol. abs/2002.06460, 2020. arXiv: 2002.06460. [Online]. Available: https://arxiv.org/abs/2002.06460.

[32] Van Hamme, David and Veelaert, Peter and Philips, Wilfried, "Robust monocular visual odometry by uncertainty voting," eng, in IEEE Intelligent Vehicles Symposium, Baden-Baden, Germany: IEEE, 2011, 643–647, ISBN: 9781457708916. [Online]. Available: %7Bhttp://dx.doi.org/10.1109/IVS.2011.5940453%7D.

[33] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2, ser. IJCAI'81, Vancouver, BC, Canada: Morgan Kaufmann Publishers Inc., 1981, pp. 674–679.

[34] Rwendland, Railway track expansion joint in continuous welded rail within hayle railway station, near to the start of the hayle viaduct, on the track to penzance, © CC-BY-SA 3.0, 2011. [Online]. Available: https://commons.wikimedia.org/wiki/File:Expansion_joint,_Hayle.jpg.

[35] L. Chatfield, A portrait of a fishplate. on the bluebell railway, © CC-BY 2.0, 2006. [Online]. Available: https://commons.wikimedia.org/wiki/File:Fishplate_UK_2006.jpg.

[36] Bas, A welding in the rails between groningen and delfzijl in the netherlands, © CC-BY-SA 3.0, 2012. [Online]. Available: https://commons.wikimedia.org/wiki/File:Rail_welding.JPG.

[37] F. Melchior, Belgian railway signal with crocodille cab signalling in the track, © CC-BY-SA 3.0, 2007. [Online]. Available: https://commons.wikimedia.org/wiki/File:CX-D.45.jpg.

Glossary

B | C | F | G | I | J | L | M | P | R | S | T | U | Y

B

breathing switch (Nederlands: Uitzettingsvoeg) An intentional gap between adjacent rails, to account for the expansion of the rails with changing temperatures, which would otherwise kink under the extreme force. 47

C

CNN Convolutional neural network 22, 49, 50

convoy ("Konvooi" or "Convoi" in Dutch or French resp.) Refers to what people describe as 'the entire train': locomotive and carriages. 51

F

FIFO First in, first out 45

film speed Not to be confused with frame rate, the film speed is a measure that describes how sensitive a camera is to light. In practice, this is known as ISO settings, which can be configured. 20

fishplate (Nederlands: Lasplaat) A metal bar, used to join two rails to each other so they form a continuous track. 47

G

GPU Graphical processing unit 50

I

IPI Ghent University Image Processing and Interpretation department 16

IR Infrared 15, 16

J

jointless track (Nederlands: Voegloos spoor) A type of railway track where the expansion joints TODO 47

L

LED Light-emitting diode 14, 15, 21

M

MFSR Multiple-frame super-resolution 39

miszending Dutch railway jargon. Describes a situation where a convoy is guided to a track that puts it on another route. This can have multiple causes, and since one can't know which one, the train conductor must perform an emergency stop as soon as he thinks a miszending has occurred. 51

MSE Mean squared error 36

P

PSNR Peak signal to noise ratio 36

R

RCM Rail contour matcher 33

RGB Red/Green/Blue 16

ROI Region of interest 35, 44, 45, 47

S

SR Super-resolution 10, 39, 40, 51

SSIM Structural similarity image metric 36

SWIR Short-wave infrared 15

T

teething trouble (Nederlands: Kinderziekte) Sometimes called 'teething issue' or 'teething problem', this term is applied to problems that are to be expected for new systems or products, but are not inherent problems and are typically solved by (relatively) little additional work. 44

track trespasser ("Spoorloper" or "Intruse de voie" in Dutch or French resp.) A person that comes in zones of the national railway company for which said person does not have the proper clearance. In practice, this mostly comes down to non-railway personnel being in the vicinity of railroad tracks. 11, 12

U

UV Ultraviolet 14

UVC Ultraviolet camera 8, 14

Y

YOLO Short for "You Only Look Once", this term refers to a family of related convolutional neural networks, designed for real-time object detection. It is frequently applied in research papers proposing solutions for rail defect detection. 22

A Elektromagnetic spectrum

B Line study card

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf

../papers/l2.pdf