Machine learning and computer vision for urban drainage inspections

Meijer, D.W.J.

Machine Learning and Computer Vision for Urban Drainage Inspections

"What do such machines really do? They increase the number of things we can do without thinking. Things we do without thinking — there's the real danger."

Frank Herbert, *God Emperor of Dune*

Machine Learning and Computer Vision for Urban Drainage Inspections

Proefschrift

ter verkrijging van
de graad van doctor aan de Universiteit Leiden,
op gezag van rector magnificus prof.dr.ir. H. Bijl,
volgens besluit van het college voor promoties
te verdedigen op dinsdag 7 november 2023
klokke 15.00 uur

door

Dirk Willem Johannes Meijer
geboren te ‘s-Gravenhage
in 1989

Promotores:

dr. A. J. Knobbe

prof.dr. T. H. W. Bäck

Promotiecommissie:

prof.dr. M. S. K. Lew

prof.dr.ir. F. J. Verbeek

dr. L. Scholten (TU Delft)

prof.dr.ir. F. H. L. R. Clemens (TU Delft, NTNU Trondheim)

dr. K. J. Wolstencroft

This work is part of the Cooperation Programme TISCA (Technology Innovation for Sewer Condition Assessment) with project number 15343, which is (partly) financed by NWO domain TTW (the domain applied and Engineering Sciences of the Netherlands Organisation for Scientific Research), the RIONED Foundation, STOWA (Foundation for Applied Water Research) and the Knowledge Program Urban Drainage (KPUD).

Cover design by Dirk W. J. Meijer, from data collected by Rianne A. Luimes.

Typeset with pdfLaTeX, style based on `tufte-latex`.

Printed by NBD Biblion Services.

Dedicated to the memory of Ilundi
for helping me rediscover meaning.

Contents

Introduction

1

This thesis aims to expand the available knowledge on how machine learning and computer vision techniques can be used to improve efficiency and quality of urban drainage inspections. This chapter will outline the motivation for the research and the contents of the rest of the thesis.

1.1 Motivation

1.1.1 Sewer Asset Management

Properly operating urban drainage systems are essential to ensure public health, safety, and productivity in cities, but not enough is known about the failure mechanisms that lead to decreased performance or loss of functionality [1]. To understand the condition of the system and to assess which assets need repair or rehabilitation, inspections are performed.

The largest share of the operation and maintenance cost across the technical assets in the system is usually spent on the sewer pipes. For their inspection, CCTV inspection is commonly performed: a 'pipe inspection vehicle' (*PIG*) is lowered into a manhole, where it records photo or video footage which is reviewed by trained operators. The operators identify defects and possible indications of defects in the footage, and assign this a severity rating between 1 (no intervention necessary) and 5 (immediate intervention necessary).

One of the major shortcomings of this method is that these severity ratings and the defect identification prior to it are highly subjective, and have been shown to differ not only between operators, but also for the same operator at different time points [2] [3].

[1] Stanić, N., Langeveld, J. G., and Clemens, F. H. 2014. Hazard and operability (hazop) analysis for identification of information requirements for sewer asset management. *Structure and Infrastructure Engineering 10,* 11, 1345–1356

[2] Dirksen, J., Clemens, F., Korving, H., Cherqui, F., Le Gauffre, P., Ertl, T., Plihal, H., Müller, K., and Snaterse, C. 2013. The consistency of visual sewer inspection data. *Structure and Infrastructure Engineering 9,* 3, 214–228

[3] Wirahadikusumah, R., Abraham, D., and Iseley, T. 2001. Challenging issues in modeling deterioration of combined sewers. *Journal of infrastructure systems 7,* 2, 77–84

A second shortcoming is that these urgency ratings assigned by operators do not necessarily reflect the actual urgency of intervention in a broader sense. It is only a measure of how advanced a specific defect appears to be, and does not take into account any other factors, such as location of the pipe or co-occurrence with other defects. As such, it is not a viable measure of the risk or costs incurred by neglecting to intervene.

1.1.2 Machine Learning and Computer Vision

At the start of the SewerSense project [4], recent advances in machine learning and computer vision had for the most part not yet been introduced to the field of urban drainage. The SewerSense project aimed to incorporate such recent advances to the sewer inspection task.

[4] TISCA programme funded by NWO-TTW. 2016-2020. Sewersense – multi-sensor condition assessment for sewer asset management

Machine learning techniques such as neural networks and other classification algorithms can potentially automate parts of the inspection by processing the photo and video footage faster and more precisely than a trained operator could. Automating parts of the inspection process can on the one hand objectify the inspection results, and on the other hand facilitate decision making in sewer asset management by providing more accurate information than an arbitrary urgency scale.

In the short term, these techniques may also be integrated into current inspection practices to increase the inspection efficiency and quality, while the industry slowly moves toward fully automated inspections. The visual inspections that are the current practice can be used to train computer vision algorithms that can extract knowledge from images, which can bring short term benefit even if the eventual fully automated solution does not rely on visual inspection.

Full automation may become possible in the future, but

will require more and higher quality data than is available at the time of writing this thesis. Some defect types are exceedingly rare, difficult to spot, and experts may not agree on them. Expecting an algorithm to accurately detect such defects is currently not realistic. In the long term, however, trained machine learning and computer vision algorithms may reshape the sewer inspection practices to rely less on human inspections.

1.1.3 Scope

The scope of this thesis then, is to perform preliminary research into the possibilities of machine learning for automation in the asset management industry. We attempt to bridge the gap between current sewer inspection practices and state-of-the-art machine learning and computer vision techniques that promise to automate (parts of) the process. We explore how well machine learning and computer vision algorithms can perform on existing data from previous inspections, as well as how an additional mode of data collection can improve this performance while remaining compatible with current inspection practices.

This thesis will not cover the collection of visual inspection data that is used as input for our models, nor the decision making process of when and whether to repair or replace. These aspects are both covered extensively in their respective domains.

1.2 Research Questions

We pose several research questions to be answered in this thesis. These question roughly correspond with chapters of the thesis, and the answers will be summarised in the conclusion.

Q1

What knowledge can be obtained from available inspection data without the utilization of expert classification, which might be inconsistent or unavailable?

We know expert classification to be limited in reliability [5], and much data collected from inspections may not have the expert classifications available in machine-readable format. Unsupervised learning techniques can obtain knowledge from this data simply by clustering it in similarity to itself, and may circumvent these two issues altogether.

[5] Dirksen, J., Clemens, F., Korving, H., Cherqui, F., Le Gauffre, P., Ertl, T., Plihal, H., Müller, K., and Snaterse, C. 2013. The consistency of visual sewer inspection data. *Structure and Infrastructure Engineering 9,* 3, 214–228

Q2

How can the data collected with current inspection practices be analysed with machine learning techniques in order to improve processing efficiency and accuracy?

While fully automated sewer inspections might not be a pipe dream, a more timely benefit may be found in incremental progress. Instead of rebuilding the industry from the ground up for automation, we investigate whether analysis of existing data with modern computation can improve quality today.

Q3

How do we assess the quality and operational impact of (partial) automation of the current inspection practices?

Training a model to make predictions is of limited use if we have no reference to compare the modelled predictions to reality. For such a model to be used in practice, we must have an assessment framework with which to assess its performance. Different types of errors will have differing consequences, and the impact on the industry might not be measured by metrics that are common in machine learning.

Q4

To what extent are the current inspection practices automatable?

Q1, Q2, and Q3 focus on building and verifying models from existing data, we must also look beyond the current practices and investigate the limits of such approaches. Restricting models to the existing data may hinder progress or reinforce existing biases from current practices. It is important then, to decide whether building on the current practices is worthwhile, compared to a bottom-up design approach focused on automation entirely.

Q5

Does introducing depth information through computer stereovision improve the data quality and analysis capabilities?

Extending Q4, we investigate a specific additional mode of data collection, that of stereovision, the use of two cameras to obtain not only a two-dimensional image, but a depth component as well. Such advanced modes of data collection are not exactly novel, but have not seen much use in practice because of the additional training required for human operators to interpret the results. From our scope of preparing the industry for automation, it is then again an interesting question to see what the added value of such data modes is.

Q6

How can we employ machine learning and computer vision to improve the efficiency and quality of urban drainage inspections?

Using the answers to the previous five questions, we conclude having outlined the possibilities of enriching sewer asset management with machine learning and computer vision techniques, and highlight the areas that still require more research.

1.3 CONTRIBUTION

Many of the ideas and text in this thesis have appeared earlier in four publications, written over the course of five years. These papers are an extended abstract in a regional conference, a continuation of that extended abstract in the proceedings of an international conference, and two articles published in a high impact journal.

This section presents an overview of each of the main chapters in this thesis, and lists the main contributions made for that chapter and in which publication the contributions first appeared.

CHAPTER 2 goes over preliminary knowledge required for a complete understanding of this thesis.

CHAPTER 3 approaches the defect detection problem from an unsupervised learning perspective, based on how commonly patterns appear in a set of images. The approach leverages principal component analysis or a convolutional autoencoder for their ability to generalise only those patterns that appear frequently in the training set, interpreting poor generalization as a signal of anomalous information. The main contribution is the comparison of the original image to an image partially reconstructed by the autoencoder or a limited number of principal components. Parts of this chapter have previously appeared in [6] and [7]. This chapter explores research questions Q1 and Q4.

CHAPTER 4 approaches the defect detection problem from a supervised learning perspective, using a convolutional neural network. The article this chapter is based on became highly influential, having been cited over two dozen times at the time of writing this thesis. This might be attributed to timeliness or novelty, but we feel that the main contribution

[6] MEIJER, D. W. AND KNOBBE, A. J. 2017. Unsupervised region of interest detection in sewer pipe images: Outlier detection and dimensionality reduction methods (extended abstract). In *Benelux Conference on Machine Learning (BeneLearn)*

[7] MEIJER, D. W., KESTELOO, M., AND KNOBBE, A. J. 2018. Unsupervised anomaly detection in sewer images with a PCA-based framework. In *International Conference on Pattern Recognition and Artificial Intelligence (ICPRAI)*. 354–359

is the methodological groundwork that explores the results as relevant to the target domain: the realistic conditions in which the experiment was performed, to ensure a realistic assessment of real-world value of such a classifier. Parts of this chapter have previously appeared in [8]. This chapter explores research questions Q2, Q3, and Q4.

Chapter 5 introduces a new data modality by using two cameras to capture sewer pipes in stereovision. The added depth channel is combined with knowledge of the physical properties of the setup to reconstruct the three-dimensional pipe geometry virtually. Anomaly detection is performed through robust regression of a model that is informed by the expected geometry of a sewer pipe, under the assumptions that certain types of surface damage will deviate from this model. The main contributions are the adaption of stereovision techniques to this unique use case of an object that is positioned perpendicular to the image plane, and the sewer pipe model. Parts of this chapter have previously appeared in [9]. This chapter explores research question Q5.

Chapter 6 concludes with a summary of the main content of the thesis and answers to the six research questions.

[8] Meijer, D. W., Scholten, L., Clemens, F. H., and Knobbe, A. J. 2019. A defect classification methodology for sewer image sets with convolutional neural networks. *Automation in Construction 104*, 281–298

[9] Meijer, D. W., Luimes, R. A., Knobbe, A. J., and Bäck, T. H. W. 2021. RADIUS: Robust anomaly detection in urban drainage with stereovision. *Automation in Construction 139*, 104285

2 Preliminaries

This chapter covers preliminary knowledge that is required to understand the main content of the thesis. Depending on the reader's proclivities, this chapter may be skipped in its entirety, or referred back to when needed.

2.1 Machine Learning

Machine learning can be broadly summarised as using statistical methods to extract *knowledge* from *data*. The general case considers data which is comprised of *instances*, each containing the same *attributes*, one of which may be the *target* attribute. The goal is then to find some pattern in the non-target attributes that can predict the target attribute as best as possible, meaning that the prediction error across all instances is limited in some way. Machine learning can be supervised, unsupervised, or semi-supervised. This distinction indicates whether the value of the target attribute is known for all, none, or some of the instances, respectively.

This section will go over several use cases, methods, and concepts relevant to the remainder of the thesis and may be skipped or referred back to at the reader's discretion.

2.1.1 Classification

Supervised classification (henceforth simply classification) is a machine learning task where the goal is to infer a relationship between instances and target 'labels' by generalizing from a training set, a collection of such instances for which the label is known. This generalization can then be used to classify objects for which the label is not known.

We consider a dataset $\mathbf{X}$ consisting of N real-valued vec-

tor instances of equal length:

$$\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \tag{2.1}$$

$$\mathbf{x}_i \in \mathbb{R}^d \qquad \forall i \in [1, N] \tag{2.2}$$

$$\mathbf{x}_i = \left\{x_{i,1}, x_{i,2}, \ldots, x_{i,d}\right\} \qquad \forall i \in [1, N] \tag{2.3}$$

Each of these vectors also has an associated target, y_i, belonging to one of m distinct classes:

$$\mathbf{Y} = \left\{y_1, y_2, \ldots, y_N\right\} \tag{2.4}$$

$$y_i \in \{c_1, c_2, \ldots, c_m\} \qquad \forall i \in [1, N] \tag{2.5}$$

The goal of classification is then to find the function F that relates the vectors to their label:

$$F(\mathbf{x}_i) = y_i \qquad \forall i \in [1, N] \tag{2.6}$$

We can state that $F(\cdot)$ should be a function between the two domains:

$$F : \mathbb{R}^d \mapsto \{c_1, c_2, \ldots, c_m\} \tag{2.7}$$

In reality, we will have a *model*, $f(\cdot)$, which does not return the real label, but rather an estimation, $f(\mathbf{x}) = \hat{y}$. Finding and improving this $f(\cdot)$ is done through a *loss function* L, a function that is minimised when $y = f(\mathbf{x})$.

The way in which L is minimised depends on the choice of model. For some combinations of loss function and model there may exist an analytical solution that minimises L. For most combinations however, this will be a heuristic process without any analytical solution.

2.1.2 Regression

Regression is a learning problem very similar to classification, but one where the target attribute is on a spectrum. We are still dealing with a dataset $\mathbf{X}$ consisting of N real-valued

vector instances of equal length and associated targets $\mathbf{Y}$. However, instead of the target being a discrete label, y_i is now a real-valued scalar:

$$y_i \in \mathbb{R} \qquad \forall i \in [1, N] \tag{2.8}$$

$$F : \mathbb{R}^d \mapsto \mathbb{R} \tag{2.9}$$

The most common form that the model $f(\cdot)$ will take is a *linear regression model*:

$$f(\mathbf{x}_i) = \beta_0 + \sum_{k=1}^{d} \beta_k x_{i,k} \tag{2.10}$$

$$\beta = \{\beta_0, \ldots, \beta_d\} \tag{2.11}$$

Where the collection of parameters β define the model.

The most common choice of loss function for a linear regression model is the Euclidean distance, known also as the squared error:

$$L = \left\| y - f(\mathbf{x}) \right\|^2 \tag{2.12}$$

With this combination of model and loss function, the values of β that minimise the squared error as defined in equation (2.12) for given values of $\mathbf{X}$ and $\mathbf{Y}$ can be found analytically through the *ordinary least squares* method.[1]

[1] GOLDBERGER, A. S. 1964. *Classical Linear Regression, Econometric Theory.* New York: John Wiley & Sons

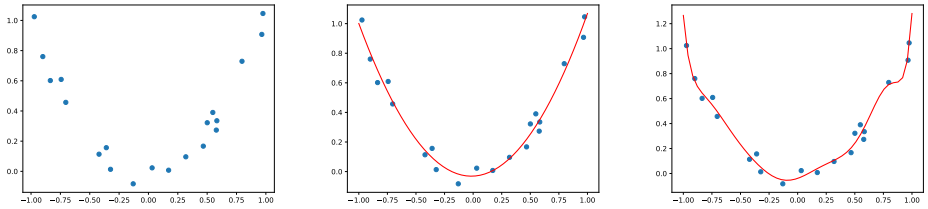
2.1.3 Overfitting, Regularization, Cross Validation

When performing classification or regression, we assume that the N vectors in $\mathbf{X}$ are sufficient for a model $f(\cdot)$ to estimate the targets $\mathbf{Y}$. Often, this is not the case, and the model $f(\cdot)$ may perform well on the training set, without generalizing well to new data from the same distribution. This effect is called *overfitting*, as the model is fitted to the training data so well that it hinders generalization performance.

To further appreciate the risks of overfitting, consider a polynomial regression model, fit onto one-dimensional data $\mathbf{X}$ with target $\mathbf{Y}$. We define the model as:

$$f(x_i) = \sum_{k=0}^{\kappa} \beta_k \cdot (x_i)^k \tag{2.13}$$

The value of κ is called the order of the polynomial, and is a direct measure of the complexity of the model, as the model has $\kappa + 1$ parameters. As an example, we create an artificial dataset by sampling $N = 20$ datapoints on a parabola and adding uniform noise to the target attribute, as shown in figure 2.1(a). Then compare the difference between a fit of a polynomial of order $\kappa = 2$ in figure 2.1(b), and a fit of a polynomial of order $\kappa = 10$ in figure 2.1(c).



(a) Scatterplot of artificial data (b) Polynomial fit of order $\kappa = 2$ (c) Polynomial fit of order $\kappa = 10$

Figure 2.1: Example of overfitting in one-dimensional regression

The more complex model with $\kappa = 10$ has a smaller error than the model with $\kappa = 2$, but an important distinction is that the smaller error was achieved on the data that the model was fit on, but not necessarily on a future sample from the same distribution (a parabola with uniform noise). The model with $\kappa = 10$ has in fact *overfitted* on the training data: the fit is so precisely tailored to this sample, that it will generalise less well than the $\kappa = 2$ model to other samples from the same distribution. One approach to prevent overfitting is to simply limit the complexity of the model we use. A good rule of thumb is that the number of model parame-

ters should be at least one order of magnitude smaller than the sample size, in this case e.g. $\kappa = 2$.

REGULARIZATION is a different approach to prevent overfitting: instead of limiting our model complexity, we solve a proxy problem that forces constraints on the solution. A regularization term is added to the loss function, which grows larger in size for more complex models. The new loss function is defined as:

$$L = L_p(\mathbf{x}, y) + \lambda L_r(f) \tag{2.14}$$

Where L_p is the *problem loss*, L_r is the *regularization loss*, and λ is a scaling factor.

The problem loss is the loss function that was described in section 2.1.1, which when minimised gives the optimal fit. The regularization loss is a value that scales with the complexity of the model. The scaling factor λ is used to weight the importance of the regularization.

The rationale is that a compromise has to be made between a model that fits the training data perfectly, and the complexity of that model. In our one-dimensional regression example, we could determine the complexity of the model by taking $L_r = \sum_{k=0}^{\kappa} ||\beta_k||$, known as L_1 regularization, which prefers that only some β_k have non-zero values.

CROSS VALIDATION does not prevent overfitting, but does give us an indication of whether overfitting is happening with our model. To validate a model, it is important to assess the performance on data that was not part of the training set. A common option is to split the available data into a training set and a testing set, using the first part to fit the model and the second to test it.

This method of splitting data becomes problematic when the amount of available data is small, as any data point not in the training set will make the fit worse, and any data point

not in the test set will make the estimated performance less reliable.

Cross validation offers a middle ground by having a small test set and a large training set, but repeating the process to better estimate the performance. The data set is divided into k non-overlapping *folds*. One fold is used as the test set and the remaining folds are concatenated into the training set. This process is repeated k times, each time using a different fold as the test set, such that every item in the data set has been tested once. $k = 10$ is a customary value, which we call '10-fold cross validation'.

2.1.4 Model Validation

Perhaps just as important as obtaining a trained model, is knowing the model's boundaries. In the context of urban drainage inspection, if a classifier predicts some defect, it is important to know how trustworthy this result is to put it into context before acting on it. To properly assess the quality of our models, we examine several different performance metrics to discuss their value.

Commonly, the classification *accuracy* is used as a performance metric, which is the ratio of correctly classified samples out of all samples. It can be observed that this is not a very useful measure for extremely imbalanced datasets, or use cases where different types of errors do not carry the same cost. For an extremely imbalanced dataset, we could create a classifier that classifies every datum as the majority class, and it would have a high accuracy. When different types of errors carry different costs, we may want to use a metric that is aware of these costs, which accuracy is not.

Table 2.1 shows a *confusion matrix* for a binary classification problem, which illustrates the types of errors we can make when misclassifying instances. True positives (TP) and true negatives (TN) are correct classifications, false negatives (FN, sometimes called Type II Errors) and false pos-

Table 2.1: Confusion matrix for a binary classification scenario

Actual \ Predicted	Defect	No Defect
Defect	True Positive	False Negative (Type II error)
No Defect	False Positive (Type I error)	True Negative

itives (*FP*, sometimes called Type I Errors) are misclassifications. We might not always want these types of errors to count equally in our performance assessment.

We define the false positive rate (*FPR*), true negative rate (*TPR* or *specificity*), false negative rate (*FNR*), and true positive rate (*TPR*, or *recall*) by dividing a quadrant in the confusion matrix with the row total:

$$FPR = \frac{FP}{FP + TN} = 1 - TNR \tag{2.15}$$

$$TNR = \frac{TN}{FP + TN} = 1 - FPR \tag{2.16}$$

$$FNR = \frac{FN}{FN + TP} = 1 - TPR \tag{2.17}$$

$$TPR = \frac{TP}{FN + TP} = 1 - FNR \tag{2.18}$$

These rates hold some significance as they are equivalent to the chances that a classifier will make a certain error. For example, if $FNR = 0.2$, this means in practice that 20% of actual defects are not recognised as defects by the classifier.

A binary probabilistic classifier may output real-valued predictions in the interval $[0, 1]$, while the actual labels are always either 0 or 1. This means we have some freedom in choosing a threshold τ, that separates a predicted 0 from a predicted 1. We write this as:

$$\hat{y} = \begin{cases} 0 & \text{if } f(x) < \tau \\ 1 & \text{if } f(x) \geq \tau \end{cases} \tag{2.19}$$

where $\hat{y}$ is the predicted label and $f(x)$ is the classifier's real-valued output for instance x. Setting a specific threshold τ for the classifier's output results in each classified instance being either a true positive ($y = \hat{y} = 1$), a true negative ($y = \hat{y} = 0$), a false positive ($y = 0; \hat{y} = 1$), or a false negative ($y = 1; \hat{y} = 0$).

We have some freedom on how to choose τ, which gives us a way to balance the false negatives and false positives. Any increase in τ leads to an increase in *FNR* and a decrease in *FPR* (and vice versa). If we decide that a false negative is 20 times as costly as a false positive, we could set τ at an optimum such that $\frac{FPR}{FNR} = 20$.

The trade-off leads to the construction of the *receiver operating characteristic* (ROC) curve [2], as shown in figure 2.2, showing values of the *TPR* and *FPR* on the vertical and horizontal axes respectively. Every value of τ corresponds to a point in the ROC curve, and it is common to use the area under the ROC curve (AUROC) as a measure of classifier performance that is independent of the particular setting of threshold τ.

[2] Bishop, C. M. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg

The reason accuracy is not a very useful measure for urban drainage inspections specifically, is that it is a particularly unbalanced problem: defects are very uncommon [3]. The *FPR* (and the equivalent *TNR*) are dominated by the *TN* term in this unbalanced classification scenario, which isn't very interesting, as it is easy to achieve a very high *TN* by classifying everything as negative. In practice, this would mean that the detection system would never detect a defect, and in fact would be correct in that detection for 99% of cases. As the *FPR* is one of the dimensions of the ROC, this leads to the (AU)ROC only having limited usefulness. Instead of the *FPR*, we use *precision*, defined as:

[3] In the real-world datasets used in this thesis, we found defects to appear in approximately 1% of images.

$$Pr = \frac{TP}{TP + FP} \tag{2.20}$$

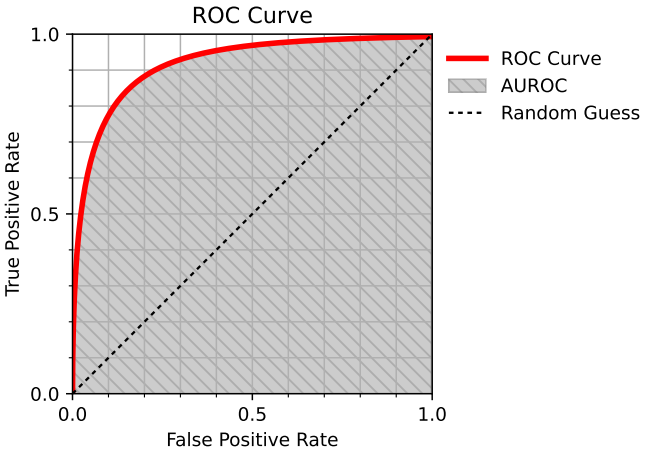


Figure 2.2: Example of a Receiver-Operating Characteristic (ROC) curve. Every point on the red line corresponds to a possible threshold τ, that defines the TPR and FPR. The shaded area shows the area under the ROC curve (AUROC), and the dashed line is the ROC curve one would obtain by randomly guessing the label for each instance in a binary classification scenario.

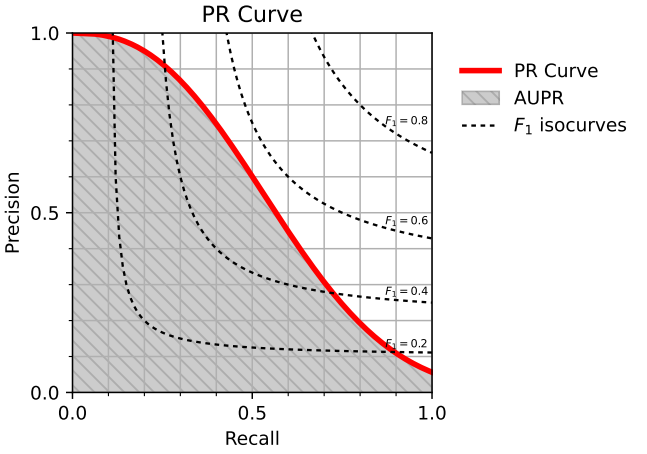


Figure 2.3: Example of a Precision-Recall (PR) curve. Every point on the red line corresponds to a possible threshold τ, that defines the precision and recall. The shaded area shows the area under the PR curve (AUPR), and the dashed lines are curves with a constant F_1-score.

Both the *FPR* and the precision are measures of the number of type I errors (detecting a defect when there is none) a classifier makes. The difference is that the *FPR* compares this to the total negative cases (*"How many of the sewer pipes without defects did we label as defective?"*), whereas the precision compares this to the total cases that were classified as positive (*"How many of the sewer pipes labeled as defective are false alarms?"*). The former is heavily skewed towards the appearance of a good performance, because of the prevalence of negative cases, but the latter does not have this issue.

If we now combine precision with *TPR*, we can construct a curve analogous to the ROC curve, called the Precision Recall [4] curve, or PR curve, as shown in figure 2.3. The area under the PR curve is more meaningful than the AUROC and also independent of τ. Figure 2.3 also displays F_1-score isocurves, curves where the harmonic mean of the precision and recall are constant. Being a function of precision and recall, the F_1-score is also a metric of performance that is not influenced by the overwhelming amount of false positives that rule the accuracy metric.

[4] Recall is a synonym for TPR.

2.1.5 Anomaly Detection

Anomaly detection, sometimes referred to as outlier detection, is a machine learning problem aimed at finding instances in a dataset that deviate from the majority [5]. It has many applications, from fraud detection to noise removal.

Unsupervised anomaly detection relies in most cases on robust [6] regression. This means that we look for some model that explains the behaviour of *most* instances in our dataset. Any instances not explained by this model are considered to be anomalies or outliers.

An important quality of the model we fit on our data is that it has limited complexity. If the model's complexity is too high, it may fit the anomalies that we are trying to detect as well, meaning they become inliers and are no longer detected as anomalies. Still, a certain degree of complexity may be required in order to account for variation in the data. This implies a trade-off between how complex we allow patterns to be, and when complexities become anomalies. This is similar in concept to regularization as described in section 2.1.3: a regularised model may be well suited to detect anomalies by finding data that it does not generalise well to.

[5] Zimek, A. and Schubert, E. 2017. *Outlier Detection*. Springer New York, New York, NY, 1–5

[6] *Robust* in this context refers to a reduced sensitivity to noise or outliers.

2.1.6 Principal Component Analysis

Principal Component Analysis (PCA) [7] is a popular tool in statistics, data science and many other scientific fields, used to reduce the dimensionality of data to facilitate data exploration and the use of algorithms that are sensitive to high dimensionality. It may be thought of as a form of unsupervised learning.

Given a dataset $\mathbf{X}$, consisting of N instances with d real-valued attributes each, we express this as an $[N \times d]$ matrix. PCA is performed by calculating the covariance matrix $\mathbf{C}$ of this matrix and performing eigenvalue analysis on $\mathbf{C}$. This results in d eigenvalues and d eigenvectors (or 'principal components') of length d. These eigenvectors form an orthonormal basis for the space in which our dataset $\mathbf{X}$ exists and the corresponding eigenvalues are proportional in magnitude to the variance of dataset $\mathbf{X}$ explained by each eigenvector (their sum being equal to the total variance present in $\mathbf{X}$). This establishes a transformation from the original space of d dimensions to a new space that also consists of d dimensions.

When using PCA for dimensionality reduction, we choose a dimensionality $\theta \leq d$ and project $\mathbf{X}$ [8] on the first θ eigenvectors (in order of descending eigenvalues). The projected matrix $\mathbf{P}$ retains as much variance as is possible [9] in θ dimensions. This allows researchers to view high-dimensional data in two or three-dimensional visualisations, or employ algorithms that are not designed for high-dimensional data.

When $\theta = d$, we can return to the original space by inverting the projection matrix and adding the mean values of each feature after transformation, without any loss of information. In chapter 3 we will utilise a partial projection as an unsupervised anomaly detection method.

[7] Pearson, K. 1901. LIII. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2*, 11, 559–572

[8] The covariance matrix and the newly-found orthonormal basis do not contain the mean values of the original dataset $\mathbf{X}$, so the dataset should be centred around zero before projecting onto the basis.

[9] Barring non-linear embeddings

2.2 Digital Image Processing

When using visual data for statistical learning, we will often convert this visual data to digital images, suited for computer processing. Digital images are represented as two-dimensional matrices of pixels. The pixels themselves may be scalar or vector values, for greyscale or colour images respectively.

In the case of scalar pixels that take binary values, we speak of a binary image or a *mask*. In the case of colour images, the most common representation is RGB [10], corresponding to digital screens, although HSL/HSV [11] and CMYK [12] are also common, depending on the application. By convention, for greyscale (and binary) images, higher values correspond to lighter tones, the maximum value corresponding to white and the minimum value corresponding to black. For convenience, the colourspace is often scaled to a range of $[0, 1]$.

[10] Red, Green, Blue

[11] Hue, Saturation, Lightness/Value

[12] Cyan, Magenta, Yellow, blacK

Sometimes the individual values of vector pixels may be called *channels*. We might speak of the 'hue' channel of an HSV image for example, which is itself an image with scalar values.

We write an image as $I(x, y)$, where x and y are the horizontal and vertical locations of the pixel in the image, and the value of I is the pixel value at that location.

2.2.1 Convolution

In signal processing, it is common to apply filters to signals through the use of a convolution operation. In the case of images, these filters can be used to smooth or sharpen certain patterns in images, like edges, corners, or textures. In convolution, a filter is moved across the image, and at

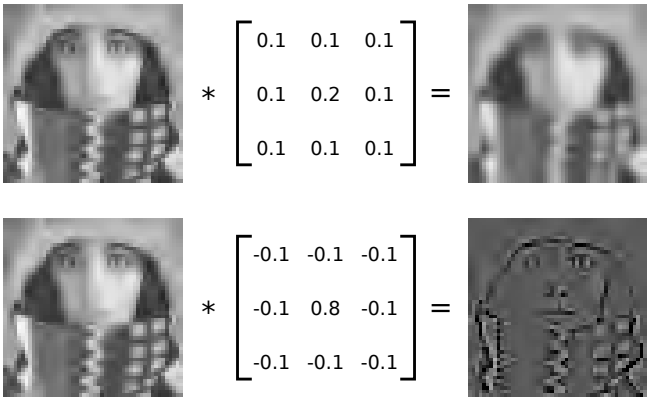


Figure 2.4: Example of how convolution with different kernels can be used to smooth an image or emphasise its edges.

each position the 'overlap' between the image and the filter is calculated.

Discrete [13] convolution in two dimensions is a mathematical operation defined as:

$$I(x,y) * k(x,y) = \sum_{u=-\infty}^{\infty} \sum_{v=-\infty}^{\infty} I(u,v)k(x-u, y-v) \tag{2.21}$$

In practice we might smooth image $I(x,y)$ by convolving it with a Gaussian *kernel* $k(x,y)$. It should be noted that while convolution is defined on an infinite domain, in practice both the image and the kernel will be non-zero only within limited domains and the convolution can be performed by summing only over those domains. The kernel itself is commonly a scalar image, and as such the output of the convolution has the same number of channels as the image $I(x,y)$. Figure 2.4 shows two examples of how convolution can be used to smooth an image or to detect edges in an image.

[13] As opposed to continuous convolution, which is defined for continuous signals. Our signals are images, consisting of pixels at discrete locations, and as such continuous convolution is beyond the scope of this thesis.

2.3 Convolutional Neural Networks

Neural networks are a type of machine learning algorithm modelled after the way synapses in the human brain activate and pass information along [14]. Convolutional neural networks are a type of neural network that has been shown to work particularly well with image data [15]. This section will explain them into as much detail as is required for the context of this thesis.

[14] Goodfellow, I., Bengio, Y., and Courville, A. 2016. *Deep learning*. Vol. 1. MIT press Cambridge

[15] Szeliski, R. 2010. *Computer Vision: Algorithms and Applications*, 1st ed. Springer-Verlag, Berlin, Heidelberg

2.3.1 The Perceptron

Neural networks are composed of neurons, smaller elements that perform a simple function. The network itself performs functions much more complicated than the neurons are capable of. The simplest neuron is Rosenblatt's perceptron [16]. A perceptron has inputs, weights, a bias, and an activation function. Each input is multiplied with its assigned weight, and all are added together with the bias. The single scalar resulting from this operation is passed through the activation function, which can be any function, but is usually a non-linear non-decreasing function [17]. (The reason it is non-linear is that if it were linear, the network itself would learn a linear function in the inputs, which means the network *would* be limited to a function no more complex than the neurons themselves.) Traditionally, the step function was used, but sigmoidal functions or piecewise linear functions are also common [18].

The perceptron itself is a neural network, and can be employed as a classifier as defined in section 2.1.1, by using it as a function to describe a relationship between data and

[16] Goodfellow, I., Bengio, Y., and Courville, A. 2016. *Deep learning*. Vol. 1. MIT press Cambridge

[17] Bishop, C. M. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg

[18] Goodfellow, I., Bengio, Y., and Courville, A. 2016. *Deep learning*. Vol. 1. MIT press Cambridge

labels:

$$f(\mathbf{x}) = \hat{y} = \phi\left(w_0 + \sum_j x_j w_j\right) \tag{2.22}$$

where x_j are the inputs, w_j are the weights, w_0 is the bias, $\phi(\cdot)$ is the activation function, and $\hat{y}$ is the output.

The weights and bias are initialised to random values, which means its output $\hat{y}$ is initially unlikely to resemble its target y much at all. The perceptron is trained through an iterative process called *backpropagation*, which brings the output $\hat{y}$ closer to the target y with every iteration, by changing the weights and bias by slight increments.

Backpropagation works by feeding a single instance x into the perceptron, which returns output $\hat{y}$. We then compare $\hat{y}$ to the expected output y with the loss function, and determine the derivative of the loss with regards to $\hat{y}$. We use the differentiation chain rule [19] to calculate the derivative of the loss with respect to each weight. A gradient step towards minimizing the loss is calculated by multiplying each weight's derivative by a (typically small) learning rate α and the weights are adjusted.

After sufficiently many iterations of the backpropagation process, the perceptron will have converged to a state where the backpropagation process cannot further reduce the loss function for the data it was trained on. This process often requires multiple *epochs*, the amount of iterations it takes for the entire dataset to be processed.

[19] Differentiation chain rule:

$$\frac{dL}{dw_j} = \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dw_j}$$

Multi-Layer Perceptron

To make a slightly more complex neural network, capable of learning more complex relations between input and output, we can chain multiple perceptrons together to create the multi-layer perceptron [20]. The perceptrons are ordered in several *layers*. Only the first layer has the dataset as its input, and each perceptron in the layer processes this data in parallel, each outputting a single scalar. The perceptrons in the

[20] GOODFELLOW, I., BENGIO, Y., AND COURVILLE, A. 2016. *Deep learning*. Vol. 1. MIT press Cambridge

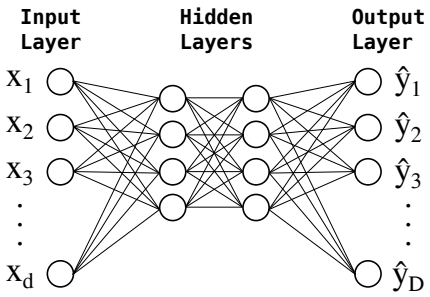


Figure 2.5: An illustration of a multi-layer perceptron with 2 hidden layers.

next layer then each take these outputs from the previous layer as inputs, applying their weights, bias, and activation function as if these were input data, et cetera for consecutive layers. The final layer has as many neurons as the dimensionality of the output y.

Such a network in which data flows from input to output without any loops is called a *feed-forward* neural network. The backpropagation process still works the same way, except that the gradients require more complex calculations with each layer added. This way, adding additional layers allows us to model more complex functions by adding the same elementary neurons. An illustration is shown in figure 2.5.

The layers of neurons described in this section are called *dense* or *fully-connected* layers, as the output of each neuron in a layer is connected to the input of each neuron in the next layer. To construct a convolutional neural network, two additional layer types are required: the *convolutional* layer, a layer in which perceptrons have limited connections with the previous and next layers, and the *pooling* layer, a layer in which the neurons also have limited connections, but more importantly, the neurons perform a significantly different function from perceptrons.

2.3.2 CONVOLUTIONAL LAYERS

Convolutional layers are generally used for image processing and perform the convolution operation as described in section 2.2.1.

We can simplify the perceptron somewhat to simulate this function. The advantage here is that we can use the backpropagation process to learn image filters, instead of having to design the filters based on what structure we expect the images to contain.

A simple convolutional layer consists of as many neurons as the previous layer in the network, but neurons are connected only to neurons in the previous layer that *surround* the neuron in the same location. In a one-dimensional setting, this would mean that neuron n_i in the convolutional layer receives as input the output from neurons n_{i-s} to n_{i+s} in the previous layer. We call $2s+1$ the *filter size*, as each neuron in the convolutional layer has $2s+1$ inputs. In a higher-dimensional setting the neighbourhood around neuron n_i that is connected to neuron n_i in the next layer extends in all dimensions.

Additionally, each neuron in the convolutional layer has the same weights, just different connections. This is called *weight sharing*, and it results in an equivalent of the convolution operation performed by the network, as each neuron applies the same filter, but at a different location in the image.

Three additional hyperparameters are used to define a convolutional layer, the *stride*, the *depth*, and the *edge condition*.

The stride, or step size, incorporates a subsampling mechanism into the layer. If a stride of x is chosen in a one-dimensional setting, the convolutional layer contains N/x neurons, where N is the number of neurons in the previous layer. In other words, $N \cdot (x-1)/x$ neurons in the convolutional layer are discarded. The reasons one might do this,

is that if neither the kernel nor the image consist entirely of white noise, adjacent samples in the result will be highly correlated, and we can reduce the neurons in the layer (and with it the computational requirements of backpropagation) without losing much of the information present in the input data. In higher-dimensional settings, the stride has as many dimensions as the data because it can be defined for every dimension individually.

The depth of a convolutional layer determines how many filters are applied in parallel. With a depth of 1, every neuron is just a perceptron with limited connections, as mentioned earlier in this section. However, when the depth increases, each neuron becomes a collection of these limited perceptrons, and the output of the neuron is simply a vector of these perceptrons' outputs.

Finally, the edge condition determines what happens at the edges of an image, when the input does not fully overlap with the filter size. For example, with a filter size of $2s + 1$, the behavior for the first and last s samples of the input is poorly defined. It is an option to simply discard these cases, but this results in a layer size that is dependent on the filter size. Another option is to pad the input by half the filter size (rounded down) and then discard the violating cases, which does ensure the convolutional layer to be of the same size as the input layer. In this thesis we have chosen to apply zero-padding, black pixels are added to the edges of the images.

2.3.3 Pooling layers

A pooling layer is similar in structure to a convolutional layer, but instead of consisting of perceptrons, it consists of neurons that perform a non-linear function. They are often placed immediately after a convolutional layer, to introduce a non-linearity that allows the network to learn more complex structures than it would with just convolutional and

dense layers [21].

The most common example is the max-pooling layer. It has the same structure as the convolutional layer, taking as its input only the surrounding neurons from the previous layer, and has a filter size and stride as well. However, instead of multiplying the input by weights and summing, it simply outputs the maximum value of each depth slice from the inputs, as if we were performing a morphological dilation.

By taking the maximum value over a spatial window, we can perform a dimensionality reduction when the stride is larger than 1. The reason the maximum value is used is that this works well with the convolution operation: convolution overlaps two signals (image and kernel in this case) and returns the inner product, so a high response at a location means that the image resembles the kernel at that location. By taking the maximum, we achieve a sense of how much that portion of the image resembles each kernel (each depth slice).

2.3.4 Convolutional Neural Network Design

Conventional wisdom in recent image processing techniques dictates a set of design patterns for convolutional neural network architectures that have been shown to work well. Specifically, the most common pattern is to interlace convolutional layers with max-pooling layers. The input is fed through some number of these interlaced layers successively before being passed into some number of dense layers, before being passed to the output.

The issue with designing and optimizing these networks is that the search space is infinite, and while there exists a lot of knowledge on what does and does not work for common datasets and tasks, still much research has to be done on *why* these are good design patterns [22].

[21] Goodfellow, I., Bengio, Y., and Courville, A. 2016. *Deep learning*. Vol. 1. MIT press Cambridge

[22] Goodfellow, I., Bengio, Y., and Courville, A. 2016. *Deep learning*. Vol. 1. MIT press Cambridge

Commonly, networks are used that have been shown to work well on difficult datasets (such as ImageNet), often pre-trained to reduce the time it takes to train on the new dataset, but the "no free lunch" theorem [23] dictates that there cannot be a single architecture that works best on all different tasks and datasets.

[23] Wolpert, D. H. and Macready, W. G. 1997. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation 1,* 1, 67–82

2.4 Computer Stereovision

Computer stereovision, or simply 'stereovision', is a computer vision technique in which two side-by-side cameras simultaneously record an image. The correspondence between points that appear in both images give us information on the distance from the cameras to that point, similar to how the correspondence between the left and right eyes allows humans to perceive depth [24].

To illustrate the principle, we examine the *epipolar plane* [25] of two horizontally aligned cameras and an object that is visible to both cameras, as shown in figure 2.6. The problem is significantly simplified by the camera axes being parallel, which is an achievable situation for the application of urban drainage inspections. $\mathbf{C}_1$ and $\mathbf{C}_2$ are the two cameras, and $\mathbf{P}$ the point of interest. Both cameras have identical physical properties, and we consider $\mathbf{C}_1$ to be the *reference camera*. f is the focal distance of the cameras, b is the baseline distance between the cameras, two physical distances that we know precisely. $\mathbf{I}_1$ and $\mathbf{I}_2$ are the *virtual* image planes, one focal length distance in front of the cameras.

We wish to calculate X and Z, the physical location of $\mathbf{P}$ in the epipolar plane, from the perspective of our reference camera $\mathbf{C}_1$. Consider d_1 and d_2, the projected locations of $\mathbf{P}$ onto $\mathbf{I}_1$ and $\mathbf{I}_2$, relative to the centres of the image planes [26].

[24] Howard, I. P. and Rogers, B. J. 2012. *Perceiving in depth, Volume 2: Stereoscopic vision*. Oxford University Press

[25] Szeliski, R. 2010. *Computer Vision: Algorithms and Applications*, 1st ed. Springer-Verlag, Berlin, Heidelberg

[26] Note that we take d_2 to be a negative value in this case, as it is to the left of the centre of $\mathbf{I}_2$. If $\mathbf{P}$ would be on the same side of both camera axes, d_1 and d_2 would have the same sign.

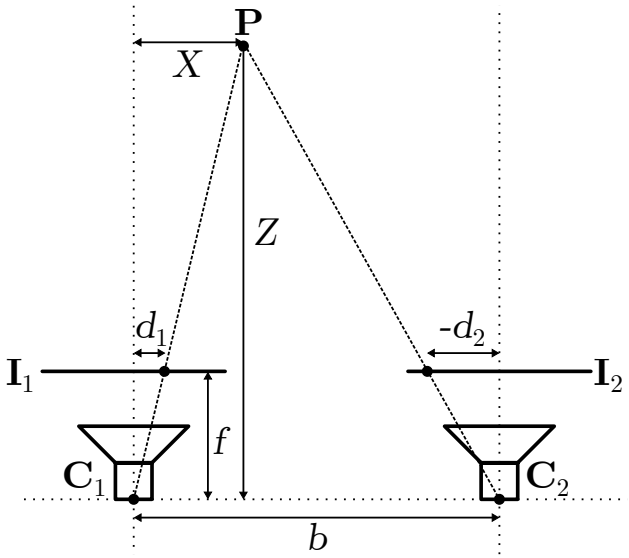


Figure 2.6: Epipolar geometry with parallel camera axes

Similar triangle geometry allows us to solve for Z and X:

$$d_1/f = X/Z \tag{2.23}$$
$$-d_2/f = (b - X)/Z \tag{2.24}$$
$$(d_1 - d_2)/f = b/Z \tag{2.25}$$
$$Z = b \cdot f/(d_1 - d_2) \tag{2.26}$$
$$X = d_1 \cdot b/(d_1 - d_2) \tag{2.27}$$

Two important things should be noted at this point:

◇ The Y coordinate of $\mathbf{P}$, which we neglected in this planar example for simplicity, also has to be computed.

$$Y = d_y \cdot b/(d_1 - d_2) \tag{2.28}$$

where d_y is the vertical position relative to the centre of the projection of $\mathbf{P}$ on $\mathbf{I}_1$. Since the cameras are aligned in the horizontal plane, there is no need to take a vertical shift into consideration, as any point

will be projected on both virtual image planes at equal height.

◇ For the calculated coordinates to be represented in physical units, we can either express d_1 and d_2 in physical units, or we can express f in pixels instead of physical units. Either conversion is done by finding the physical size of a pixel on the camera's sensor array. In this thesis, we will assume the focal length f is expressed in pixels.

Stereovision algorithms apply this principle to all pixels in an image: each pixel is considered to be a projection of some point with physical coordinates that we try to find. Each pixel in the image produced by the reference camera is matched to a pixel in the second image, and the difference in horizontal positions of these pixels produces the *disparity* $(d_1 - d_2)$. More specifically, for each pixel in the reference image, a local neighbourhood around the pixel is compared to a patch of the same size as this neighbourhood, in the same position in the other image, but shifted horizontally. The horizontal shift that minimises the difference between the two image patches is considered the best match.

This introduces multiple difficulties, as finding the correspondence between pixels is a heuristic search process with multiple local optima, as exhaustive search is often infeasible. Images that have some periodicity in the horizontal direction may result in the correspondence being off by a multiple of the period. An even bigger challenge arises when the selected neighbourhood patch is entirely smooth: matching the exact location will become difficult, as small shifts lead to little difference in matching quality. Practically, an exact alignment of the cameras is also difficult to achieve, and any physical camera and lens are going to introduce distortion to the recorded images [27], both of which will have to be corrected before the matching process commences.

[27] Hecht, E. et al. 2002. *Optics*. Vol. 5. Addison Wesley San Francisco

3 Image-Based Unsupervised Anomaly Detection

In this chapter, we propose a three-part framework to detect anomalies in *aligned image sets*, such as static camera video or photographs, or registered images. The framework is based on principal component decomposition and partial reconstruction, but accounts for the fact that not all common elements in image sets can be accounted for by a linear model (such as PCA is) by first extracting possibly non-linear features from the image sets. We also foray into the field of deep learning and investigate the possibility of using convolutional autoencoders (CAEs) to fill the role of several parts of the framework.

We would like to emphasise that while this framework originated from the need to automatically process sewer pipe images, no assumptions are made specific to this problem. The only requirement is that *the images in a set are aligned*, so other possible applications include video surveillance, autonomous vehicles and medical image processing.

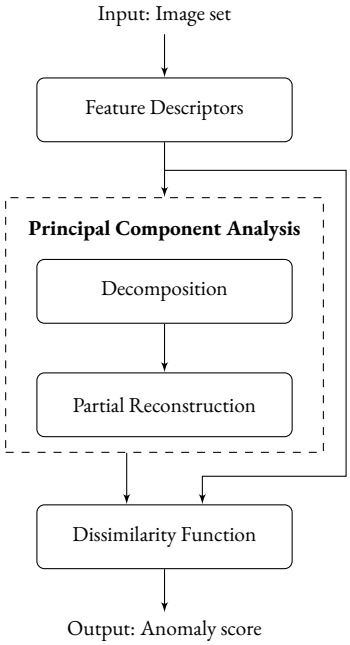


Figure 3.1: The proposed three-part anomaly detection framework.

3.1 Framework

We propose a simple three-part framework to detect local anomalies in aligned image sets and videos, as shown in figure 3.1 and described in more detail in algorithm 1. The three parts are: (i) feature descriptors, (ii) PCA decomposition and partial reconstruction, (iii) a dissimilarity function to compare the PCA reconstructed feature to the extracted features.

Algorithm 1: Anomaly Detection Framework

Input : Image set $\{\mathbf{I}_1, \ldots, \mathbf{I}_N\}$
Input : Feature descriptors $\mathbf{F} : \mathbf{I} \mapsto \mathbb{R}^d$
Input : Number of principal components to use in reconstruction: θ
Input : Dissimilarity function $\mathbf{D} : \left\{\mathbb{R}^d, \mathbb{R}^d\right\} \mapsto \mathbb{R}$
Initialise: Featurespace $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ with $\mathbf{x}_i \in \mathbb{R}^d \quad \forall i \in [1, N]$
Initialise: Featurespace $\mathbf{P} = \left\{\mathbf{p}_1, \ldots, \mathbf{p}_N\right\}$ with $\mathbf{p}_i \in \mathbb{R}^d \quad \forall i \in [1, N]$
Initialise: Featurespace $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N\}$ with $\hat{\mathbf{x}}_i \in \mathbb{R}^d \quad \forall i \in [1, N]$
1 $\mathbf{x}_i \leftarrow \mathbf{F}(\mathbf{I}_i) \quad \forall i \in [1, N]$ `// Extract per-image features`
2 $\mathbf{P} \leftarrow \mathrm{PCA}(\mathbf{X})$ `// Decompose X into PCs`
3 $[\mathbf{p}_i]_j \leftarrow 0 \quad \forall i \in [1, N] \quad \forall j \in (\theta, d]$ `// Discard low variance PCs`
4 $\hat{\mathbf{X}} \leftarrow \mathrm{PCA}^{-1}(\mathbf{P})$ `// Reconstruct to orig. space`
5 $A_i \leftarrow \mathbf{D}\left(\mathbf{x}_i, \hat{\mathbf{x}}_i\right) \quad \forall i \in [1, N]$ `// Calculate anomaly scores`
Output : Anomaly scores $\{A_1, \ldots, A_n\}$

3.1.1 PCA Decomposition and Partial Reconstruction

The core of this approach is PCA decomposition and partial reconstruction. The rationale is as follows: Common structure within the image set will account for a large amount of the variance present in the set. By decomposing the feature vectors into principal components and discarding components that represent less common variations before performing partial reconstruction, we are using PCA akin to a *trained image smoother*, which keeps common and discards uncommon structure.

This step requires a parameter θ, the number of principal components used for reconstruction. This parameter corresponds roughly to a bias/variance trade-off. A very high θ might mean the difference between original and reconstructed feature vectors mostly constitutes noise. A very low θ means the method relies more on low-order deviations

from the mean feature vector, and less on the specific deviations it might learn from the entire set. It is also possible to replace this abstract parameter θ with a more interpretable concept by choosing a percentage of explained variance that the model should learn, and setting θ to the lowest number of principal components that explain at least that amount of variance, or even a specific fraction of d.

3.1.2 Feature Descriptors

The choice of feature descriptor depends on the type of anomaly that has to be detected in the images. For example, to detect abnormal texture, we might use a feature that is known to work well in texture classification such as wavelet responses [1]. Or to detect motion in otherwise static camera images, we might calculate the difference between a frame and the previous frame at each position and use these as features. The simplest choice is an identity function, i.e. the features are the original pixel values in the image.

[1] UNSER, M. 1995. Texture classification and segmentation using wavelet frames. *IEEE Transactions on image processing 4*, 11, 1549–1560

The reason for using feature descriptors instead of simply the images themselves stems from the fact that PCA is a linear model, and the resulting principal components will be combined linearly to reconstruct each image. The problem is that images are not like typical feature vectors, in the sense that (for example) translating an image by a single pixel will result in an almost identical image to the human eye, but a very different feature vector. Moreover, images with texture may look similar to the human eye, but the pixel values are hardly comparable. Extracted features, unlike the images they were extracted from, may have invariances to transformations that makes them more suited to compare images of a certain type than the original pixel values would.

A feature can be used to describe an entire image, a specific location, or portions of an image, depending on the descriptor used. This determines how 'localised' the anomaly

detection is. For example, we might calculate a locally windowed greyscale histogram, resulting in as many feature vectors as we have windows for each image in the set. We might want to detect entire images as being anomalous, or we might want to focus on specific regions within the image. When using localised features, we have the option to either treat all resulting feature vectors as if they came from the different images (treating each window location as an image in itself) or perform the framework for each window location individually.

3.1.3 Dissimilarity Function

To determine whether something is or isn't an outlier, the decomposed and reconstructed feature vector is compared to the feature vector before decomposition by means of some dissimilarity function. This might be Euclidean distance, one minus a normalised Pearson correlation, or however the chosen feature descriptors are usually matched in other applications [2]. It can be any function $D(f_1, f_2)$ that compares two feature vectors f_1 and f_2, with the restrictions that $D(f, f) = 0$, the dissimilarity of any vector to itself is zero, $D \geq 0$ for all inputs, and the function is symmetric: $D(f_1, f_2) = D(f_2, f_1)$. Triangle inequality, $D(f_1, f_3) \leq D(f_1, f_2) + D(f_2, f_3)$, is a property that we might want a dissimilarity function to have, but is not required. If a dissimilarity function does satisfy triangle inequality, it may also be called a *distance function* or *metric*.

[2] It should be noted that PCA minimises the mean squared reconstruction error, so this is also minimised for the anomalies we want to detect.

We call the dissimilarity of the feature vector to its partial reconstruction the *anomaly score*. This anomaly score can then be thresholded to determine whether each feature vector represents an anomalous image or region.

Because the optimal value for thresholding will vary depending on feature descriptor, dissimilarity function and number of principal components used to reconstruct, we will evaluate an AUROC of a manually labeled test set to

Figure 3.2: A sample of each digit from the MNIST dataset.

Figure 3.3: The mean values (left) and first 9 principal components of the MNIST dataset. (Greyscale ranges have been rescaled for maximum visibility.)

assess the quality of this method. The ROC curve itself is also a useful chart to have, as in the case of defect detection for industrial processes (such as sewer inspections) we often have a higher tolerance for false positives than we do for false negatives.

3.2 Proof of Concept

To illustrate our method, we look at the MNIST reference dataset [3], consisting of 70,000 handwritten digits in greyscale images of dimensions $[28 \times 28]$, see figure 3.2 for some examples. We use the identity function as feature descriptor, so that the feature vector is identical to the pixel vector. This means our feature matrix is shaped $[70000 \times 784]$. When we apply PCA to the MNIST dataset, we obtain 784 principal components, which we can reshape into $[28 \times 28]$ images for visual inspection (also known as *eigenimages*), as shown in figure 3.3 for the first 9 principal components.

Now when we project an image onto the basis spanned by the principal components, we express the image as a linear combination of the eigenimages. Since the eigenimages are sorted in order of decreasing explained variance, an image that is similar to the images in the set (in this case also a handwritten digit, for example) is expected to have a larger (absolute) projected component (or eigenvalue) onto earlier

[3] LeCun, Y., Cortes, C., and Burges, C. J. 1998. The MNIST database of handwritten digits

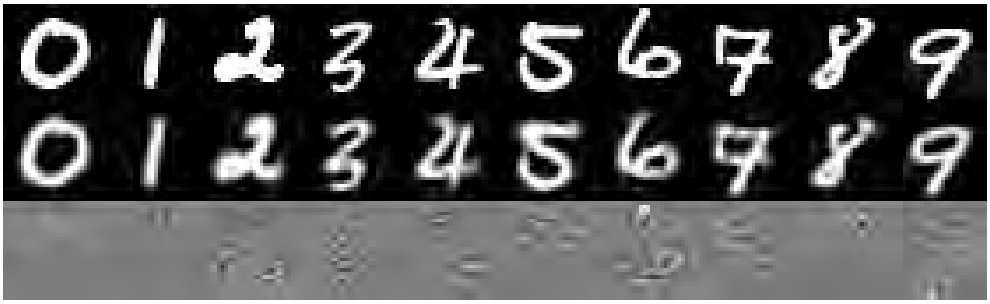


Figure 3.4: 10 sample digits from the MNIST dataset (top row) are reconstructed with the first 50 principal components (middle row) and the difference images between the original and the reconstructions (bottom row).

principal components, than onto later principal components.

The goal when PCA is employed is often dimensionality reduction: we project onto the first 2 or 3 principal components for inspection, or we use it to reduce the dimensionality by one or more orders of magnitude, while reducing the variance by only a fraction. (To illustrate: 90% of the variance in the MNIST dataset is in the first 87 principal components, a dimensionality reduction of about 89%.)

When we project an observation onto all principal components, we can perfectly recreate the original observation by inverting the projection matrix and adding the mean values, but we also know that principal components with lower eigenvalues are expected to be less important, because less of the variance present in the dataset is explained by these components. This leads to the following experiment: an observation is projected onto the first θ principal components, and this projection is augmented with zeroes for all less significant principal components we did not project onto. This augmented projection is then projected back. What we get is an approximation of the original observation, as can be seen in figure 3.4 for the MNIST dataset and $\theta = 50$ (a dimensionality reduction of over 95%).

As we can see, the approximations with only 50 principal components are very close to the original images. This is because the PCA was trained on these types of images, and

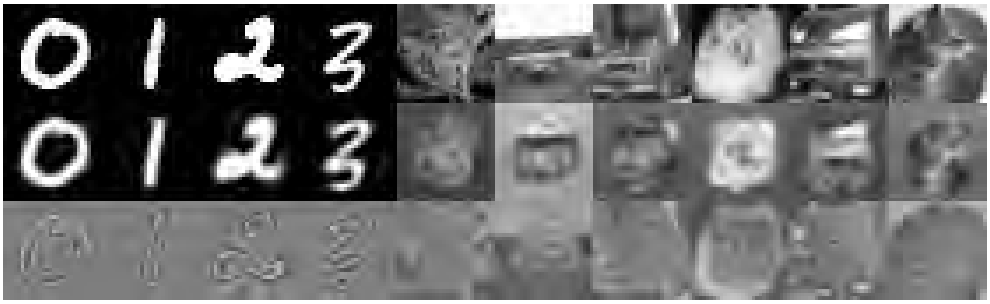


Figure 3.5: Sample images from the MNIST and CIFAR-10 datasets (top row) are reconstructed with the first 50 principal components after PCA was performed on 70,000 MNIST images and 1,000 CIFAR-10 images (middle row) and the difference images between the original and the reconstructions (bottom row). Below each difference image is the mean absolute value, which is used as the anomaly score.

[4] KRIZHEVSKY, A. AND HINTON, G. 2009. Learning multiple layers of features from tiny images. `https://www.cs.toronto.edu/~kriz/cifar.html`

[5] The images from CIFAR-10 are converted to greyscale and cropped to $[28 \times 28]$ pixels to conform to the images in the MNIST set.

the digits in the example are similar to the rest of the dataset.

Now what happens when our dataset contains anomalies? To illustrate, we add the first 1,000 images of the CIFAR-10 dataset of natural images [4] to the MNIST dataset [5]. These images are very different from the digits in the MNIST set, and since there are so few of them compared to the total size of the dataset, they can be considered anomalies. We perform PCA on the combined dataset and then recreate all images using only the first 50 principal components. We show the reconstruction of some sample images in figure 3.5.

It can be seen that the images from the CIFAR-10 set reconstruct poorly at the edges, which makes sense as 98.5% of the images are from the MNIST dataset, which does not contain any structure on the edges of the images. As a result, the difference images contain more structure at the edges and the CIFAR-10 images will be easier to distinguish from the MNIST images with our dissimilarity function.

As dissimilarity function, we take the mean absolute value of the pixels in the difference images, which gives us an anomaly score for each image in the set. This is going to be categorically higher for images from the CIFAR-10 dataset than images from the MNIST dataset (see for example the anomaly scores of the example images in figure 3.5). We can now predict which images are anomalies by thresholding

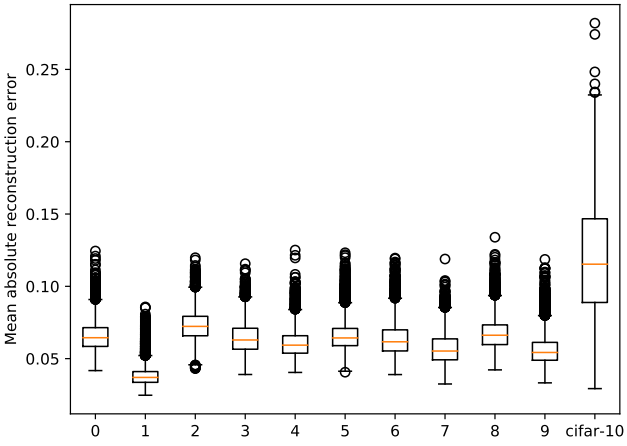


Figure 3.6: Comparison of the absolute reconstruction error of the 70,000 digits in the MNIST dataset and the first 1,000 images of the CIFAR-10 dataset, using the first 50 principal components to recreate the images.

the anomaly score. Figure 3.6 shows the spread of the reconstruction errors for different digits and images from the CIFAR-10 set. As can be seen, the error of the CIFAR-10 images tends to be significantly larger.

This illustrates the basic principle of the framework: the reconstruction error with a limited number of principal components can find anomalies in an image set of otherwise similar appearance. Although no feature descriptors were used for this simple example, the need for this will become clear in the next section.

3.3 Application in Sewer Pipe Images

Dutch urban drainage inspection company vandervalk+degroot has provided us with a dataset of images from a front-facing camera on a PIG (pipe inspection gadget), from ten different streets within different municipalities in the Netherlands. These images are already spatially aligned, as the inspector

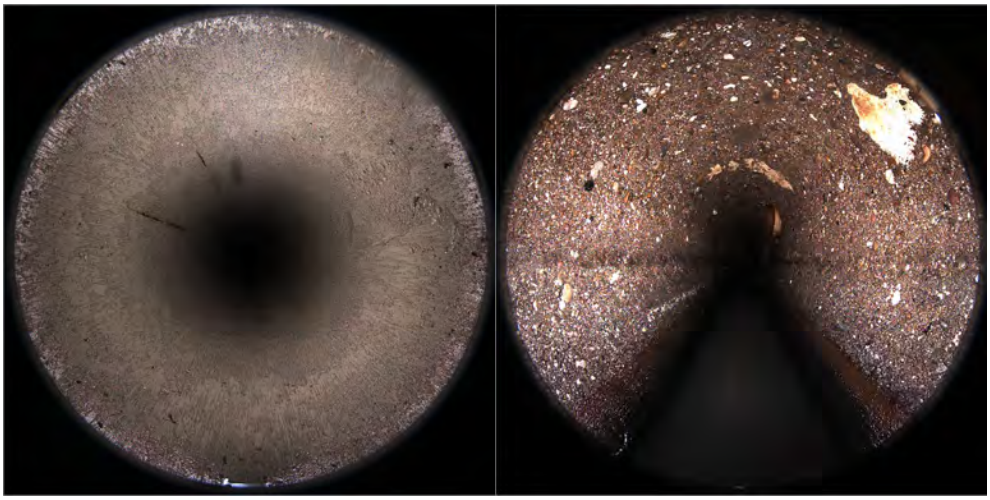


Figure 3.7: Sample images from the two labeled datasets: on the left the more smooth concrete pipe, on the right the more roughly textured granulate.

has aligned the camera to the centre of the pipe before starting the recording.

The two subsets correspond to two different types of pipe: (1) smooth concrete and (2) more rough and textured granulate, exposed over time. Figure 3.7 shows an example of each. Henceforth, we will refer to these two image sets as 'smooth' and 'coarse'. The image sets contain 684 and 698 images respectively. Each individual image is composed of $[1080 \times 1080]$ RGB pixels.

The images are processed by the framework on a per-street basis. The reason for this is that the material used varies for different municipalities and date of installation, as will the effects of age. When using images from a single street, we can be reasonably certain that all images in such a set are of similar manufacturing and age, which means that anomalies are more easily detected, because we do not have to account for a possible multimodal distribution in appearance.

The images are divided into 676 non-overlapping patches of $[40 \times 40]$ pixels, and we select 324 of such patches per image, corresponding to the regiosn of the images that are

in focus. Different patch locations are processed separately, this allows us to compare the portions of the images that are spatially aligned while high anomaly scores can still be pinpointed to a specific image patch, rather than an entire image.

The images are not accompanied by labels or annotations, so a method of verifying that the unsupervised method correctly finds anomalies is required. To this end, we selected two different subsets that are somewhat representative of all the sewer pipes from the different municipalities present in the datasets and hand-labeled 22 images from these sets. Each patch in the 22 validation images was labeled as 'anomaly' or 'normal', in the context of the rest of the pipe. This includes both actual defects, such as discolouration as a result of leakage, as well as physical features that are simply less common than others, such as pipe joints and refuse.

The images in the labeled subsets sets are divided into the same 324 patches as the labeled images, and for each patch location features are extracted and PCA is applied to the feature vectors at a specific location. This means the framework is applied 324 times and each patch location across the images is treated as a separate image set. We construct an ROC curve by thresholding the anomaly scores at various levels and obtaining true and false positive rates for our labeled validation images. We report the area under the ROC curve (AUROC) as a measure of how well the resulting anomaly score performs.

The parameter θ, the cutoff value for the number of principal components to use in reconstruction, was chosen to maximise the AUROC. In our experiments, we found that the optimal value for θ corresponds to approximately 99% explained variance for the smooth image set and 95% explained variance for the coarse image set.

The AUROC allows us to compare the performance of different methods regardless of what costs or restrictions we

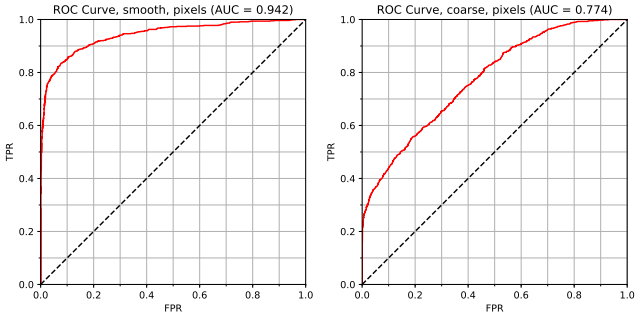


Figure 3.8: ROC curves we obtain from the anomaly detection framework on our manually labeled validation set, using pixels as features to be analysed by PCA. On the left the smooth dataset, on the right the coarse dataset.

assign to types of misclassification. It should also be noted that since this anomaly detection step might be followed by a classification into a taxonomy of defect classes, a high false positive rate might be salvaged by the later classification.

When using pixels as features and the mean absolute difference as a dissimilarity measure, we obtain results as shown in figure 3.8. The AUROC for the smooth set is 0.942, the AUROC for the coarse set is 0.774.

3.3.1 Feature Extraction

A possible reason that the framework performs less well on the coarse set when using pixels as features, is the texture present in the surface of the pipe in those images. The variance between pixel values is far greater than it is in the smooth set, where the entire pipe is more or less a single colour, and as a result the image are difficult to capture in a linear model such as PCA.

To alleviate this issue, we extract features that are more robust to textured images. The feature vectors are then decomposed, reconstructed and compared in the same way that the images would be, as shown in the framework in figure 3.1. In this section, we propose five higher-level features. An overview of each feature's invariances is given in table 3.1. The performances of each can be easily compared

Feature	Invariances
Pixel Values	None
Colour Histogram	Translation, rotation, scaling
Fourier Transform	Translation [6]
Histogram of Oriented Gradients	None
Local Binary Patterns	Translation, rotation
Homogeneous Texture Descriptor	Translation, rotation

Table 3.1: Overview of feature extractors invariances

[6] After discarding phase component

in table 3.2.

Colour Histograms

A simplistic but quite useful feature is a colour histogram of the pixel values. The 1600 values in each colour channel of a patch are binned into 20 equally sized bins per colour channel and concatenated to form a feature vector of length 60. These (in comparison) small vectors are decomposed into principal components and reconstructed with fewer than 60 principal components. The histogram is compared to the reconstructed histogram again by mean absolute difference. We see a slight improvement when using the histograms on the coarse set, an AUROC of 0.790, whereas performance on the smooth set is similar with an AUROC of 0.942.

Fourier Transform

We perform a two-dimensional Fourier transform on the $[40 \times 40]$ image patches, obtaining the frequency representation of the image patches. We discard the phase component by taking the absolute value and discard half the frequency plane because of symmetry. Again we decompose and try to reconstruct the feature vector, using the mean absolute difference as dissimilarity measure. The Fourier transform does not provide an improvement over using the pixel values, as we obtain an AUROC of 0.928 on the smooth set and 0.715 on the coarse set.

Histogram of Oriented Gradients

Often abbreviated as HOG, histograms of oriented gradi-

ents [7] describe an image by determining gradient directions at each pixel location, and binning these locally into histograms over a patch of specified size. It seems that this feature does not suit our purpose too well, as the AUROC for the smooth set becomes 0.886 and for the coarse set becomes 0.588. This might be explained by the fact that this feature is meant for object detection, and our image patches contain mostly texture.

[7] DALAL, N. AND TRIGGS, B. 2005. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE, 886–893

LOCAL BINARY PATTERNS

Local binary patterns are a feature used to describe points as being edges or corners [8]. Each pixel is compared to its neighbouring n pixels (usually $n = 8$) and for each of these neighbours, it assigns a 1 or 0 depending on whether the pixel has a higher greyscale value than that particular neighbour. The resulting 8-bit numbers are locally binned to summarise the texture of a cell as containing corners, edges, or otherwise. The concatenated histograms are used as a feature vector. We obtain AUROCs of 0.865 for the smooth set and 0.705 for the coarse set.

[8] OJALA, T., PIETIKÄINEN, M., AND HARWOOD, D. 1996. A comparative study of texture measures with classification based on featured distributions. *Pattern recognition 29,* 1, 51–59

HOMOGENEOUS TEXTURE DESCRIPTOR

Part of the MPEG-7 multimedia description standard, homogeneous texture descriptors are shown to perform well on image retrieval tasks, especially for images with much texture [9]. The HTD features are comprised of logarithmically scaled mean values and standard deviations of Gabor wavelet responses. We obtain AUROCs of 0.941 for the smooth set and 0.785 for the coarse set.

[9] RO, Y. M., KIM, M., KANG, H. K., MANJUNATH, B., AND KIM, J. 2001. MPEG-7 homogeneous texture descriptor. *ETRI journal 23,* 2, 41–51

3.3.2 CONCATENATING FEATURE VECTORS

One of the strengths of the framework is that we can concatenate multiple feature vectors and the framework will

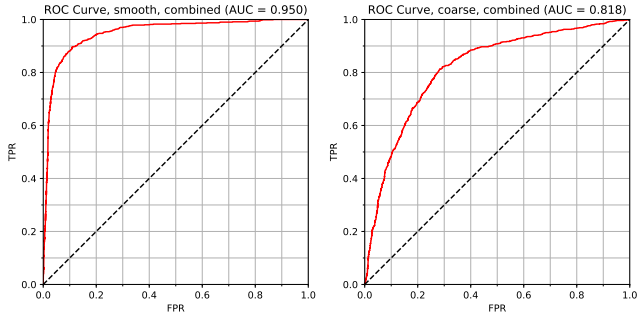


Figure 3.9: ROC curves from the anomaly detection framework on the validation set, using both pixel values and the high-level features described in this section (except for HOG and Fourier transform) combined as features to be analysed by PCA.

still function identically. This allows us to combine the strengths of multiple feature types, and even combine these with the raw pixel values if we wish to do so.

After examining every possible permutation of the features previously described, we found that excluding only the HOG and Fourier transform from the feature vector gave the best result on both image sets. Figure 3.9 shows the resulting ROC curves when we use the other high-level features described in this section, as well as the raw pixel values, giving us the highest AUROCs so far, 0.950 for the smooth set and 0.818 for the coarse set. The ROC curves are shown in figure 3.9.

Leaving out the HOG features seems reasonable, as these performed worse than most other features individually. As both PCA and the Fourier transform are linear operations, performing PCA on the Fourier transform would provide identical results to performing PCA on the pixels (excluding an arbitrary phase shift). We discarded the phase component of the Fourier transform before performing PCA, so the result is not identical, but this might explain why including it when already using the pixel values does not improve the AUROC.

3.4 Convolutional Autoencoder

Principal component analysis is not the only available method for this task. We compare the performance of this method when using a convolutional autoencoder as a drop-in replacement.

An autoencoder is a neural network that tries to learn the identity function [10], and a convolutional autoencoder combines this with image filter learning [11]. Analogous to our framework, this means we can learn the feature representation, perform non-linear dimensionality reduction (replacing the PCA) and reconstruct the input images. As we train this network on an image set, we should be similarly able to use it to detect anomalous regions by inspecting the difference image.

We designed a convolutional autoencoder consisting of:

- Input layer: [1040 × 1040] resolution
- Convolutional layer 1: 10 [20 × 20] filters, stride [10 × 10]
- Pooling layer 1: [2 × 2] max pooling, stride [2 × 2]
- Convolutional layer 2: 10 [20 × 20] filters, stride [10 × 10]
- Pooling layer 2: [2 × 2] max pooling, stride [2 × 2]
- Autoencoder: 1690 → 845 → 422 → 845 → 1690 units
- Unpooling layer 1: uniform, [2 × 2]
- Deconvolutional layer 1: Weights shared Conv. layer 2
- Unpooling layer 2: uniform, [2 × 2]
- Deconvolutional layer 2: Weights shared Conv. layer 1
- Output layer: [1040 × 1040] resolution

[10] Baldi, P. and Hornik, K. 1989. Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks 2*, 1, 53–58

[11] Chen, M., Shi, X., Zhang, Y., Wu, D., and Guizani, M. 2017. Deep features learning for medical image analysis with convolutional autoencoder neural network. *IEEE Transactions on Big Data*

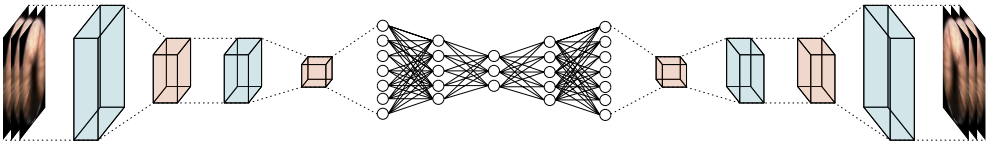


Figure 3.10: Schematic overview of the convolutional autoencoder used. Convolutional and deconvolutional layers are shown in blue, pooling and unpooling layers are shown in orange.

A schematic overview of the network can be found in figure 3.10.

Using this network, trained on the same image sets, we obtained the following results: an AUROC of 0.946 on the smooth set and 0.714 on the coarse set, figure 3.11 shows the ROC curves. The results on the smooth set are rather similar to those obtained by the PCA framework, the AUROC results on the coarse set are noticeably worse, as can be seen when comparing with the PCA-based method in table 3.2.

Still, urban drainage inspections might be an application where the convolutional autoencoder could outperform the PCA-based method, when we cannot afford to miss any potential defects. We can see from comparing the ROC curves that the convolutional autoencoder reaches a true positive rate of 1.0 at a lower false positive rate than the PCA-based method. Overall performance is still expected to be worse, as indicated by the AUROC.

We expect that the reason for this reduced performance is the reconstruction of the full images. In the PCA framework, we are extracting features, decomposing and reconstruction these features, and comparing the reconstruction to the *extracted features*. In the convolutional autoencoder, we try to reconstruct the image itself out of necessity, as we do not know what the features should be. But this means that the reconstructed images are compared to the original images, instead of the reconstructed features to the original features.

The fact that the convolutional autoencoder has to reconstruct the original image, means it can't learn features we might describe as 'texture descriptors,' as these are inher-

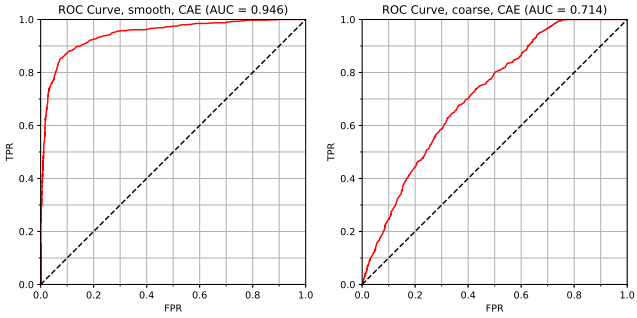


Figure 3.11: ROC curves from convolutional autoencoder.

Table 3.2: Results for the methods and datasets described in this work.

	AUROC	
Feature type	smooth	coarse
Pixels	0.942	0.774
Colour Histogram	0.942	0.790
Fourier Transform	0.928	0.715
Histogram of Oriented Gradients	0.886	0.588
Local Binary Patterns	0.865	0.705
Homogeneous Texture Descriptor	0.941	0.785
Pixels + Histogram + LBP + HTD	**0.950**	**0.818**
Convolutional Autoencoder	0.946	0.714

ently rotation and translation independent, so reconstructing the original pixel values from such features would be impossible for patches containing a lot of texture. But these are the types of features we expect (and confirmed for the PCA-based approach) to perform well, so the comparison is not entirely fair.

To make the systems more similar, we could try to discard the unpooling and deconvolutional layers, and compare the output of the fully connected autoencoder to the input of the fully connected autoencoder (after the network was trained *with* the unpooling and deconvolutional layers), but this is beyond the scope of our current research.

It should also be noted that the network's many hyperparameters are more difficult to optimise than the singular parameter θ our framework relies on, and a network better optimised for this specific task may perform better [12].

[12] SUN, Y., XUE, B., ZHANG, M., AND YEN, G. G. 2018. An experimental study on hyper-parameter optimization for stacked auto-encoders. In *2018 IEEE Congress on Evolutionary Computation (CEC)*. 1–8

3.5 Summary

We have proposed a framework for unsupervised anomaly detection in aligned image sets, relying on feature extraction, PCA decomposition and partial reconstruction, and classification of the reconstruction error, and tested this framework on sewer pipe images. Table 3.2 summarises the results obtained by the different feature types. We see that while raw pixel values perform quite well on the 'smooth' dataset, improvement can be made by combining different feature descriptors. For the 'coarse' dataset, the difference is larger: drastic improvements are made by combining features, as is expected when we consider that the images are more defined by texture than by individual pixel values.

We conclude that our PCA-based approach, which could be considered a more 'traditional' statistical approach to computer vision using combinations of hand-crafted features, outperforms the more 'modern' convolutional autoencoder in this setting, but we must also admit that the comparison is not entirely fair as we are in one case reconstructing high-level features and in the other case pixel values.

4 Convolutional Neural Network Classification

The process of CCTV sewer pipe inspections is both labour-intensive and error-prone. Other researchers have suggested machine learning techniques to (partially) automate the human review of this footage, but the automated classifiers are often validated in artifial testing setups, leading to biased results that do not translate well to practice.

In this chapter, we design a convolutional neural network (CNN) and apply this validation methodology to automatically detect the twelve most common defect types in a dataset of over 2 million CCTV images. We also discuss suitable evaluation metrics for this specific classification task — most notably 'specificity at sensitivity' and 'precision at recall' — and the importance of using a validation setup that includes a realistic ratio of images with defects to images without defects, and a sufficiently large dataset. We also introduce *'leave-two-inspections-out'* cross validation, designed to eliminate a data leakage bias that would otherwise cause an overestimation of classifier performance.

With this dataset and our validation methodology, our CNN outperforms the state-of-the-art. Classification performance was highest for intruding and defective connections and lowest for porous pipes. While the CNN is not capable of fully automated classification at sufficient performance levels, we determined that if we augment the human operator with the CNN, this may reduce the required human labour by up to 60.5%.

4.1 Introduction

In this chapter a possible method to automate the inspection process is demonstrated and shown to be viable. While the performance of the method is noteworthy, we consider the most important contribution of this chapter not to be this method itself, but rather the methodology used to validate these results and assess their impact if used in practice.

4.1.1 Image Classification

Image classification is the primary way in which we attempt to address the automation of the inspection process. This classification assumes that we have *training data*, consisting of a set of images of CCTV footage, each of which has an assigned *label*, which indicates whether specific types of defects are present and visible in the image. The classifier infers a statistical relation between the images and the labels, which allows it to make predictions about the labels of images that we do not know the true labels for, such as recently recorded images that still require assessment.

Traditionally, the automated classification of images is done with extracted image features [1], which are known to capture information that is less visible in raw pixel values. Recently, this approach has been mostly replaced by convolutional neural networks [2] (CNNs, explained in more detail in section 2.3). CNNs employ *end-to-end* learning: the original pixel values are used as inputs, and the CNN learns the feature extractions as well as how these features relate to the labels. This allows for extracted image features that are more specialised to the classification task. There is one main downside to this approach: there are a lot more parameters to fit, as the extracted features also need to be inferred from the image data. Two resulting limitations are that a lot more data is required to fit all these parameters,

[1] Szeliski, R. 2010. *Computer Vision: Algorithms and Applications*, 1st ed. Springer-Verlag, Berlin, Heidelberg

[2] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. S., Berg, A. C., and Li, F. 2014. Imagenet large scale visual recognition challenge. *CoRR abs/1409.0575*

and hyperparameter optimization becomes more difficult as the hyperparameter search space (how many filters and of what shape) increases drastically compared to traditional methods.

The impact of the data availability problem can be lessened with *transfer learning*, by using a network that has been pre-trained on a different set of images [3], but then we may also reduce the benefit that the CNN may have in training the convolutional filters specifically to the data and the task at hand. Still, this approach is often favored over a random initialization of the network parameters to save time [4].

[3] Hoo-Chang, S., Roth, H. R., Gao, M., Lu, L., Xu, Z., Nogues, I., Yao, J., Mollura, D., and Summers, R. M. 2016. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging 35*, 5, 1285

[4] Oquab, M., Bottou, L., Laptev, I., and Sivic, J. 2014. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1717–1724

4.1.2 Classification Result Validation

To assess the performance of a trained classifier, we need a test set that is independent of the training set. To use (part of) the same training set as the test set introduces a bias, which means we are not measuring how well the classifier performs, but only how well it can recognise before-seen data. Since two independent data sets may be difficult to come by, often a portion of the training set is set apart to be used as the test set. [5] The training and test set are not independent in such a scenario and likely contain the same sampling bias, but it is often the best we can do.

To assess the performance accurately, some variance in the samples in the test set is required, which means many samples are required, and a significant portion of the training set may have to be set apart. A significant reduction in size of the training set could itself impact the performance negatively, leading us to underestimate the actual performance of the classifier due to lack of training data. An often used technique to circumvent this problem is *k*-fold cross validation, as outlined in section 2.1.3.

[5] Bishop, C. M. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg

Besides a test set, the performance metrics have to be defined. The most common performance metric used for classification is the *accuracy*, the percentage of correctly classified samples. However, the performance metric should be chosen based on the task at hand, and accuracy is not a good choice for unbalanced classification problems, such as this particular problem, as it favors correct classification of the majority class. [6] Most performance metrics can be thought of as some function of the false positive rate (FPR) and the false negative rate (FNR). A classifier can often be tuned after it has been trained, making it essentially a family of classifiers. In such cases the performance may change as a function of this tuning, and it can be worthwhile to use performance metrics that are independent of which member of the family of classifiers is used. Examples of such metrics are the receiver operating characteristic, or the Pareto-boundary of any combination of metrics [7].

[6] further discussion on this can be found in section 2.1.4

[7] Bishop, C. M. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg

4.1.3 Related Work

Researchers have already applied machine learning techniques to the task of automating sewer inspections. But realistic validation of such methods is often of less note in such articles. As actual defect rates are often very low, in the order of magnitude of 1% of images captured by CCTV — in our dataset we found 0.8% images with defects — it is curious that many authors test their methods on artificial test sets that contain 50% defects. We feel that such a result might be interesting in a vacuum, but gives no indication of the actual 'real-world performance' of a classifier. A relevant selection of research is discussed in this section.

Chae and Abraham [8] use a (non-convolutional) neural network to learn various attributes in relation to the existence and severity of cracks from images of the inner surface of sewer pipes. Their neural network is trained on 20 images

[8] Chae, M. J. and Abraham, D. M. 2001. Neuro-fuzzy approaches for sanitary sewer pipeline condition assessment. *Journal of Computing in Civil engineering 15*, 1, 4–14

and tested on 13 images, so the actual applicability remains unclear.

Yang and Su [9] compare two SVM approaches and a neural network, trained on wavelet filter responses of images. The classifiers were only applied to images containing defects, and subsequently used to classify *what* defect was present in the image. This means no information is available regarding the false detections in images without defects.

Guo et al. [10] use image registration (alignment of pixel locations) and the absolute pixelwise difference between images to classify image regions as defective or healthy. The method is tested on a dataset consisting of 51 images of defective pipes and 52 images of healthy pipes, with reported accuracy and false alarm rates.

Halfawy and Hengmeechai [11] present an algorithm for crack detection in CCTV inspections, based on a Sobel filter and morphological operations. As the model is partially based on expert knowledge, it does not require a large dataset to train, and it was tested on a dataset with 50 images containing cracks and 50 images not containing cracks.

Halfawy and Hengmeechai [12] improve on their previous work, now training an SVM with varying kernels with HOG features extracted from CCTV images, and report more meaningful performance metrics such as precision and AUROC. The experiments are still performed on a test set that consists of 50% images with defects, so it still tells us very little about real-world performance.

Kumar et al. [13] are one of the first to use convolutional neural networks to exploit end-to-end learning in sewer CCTV defect detection. They focus on three different defect types and train the network three times, once for each defect. They also report the precision as one of their performance metrics and use a training set consisting of 12,000 images, but their test sets also consist of 50% images with defects, again limiting their obtained results to such an artificial scenario. In our work, we reimplemented their sug-

[9] Yang, M.-D. and Su, T.-C. 2008. Automated diagnosis of sewer pipe defects based on machine learning approaches. *Expert Systems with Applications 35,* 3, 1327–1337

[10] Guo, W., Soibelman, L., and Garrett Jr, J. 2009. Automated defect detection for sewer pipeline inspection and condition assessment. *Automation in Construction 18,* 5, 587–596

[11] Halfawy, M. R. and Hengmeechai, J. 2013. Efficient algorithm for crack detection in sewer images from closed-circuit television inspections. *Journal of Infrastructure Systems 20,* 2, 04013014

[12] Halfawy, M. R. and Hengmeechai, J. 2014. Automated defect detection in sewer closed circuit television images using histograms of oriented gradients and support vector machine. *Automation in Construction 38*, 1–13

[13] Kumar, S. S., Abraham, D. M., Jahanshahi, M. R., Iseley, T., and Starr, J. 2018. Automated defect classification in sewer closed circuit television inspections using deep convolutional neural networks. *Automation in Construction 91*, 273–283

gested convolutional neural network and performed tests on our dataset, which more accurately represents a real-world scenario.

Myrans et al. [14] train an SVM and a random forest on extracted GIST features from CCTV images. They use 25-fold cross validation and provide the ROC curve along with the misclassification rates for various defect types, but unfortunately work with a dataset that consists of approximately 37% images with defects, which is not representative of a realistic scenario.

[14] Myrans, J., Everson, R., and Kapelan, Z. 2018. Automated detection of faults in sewers using cctv image sequences. *Automation in Construction 95*, 64–71

In later work, Myrans et al. [15] combine both the SVM and the random forest on a dataset in which 'approximately half' the images contained defects, and obtain results superior to either individual classifier. Again, unfortunately the validation results are not representative of a real-world scenario because of the high prevalence of defects in the test set.

[15] Myrans, J., Kapelan, Z., and Everson, R. 2018a. Combining classifiers to detect faults in wastewater networks. *Water Science and Technology 77*, 9, 2184–2189

4.2 Data Exploration

A dataset has been kindly provided to us by Dutch sewer inspection company *vandervalk+degroot*. The data has two components: the images themselves, and the accompanying inspection reports. The data encompasses 30 inspections from 11 Dutch municipalities, for a total of 2,202,582 images from 3,350 different concrete pipes ranging in diameter between 300 mm and 1000 mm.

4.2.1 Image data

The images have been collected with the RapidView IBAK Panoramo® pipeline inspection system [16]. While the Panoramo software can be used to inspect the pipe in a virtual 3D environment, for this study we merely used the 2D images used

[16] IBAK Helmut Hunger GmbH & Co. KG. 2015. Panoramo® 3d optical pipeline scanner. `http://www.rapidview.com/panoramo_pipeline.html`. Accessed: 2018-12-05

to create these 3D environments as input. The Panoramo system does not record video, but still images with a strobe light, spaced 50 mm apart. This allows for improved image quality, without the need to stop the inspection vehicle from moving. The system is equipped with a front-facing and back-facing camera, each with a 185° wide angle lens. The images from the back-facing camera are slightly occluded by parts of the inspection vehicle and the chain that lowered it into the pipe, so our dataset contains only the images from the front-facing camera.

The images are in 24-bit RGB format, 1040×1040 pixels, JPEG images. No information was given about the compression level, but the images range from 19 KiB to 447 KiB. Four randomly selected sample images are shown in Figure 4.1.

An important feature of the images recorded by the Panoramo system is that the images are spatially aligned. After the device is lowered into the sewer pipe, the operator aligns the camera with the centre of the pipe before starting the recording. This allows the Panoramo software to stitch the images together into a three-dimensional, virtual environment, but it also allows us to consider the images to be of the same modality, allowing a 1-on-1 comparison between two images.

As the images from the Panoramo system are meant for offline processing, the operator does not pan, rotate, or zoom the camera during the recording, as they might with other CCTV feeds. This is a very important distinction, because being able to automatically classify images where a human operator has already isolated, centred, and zoomed in on the defects, as is apparently the case in some previous studies, does not achieve much in terms of "automating classification". To take steps towards fully automated inspection, we should aim to classify images that were recorded without human intervention, other than starting the system.

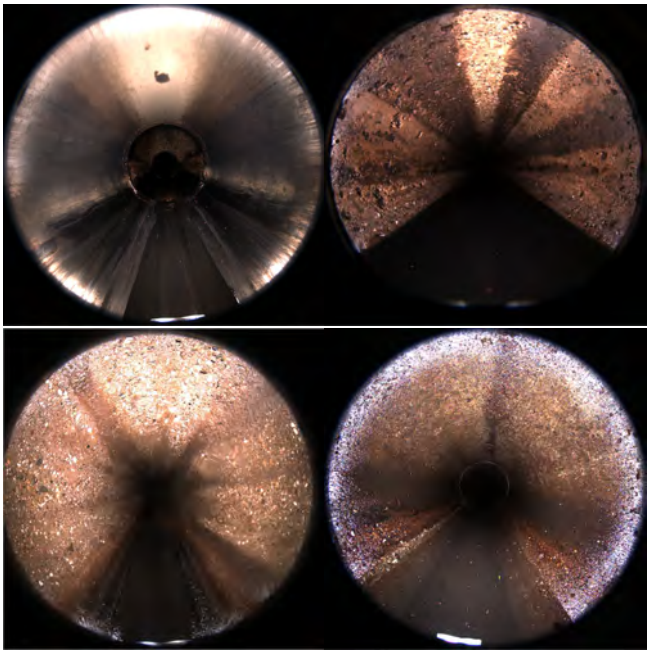


Figure 4.1: Randomly selected sample images from the dataset.

4.2.2 Inspection Reports

Besides the images, each of the 30 inspections is accompanied by an inspection report, containing all points of interest along the pipes referenced according to the European standard coding norm EN 13508-2 [17], as annotated from visual review by human operators. An example of an entry from the tabular datafile that generated this report could be

```
38.40m  BBAC2  @Blick=38.38;91;72;90;0;
```

This indicates that at 38.40 meters from the start of the pipe, a defect was found with main code `BBA` (roots), characterization `C` (complex mass of roots), and quantification `2` (pipe diameter reduced by $\leq 10\%$)

From these entries, we can assign contextual labels to the images. We have selected the twelve most commonly occurring defects (that are not simply expected landmarks

[17] European Committee for Standardization. 2003. En 13508-2: Condition of drain and sewer systems outside buildings, part 2: Visual inspection coding system, european norms

Table 4.1: Defect types and occurrences

Defect Type \ Total:	Pipes 3,350	Images 2,202,582
Fissure	586	1,442
Surface Damage	1,242	2,507
Intruding Connection	375	1,004
Defective Connection	506	838
Intruding Sealing Material	74	173
Displaced Joint	1,509	4,988
Porous Pipe	117	187
Roots	273	629
Attached Deposits	183	338
Settled Deposits	164	219
Ingress of Soil	536	1,249
Infiltration	1,353	7,565

such as pipe joints, an overview is shown in table 4.1) and matched these to specific images, using the location of the entries. Because we know the Panoramo system's images are spaced exactly 50 mm apart, it is a relatively simple task to determine which entries in the report should be visible in each image. It is important to note that the best performance we can reasonably expect to achieve on such a dataset, is to label the images as well as (and no better than) a human operator would.

The `@Blick` entry is added by the Panoramo software and can be used to recreate the exact view in the virtual environment the operator was looking at when this defect was recorded.[18] In this research, the `@Blick` entry was not used.

In the end, we have a set of roughly 2.2 million images, and for each image a list of twelve Boolean values, telling us whether or not specific defects are present in the image. Table 4.1 gives an impression of how common these defects are in the dataset. In the next section, we will go into detail on how this data is modelled.

[18] These parameters are: the location along the pipe wall, the azimuthal angle, the polar angle, the field of view angle, and the rotation of the virtual camera with respect to the water level.

4.3 Methodology

The classifier used in this research is a convolutional neural network, and the model is approximated through backpropagation. This processs is explained in more detail in section 2.3.

4.3.1 Loss Function for Multi-Label Classification

In a standard classification setting, we differentiate between different classes. Each entry in the dataset is assigned to a single class. In the case of defect detection in sewers, this leads to a problem: several defects often co-occur. Infiltration, for example, almost always has a cause that is defined as a separate defect, such as a fissure. This co-occurrence can be a result of the definitions used in the EN13508–2 guidelines [19], or it might be an effect of cascading failures [20]. In our dataset there are 17,662 out of 2,202,582 images (0.802%) that contain defects, totalling 21,139 different defects, but 6,494 of these (30.7%) co-occur with another defect in the same image. When considering entire pipes, 2,512 out of 3,350 pipes (75.0%) contain defects, 6,918 defects are found in total, and 6,171 (89.2%) of these defect types are found co-occurring with other defect types in the same pipe.

As a result of this multi-label problem,[21] we have decided to label the images with a Boolean vector, each consisting of twelve Boolean values, representing the presence or absence of a particular defect. This means that images that do not contain a defect at all will have a vector of all negatives.

Not all misclassifications should be treated equally. If we correctly classify the presence or absence of eleven defects, but misclassify the presence or absence of the last defect, this is less severe than misclassifying multiple defects.

[19] European Committee for Standardization. 2003. En 13508-2: Condition of drain and sewer systems outside buildings, part 2: Visual inspection coding system, european norms

[20] Sitzenfrei, R., Mair, M., Möderl, M., and Rauch, W. 2011. Cascade vulnerability for risk analysis of water infrastructure. *Water Science and Technology 64,* 9, 1885–1891

[21] We distinguish *multi-label* classification (multiple classes per object), as opposed to *multi-class* classification, which might also refer to a non-binary classification case, i.e. an object has a single class, but there are more than two classes.

For each of the twelve defects we calculate an individual loss function, namely the cross entropy [22] between the actual value for a defect, y_c (0 for absence, 1 for presence), and the predicted value output by the network for that defect, $\hat{y}_c$ (a real value in the interval $[0, 1]$):

[22] SHORE, J. AND JOHNSON, R. 1980. Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. *IEEE Transactions on information theory 26*, 1, 26–37

$$L(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{c=1}^{12} y_c \log \hat{y}_c \tag{4.1}$$

As written, only false negatives contribute to the cross entropy loss, as y_c is zero for false positives. This means that we penalise the classifier for not detecting a defect, but not for seeing a defect where there is none. To make sure that the network does not simply output 1 for all defects, $\hat{y}$ is commonly normalised so that $\sum_c \hat{y}_c = 1$, which is called *soft-max* normalization. [23] Alternatively, it is also possible to account for false positives by adding contributions both for y_c and its complement:

[23] GOODFELLOW, I., BENGIO, Y., AND COURVILLE, A. 2016. *Deep learning*. Vol. 1. MIT press Cambridge

$$L(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{c=1}^{12} y_c \log \hat{y}_c + (1 - y_c) \log(1 - \hat{y}_c) \tag{4.2}$$

This is what we will use, as normalizing $\hat{y}$ does not make much sense when we expect defects to co-occur.

4.3.2 Class Imbalance and Oversampling

Our dataset consists of 3,350 pipes with a total of 2,202,582 images. While every pipe contains at least one defect of some type in one of its images, only 17,663 images, [24] roughly 0.8% of all images contain one or more defects. It should also be noted that the percentage of pipes that contain a specific defect is not the same as the percentage of images

[24] This number is not equal to the sum of the numbers in the rightmost column of table 4.1 because defects often co-occur in the same image, and some images are counted multiple times if we sum the column.

that contain that defect, as a pipe is said to contain a defect if at least one image from that pipe contains the defect.

The extreme class imbalance of images with and without defects in our dataset means that if we train a classifier without accounting for the imbalance, it will err on the side of false negatives, as these are simply more likely. If we make some number of misclassifications, we expect these to be distributed the same as the prior probabilities of the classes, simply put: our set has about 1% defects and 99% non-defects, so of the errors, that a naive classifier will make, 1% will be a false positive and 99% will be a false negative. It can be safely assumed that a false negative is more costly than a false positive (the latter costs labour hours, the former might pose a health hazard or incur additional costs e.g. through property damage or disruption of traffic). This has some important implications for the quality assessment of a classifier as well, which have been discussed in section 2.1.4.

As noted in section 4.1.3, a lot of previous work completely disregards the class imbalance when training and testing their classifier, and instead opts for a more manageable 50% split. This approach has a major issue: the test results are not representative of a real-world scenario, and only indicative of the quality of the classifier in a general case, not for this specific classification scenario. Instead, we require the test set to have a realistic ratio of images with defects to images without defects, as this means our results translate more directly to the results we would obtain when applying our classifier on newly obtained data.

While the area-under-the-curve for an ROC-curve or a PR-curve provide a metric independent of hyperparameter selection, they still take all levels of recall into account, whereas we are likely interested only in higher levels of recall, as we have assumed that a false negative is far more costly than a false positive.

To this end, we introduce two more metrics: *specificity at recall* and *precision at recall*. Neither of these metrics

require us to manually choose a value for τ [25]. Instead they dictate τ to be chosen such that recall is at a certain level, and report the specificity (TNR) or precision at this τ. This is the same as taking a point on the ROC and PR curves that corresponds to a particular value on the recall axis, and reading the point it corresponds to on the other axis.

[25] as defined in equation (2.19)

We feel that especially the specificity at recall and precision at recall metrics are useful to put the results into real-world context: for public health reasons we might be restricted to a minimum value for recall (as a lower value would allow too many defects to slip by unnoticed and increase the risk), and we simply want to know how efficient the system is *at least* at that level. For both of these metrics, we evaluate at the recall levels $\{0.90; 0.95; 0.99\}$, as we are mostly interested in high recall. An overview of the performance metrics we are using is given in table 4.2.

4.4 Aggregating performance on pipe level

The previous section outlined performance metrics for classifying single images, but it is not an uncommon scenario to classify entire pipes as a whole for a certain defect, as the

Metric	Description
AUROC	Area under the ROC curve
AUPR	Area under the PR curve
Specificity at Recall	Percentage of non-defects detected as defects when we require a minimum percentage of defects to be detected
Precision at Recall	Percentage of detected defects that are actually non-defects when we require a minimum percentage of defects to be detected

Table 4.2: An overview of the performance metrics used

decision to intervene with repair or replacement is made on a larger scale. To achieve this, we aggregate the real and predicted labels on the images with some *aggregation rule*, and calculate the same metrics from table 4.2 on the aggregated labels.

An obvious choice for an aggregation rule is the maximum: This would be analogous to determining whether any of the images is labeled as a defect, compared to whether any of the images actually contains a defect. Importantly, this aggregation rule does not depend on the size of the pipe, like the average value would. A downside to this rule is that we might actually be detecting a defect in an image where there is none and missing a defect that is in another image, but we still count this as a true positive, because we only care to know if we found the defect in the correct pipe.

Maximum aggregation performance metrics on pipe level will be reported alongside performance metrics for single images.

4.5 Leave-two-inspections-out Cross Validation

To accurately assess performance of a classifier on a dataset, we might use k-fold cross validation [26], as outlined in Section 2.1.3. The folds are often divided either randomly or *stratified*, meaning that the classes are divided as equally as possible among the folds. Because of how our dataset was sampled, we expect a large overlap in construction material and age within an inspection, which is often performed within a single geographical neighbourhood. In this case a random or stratified split might lead to *data leakage*, information from outside the training set being implicitly part of

[26] Bishop, C. M. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg

the training set: two points in a single pipe might exhibit the same defect, as they are subject to the same conditions, are of the same build material, and have the same age. But these factors also mean that the pipes themselves might appear very similar. As a result, it might be that our classifier is simply classifying the appearance of a pipe, and not the defects themselves. If we then use random or stratified splitting, we might overestimate the actual performance.

Instead, we introduce *leave-two-inspections-out* cross validation. This is inspired by *leave-one-subject-out* cross validation, used in the medical field. Since the data is already categorised into 30 inspections, we use these same inspections as folds for cross validation. We take 28 folds as the training set, 1 fold as the testing set, and 1 fold as a validation set, to prevent overfitting on the training set. These folds are rotated 30 times, until each fold has been used as the training set once, and we have a prediction for each image. This provides a more realistic scenario, where the classifier would be used to predict the presence of defects in a pipe it has never seen before.

A possible downside of this method is that for any given fold, we might not have every defect present in both the test and validation sets. Since there is no defect that appears in fewer than three inspections, at the very least every training set will contain every defect.

4.5.1 Overfitting

Overfitting is what happens when a model is trained on the training set so well that it loses generalisability on other datasets. All data that has been sampled from real world measurements (such as photographs in our case) is expected to have some amount of noise in it. This means that any model that can describe this data to a 100% accuracy has incorporated this noise in its model. The model's performance on a different dataset (with different noise, perhaps

from different measuring instruments) will be worse than that of a model which has learnt to model the data, but not the included noise.

The risk of overfitting is exacerbated when the noise in the training set is systemic, for example through a sampling bias, as this becomes another pattern the model might detect and learn, when it is in fact noise that will not be present in future datasets. Neural networks are also more prone to overfitting than a lot of other models, because of the large number of parameters that are subject to change when learning from the training data. [27] To prevent overfitting, we employ two methods: the use of a validation set and dropout.

[27] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. 2012. Improving neural networks by preventing co-adaptation of feature detectors

The use of a validation set is the more general approach of the two. Instead of training on all the data in the training set, we keep a subset of the training set apart, which is not used to train on. Periodically during the training phase, we calculate the loss function on this validation set. At some point the classifier will start overfitting, meaning the loss function on the training set will keep decreasing, but the loss function on the validation will either stagnate or start increasing. At this point we choose to stop the training and take the classifier as trained up to that point as the final classifier. We have chosen to calculate the loss on the validation set after every epoch and stop early if the loss on the validation increased significantly, or hasn't decreased for several epochs in a row.

Dropout is another way to prevent overfitting specifically for neural networks. [28] The idea is that to prevent a network that is too specifically catered to the input data, we should assure some stability with regards to small changes in the network structure. If the correct classification of a sample depends on a single specific path through the network, that classification would not be stable, as only one of the neurons in that path has to change some weights for the classification result to change. To force the network to not rely on a single path through the network, we randomly

[28] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research 15*, 1, 1929–1958

disable neurons in the dense layers during the training step, setting their output to zero. This forces the network to create a path to the correct classification result with the still enabled neurons. Since a different set of neurons will be deactivated for every batch of data, this increases the overall stability of the network by ensuring the correct result can be reached through different paths. As the dropout is disabled after training is complete, all the paths that lead to correct classification will work together, and small changes in any one of these paths should not change the end result.

4.5.2 Averaging performance metrics across folds

While the leave-two-inspections-out cross validation should prevent data leakage and give a more accurate performance indication, it also means we are training 30 different CNNs, and combining the performance results of these into a single metric is not straightforward. There is no guarantee that the trained networks have at all similar weights at any given point or that the outputs of the networks is similar. As noted in the previous section, the distribution of defects among folds can also be skewed, with some inspections containing a lot more or fewer defects than others.

As such, it does not make sense to average the metrics as calculated on the folds. We could set a single threshold τ for each defect and fold, but since the outputs of the different networks could behave very differently, this is also not desirable.

As we have argued that it is not unlikely for defect detection systems to be tuned to achieve some minimum recall, we have decided to construct the ROC and PR curves for each fold and each defect individually, and combine the curves for different folds by equating the recall axis, and combining the values on the complimentary axis. For the

ROC curve, this is called *horizontal averaging* [29], for the PR curve, we might call it *vertical averaging*, as the recall axis is the horizontal axis, but there is no previous use of this term in literature that we know of.

It should also be noted that the averages for the specificity and precision are not calculated identically. Both are calculated with a weighted average, but the results for specificity in each fold should be weighted such that the combined result represents the specificity of the entire set, and the results for precision should be weighted such that the combined result represents the precision of the entire set. In practice, this means that the results are weighted with the relevant denominator from equation (2.16) or (2.20). As a result, a fold with no occurrences of a particular defect will have no impact on the combined specificity of that defect, and a fold with no *predicted* occurrences of a particular defect will have no impact on the combined precision of that defect.

[29] Millard, L. A. C., Kull, M., and Flach, P. A. 2014. Rate-oriented point-wise confidence bounds for roc curves. In *Machine Learning and Knowledge Discovery in Databases*, T. Calders, F. Esposito, E. Hüllermeier, and R. Meo, Eds. Springer Berlin Heidelberg, Berlin, Heidelberg, 404–421

4.5.3 Class Imbalance

Two choices have been made to adjust the classifier and make it more able to handle this imbalance: oversampling and a class-weighted loss function.

Oversampling is done from a more practical perspective: to train the CNN we have to load a *batch* of input images into memory and the backpropagation step happens for all images in the batch at once. Because of computational limitations, we found that our experimental setup could handle batches of about 50 images at a time. This means that it is extremely likely for a batch not to contain any defects at all. The gradient of such a batch can not be used to accurately estimate the gradient of the entire training set [30]. To remedy this, each image with a defect in the dataset is added not once but five times to the training set, to increase the odds of having at least one defect in every batch.

[30] Bengio, Y. 2012. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*. Springer, 437–478

As oversampling the defects by a factor five raises the imbalance from 0.8% to about 4% of the images in the training set containing a defect, we should still be wary of training a network with such an imbalance. To shift the error that the network makes more towards false positives than false negatives, we weight the cross entropy loss function from equation (4.2) as follows:

$$L(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{c=1}^{12} W \cdot y_c \log \hat{y}_c + (1 - y_c) \log(1 - \hat{y}_c) \quad (4.3)$$

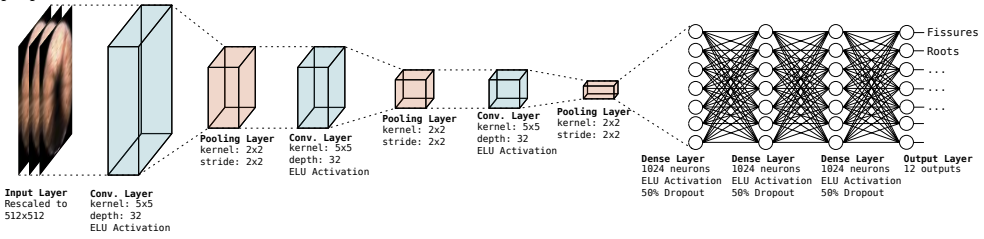
where W is a weight that represents how much more important false negatives are compared to false positives. If we consider a false negative to be 100 times more costly than a false positive, we should set W to 100.

4.6 Implementation Details

We have implemented two different CNNs, one designed by us for this task, and one reimplementation of the network used by Kumat et al. [31], which was the state-of-the-art at the time of writing (with the first layer adapted to our image sizes). This is of course not an entirely fair comparison, as we failed to reproduce their entire pipeline, but instead only replicated the network itself, but it does put the performance into context.

[31] Kumar, S. S., Abraham, D. M., Jahanshahi, M. R., Iseley, T., and Starr, J. 2018. Automated defect classification in sewer closed circuit television inspections using deep convolutional neural networks. *Automation in Construction 91*, 273–283

Figure 4.2: Network structure of our proposed CNN.

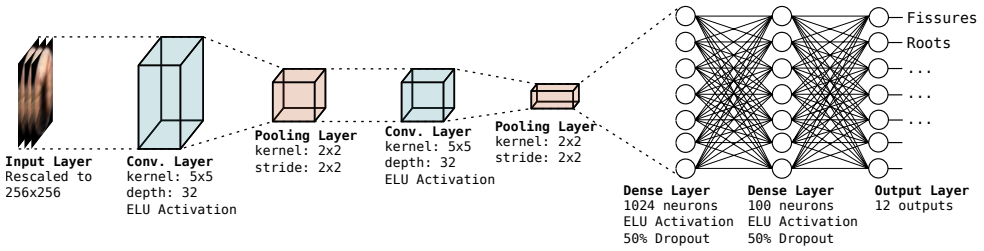


The network topologies are shown in figures 4.2 and 4.3. The network topology of our proposed CNN was designed through experimentation with different layer sizes, filter sizes, and numbers of layers on a smaller subset of the dataset.

The CNNs were built and run with TensorFlow (version 1.8.0) and Python (version 3.4.8), running on a Linux system with sixteen NVIDIA Tesla K80 GPUs and CUDA (version 9.2.148). Each network was trained using a single GPU, with several networks (one for each validation fold) being trained simultaneously on multiple GPUs. Training a single network took on average roughly five hours (per fold). Testing the different networks with each different testing fold took on average roughly 1 hour (for all 30 folds).

Each of the networks was trained with a batch size of 50 images. After every 500 batches, the performance on the validation fold was assessed. The network stopped training when the AUROC on the validation fold had not increased for 25 consecutive assessments, or when the AUROC on the validation fold decreased by more than 1%, with a minimum of 1,000 batches processed.

Figure 4.3: Network structure of the CNN proposed in Kumar et al.



4.7 Results

In this section, we present the results achieved by our proposed CNN, as well as our reimplementation of the CNN proposed by Kumar et al.,[32] on the performance metrics outlined in section 4.3.2.

Tables 4.3, 4.4, 4.5, and 4.6 show the specificity (TNR) and precision at recall (TPR) values of 0.90, 0.95, and 0.99, for our proposed CNN and our reimplementation of the CNN proposed by Kumar et al. The better result for each scenario is displayed in bold when it is significantly better, determined by a paired sample t-test (at a significance level of $\alpha = 0.05$) across the validation folds.

Figures 4.8.1, 4.8.1, 4.8.1, and 4.8.1 show the ROC and PR curves for our proposed CNN for classification in images and entire pipes.

[32] Kumar, S. S., Abraham, D. M., Jahanshahi, M. R., Iseley, T., and Starr, J. 2018. Automated defect classification in sewer closed circuit television inspections using deep convolutional neural networks. *Automation in Construction 91*, 273–283

4.8 Discussion

Looking at tables 4.3, 4.4, 4.5, and 4.6, we see that in each of the shown scenarios, our proposed CNN either outperforms Kumar et al.,[33] or it does not perform significantly worse. Out of 144 scenarios, our proposed network wins significantly 81 times. Additionally, it wins 44 times, but not by a significant margin, and looses 19 times, but never significantly.

[33] Kumar, S. S., Abraham, D. M., Jahanshahi, M. R., Iseley, T., and Starr, J. 2018. Automated defect classification in sewer closed circuit television inspections using deep convolutional neural networks. *Automation in Construction 91*, 273–283

4.8.1 Classifying individual images

When we take a closer look at the ROC and PR curves for the classification of individual images in figures 4.8.1 and 4.8.1, there are a few observations to be made.

The ROC curves in figure 4.8.1 generally look quite good, with the exception of those for porous pipes, and to a lesser

degree attached deposits and settled deposits. The class imbalance is quite important here: the AUROC does not take into account that a false positive and false negative are not comparable in this context. As noted earlier, we are mostly interested in the scenario with high recall (TPR), as these are a requirement for any kind of automated sewer inspection, which means the top portion of each plot is more important than the bottom portion. It must also be noted that while both axes go from 0 to 1, the horizontal axis represents many more images than the vertical axis does, because of the class imbalance. One interesting feature of these curves, is that they seem to have a 'plateau' near the top. This indicates that a specific threshold exists where it is no longer advantageous to further increase the threshold, as this will only increase the false positive rate, but not the true positive rate. The false positive rate at this interval (which is approximately equal to 1 minus the specificity at 99% recall, as noted in table 4.3) can be regarded as the best specificity we can achieve for a certain defect.

The PR curves in figure 4.8.1 paint a different picture: the PR curves are mostly below an F_1-score of 0.2, seeming very unimpressive. Similar to the ROC curves, we are mostly interested in high recall, i.e. the rightmost portion of each plot. In this case, the precision seems to be quite low, but unlike the specificity, the precision axis *is* scaled with the prior probability of the defects. We will go into more detail on how to interpret these precision scores in the next section, but it should be noted that a small precision is expected when we have small prior probabilities.

4.8.2 Classifying entire pipes

When we observe the ROC and PR curves for classification of pipes in figures 4.8.1 and 4.8.1, they paint a rather different image. The ROC curves in figure 4.8.1 do not look very good, but it should be noted that the 'plateaus' are

Table 4.3: **Specificity at recall** values when classifying **single images**. Numbers displayed in bold indicate that performance is significantly better than the performance achieved by the other network, as determined by a paired sample t-test at significance level $\alpha = 0.05$.

	Specificity at 0.90 recall		Specificity at 0.95 recall		Specificity at 0.99 recall	
Defect	This work	Kumar et al.	This work	Kumar et al.	This work	Kumar et al.
Fissure	**0.754**	0.375	**0.683**	0.290	**0.550**	0.208
Surface Damage	**0.702**	0.240	**0.548**	0.107	**0.291**	0.045
Intruding Connection	**0.916**	0.448	**0.809**	0.392	**0.741**	0.370
Defective Connection	**0.901**	0.553	**0.811**	0.460	**0.703**	0.372
Intruding Sealing Material	**0.780**	0.061	**0.731**	0.057	**0.706**	0.057
Displaced Joint	**0.691**	0.441	**0.532**	0.306	**0.262**	0.145
Porous Pipe	0.349	0.207	**0.322**	0.179	0.307	0.171
Roots	**0.728**	0.209	**0.633**	0.166	**0.561**	0.159
Attached Deposits	**0.388**	0.142	**0.313**	0.116	**0.281**	0.115
Settled Deposits	**0.510**	0.114	**0.459**	0.102	**0.442**	0.097
Ingress of Soil	**0.762**	0.322	**0.670**	0.237	**0.532**	0.180
Infiltration	**0.622**	0.220	**0.486**	0.160	**0.253**	0.092

Defect	Precision at 0.90 Recall		Precision at 0.95 Recall		Precision at 0.99 Recall	
	This work	Kumar et al.	This work	Kumar et al.	This work	Kumar et al.
Fissure	**0.036**	0.006	**0.019**	0.004	**0.005**	0.003
Surface Damage	**0.011**	0.003	**0.005**	0.002	**0.003**	0.002
Intruding Connection	**0.071**	0.007	**0.011**	0.005	**0.006**	0.004
Defective Connection	**0.014**	0.002	**0.008**	0.002	**0.004**	0.001
Intruding Sealing Material	0.002	0.000	0.002	0.000	0.001	0.000
Displaced Joint	**0.015**	0.006	**0.010**	0.005	**0.005**	0.004
Porous Pipe	0.000	0.000	0.000	0.000	0.000	0.000
Roots	**0.003**	0.001	**0.002**	0.001	**0.002**	0.001
Attached Deposits	**0.001**	0.000	**0.001**	0.000	**0.001**	0.001
Settled Deposits	0.001	0.000	0.000	0.000	0.000	0.000
Ingress of Soil	**0.008**	0.002	**0.005**	0.002	**0.003**	0.002
Infiltration	**0.024**	0.012	**0.017**	0.012	**0.013**	0.012

Table 4.4: **Precision at recall** values when classifying **single images**. Numbers displayed in bold indicate that performance is significantly better than the performance achieved by the other network, as determined by a paired sample t-test at significance level $\alpha = 0.05$.

Table 4.5: **Specificity at recall** values when classifying **entire pipes**. Numbers displayed in bold indicate that performance is significantly better than the performance achieved by the other network, as determined by a paired sample t-test at significance level $\alpha = 0.05$.

	Specificity at 0.90 recall		Specificity at 0.95 recall		Specificity at 0.99 recall	
Defect	This work	Kumar et al.	This work	Kumar et al.	This work	Kumar et al.
Fissure	**0.428**	0.357	**0.365**	0.279	0.306	0.261
Surface Damage	0.250	0.226	0.193	0.155	0.128	0.107
Intruding Connection	**0.414**	0.250	**0.371**	0.224	**0.354**	0.220
Defective Connection	**0.230**	0.186	0.186	0.147	0.172	0.132
Intruding Sealing Material	0.411	0.406	0.411	0.406	0.411	0.406
Displaced Joint	0.294	0.268	**0.219**	0.169	**0.123**	0.085
Porous Pipe	0.363	0.399	0.346	0.377	0.344	0.372
Roots	0.403	0.338	0.338	0.290	0.317	0.267
Attached Deposits	0.518	0.481	0.496	0.454	0.486	0.444
Settled Deposits	**0.386**	0.305	**0.364**	0.282	**0.363**	0.282
Ingress of Soil	0.285	0.315	0.237	0.286	0.209	0.245
Infiltration	0.284	0.298	0.218	0.207	0.141	0.133

Defect	Precision at 0.90 Recall		Precision at 0.95 Recall		Precision at 0.99 Recall	
	This work	Kumar et al.	This work	Kumar et al.	This work	Kumar et al.
Fissure	**0.414**	0.379	**0.393**	0.364	0.363	0.358
Surface Damage	0.582	0.589	0.573	0.570	0.557	0.552
Intruding Connection	**0.369**	0.333	**0.360**	0.324	**0.348**	0.315
Defective Connection	**0.294**	0.276	**0.291**	0.273	**0.285**	0.272
Intruding Sealing Material	0.068	0.062	0.068	0.062	0.068	0.062
Displaced Joint	0.600	0.602	0.585	0.582	0.565	0.561
Porous Pipe	0.130	0.137	0.129	0.136	0.129	0.135
Roots	**0.208**	0.189	0.185	0.184	0.176	0.175
Attached Deposits	0.190	0.196	0.185	0.182	0.183	0.179
Settled Deposits	0.125	0.118	0.117	0.108	0.117	0.109
Ingress of Soil	0.363	0.364	0.348	0.358	0.339	0.335
Infiltration	0.584	0.602	0.579	0.584	0.560	0.563

Table 4.6: **Precision at recall** values when classifying **entire pipes**. Numbers displayed in bold indicate that performance is significantly better than the performance achieved by the other network, as determined by a paired sample t-test at significance level $\alpha = 0.05$.

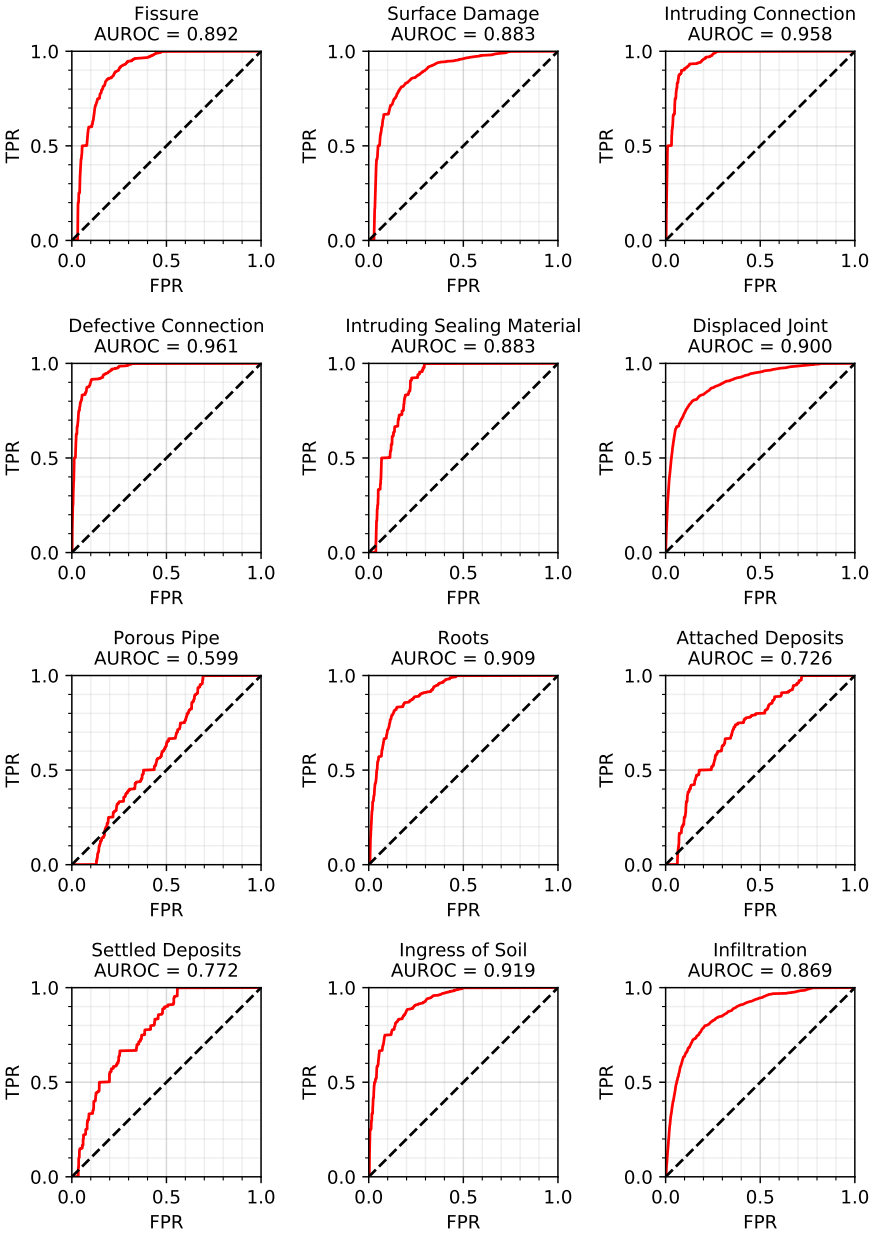


Figure 4.4: **ROC Curves** for the proposed CNN when classifying **single images**.

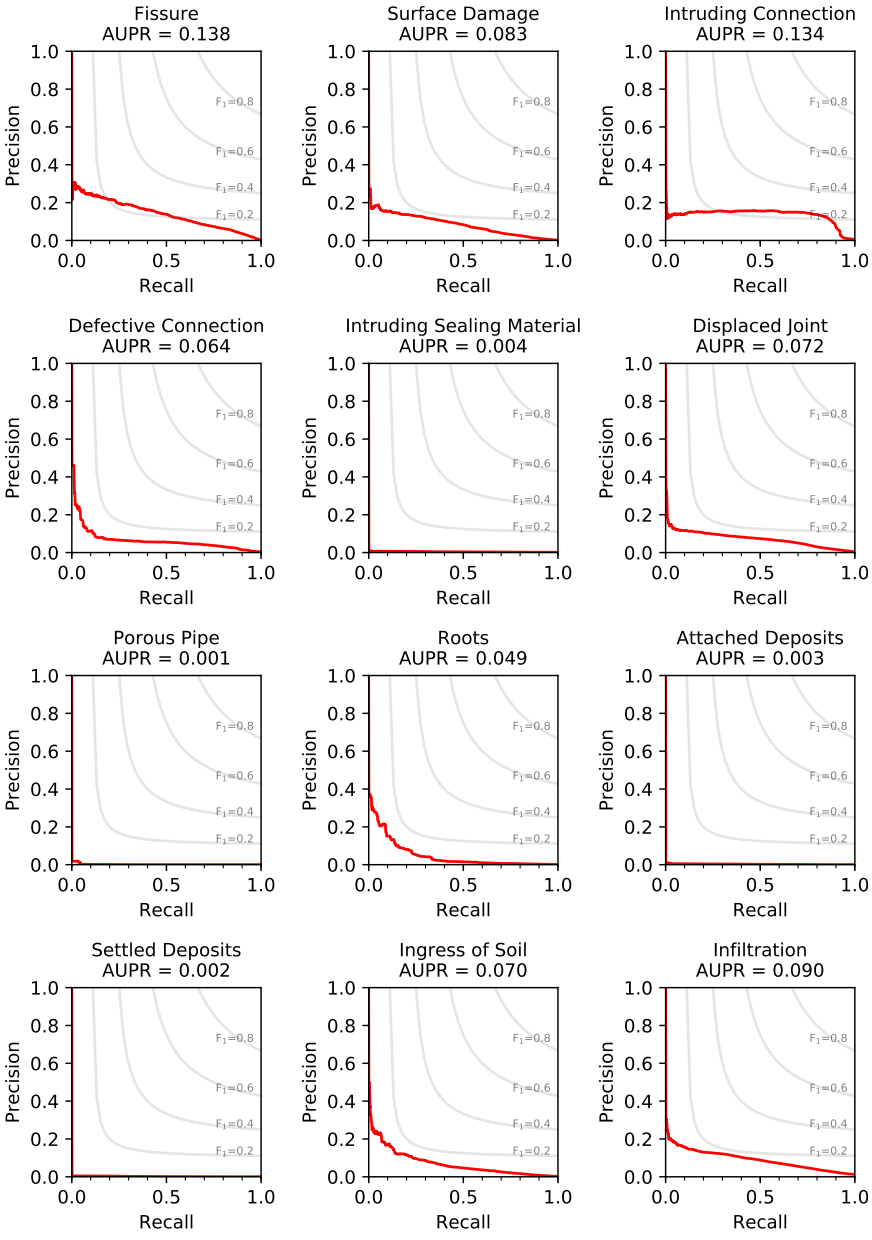


Figure 4.5: **Precision-Recall Curves** for the proposed CNN when classifying **single images**.

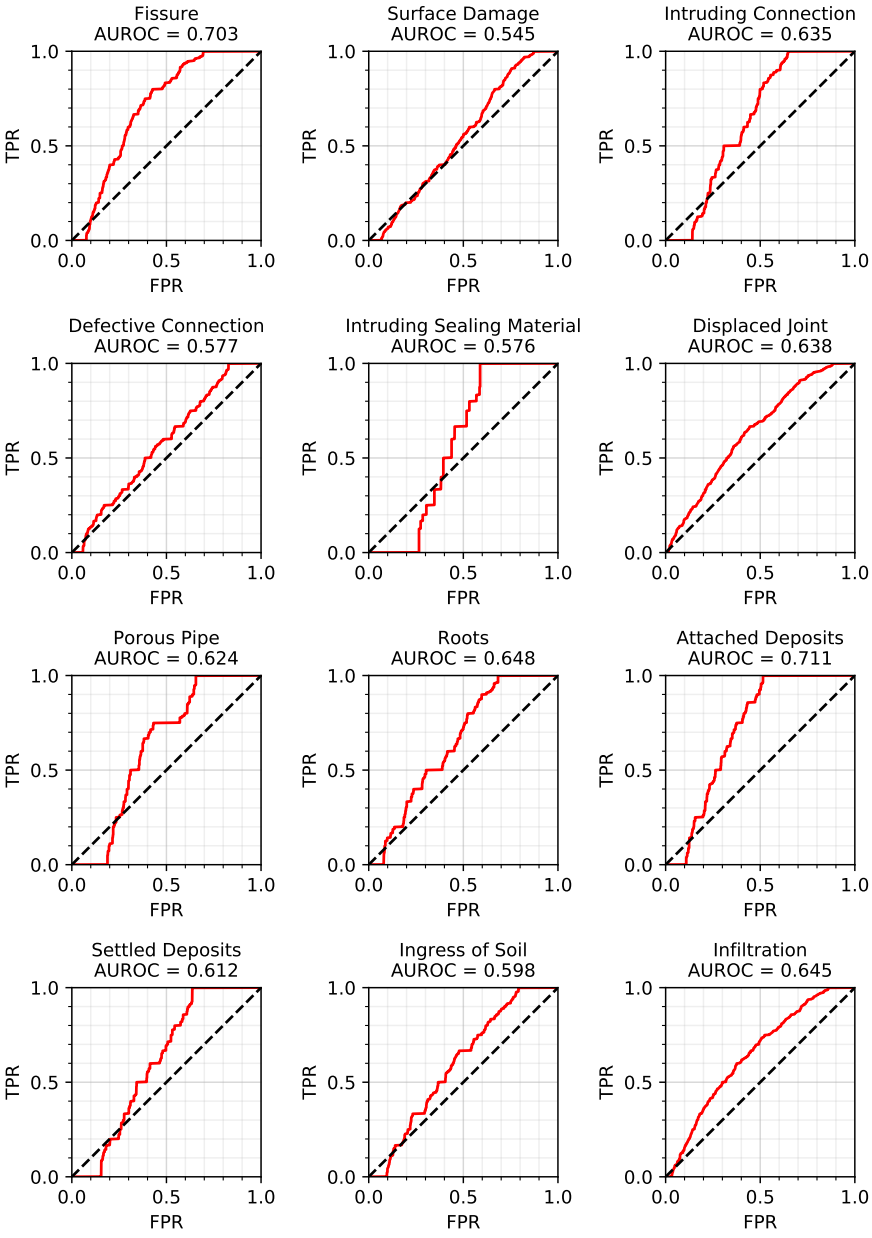


Figure 4.6: **ROC Curves** for the proposed CNN when classifying **entire pipes**.

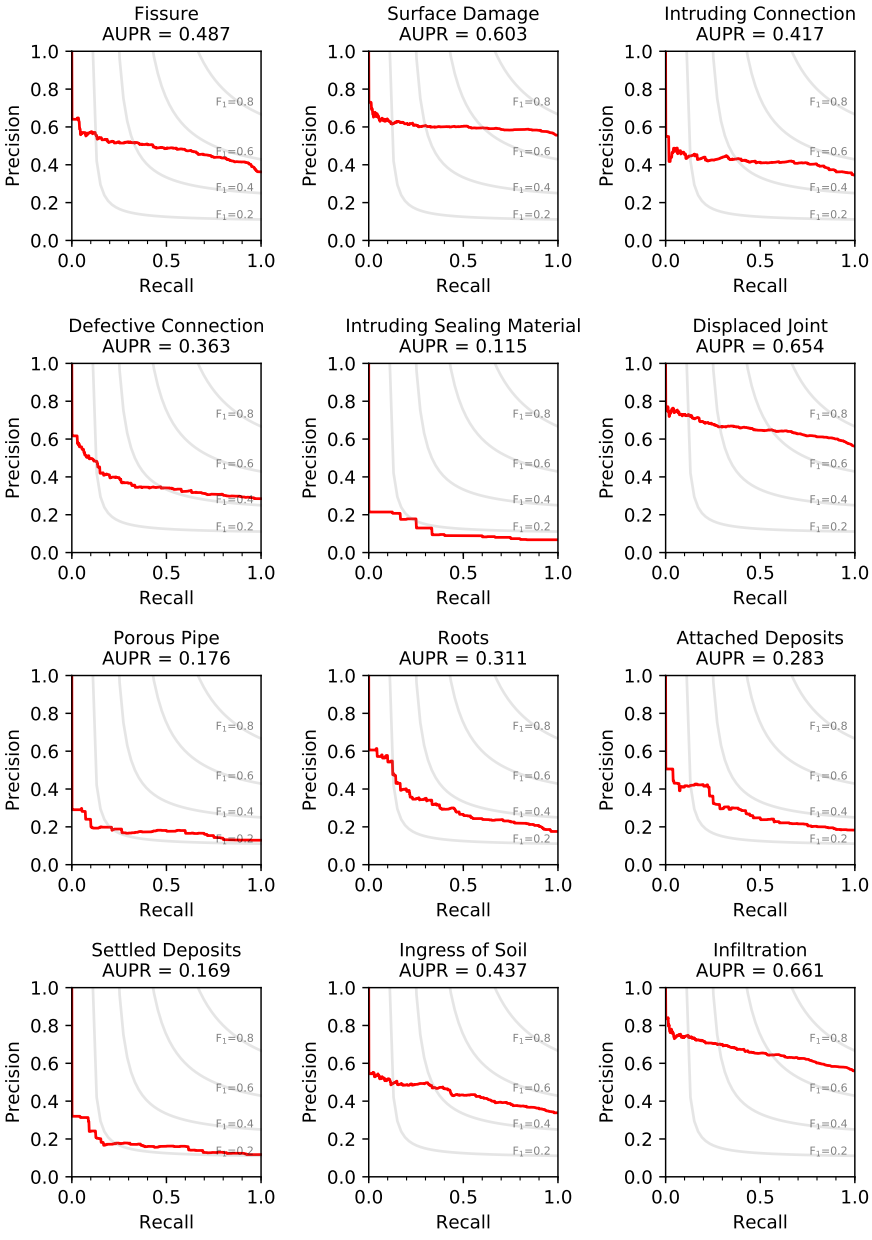


Figure 4.7: **Precision-Recall Curves** for the proposed CNN when classifying **entire pipes**.

again present in a lot of the curves, which again indicates that there are some pipes of which we are confidently sure they *do not* contain defects. Rather interestingly, among the better ROC curves are those that underperformed on single-image classification: porous pipes, attached deposits, and settled deposits. This might indicate that some labels were missing in our dataset: if a pipe has these defects at multiple locations but only a few locations were marked in the inspection report, we would overestimate the false positives our classifier finds in single-image classification, but we can be more sure when deciding whether a pipe has or does not have this defect.

The PR curves for the classification of pipes in figure 4.8.1 looks a lot better than that of single images. This is because the class imbalance is much less present on pipe-level. Still the worst results are obtained for classes that have a low prior on pipe-level (intruding sealing material, porous pipe, roots, attached deposits, settled deposits), as expected for the precision.

4.8.3 Result Interpretation

To put our findings into context, we will take a closer look at their impact on the day-to-day operation of sewer inspections aided by our automated system. When looking at our results superficially, they are easily misinterpreted. It is important to keep in mind that we are dealing with a very imbalanced dataset, which makes the precision the more interesting of these metrics (as described in section 4.3.2). Let's consider the class *Fissure* in more detail. From table 4.1 we can tell that approximately 0.065% of the images (1,442 out of 2,202,582 images in total) contain a fissure, which makes for a very imbalanced target. For fissures at 0.90 recall we achieve a specificity of 0.754 and a precision of 0.036 (see tables 4.3 and 4.4, top left cell). [34]

[34] It should be noted that these numbers do not add up perfectly to the 1,442 fissures out of 2,202,582 images, as the specificity and precision are aggregated over 30 different folds, each with its own specificity and precision.

The specificity of 75.4% indicates that, of all the images that do *not* contain fissures, we identify 75.4% as such, and the remaining 24.6% are suspected of containing fissures, meaning they still have to be inspected by an operator. The precision indicates that of all fissure detections, 3.6% are true positives, while the remaining 96.4% are false alarms.

If we assume that fissures are randomly distributed, an unsupported operator would have to inspect 90% of all images ($0.9 \times 2{,}202{,}582 = 1{,}982{,}324$ images) to find 90% of the fissures. Our proposed classifier detects 90% of all images with fissures with a specificity of 75.4%.

To detect 90% of all fissures, an operator would have to inspect all detections the system made: $0.246 \times (2{,}202{,}582 - 1{,}442) + 0.9 \times 1{,}442 = 542{,}778$ images. In comparison to the situation without a classification system, this is equal to a reduction of 72.6%. In an ideal situation, this means that the time an operator spends on inspecting fissures is reduced by almost a factor 4. Table 4.7 lists these reduction numbers (derived from tables 4.3 and 4.4) for all defect types considered. The reduction of 72.6% for Fissure appears as the top-left cell. The highest reduction (at 0.90 recall) is attained for Intruding Connection, with a 90.7% reduction (a factor 10). Not surprisingly, this defect type scores well both in the ROC as in the PR plots. It ranks 6th in terms of frequency of defect type, with 1,004 observed cases.

We can perform the same calculations with the results from classification on *entire pipes*, but the interpretation is a little less clear, as we cannot assume different pipes take the same amount of time for review; especially pipes with a lot of defects will be more labour-intensive to inspect. From table 4.1 we can tell that approximately 17.5% of pipes contain fissures (586 out of 3,350 pipes). Let us for this case assume 99% of all pipes containing fissures need to be detected, this means $0.99 \times 3{,}350 = 3{,}317$ pipes have to be inspected for fissures. Our classifier achieves 99% recall with a specificity of 30.6% (table 4.5) and a precision of 36.6% (table 4.6). By

Table 4.7: Reduction of *images* that need to be reviewed by a human after inspection with our classifier, expressed in percentage compared to images that would need to be inspected without our classifier.

	Recall		
Defect Type	**0.90**	**0.95**	**0.99**
Fissure	72.6%	66.6%	54.5%
Surface Damage	66.8%	52.4%	28.3%
Intruding Connection	90.7%	79.8%	73.8%
Defective Connection	89.0%	80.1%	70.0%
Intruding Sealing Material	75.5%	71.7%	70.3%
Displaced Joint	65.5%	50.7%	25.4%
Porous Pipe	27.7%	28.7%	30.0%
Roots	69.8%	61.4%	55.6%
Attached Deposits	32.0%	27.7%	27.4%
Settled Deposits	45.5%	43.1%	43.7%
Ingress of Soil	73.5%	65.3%	52.7%
Infiltration	57.8%	45.8%	24.4%

the same calculations as before, this means we now have to inspect $0.694 \times (3{,}350 - 586) + 0.99 \times 586 = 2499$ pipes. This is a reduction of 24.7%. Table 4.8 shows similar reductions for all the defect types, for pipes.

In table 4.7 we see that intruding and defective connections are best classified by our CNN and have the largest reduction rate in images or pipes that still require human review, while porous pipes are the more difficult to classify and these have the lowest reduction rates.

Realistically, the defects are not randomly distributed throughout the image set and operators would not inspect single images, but rather a sequence of images with a clear spatial relationship (a 5 cm shift). This means that the reduction by a factor of 4 is almost certainly an overestimation. On the other hand, we know defects can often co-occur [35] and this estimation was only for fissures, which has one of the higher prior probabilities of the defects we consider. For defects with a lower prior probability, there is a larger potential for improvement.

It should also be noted that with the reported false negative probability of about 25%[36] in the labels of our data set,

[35] SITZENFREI, R., MAIR, M., MÖDERL, M., AND RAUCH, W. 2011. Cascade vulnerability for risk analysis of water infrastructure. *Water Science and Technology 64,* 9, 1885–1891

[36] DIRKSEN, J., CLEMENS, F., KORVING, H., CHERQUI, F., LE GAUFFRE, P., ERTL, T., PLIHAL, H., MÜLLER, K., AND SNATERSE, C. 2013. The consistency of visual sewer inspection data. *Structure and Infrastructure Engineering 9,* 3, 214–228

Defect Type	Recall 0.90	0.95	0.99
Fissure	30.1%	27.4%	24.7%
Surface Damage	10.5%	9.5%	7.5%
Intruding Connection	31.0%	30.0%	30.9%
Defective Connection	12.2%	12.1%	13.9%
Intruding Sealing Material	33.8%	37.2%	39.7%
Displaced Joint	11.9%	9.8%	6.3%
Porous Pipe	28.3%	30.1%	32.6%
Roots	30.9%	27.8%	28.5%
Attached Deposits	43.9%	44.3%	45.4%
Settled Deposits	30.3%	31.5%	33.9%
Ingress of Soil	17.2%	16.5%	16.9%
Infiltration	12.2%	10.5%	7.9%

Table 4.8: Reduction of *pipes* that need to be reviewed by a human after inspection with our classifier, expressed in percentage compared to pipes that would need to be inspected without our classifier.

the actual precision and specificity are likely higher than we report. For any given defect, there is approximately a 1 in 4 chance that the operator missed it and it was labeled in our dataset as not being a defect (whereas the probability of a false positive was estimated "in the order of a few percent"). The 1,442 images that are labeled as fissures, are possibly only 75% of all images labeled containing fissures, meaning there would be approximately 480 images among the images not labeled as fissures.

4.8.4 Combining Defect Outputs

Because of the co-occurrence of defects, it can be interesting to combine the classifier outputs for different defects into a single decision: *"Does this image/pipe need further (human) review?"*

As discussed in section 4.3, in our dataset 30.7% of defects in images co-occur with other defects in the same image and 89.2% of defects in pipes co-occur with other defects in the same pipe. To treat the problem as a binary classification problem, we simply take the maximum value of the true

Table 4.9: Specificity and precision at recall values for binary classification on either single images or entire pipes.

Metric and Classification type	Recall 0.90	0.95	0.99
Specificity for Images	0.649	0.452	0.180
Specificity for Pipes	0.372	0.284	0.113
Precision for Images	0.021	0.014	0.009
Precision for Pipes	0.717	0.703	0.668

label over the classes (a 1 if at least one defect is present, a 0 if no defects are present), and the average of the predicted labels over the classes (a real-valued number between 0 and 1). This gives us the curves as shown in figure 4.8.

For classification on images, reducing this problem to a binary classification case does not improve things much. The overall result is approximately equal to the average of the classification results on individual classes. This is not unexpected, as the co-occurrence of defects in individual images is rare.

For classification on pipe level though, the results are

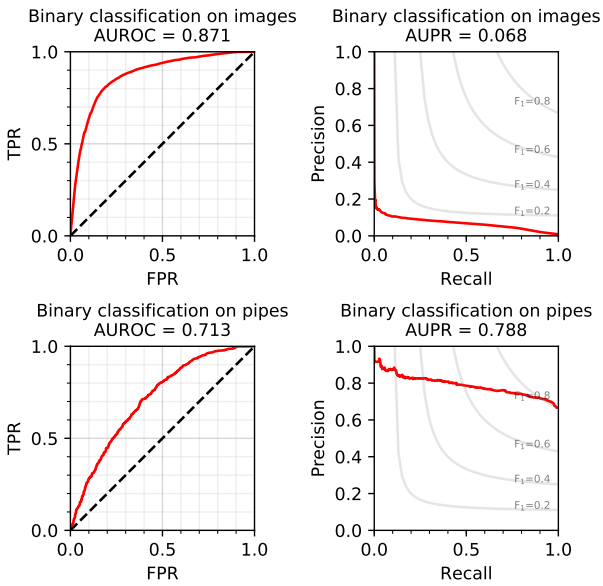


Figure 4.8: ROC and PR curves obtained when treating the problem as a binary classification problem, for image level or pipe level.

Classification Type	Recall 0.90	0.95	0.99
Images	60.5%	42.0%	17.0%
Pipes	7.6%	6.2%	2.6%

Table 4.10: Reduction of images or pipes that need to be inspected with our combined binary classifier, expressed in percentage compared to pipes that would need to be inspected without our combined binary classifier.

more interesting than a simple averaging. The PR curve is strictly better than the PR curves of individual defects. The ROC curve at high recall is slightly worse than some individual defects, but the overall AUROC is higher.

Table 4.9 shows the specificity and precision at specific recall values, for comparison with the multi-label classification results in tables 4.3, 4.4, 4.5, and 4.6. Using these values we can again calculate the reduction in images or pipes that require review to achieve a certain recall, as shown in table 4.10.

The reductions on pipe level are quite low, this is because the transition to a binary classification scenario results in the class imbalance disappearing on pipe-level: 75.0% of pipes contain *at least one* defect, and fall into the positive class. This means a high precision is required, and while precision had increased by combining the defect types, the reduction has decreased.

4.9 Conclusion

In this chapter, we have approached the task of automated defect detection in sewer image sets as a supervised classification task. The focus has been on the validation methodology used to interpret the results achieved by a classifier. While we feel that there is a lot of potential for future improvement of classifiers trained for this task, with the data and computational resources available, the proposed convolutional neural network performed reasonably well.

While our proposed classifier does not perform well enough for fully autonomous classification, it can be used to signif-

icantly reduce the amount of images that require human review by eliminating images which are highly unlikely to contain defects according to the classifier. We estimate the amount of images that require human review can be reduced by 60.5%, given that detecting 90% of all defects is sufficient.

We compared the results of our proposed classifier to that of Kumar et al.,[37] and our proposed classifier outperforms their proposed classifier, but we did not implement their classification pipeline beyond the network structure, such as for example, the oversampling outlined in their work. Our dataset also differs significantly from theirs. As noted in section 4.2, no human inspector has changed the camera settings during the inspection, as is common with other CCTV inspection datasets.

[37] Kumar, S. S., Abraham, D. M., Jahanshahi, M. R., Iseley, T., and Starr, J. 2018. Automated defect classification in sewer closed circuit television inspections using deep convolutional neural networks. *Automation in Construction 91*, 273–283

A major topic of this chapter was the validation methodology. We have discussed our reasons for choosing the "specificity at recall" and "precision at recall" metrics for this specific task in section 4.3.2: these give us easily interpretable measures of the possible improved efficiency at realistic scenarios. We have also explained why "leave-two-inspections-out cross validation" is an appropriate way to prevent data leakage, and applied this technique in our experiments. These methods provide us with less biased and more easily interpretable results.

4.9.1 Future Work

Not all information in the inspection reports was used to its full potential and we feel that using the information pertaining to where in an image a defect is visible (with a classifier capable of processing this information of course) could lead to further performance improvement. Additionally, the use of other types of sensors, either already present on or easily added to the pipe inspection vehicle, may prove to be useful for further improvement.

Since we know there are likely undetected defects in our dataset,[38] it would be an interesting experiment to see if a classifier trained on data where these are unlabeled, is still able to find these defects in its own training set. To achieve this, the false positive detections would have to be re-classified by a human operator. Hopefully, this would indicate that the classifier detected defects that we *thought* were false positives, but were in fact true positives. Unfortunately, this is beyond the scope of this thesis.

[38] Dirksen, J., Clemens, F., Korving, H., Cherqui, F., Le Gauffre, P., Ertl, T., Plihal, H., Müller, K., and Snaterse, C. 2013. The consistency of visual sewer inspection data. *Structure and Infrastructure Engineering 9,* 3, 214–228

5 Stereovision and Geometry Reconstruction

So far, we have looked at unsupervised and supervised machine learning methods applied to image data generated by current inspection practices in urban drainage. In this chapter, we move beyond current inspection practices, and extend image acquisition to a stereovision system, consisting of two cameras, in order to reconstruct the three-dimensional pipe geometry and recognise potential defects from that. We consolidate several techniques into RADIUS (Robust Anomaly Detection In Urban drainage with Stereovision), a framework for anomaly detection in sewer pipes.

5.1 Introduction

The RADIUS framework is designed to be compatible with the existing workflow of the trained operator, as well as be future-proof for a completely automated sewer inspection system, for which advances are being made rapidly. The framework also has a low up-front investment in terms of equipment, as it uses two cameras for data collection, and image processing steps that can be performed on a consumer-grade computer within a reasonable amount of time.

Our proposed method revolves around the technique of *computer stereovision*, which uses two or more calibrated cameras placed side by side, in order to create a sense of depth, similar to how the binocular vision in humans is used to capture the spatial configuration of one's surrounding. In that sense, the proposed method is a type of 3D ranging technique that promises to produce a faithful 3D reconstruction of the interior of a sewer pipe in the form of a 3D point cloud with associated colour information.

Since the raw output of our stereovision setup captures the pipe's surface in considerable detail (the extent of this is determined by the resolution of the cameras), it can theoretically be used to recognise various different categories of pipe defects that have a spatial nature. These include deposits, holes, fissures, intrusions and exposed granulates. Some of these may in fact be harder to correctly classify using traditional single CCTV setups, since without further spatial clues, they cannot be distinguished (for example, a fissure (*on* the surface) vs. an intruding root (*away* from the surface)).

While in terms of types of defects to be recognised, the framework is open-ended, we focus in our experiments on the recognition of two specific types of defects, namely deposits and exposed granulates. For other defect types (such as misaligned joints) the larger part of our proposed pipeline remains unchanged, but in the final stages provisions need to be made to account for the different spatial phenomena at hand. When focusing on deposits and exposed granulates, we need to recognise areas in the pipe where the surface is further inward or outward than one would expect. In other words, there is an *expected* pipe geometry (the pipe model) and a *measured* pipe geometry, and any deviations between these will be expressed in an anomaly score. Although this form of anomaly detection sounds fairly straightforward, it relies on the availability of a pipe model, which is actually non-trivial. First of all, the precise location and orientation of the camera pair inside the pipe is uncertain. We can only assume that the cameras are pointing roughly along the main axis of the pipe, not too far removed from the centre of the pipe. Second, our method should be robust with respect to the shape of the pipe, such that different pipe topologies can be dealt with, without having to reconfigure the recognition system. This means we will take a data-driven approach that assumes that the larger part of the pipe is unaffected, such that a 'normal' geometry can be derived from that, and the

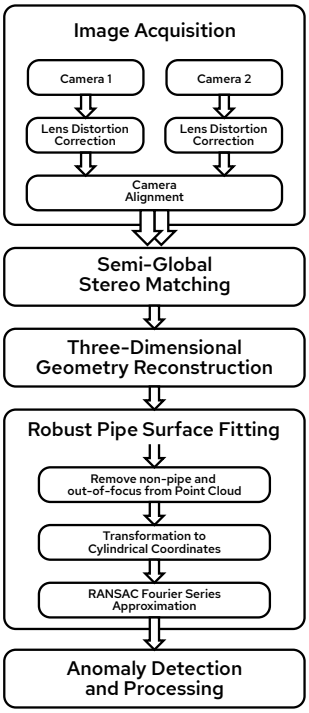


Figure 5.1: Overview of the proposed framework

outliers with respect to this geometry are the anomalies.

In broad strokes, our framework works as follows (see also figure 5.1). In the image acquisition stage, any radial lens distortion is removed from the images, and misalignment between the left and right image are corrected. An existing algorithm for stereo matching [1] then produces pairs of corresponding pixels in both images. The computed disparity between each pair can be translated into a distance for this pixel: points closer to the cameras will appear further apart in the two images. The resulting point cloud already captures the pipe geometry, but needs to be further processed in order to automatically identify the defects of interest, in our case deposits and exposed granulate. The next stage of surface fitting combines a parameterised surface model, one that can adapt to a wide range of pipe types, with the robust regression algorithm RANSAC [2]. This algorithm assumes that the majority of the data fits a predefined model class (our parameterised pipe model), but also allows for a fraction of the data to constitute outliers. This allows us to use the parameterised pipe model without the fit being influenced by these outliers. The deviation of the measured geometry from the expected geometry as predicted by the RANSAC model is used as an anomaly score.

The main contributions of this chapter are:

- ◇ We demonstrate how a faithful and high-resolution reconstruction of the pipe surface, including its defects, can be obtained with stereo cameras and a stereo matching algorithm.

- ◇ We propose a generic pipe surface model that is able to model the pipe geometry of a range of pipe shapes (including circular and egg-shaped), captured under various angles. This pipe surface model has the attractive property that it falls in the category of functions that can be statistically fit with the Ordinary Least Squares method [3], making it computationally efficient.

[1] HIRSCHMÜLLER, H. 2005. Accurate and efficient stereo processing by semi-global matching and mutual information. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 2. IEEE, 807–814

[2] FISCHLER, M. A. AND BOLLES, R. C. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM 24,* 6, 381–395

[3] GOLDBERGER, A. S. 1964. *Classical Linear Regression, Econometric Theory.* New York: John Wiley & Sons

- ◇ We propose a method based on the RANSAC algorithm to fit point cloud data that is a mixture of regular pipe surface and anomalies.
- ◇ We define a global anomaly score that quantifies the amount of deviation from the pipe model per image pair.

Section 5.2 outlines relevant prior work. The framework itself is described in full detail in section 5.3. Section 5.4 gives an overview of the data and the experiments, the results of which are summarised and discussed in section 5.5. Section 5.6 discusses the limitations, envisioned applications, and possible future work.

5.2 Prior Work and Motivation

In the field of 3D ranging techniques, where our approach belongs, the use of laser scanners for sewer pipe inspections has been thoroughly researched, see for example [4,5,6,7]. The reasons we have opted to go with stereovision instead of laser scanning are threefold, i) the equipment cost of two cameras versus that of a laser scanner is significantly lower, making this approach more accessible, ii) a stereovision setup has no moving parts, which matters in real-world scenarios, where the environment of a sewer pipe can be very abrasive to moving parts in particular, and iii) the point cloud obtained from stereovision will be linked directly to images with a colour component, whereas a laser scanner only provides the geometry.

While not much research has been done on the use of stereovision in the context of sewer condition assessment, the use of stereovision in the general context of sewer maintenance is not new. Most works restrict their approach to

[4] Lepot, M., Stanić, N., and Clemens, F. H. 2017. A technology for sewer pipe inspection (part 2): Experimental assessment of a new laser profiler for sewer defect detection and quantification. *Automation in Construction 73*, 1–11

[5] Stanić, N., Clemens, F. H., and Langeveld, J. G. 2017. Estimation of hydraulic roughness of concrete sewer pipes by laser scanning. *Journal of Hydraulic Engineering 143,* 2, 04016079

[6] Duran, O., Althoefer, K., and Seneviratne, L. D. 2007. Automated pipe defect detection and categorization using camera/laser-based profiler and artificial neural network. *IEEE Transactions on Automation Science and Engineering 4,* 1, 118–126

[7] Bahnsen, C. H., Johansen, A. S., Philipsen, M. P., Henriksen, J. W., Nasrollahi, K., and Moeslund, T. B. 2021. 3d sensors for sewer inspection: A quantitative review and analysis. *Sensors 21,* 7, 2553

cylindrical pipes, as these our fairly common, but our work does not impose that limitation.

Ahrary et al. (2005) [8] propose an algorithm for navigation of an autonomous vehicle through a sewer network based on stereovision. Later, Ahrary et al. (2008) developed a computationally efficient stereo matching algorithm specifically for sewer pipes [9]. We do not use either of these algorithms, as processing power is not a limitation in our research, and navigation of an autonomous vehicle is outside our scope.

Tangentially related to this work, Koodtalang et al. [10] use stereovision to determine manufacturing defects in pipes prior to installation.

Gunatilake et al. [11] combine a stereovision setup with two laser profilers to map the images recorded by the cameras onto the potentially more accurate point cloud produced by the laser profilers. This produces a high-resolution RGB-D dataset for later inspection, either by a trained expert or another algorithm. The method is tested on a single, heavily corroded pipe, as well as an artificial pipe.

Most closely related to our work, Huyhn et al. have published two works [12,13] on anomaly detection in sewer pipes with stereovision. They demonstrate the visibility of artificial defects in point clouds generated from stereovision. A critical difference to their approach is that our approach requires no human operator to centre the defect into the camera's field of view, but instead is able to highlight anomalies in the entire pipe from a set of images, and is thus more suitable for automated defect detection.

Anomaly detection has been used extensively in sewer condition assessment as a stepping stone from "traditional" data that is gathered for manual classification, towards automation of the inspection process. Meijer et al. [14] performed principal component analysis (PCA) on various feature descriptors of a labeled set of CCTV images and compared the partial reconstruction with the actual val-

[8] Ahrary, A., Tian, L., Kamata, S.-i., and Ishikawa, M. 2005. An autonomous sewer robots navigation based on stereo camera information. In *17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05)*. IEEE, 6–pp

[9] Ahrary, A. and Ishikawa, M. 2008. A fast stereo matching algorithm for sewer inspection robots. *IEEJ transactions on electrical and electronic engineering 3*, 4, 441–448

[10] Koodtalang, W., Sangsuwan, T., and Noppakaow, B. 2018. A design of automated inspections of both shape and height simultaneously based on stereo vision and plc. In *18th International Conference on Control, Automation and Systems (ICCAS)*. IEEE, 1290–1294

[11] Gunatilake, A., Piyathilaka, L., Kodagoda, S., Barclay, S., and Vitanage, D. 2019. Real-time 3d profiling with rgb-d mapping in pipelines using stereo camera vision and structured ir laser ring. In *14th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. IEEE, 916–921

[12] Huynh, P., Ross, R., Martchenko, A., and Devlin, J. 2015. Anomaly inspection in sewer pipes using stereo vision. In *IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*. IEEE, 60–64

[13] Huynh, P., Ross, R., Martchenko, A., and Devlin, J. 2016. 3d anomaly inspection system for sewer pipes using stereo vision and novel image processing. In *IEEE 11th Conference on Industrial Electronics and Applications (ICIEA)*. IEEE, 988–993

[14] Meijer, D. W., Kesteloo, M., and Knobbe, A. J. 2018. Unsupervised anomaly detection in sewer images with a PCA-based framework. In *International Conference on Pattern Recognition and Artificial Intelligence (ICPRAI)*. 354–359

ues for an unsupervised approach, and compared this with a convolutional autoencoder. Myrans et al. [15] performed anomaly detection on sewer CCTV images by training a random forest and a support vector machine on GIST-features. Myrans et al. [16] later expanded on this by exploring the use of a one-class support vector machine, a type of support vector machine designed specifically for anomaly detection. Moradi et al. [17] similarly used a one-class support vector machine to detect anomalies from SIFT features, and combined this approach with localization of the pipe through text recognition. Fang et al. [18] performed anomaly detection on sewer CCTV video footage by performing principal component analysis on various local feature descriptors. Russo et al. [19] use a convolutional autoencoder to detect anomalies in CCTV images.

While numerous other works that use computer vision or image processing to detect defects in sewer pipe images exist, we have limited this section to only those that perform *unsupervised anomaly detection*, as they are most similar to this work. For a more broad perspective on recent advances in this field, please refer to Haurum and Moeslund [20].

[15] Myrans, J., Everson, R., and Kapelan, Z. 2018. Automated detection of faults in sewers using cctv image sequences. *Automation in Construction 95*, 64–71

[16] Myrans, J., Kapelan, Z., and Everson, R. 2018b. Using automatic anomaly detection to identify faults in sewers. In *WDSA/CCWI Joint Conference Proceedings*. Vol. 1

[17] Moradi, S., Zayed, T., Nasiri, F., and Golkhoo, F. 2020. Automated anomaly detection and localization in sewer inspection videos using proportional data modeling and deep learning–based text recognition. *Journal of Infrastructure Systems 26*, 3, 04020018

[18] Fang, X., Guo, W., Li, Q., Zhu, J., Chen, Z., Yu, J., Zhou, B., and Yang, H. 2020. Sewer pipeline fault identification using anomaly detection algorithms on video sequences. *IEEE Access 8*, 39574–39586

[19] Russo, S., Disch, A., Blumensaat, F., and Villez, K. 2020. Anomaly detection using deep autoencoders for in-situ wastewater systems monitoring data. *arXiv preprint arXiv:2002.03843*

[20] Haurum, J. B. and Moeslund, T. B. 2020. A survey on image-based automation of cctv and sset sewer inspections. *Automation in Construction 111*, 103061

5.3 Framework

We propose a framework for anomaly detection from stereovision measurements in sewer pipes, as shown in figure 5.1. The framework consists of five major steps, designed to be executed in sequence, each step's output being the next step's input. The five steps in the framework are each discussed in detail in sections 5.3.1-5.3.5.

5.3.1 Image Acquisition

The stereovision setup requires two cameras placed side-by-side, at equal height, pointed in the same direction. Perfect

alignment of the cameras is near impossible, but correcting a slight misalignment is possible. The setup is then directed into the pipe, such that the camera axes are mostly parallel to the pipe axis. For in-situ inspection, this means that the setup has to be attached to the pipe inspection vehicle, while aimed directly into the pipe.

Any optical lens introduces some radial distortion to an image [21], meaning that points at different distances from the lens axis have different levels of magnification. As the lens axes of the two cameras are parallel but translated, this may introduce a difference in magnification of a point between the two cameras, and thereby a difference in vertical position. Depending on the severity of this distortion (or if the cameras and lenses are not of identical make), correction may be required for the images to be suitable for stereo matching.

[21] HECHT, E. ET AL. 2002. *Optics*. Vol. 5. Addison Wesley San Francisco

By taking pictures of a chessboard pattern from different angles and distances, we can observe the effects of the radial distortion: without any distortion, the lines on a chessboard should be entirely straight, but slight curves may appear as a result of the radial distortion. Radial distortion can be reversed digitally by performing a second radial distortion to undo the first. The correct inverse distortion parameters can be estimated from the deviations in the images of the chessboard pattern, as outlined in more detail in [22].

[22] WU, Y., JIANG, S., XU, Z., ZHU, S., AND CAO, D. 2015. Lens distortion correction based on one chessboard pattern image. *Frontiers of Optoelectronics 8,* 3, 319–328

Once images from both cameras are free of (extreme) lens distortion, an alignment between the cameras must also take place, in order to compensate for a vertical misalignment or rotation of one camera around its axis. From a set of images with several visible landmarks (the same images of the chessboard pattern may be used), we can estimate any vertical shift between the images, as well as a possible rotation along the camera axis, and correct this with a simple affine transformation [23]. If the camera axes themselves are not perfectly aligned, this may be visible as a horizontal shift of points that are very far away, as a point on the horizon

[23] BROWN, L. G. 1992. A survey of image registration techniques. *ACM computing surveys (CSUR) 24,* 4, 325–376

should in theory have the same position in both images.

After camera alignment, the first step is complete and the images can serve as input for the second step: semi-global stereo matching.

5.3.2 Stereo Matching

As described in section 2.4, stereo matching relies on comparing projected locations of a three-dimensional point onto two-dimensional images. Stereo matching an image is generally done by comparing positions in the reference image to horizontally shifted positions in the second image, often to sub-pixel accuracy [24]. The shift that best matches a pixel in the reference image to a pixel in the second image is called the *disparity* for that pixel. The comparison may be done by minimising the difference in values of the pixels, but better results may be obtained by using a matching cost that relies less on absolute values, such as cross-correlation, Hamming distance, or Birchfield-Tomasi dissimilarity [25].

Matching single pixels from the two images is going to lead to a substantial amount of incorrect matches. Suggested solutions for this include matching a window around each pixel to a window of the same size, enforcing some type of smoothness between disparity in neighbouring pixels, and various combinations thereof [26].

Unique to our problem is the fact that the sewer pipe axis is parallel to the camera axes. The surface we are most interested in, the pipe wall, is perpendicular to the image plane. This causes the Z-distance and disparity to gradually change and not be constant anywhere inside the pipe. Window-based stereo matching methods are designed to perform best when large patches of the reference image have the same disparity, which is not the case in this scenario.

This gradual change requires the window around the pixel that is to be matched to be small (we suggest < 10 pixels on either side). The larger the window is, the more

[24] Hirschmüller, H. and Gehrig, S. 2009. Stereo matching in the presence of sub-pixel calibration errors. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 437–444

[25] Birchfield, S. and Tomasi, C. 1998. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence 20,* 4, 401–406

[26] Lazaros, N., Sirakoulis, G. C., and Gasteratos, A. 2008. Review of stereo vision algorithms: from software to hardware. *International Journal of Optomechatronics 2,* 4, 435–462

difficult it will be to match it properly. The extremes of the window are expected to have a different disparity from the centre pixel, so too large a window will be impossible to match correctly.

To enforce smoothness, we suggest using Hirschmüller's *semi-global matching* algorithm [27], which adds two regularisation parameters. These parameters, P_1 and P_2, penalise a pixel having a different disparity from its neighbouring pixels. The best match for each pixel in the reference image is chosen as the match that minimises the sum of the matching cost M and the regularisation cost R. This introduces a circular dependency, as the regularisation cost depends on the best match of neighbouring pixels, which itself depends on the regularisation cost. Because of this, the algorithm usually requires multiple passes to stabilise.

[27] HIRSCHMÜLLER, H. 2005. Accurate and efficient stereo processing by semi-global matching and mutual information. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 2. IEEE, 807–814

Each neighbouring pixel contributes 0, P_1, or P_2 to the regularisation cost, depending on whether the neighbour has the same disparity, an absolute difference in disparity of 1, or a larger difference in disparity with the current pixel, respectively. Specifically, each match is given the regularisation cost:

$$R(d_1, d_2) = \begin{cases} 0 & \text{if } d_1 = d_2 \\ P_1 & \text{if } |d_1 - d_2| \leq 1 \\ P_2 & \text{if } |d_1 - d_2| > 1 \end{cases} \tag{5.1}$$

Again taking into account the fact that we expect the disparity to change gradually, we suggest setting $P_1 \ll P_2$. A value of $P_1 = 0$ may prove successful, but could also lead to more erratic results. With a small or no penalty on small differences in disparity, but a large penalty on larger disparities, we can enforce the type of smoothness that we expect in sewer pipe images. The best value of P_2 depends on the window size chosen for matching, the matching cost itself, and the number of neighbours that the semi-global matching algorithm considers (commonly 4 or 8).

To reduce false detections further, we also suggest using

a *uniqueness ratio*, which requires that the best disparity must have a score that is at least u times as large as the next best disparity. This may lead to correct disparities also being discarded, but this happens most commonly in very smooth regions, where the exact disparity is difficult to pinpoint anyway. For the purposes of anomaly detection, this is not an issue, as these regions are unlikely to contain anomalies.

With an accurate estimation of the disparity for each pixel, these disparities can be used to reconstruct a three-dimensional point cloud.

5.3.3 Three-Dimensional Geometry Reconstruction

As outlined in section 2.4, we can triangulate the three-dimensional location of a point visible in both cameras once we know the disparity, using equations (2.26), (2.27), and (2.28). Equations (2.27) and (2.28) may be rewritten as

$$X = (\mathrm{x} - \mathrm{x}_0) \cdot Z/f \tag{5.2}$$

$$Y = (\mathrm{y} - \mathrm{y}_0) \cdot Z/f \tag{5.3}$$

Where (x, y) is the pixel position of the triangulated point in the reference image, and $(\mathrm{x}_0, \mathrm{y}_0)$ is the pixel position of the centre of the reference image. This means that the centre of the image will be projected to some position on the Z-axis, both the X and Y coordinates of the point being zero.

Doing this for every pixel in the image (that has a valid disparity) gives us a point cloud with one point for every pixel. It may be useful to keep the RGB values of the pixels attached to the points in the point cloud for easier inspection and later processing. But a few important caveats arise when we move away from the ideal purely mathematical situation as introduced in section 2.4 though.

The measurement of baseline b will have some non-zero error, which leads to a scaling of the entire point cloud by a factor of $\frac{b'}{b}$ where b' is the measured baseline and b the actual baseline. The more accurately the baseline is measured, the closer to 1 this scaling factor is. It should be noted that if correct physical dimensions of the point cloud are not important to the application, this does not have to be taken into consideration.

If the camera axes are not entirely parallel in the epipolar plane, the calculated disparity will have a small error. This may lead to a perceived deviation in radius along the length of the pipe, meaning a cylindrical pipe may appear conical in the point cloud.

While these are both issues to be aware of, they do not hinder the pipe model we propose in this work.

5.3.4 Robust Pipe Surface Fitting

At this point, we have a three-dimensional point cloud of a pipe which can be used to estimate the original pipe geometry as a mathematical model, excluding any anomalies (henceforth simply referred to as the 'geometry').

While the image will be perfectly in-focus at a specific distance, a region known as the *depth of field* around this distance is determined to be the range with *acceptable* levels of focus. The distance range of the depth of field is a simple function of the focus distance, the focal length, and the aperture size, which is adjustable in most cases. A smaller aperture size will give a larger depth of field, at the cost of less light reaching the camera sensor, leading to more sensor noise at equal exposure times [28]. To inspect an entire pipe, we might move the pipe inspection vehicle through the pipe at small intervals, taking measurements at each interval. This means that a larger depth of field leads to fewer measurements needed to process a unit length of pipe, as a larger portion of the pipe can be captured in a single

[28] Salvaggio, N. 2009. *Basic photographic materials and processes*. Taylor & Francis

photograph. This results in a point cloud of a pipe, centred approximately around the Z-axis. Points with a Z value outside the depth of field can be discarded, as we are better off estimating the geometry of such points when the cameras are positioned at a different position along the pipe.

A transformation to cylindrical coordinates at this point allows for a more natural notation of the geometry. We define:

$$r = \sqrt{X^2 + Y^2} \tag{5.4}$$

$$\phi = \arctan_2 (Y, X) \tag{5.5}$$

where $\arctan_2$ is the two-argument arctangent, which spans the interval $(-\pi, \pi]$. We can now without loss of information express each point in (r, ϕ, Z) coordinates.

A naive approach at this point might be to fit a cylinder model of

$$r = r_0 \tag{5.6}$$

to capture the geometry of the pipe, where r_0 is the inner radius of the pipe. There are a few reasons why this is a poor approach:

i. The Z-axis may not be the precise centre of the pipe, depending on how accurately it was possible to align the reference camera's axis with the pipe axis.

ii. The pipe might not have a circular profile. In our experiments we use both cylindrical and egg-shaped pipes, but any pipe with a somewhat smooth profile should work with our approach.

iii. The radius and centre of the pipe may appear slanted in the point cloud along the Z-axis as a result of a slight misalignment of the camera axes in the epipolar plane.

We can address each of these issues in order.

To address the first issue, we assume for now that the pipe is cylindrical along the Z-axis, but not perfectly centred. Using a polar coordinate representation of an off-centre circle, we may express the geometry as

$$r = \sqrt{r_0^2 - d^2 \sin^2(\phi - \rho)} + d\cos(\phi - \rho) \qquad (5.7)$$

where d is the distance between the axis of the pipe and the Z-axis, and ρ is the angle at which the distance to the Z-axis is maximal. It can be observed that if $d \ll r_0$ (the centre of the pipe is close to the centre of the image), we may simplify equation (5.7) to

$$r \approx r_0 + d\cos(\phi - \rho) \qquad (\text{for } d \ll r_0) \qquad (5.8)$$

At this point, we take a small sidestep to rewrite equation (5.8) using a trigonometric identity into

$$r = r_0 + a\sin(\phi) + b\cos(\phi) \qquad (5.9)$$

It can be seen that these two forms are identical when

$$d = \sqrt{a^2 + b^2} \qquad (5.10)$$

$$\rho = \arctan_2(b, a) \qquad (5.11)$$

The reason for this rewrite is entirely practical: we still have two unknowns to solve for, but both unknowns are now parameters of a linear function, meaning that we can now solve a and b with Ordinary Least Squares regression [29], whereas that would not be possible for the parameter ρ, because it is inside a cosine in equation (5.8).

[29] GOLDBERGER, A. S. 1964. *Classical Linear Regression, Econometric Theory.* New York: John Wiley & Sons

To address the second issue, the possibility of pipes with non-circular profiles, we need a more complex function to describe the radius r as a function of the angle ϕ. To prevent modeling possible defects into the geometry, making them impossible to detect as anomalies, we use a limited approximation of r in terms of functions of ϕ. Different approximations can be used, but we suggest the use of a

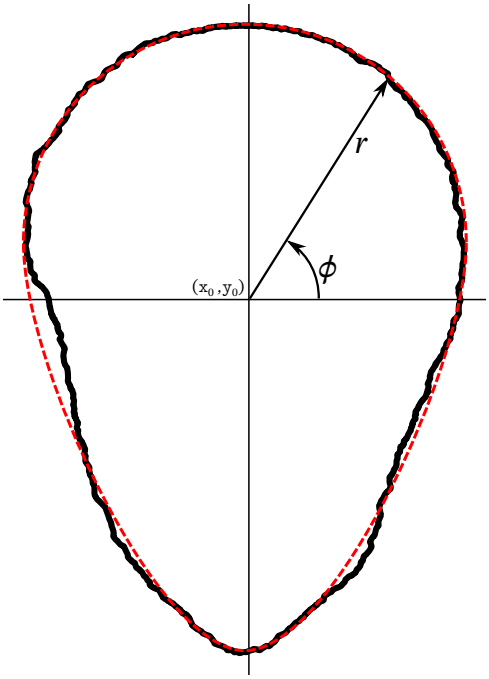


Figure 5.2: Cross-sectional profile of a (fictional) deformed 'egg'-shaped pipe (black, solid), with a fit of equation (5.12) for $K = 6$ (red, dashed).

Fourier series approximation, as the radius is inherently periodic as a function of the angle. [30] We redefine the model as

$$r = r_0 + \sum_{k=1}^{K} \Big(a_k \sin(k\phi) + b_k \cos(k\phi) \Big) \qquad (5.12)$$

where K dictates how many Fourier components are used to approximate the radius. It may be seen that equation (5.9) is an instance of equation (5.12), with $K = 1$. For egg-shaped pipes, we find a value of $K = 6$ to be generally sufficient, but any pipe profile with corners or otherwise non-smooth sections may require a higher value of K. Figure 5.2 illustrates how an egg-shaped pipe may be expressed in these Fourier components.

To address the third and final issue, we allow the radial parameters to change along the Z-axis, that is, along the pipe

[30] Angles are periodic by definition:

$$\phi \equiv \phi + 2\pi$$

axis. To account for both a translation and scaling of the profile along the Z-axis—corresponding to a misalignment of camera axis and pipe axis, and a measurement error in the baseline distance, respectively—we allow each of the previously introduced parameters to vary linearly along the Z-axis. For every term in equation (5.12), we add another term with a different parameter, and multiply by Z, giving us:

$$r = (r_0 + \rho_0 Z) + \sum_{k=1}^{K} \begin{pmatrix} a_k \sin(k\phi) + b_k \cos(k\phi) + \\ \alpha_k Z \sin(k\phi) + \beta_k Z \cos(k\phi) \end{pmatrix} \tag{5.13}$$

Equation (5.13) is the model we will use to fit the transformed point cloud data, but as we expect anomalies, we will have to employ a robust regression method. Note that, somewhat surprisingly, this model is linear in terms of the parameters to be fit, such that Ordinary Least Squares (multiple) regression can be applied.

The robust regression method we use is RANSAC, short for 'random sample consensus' [31]. RANSAC fits a model a large number of times on a 'minimal subset' of data, then selects a model fit that has both a large amounts of inliers, and a low error rate for those inliers. In this context, a minimal subset is the minimum number of points we need to fit the model. As our model has $2 + 4K$ parameters, we need as many points.

[31] FISCHLER, M. A. AND BOLLES, R. C. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM 24,* 6, 381–395

Then we determine the inliers, the points that are accurately described by this fit, according to some maximum difference between the actual value of r and the one predicted by the fit, known as the *inlier threshold*. If the number of inliers meets a set minimum, the model is fit a second time, but on all inliers this time. We store this new fit, along with the error rate on its inliers. This process is repeated a large number of times, after which we select the fit with the lowest error rate on its inliers.

The value of the inlier threshold will depend on the vari-

ance of the predicted variable. The minimum number of inliers required will usually be defined as a percentage of all data points, depending on the ratio of anomalous data points we expect to have. The chance that a minimal subset will result in a good fit of the data is low, but this first fit is only used to determine which points are the inliers that we want to perform the second fit on. Depending on how likely it is that the first fit reaches the minimum number of inliers under the chosen inlier threshold, the number of times the process should be repeated can differ by orders of magnitude: 10, 100, or 1000 could all be reasonable numbers.

If we choose the RANSAC algorithm parameters reasonably, this should give us a fit of the model described in equation (5.13) that accurately describes a large portion of the data points, while not taking the actual anomalies into account.

5.3.5 Anomaly Detection and Processing

The final step in the framework, anomaly detection, is trivial at this stage: the (absolute) difference between the actual value of r and the value predicted by the best fit is an 'anomaly score'. We might threshold these scores to distinguish anomalies from non-anomalies, or consider the scores themselves as a continuous indicator.

Depending on the context in which the framework is employed, we have several suggestions for further processing of the anomaly scores:

- ◇ the ratio of anomalous pixels to regular pixels may be an aggregated indicator of anomalousness of a length of pipe,
- ◇ for further human inspection, the anomaly scores can

be visualised in either an interactive point cloud or the original images, to indicate areas that might warrant attention,

- ◇ if pixel-wise classification is the goal, the anomaly scores can augment the RGB values of a pixel in a subsequent classifier.
- ◇ the size of connected anomalous regions, as well as the absolute anomaly scores in such regions, might be used for defect identification or even severity.

In the experiments performed for this paper, we have calculated a global anomaly score per image set as follows:

$$A = \frac{1}{N} \sum_{i=1}^{N} \| \min(r_i - \hat{r}_i, 0) \| \tag{5.14}$$

where r_i is the radius of a point in the point cloud and $\hat{r}_i$ is the predicted radius for that point. The point anomaly scores are clipped into the range $(-\infty, 0]$ and we calculate the average absolute value over all points in the point cloud. The point anomaly scores are clipped to only negative values, as otherwise the many points outside of the inner pipe wall present in our point clouds would skew the global anomaly score. This clipping would not be necessary in *in-situ* inspections, because no points outside the pipe should be visible, except if caused by defects.

5.4 Experimental Setup

We evaluated the efficacy of the framework in a laboratory experiment. A total of 26 sewer pipe segments in various conditions were photographed with a stereocamera setup. Two Basler Ace2 A1920-160umBAS area scan cameras were fitted with lenses with a 16 millimeters focal length and attached side by side to a metal plate. The baseline was determined to be 29 millimeters, the lenses were focused at

approximately 1.5 meters distance, and the lens aperture was set to an f-number of 6, meaning the aperture diameter was equal to $16/6 \approx 2.667$ millimeters. A single pixel of an object at the in-focus distance corresponds to approximately 2.2 millimeters in real-world coordinates. In the ideal circumstances the algorithm can detect a shift of 1/16th of a pixel, so the maximum sensitivity we can expect to achieve is in the order of 1/10th of a millimeter.

The plate was attached to a rail, to allow for movement of the setup along the camera axis. The cameras were directed into the sewer pipe, which was covered at both ends with a piece of cloth. The pipe segments were illuminated with an LED light placed behind the cameras. The entire setup is depicted in figure 5.4.

22 of the 26 sewer pipes were photographed from both ends, the other 4 sewer pipes were photographed from one end only, giving us a total of 48 image sets.

5.4.1 Image Data

Figure 5.4.1 shows examples of stereo images sets of the sewer pipes as obtained with the experimental setup. Subfigure 5.4(a) shows a typical naturally aged cylindrical pipe, containing plenty of texture for the stereo matching. Subfigure 5.4(b) shows a typical naturally aged egg-shaped pipe, the reason we need a non-cylindrical model. Subfigure 5.4(c) shows a new cylindrical pipe, lacking sufficient texture to accurately stereo match.

5.4.2 Implementation Details and Parameters

The framework was implemented in OpenCV 4.5 [32] and Python 3.6 [33]. Using the default implementation of semi-global stereo matching in OpenCV, we chose a search range

[32] Bradski, G. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*

[33] Van Rossum, G. and Drake, F. L. 2009. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA

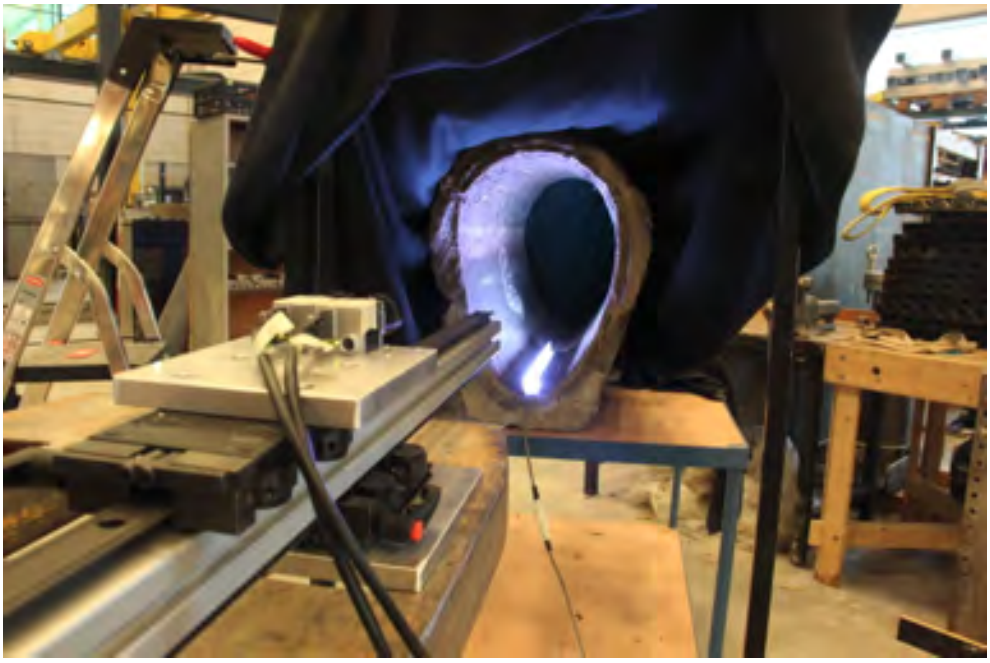


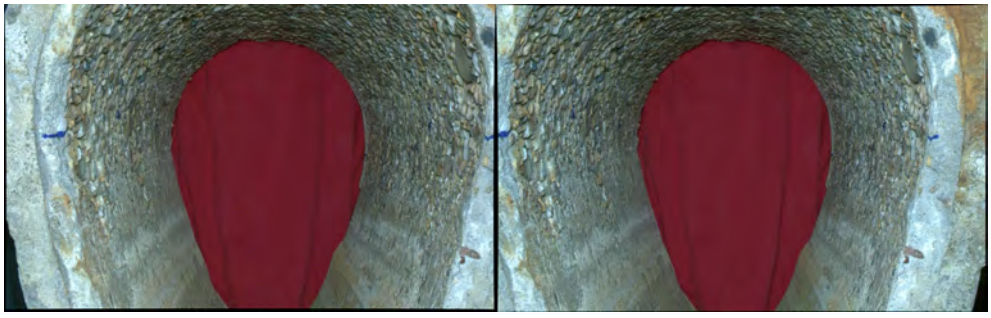
Figure 5.3: Experimental setup (not the final lighting setup).

between 20 and 220 pixels of disparity, used a block size of 7, set regularization parameters $P_1 = 100$ and $P_2 = 10,000$, and used a uniqueness ratio of $u = 10$.

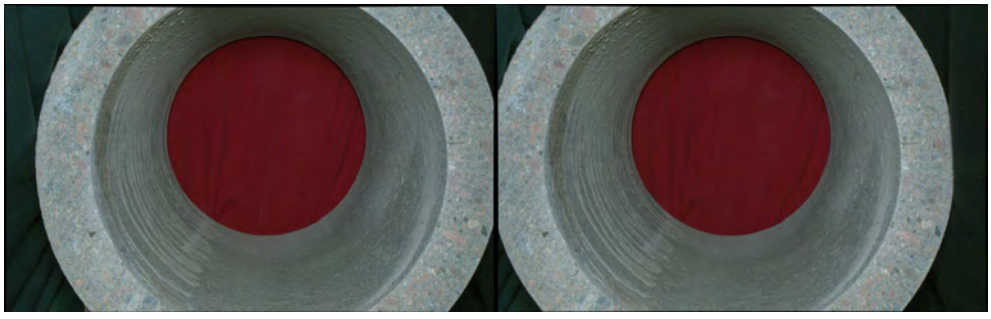
After stereo matching and geometry reconstruction, the red cloth background of the images is removed with a flood fill and a valid Z-range can be selected by the user, or a default range of $Z \in [1.5, 2.0]$ meters from the camera may be used.

After conversion to cylindrical coordinates, we fit the model in equation (5.13) with $K = 6$. RANSAC is run for 10 iterations, the initial fit is calculated on 50 randomly selected datapoints, inliers are determined by a maximum absolute difference of 0.005, and the second fit is calculated on the inliers if those inliers make up at least 90% of the datapoints. The fit with the lowest error on its inliers is

(a) A typical naturally aged cylindrical sewer pipe.



(b) A typical naturally aged egg-shaped pipe.



(c) A typical new cylindrical sewer pipe.

Figure 5.4: Three examples of stereo image sets as obtained with the experimental setup. All images have lens distortion correction, the left images also have a vertical translation correction according to the process described in section 5.3.1.

chosen, or if no initial fit had enough inliers, the entire process is repeated with 10 iterations.

The best fit from the RANSAC model is applied to all data points, including those outside the Z-range that the model was fit on, and the deviation from the fit is presented for visual inspection in both a point cloud and the reference image.

The code of our implementation is available to try as a demo. It can be found at:
`https://github.com/data-flux/StereoDemo`

5.5 Results and Discussion

5.5.1 Stereo Matching and Geometry Reconstruction

We start our discussion of the results by considering the first half of our approach, the stereo matching and geometry reconstruction. Getting an objective, unequivocal assessment of the produced point cloud is challenging, since we do not have a golden standard measurement of the 3D pipe geometry to compare the point clouds with. Because of this, we will mostly have to rely on subjective validation of the results. We asked a human assessor to judge each point cloud on how accurate and consistent the point cloud reconstructed the original 3D geometry, on a scale from 1 (very poor) to 10 (very accurate). Over the 48 image sets, an average rating of 8.4 was assigned, indicating that the 3D reconstruction was quite good. In figure 5.5.1, four examples of different types of pipes are given where the reconstruction was successful (average rating of 8.5). The left picture shows the left image of each stereo pair, and the right image shows the point cloud with points coloured by the colour from the original (left) image. The characteristics of the virtual

lens of the generated image deviate from the physical lens somewhat, and the virtual camera was deliberately placed somewhat further ahead. This allows the viewer to somewhat appreciate the 3D nature of the point cloud (rather than reproducing the images on the left without knowing the depth). Specifically, the forward camera position allows observing any occluded areas by 'seeing around' the humps on the pipe surface. Especially in the point cloud image subfigure 5.5.1(a), areas of occlusion behind the various deposits are clear. Viewed from the side, this point cloud has considerable gaps behind all deposits.

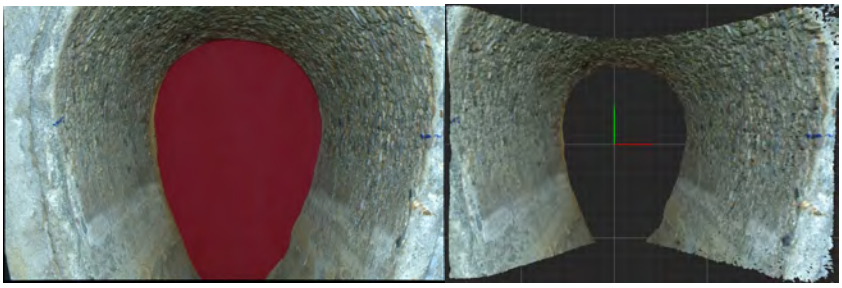
A minority of the pipes were not reconstructed correctly. Five image pairs (from four pipes) scored a value below 8 (average score 6), with the lowest being two scores of 5. What these five image pairs have in common, compared to the remaining successful image sets, is that they involve pipes with areas of smooth and monochrome surface, as can be seen in figure 5.5.1. Such areas are especially common in relatively new pipes which do not show much deterioration. The surface of such pipes will be hard to stereo-match, since few surface features can be used to match pixels in the stereo pair. The result is that entire patches of pixels have an undetermined depth. Further contributing factors to this poor matching are low lighting and lack of lens focus. With improved focus and lighting, the tiniest surface features may lead to proper matching again.

5.5.2 Surface Fitting and Anomaly Detection

Next, we consider the quality of the surface fitting and subsequent anomaly detection steps of our framework on the 48 image sets. In figure 5.5.3, the four pipes of figure 5.5.1 are shown again, with the local anomaly score assigned to the derived point cloud, colourcoded as blue being an anomaly

(a)

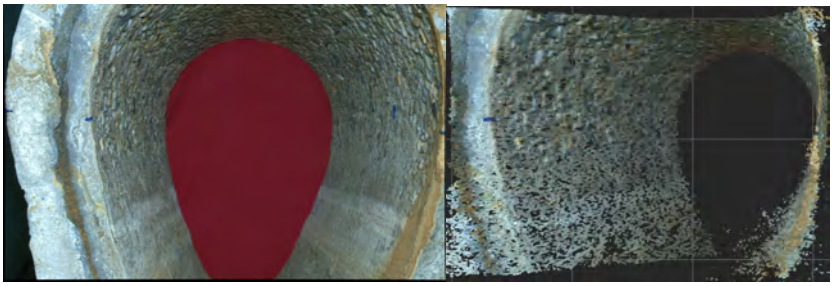
(b)



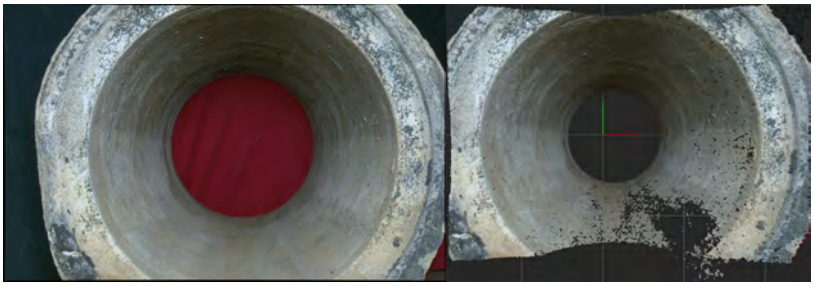
(c)

(d)

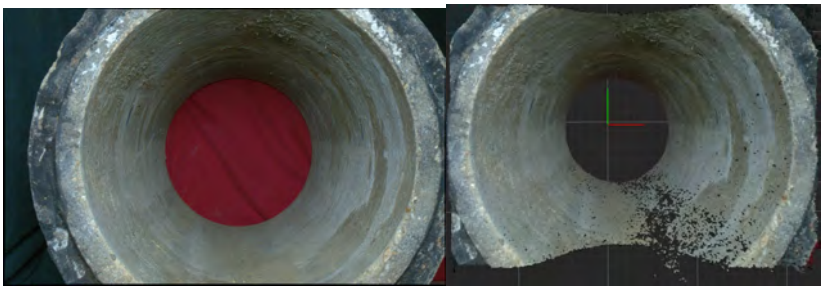
Figure 5.5: Four examples of the stereo matching. Left shows the reference image, right shows the reconstructed point cloud.



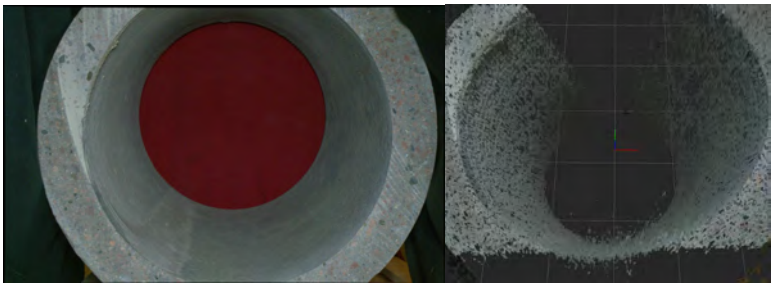
(a)



(b)



(c)



(d)

Figure 5.6: Images of pipes that showed problems with stereo matching, due to patches of pixels with mostly constant surface colour.

score of zero, red being an anomaly score of 10 mm. As can be seen, most of the clear deposits present are identified by the anomaly detection method. In subfigure 5.5.3(a), all of the clear deposits on the left have been identified, and also some of the less obvious deposits on the right can be made out, for example on the low-right at half-depth. Additionally, some of the surface roughness is indicated at the top. Note also the large occlusion areas, which do not play a role in our detection algorithm, but further strengthen the identified anomalies. The pipe in subfigure 5.5.3(b) shows some deposits hanging from the top of the pipe, as well as considerable unevenness in the surface texture throughout the pipe that is also captured by the framework as local anomalies. In this pipe, however, we also see evidence of some false positives in the detection, in the lower far corners of the pipe. It appears that here, the point cloud contains too little data in the lower regions (closer to the camera, the field of view does not contain the bottom of the pipe), such that the model parameters are possibly inaccurate. In this particular area, there indeed appear to be some deposits around the flood line, but not to the extent indicated by the model. This phenomenon, that also plays a small role in subfigure 5.5.3(c), appears to be an artefact of the (in hindsight somewhat unfortunate) choice of lens that doesn't allow a full view of the pipe. The problem is easily remedied by removing the near and far end of the pipe and focussing on the middle band of the point cloud, which incidentally also concerns the pixels with the best focus. Remember that in *in-situ* inspections, the camera will be slowly inched forward through the pipe, allowing for a complete sweep of the pipe. Our framework can thus focus on the band of data where results are the most reliable.

Of the 48 image sets, a total of six images could not be fitted properly with our RANSAC method (not reaching the required fraction of inliers). Five of these concern the cases mentioned in the previous section as suffering from

a poor stereo matching. The resulting point cloud has a non-negligible number of points that erroneously lie inside the pipe, reducing the number of inliers. The sixth pipe that could not be fitted properly had an entire section broken off, presumably during extraction from the soil. The remaining 42 cases (87.5%) were properly fitted, producing the marked point clouds demonstrated in figure 5.5.3, as well as a single global anomaly score per image, as defined in equation (5.14). Prior to validating the anomaly detection, the photographs of the pipes were also graded subjectively in terms of defect severity, to have an independent ground truth to compare the detected anomalies levels with. For the 42 cases where our framework produced a global anomaly score, a Pearson's correlation of $r = 0.65$ with the ground truth was obtained, which is, according to the customary interpretation, a moderate, positive correlation.

5.5.3 Discussion

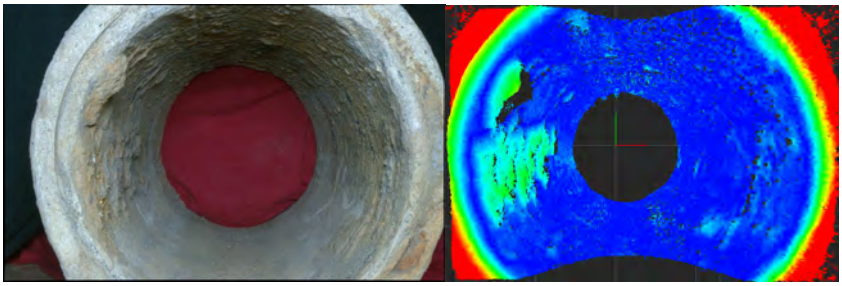
Although our framework demonstrates good results on the two defect types of exposed granulate and deposits, not all pipes are correctly assessed. The main weakness of our method appears to revolve around the images with smooth pipe surfaces. The problem with these pipes occurs in the first part of our framework, the stereo matching, which indeed is known to be problematic in images with limited texture. The upside of this limitation is that pipes with smooth surfaces (often fairly new pipes) typically do not contain any defects. The main problem here is hence one of false alarms: the method sometimes erroneously identifies defects in smooth pipes because points are incorrectly placed in the 3D space. In future work, we may investigate whether stereo matching could also produce a confidence score, indicating the quality of the stereo matching in each region of the image. If successful, the method would only

identify a defect if both the confidence is high (in other words, not a smooth surface) and the anomaly score is high.

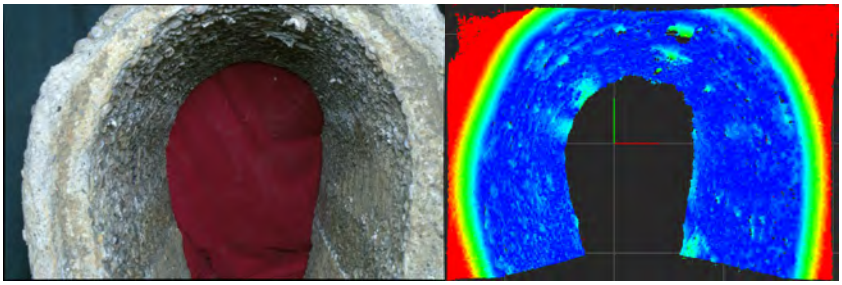
Whenever the stereo matching produces at least a reasonable result, the surface fitting and anomaly detection correctly identifies the various defects present. It should be noted that the framework even identified defects that were overlooked in the initial subjective quality grading of single images (not stereo pairs). In that sense, other than merely automating parts of the inspection process, our framework also has the potential to outperform human inspectors in certain respects.

Our experiments were performed in the lab, which may have had some minor effects on the outcome, although both in a positive and a negative sense. On the positive side, our experiments were perhaps made more challenging than necessary due to some initial choices that ended up being suboptimal. For example, the chosen lenses were not of sufficiently wide angle, such that the pipe could not be captured within the region of focus entirely, especially for egg-shaped pipes. The effect of this on the results is that for some regions of the pipe, such as the corners of the point clouds corresponding to the edges of the image, not enough evidence of the regular geometry is available in order to reliably decide on deviations from that geometry. This effect, which may cause both false positives and false negatives locally, can be observed in subfigure 5.5.3(c). This problem can be easily corrected by using a lens with a wider field of view. Another unfortunate choice concerns the low lighting conditions, which could have been corrected with a longer exposure of the images. Low lighting has not played a significant role (as the mostly good results demonstrate), but may have contributed to the lack of matching in areas with smooth surfaces.

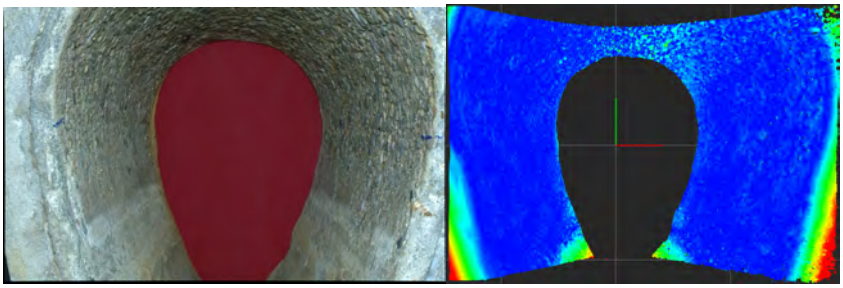
Another property of the lab set-up was that we search for anomalies in the entire pipe, which is not necessary in the field, and produces some challenges with limited focal depth and missing parts of the pipes (due to extraction). In



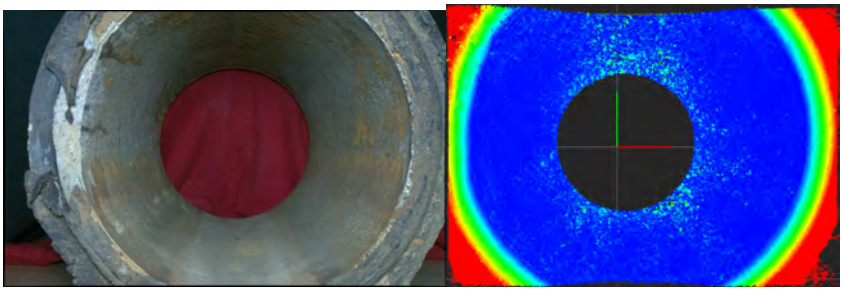
(a)



(b)



(c)



(d)

Figure 5.7: Examples of four anomaly scores. Left shows the reference image, right shows the identified anomalies. Note that the anomalies on the edges and outside of the pipes are ignored.

a more practical setting, in an actual *in-situ* pipe inspection, a single image pair would only be inspected for a somewhat narrow band, corresponding to the region of focus. After that, the pipe inspection vehicle would move forward by a small distance and the process would be repeated. Note that although our framework makes no such assumption, being in pipes of mostly the same geometry would allow one to assume a certain steady geometry, and deviations from this could be more easily recognised.

On the negative side, our lab set-up is less realistic as we only inspect individual pipe segments, not longer pipe systems. As a result, we have no images of pipe joints, which in most cases would produce a (false positive) deviation from our fitted surface. Although we expect the pipe joints to be easily matched by the stereo matching, and proper handling of these 'acceptable anomalies' would not be difficult, our current method does not include such facilities, nor have we been able to test this. As future work, it would be interesting to develop a method that uses 3D point clouds of joints in order to recognise joint-related defects such a misaligned joints or signs of leakage. A final difference between our setting and actual sewer systems is that the inspected pipes, after having been washed out, will be wet, whereas our lab pipes were recorded in a dry state. We do not expect this difference to have a significant effect on the results, but would need to set up new experiments to assess this. The experiments were performed prior and during the COVID-19 lock-down, and some of the pipes were subsequently part of destructive full-scale testing. As such, we cannot easily redo the experiments.

In future experiments it may be interesting to look at discontinuities in the points on the pipe surface as well, as these may be caused by occlusions. An occlusion of a portion of the pipe surface may be just as informative as the geometry of the visible parts of the pipe surface, but could be indicative of foreign objects present in the pipe, such

as roots. However, because poor stereo matching may also result in such discontinuities, it may be advisable to require that such discontinuities exist only in the point cloud, and not in the disparity map.

5.6 Conclusion

5.6.1 Summary

In this chapter we have proposed RADIUS, an anomaly detection framework for sewer pipes based on computer stereovision. The framework consolidates several successful techniques into a sequential process, to allow for anomaly detection in an automated fashion from stereo photographs without intermediate user input. We performed experiments to demonstrate the efficacy of the framework and conclude that it is successful in detecting defects present in physical pipes as anomalies in the three-dimensional geometry, and moves the state of the art closer towards fully automated sewer asset management.

5.6.2 Limitations

The major limitation of this work is the varying quality of the data obtained in the lab, as a result of our limited experience with the hardware. Some of the images were made in poor lighting conditions and without proper camera calibration. Repetition of the experiments was not possible due to time and budget constraints, and because a part of the pipes had been subjected to destructive full-scale testing in other experiments. A secondary limitation related to this is the total number of experiments performed.

A more intrinsic limitation of the approach of stereovision is that smooth, undamaged concrete may not contain

enough texture or markers to accurately match the reference image to the secondary image. While this is potentially an issue for a large portion of sewer pipes, we argue that such regions are unlikely to contain any anomalies. That said, since the absence of a match may also be an indicator of occlusion, we advise authors of future research to distinguish causes of a lack of a match: in the case of a too smooth pipe, the cause is likely a match that does not meet the uniqueness ratio required, as opposed to a patch in the reference image that does not appear in the secondary image due to occlusion.

In spite of these limitations, we feel the efficacy of the framework has been more than adequately demonstrated.

5.6.3 Recommendations

While anomaly detection may be a goal in itself, we hold the view that it is a stepping stone towards fully automated sewer condition assessment. To this end, we recommend future research to be performed into a follow-up step for the proposed framework: automated classification of the anomalies into defect classes. We feel that (the deviation from) the surface found through robust regression has a lot of potential for classification, as it will theoretically contain very little noise, as well as have a notion of "expected" behaviour.

It must be noted that while we have shown stereovision to be a viable tool for sewer pipe defect classification, the added value in practical settings has yet to be demonstrated. We have designed this framework to be compatible with current inspection practices: (monovision) CCTV inspection can still be performed while collecting data from two camera sources for stereovision experiments. This data may be used in parallel in order to both inch the industry towards automation of inspections, as well as to improve manual inspection techniques with an additional mode of data. State-

of-the-art sewer defect detection solely based on CCTV data may suffer from a relatively large false positive rate [34], but the additional depth information provided by an additional camera could lower this significantly. While the ambition of automated inspection is currently en vogue (again), the added value of multi-sensor inspection for more reliable, precise, and complete detection of a range of observable sewer defects, is an important added value worth researching further.

[34] Meijer, D. W., Scholten, L., Clemens, F. H., and Knobbe, A. J. 2019. A defect classification methodology for sewer image sets with convolutional neural networks. *Automation in Construction 104*, 281–298

6 Discussion and Conclusions

This thesis has explored applications of machine learning and computer vision to automate and enrich urban drainage inspections. This chapter will provide a conclusion to the thesis by answering the six research questions posed in chapter 1 and ending with some closing remarks.

Q1

What knowledge can be obtained from available inspection data without the utilization of expert classification, which might be inconsistent or unavailable?

In chapter 3, unlabeled sewer CCTV images were analysed with unsupervised learning. Image patches were classified as anomalous or non-anomalous, based on how common elements of the image patches were in the larger dataset. The use of image feature extractors and PCA decomposition allows us to detect anomalies based on the variance in the dataset they explain.

Urban drainage inspection is in fact a problem that lends itself well to unsupervised learning: because defects are very uncommon, they can be treated as anomalies in an anomaly detection problem more easily. The extreme class imbalance [1] works in our favour in this instance.

[1] Less than 1 % of images contain defects

It must be noted that these anomalies are not all defects, and that it is not trivial to separate defect and non-defect anomalies. The way we extract knowledge from unlabeled data with this approach is negative classification: images with no anomalous patches are very unlikely to contain defects. Because of this, such knowledge can be used as preselection for a later classification stage, regardless of whether that classification will be performed by humans or other algorithms.

Q2

How can the data collected with current inspection practices be analysed with machine learning techniques in order to improve processing efficiency and accuracy?

In chapter 4, we have trained a convolutional neural network to perform classifications of defects as human operators would. We have demonstrated that it may be possible to adequately perform future classification in an automated manner, provided enough varied training data is available. The problem is not a trivial one however. There is a rather extreme class imbalance and the human-labeled training data is known to have some errors.

The class imbalance leads to a dilemma: training a classification model without regard for the imbalance means the model might not adequately learn how to classify the extremely uncommon defects as they make up such a small portion of the variance present in the dataset. At the same time, adapting the training set or classification algorithm to make sure the under-represented class is properly classified introduces a bias that will decrease performance on the majority class in future, unclassified data.

And because the inspection quality is lacking, [2] the labels are not entirely reliable. The fact that the labels themselves are known to have errors leads to a limitation of a model's capabilities: we can scarcely expect the model to outperform its training data. The answer to the research question then, is that supervised machine learning could at best reach human parity with current data collection practices, which would provide an improvement to processing efficiency without (too much) reduction in quality. This in itself could be a more important improvement than it seems to be at first glance. Not only does a menial, repetitive task not have to be performed manually anymore, the consistency with which it is performed when automated is also increased.

[2] Dirksen, J., Clemens, F., Korving, H., Cherqui, F., Le Gauffre, P., Ertl, T., Plihal, H., Müller, K., and Snaterse, C. 2013. The consistency of visual sewer inspection data. *Structure and Infrastructure Engineering 9,* 3, 214–228

Q3

How do we assess the quality and operational impact of (partial) automation of the current inspection practices?

First and foremost, meaningful assessment requires that the data used to test the model be as realistic as possible: no rebalancing of datasets, no images of pipes that were present in the training set, no zooming and panning of the camera to better frame suspected defects. This might sound obvious in the context of this thesis, but published and peer-reviewed works have missed such crucial details in the past.

Secondly, and as discussed at length in chapter 4, commonly used quality metrics such as accuracy are of limited use for this problem. On the one hand, the extreme class imbalance makes some metrics difficult to interpret: an accuracy value of 99 % might look impressive in most cases, but we easily could achieve this by classifying every image as not containing a defect. On the other hand, because properly operating urban drainage systems are essential to public health and infrastructure, there are limits to the amount of false negative classifications that can be acceptable, regardless of the metric that is being used.

We have put forward two quality metrics specifically for this use case, that may provide more insight into the usefulness of classification models: the precision-at-recall and the specificity-at-recall. These metrics allow the user to define a minimum acceptable recall, and report the precision or specificity achieved if we tune the model output to achieve at least that recall value. Which of the two to use depends on the context: are we interested in knowing the amount of false positive detections as a fraction of all detections (use precision-at-recall), or as a fraction of all positives (use specificity-at-recall)?

To estimate operational impact, we need a clear picture of how the model will be used in practice. Assuming that false positive detections will be relatively common, as is to be expected with the extreme class imbalance, we might use the

specificity-at-recall to estimate a portion of data that may be discarded, such that we can still achieve the necessary true positive detections. In our research, we estimated that 60.5 % of images may be discarded as not containing defects for us to still be able to achieve 90 % detection of defects with the trained convolutional neural network. In practice this will mean a sizeable reduction in workload for this task.

To what extent are the current inspection practices automatable?

Q4

With the results obtained from the experiments outlined in chapters 3 and 4, we conclude that automation of current inspection practices is limited mostly by the available data and its quality. While we had significant amounts of image data available, these images pertained, as expected, mostly to pipes with few visible defects. In addition, the quality of data and accompanying metadata is inherently limited by the current inspection practices: not all defects can be accurately captured in CCTV footage; the defect registration standards are contentious; there seems to be limited consensus on defect severity when multiple experts independently review the same CCTV footage. [3]

Based on results obtained, we conclude that it may be possible to entirely automate current inspection practices, with a more sophisticated neural network model, provided enough images of each defect type are provided and enough time and energy is spent on the network hyperparameter optimization. Such automation would still be limited to achieving human parity in terms of quality, which is to say, less than perfect.

A proper followup question would then be: *"Should we aim to automate current inspection practices?"* Our answer to this question is that the short-term benefits of doing so might be too short-lived. The current inspection practices are outdated in terms of methodology and lagging decades behind in their use of available technology. [4] Energy, time,

[3] van der Steen, A. J., Dirksen, J., and Clemens, F. H. 2014. Visual sewer inspection: detail of coding system versus data quality? *Structure and infrastructure engineering 10,* 11, 1385–1393

[4] Tscheikner-Gratl, F., Caradot, N., Cherqui, F., Leitão, J. P., Ahmadi, M., Langeveld, J. G., Le Gat, Y., Scholten, L., Roghani, B., Rodríguez, J. P., et al. 2019. Sewer asset management–state of the art and research needs. *Urban Water Journal 16,* 9, 662–675

and money may be better spent developing new inspection workflows that are by design adaptable to future innovations such as machine learning pipelines or inspection techniques.

Q5

Does introducing depth information through computer stereovision improve the data quality and analysis capabilities?

In chapter 5 we outlined a method to recreate a three-dimensional point cloud of a sewer pipe through computer stereovision. It is immediately clear that *human* analysis capabilities of these point clouds, as compared to the images it was constructed from, are drastically increased. Inspecting the three-dimensional geometry in an interactive environment is a much easier task than inspecting a pipe based on two-dimensional images, especially for those not trained in recognizing defects from a two-dimensional image.

What we are of course more interested in, is the analysis capabilities with machine learning techniques. In the same chapter, we outlined a possible method of quantifying the degree of 'anomalousness' in a pipe, by using a robust regression method to reconstruct the original shape of the pipe. We found that this anomaly detection worked well on its own: the positions of the points in the point clouds gave us an anomaly score with a moderate, positive correlation with human-graded quality assessment of the pipes.

While the amount of data gathered in the chapter does not lend itself to a machine learning approach, we assume that the information present in the point cloud does not overlap entirely with the information in a single image. That is to say, if we would augment the data as used in chapter 4 or similar research to contain depth information from a second camera, this has a high chance to improve detection capabilities.

Q6

How can we employ machine learning and computer vision to improve the efficiency and quality of urban drainage inspections?

This thesis has provided a collection of *possible* methods to employ machine learning and computer vision to improve efficiency and quality of urban drainage inspections. We have provided examples of unsupervised learning, supervised learning, and 'classical' computer vision techniques to automate parts of the inspection process. As noted in the answer to research question 4, full automation is a while off, but applying the advances made in machine learning and computer vision to this specific problem can lead to short-term improvements in efficiency of inspections: large amounts of data may not need human classification, and for the parts that do, it might be possible to aid the human inspectors with data obtained from the machine learning and computer vision algorithms and improve the quality of their assessments.

6.1 Future Work

To speculate on possible future work that could stem from ours, we might consider a combination of the different techniques described in this work.

Unsupervised anomaly detection (as described in chapter 3) could be used as a pre- or post-processing step for convolutional neural network classification (as described in chapter 4). As a post-processing step, it might be used to estimate locations of defects within an image that was classified as containing defects. As a pre-processing step, it might be part of a semi-supervised learning approach, to select samples for active learning for example.

As mentioned in the answer to research question 5, introducing geometry information into a deep learning pipeline

has the potential to greatly improve results. While we did not have enough data to do so, adding a second camera to an inspection vehicle is inexpensive compared to more sophisticated 3D-scanning devices, and can in time provide enough data for neural network training. In this way we can imagine combining the techniques described in chapters 4 and 5.

We feel that a combination of all three techniques may be a significant step forward in the processing possibilities of urban drainage inspection data.

6.2 Closing Remarks

What this thesis has touched on is only a fraction of the possibilities for enrichment of urban drainage inspections with machine learning and computer vision. The field of urban drainage is only recently catching up on novel digital and virtual innovation, and the space for innovators to explore is virtually endless. Applying convolutional neural networks may have been an obvious first step that we and other researchers are collectively taking, but the next steps should be taken in terms of data collection and making this data accessible to researchers.

Bibliography

Ahrary, A. and Ishikawa, M. 2008. A fast stereo matching algorithm for sewer inspection robots. *IEEJ transactions on electrical and electronic engineering 3,* 4, 441–448.

Ahrary, A., Tian, L., Kamata, S.-i., and Ishikawa, M. 2005. An autonomous sewer robots navigation based on stereo camera information. In *17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05)*. IEEE, 6–pp.

Bahnsen, C. H., Johansen, A. S., Philipsen, M. P., Henriksen, J. W., Nasrollahi, K., and Moeslund, T. B. 2021. 3d sensors for sewer inspection: A quantitative review and analysis. *Sensors 21,* 7, 2553.

Baldi, P. and Hornik, K. 1989. Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks 2,* 1, 53–58.

Bengio, Y. 2012. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*. Springer, 437–478.

Birchfield, S. and Tomasi, C. 1998. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence 20,* 4, 401–406.

Bishop, C. M. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.

Bradski, G. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Brown, L. G. 1992. A survey of image registration techniques. *ACM computing surveys (CSUR) 24,* 4, 325–376.

Chae, M. J. and Abraham, D. M. 2001. Neuro-fuzzy approaches for sanitary sewer pipeline condition assessment. *Journal of Computing in Civil engineering 15,* 1, 4–14.

Chen, M., Shi, X., Zhang, Y., Wu, D., and Guizani, M. 2017. Deep features learning for medical image analysis with convolutional autoencoder neural network. *IEEE Transactions on Big Data*.

Dalal, N. and Triggs, B. 2005. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE, 886–893.

Dirksen, J., Clemens, F., Korving, H., Cherqui, F., Le Gauffre, P., Ertl, T., Plihal, H., Müller, K., and Snaterse, C. 2013. The consistency of visual sewer inspection data. *Structure and Infrastructure Engineering 9,* 3, 214–228.

Duran, O., Althoefer, K., and Seneviratne, L. D. 2007. Automated pipe defect detection and categorization using camera/laser-based profiler and artificial neural network. *IEEE Transactions on Automation Science and Engineering 4,* 1, 118–126.

European Committee for Standardization. 2003. En 13508-2: Condition of drain and sewer systems outside buildings, part 2: Visual inspection coding system, european norms.

Fang, X., Guo, W., Li, Q., Zhu, J., Chen, Z., Yu, J., Zhou, B., and Yang, H. 2020. Sewer pipeline fault identification using anomaly detection algorithms on video sequences. *IEEE Access 8*, 39574–39586.

Fischler, M. A. and Bolles, R. C. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM 24,* 6, 381–395.

Goldberger, A. S. 1964. *Classical Linear Regression, Econometric Theory.* New York: John Wiley & Sons.

Goodfellow, I., Bengio, Y., and Courville, A. 2016. *Deep learning*. Vol. 1. MIT press Cambridge.

Gunatilake, A., Piyathilaka, L., Kodagoda, S., Barclay, S., and Vitanage, D. 2019. Real-time 3d profiling with rgb-d mapping in pipelines using stereo camera vision and structured ir laser ring. In *14th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. IEEE, 916–921.

Guo, W., Soibelman, L., and Garrett Jr, J. 2009. Automated defect detection for sewer pipeline inspection and condition assessment. *Automation in Construction 18,* 5, 587–596.

Halfawy, M. R. and Hengmeechai, J. 2013. Efficient algorithm for crack detection in sewer images from closed-circuit television inspections. *Journal of Infrastructure Systems 20,* 2, 04013014.

Halfawy, M. R. and Hengmeechai, J. 2014. Automated defect detection in sewer closed circuit television images using histograms of oriented gradients and support vector machine. *Automation in Construction 38*, 1–13.

Haurum, J. B. and Moeslund, T. B. 2020. A survey on image-based automation of cctv and sset sewer inspections. *Automation in Construction 111*, 103061.

Hecht, E. et al. 2002. *Optics*. Vol. 5. Addison Wesley San Francisco.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. 2012. Improving neural networks by preventing co-adaptation of feature detectors.

Hirschmüller, H. 2005. Accurate and efficient stereo processing by semi-global matching and mutual information. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 2. IEEE, 807–814.

Hirschmüller, H. and Gehrig, S. 2009. Stereo matching in the presence of sub-pixel calibration errors. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 437–444.

Hoo-Chang, S., Roth, H. R., Gao, M., Lu, L., Xu, Z., Nogues, I., Yao, J., Mollura, D., and Summers, R. M. 2016. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging 35,* 5, 1285.

Howard, I. P. and Rogers, B. J. 2012. *Perceiving in depth, Volume 2: Stereoscopic vision*. Oxford University Press.

Huynh, P., Ross, R., Martchenko, A., and Devlin, J. 2015. Anomaly inspection in sewer pipes using stereo vision. In *IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*. IEEE, 60–64.

Huynh, P., Ross, R., Martchenko, A., and Devlin, J. 2016. 3d anomaly inspection system for sewer pipes using stereo vision and novel image processing. In *IEEE 11th Conference on Industrial Electronics and Applications (ICIEA)*. IEEE, 988–993.

IBAK Helmut Hunger GmbH & Co. KG. 2015. Panoramo® 3d optical pipeline scanner. `http://www.rapidview.com/panoramo_pipeline.html`. Accessed: 2018-12-05.

Koodtalang, W., Sangsuwan, T., and Noppakaow, B. 2018. A design of automated inspections of both shape and height simultaneously based on stereo vision and plc. In *18th International Conference on Control, Automation and Systems (ICCAS)*. IEEE, 1290–1294.

Krizhevsky, A. and Hinton, G. 2009. Learning multiple layers of features from tiny images. `https://www.cs.toronto.edu/~kriz/cifar.html`.

Kumar, S. S., Abraham, D. M., Jahanshahi, M. R., Iseley, T., and Starr, J. 2018. Automated defect classification in sewer closed circuit television inspections using deep convolutional neural networks. *Automation in Construction 91*, 273–283.

Lazaros, N., Sirakoulis, G. C., and Gasteratos, A. 2008. Review of stereo vision algorithms: from software to hardware. *International Journal of Optomechatronics 2,* 4, 435–462.

LeCun, Y., Cortes, C., and Burges, C. J. 1998. The MNIST database of handwritten digits.

Lepot, M., Stanić, N., and Clemens, F. H. 2017. A technology for sewer pipe inspection (part 2): Experimental assessment of a new laser profiler for sewer defect detection and quantification. *Automation in Construction 73*, 1–11.

Meijer, D. W., Kesteloo, M., and Knobbe, A. J. 2018. Unsupervised anomaly detection in sewer images with a PCA-based framework. In *International Conference on Pattern Recognition and Artificial Intelligence (ICPRAI)*. 354–359.

Meijer, D. W. and Knobbe, A. J. 2017. Unsupervised region of interest detection in sewer pipe images: Outlier detection and dimensionality reduction methods (extended abstract). In *Benelux Conference on Machine Learning (BeneLearn)*.

Meijer, D. W., Luimes, R. A., Knobbe, A. J., and Bäck, T. H. W. 2021. RADIUS: Robust anomaly detection in urban drainage with stereovision. *Automation in Construction 139*, 104285.

Meijer, D. W., Scholten, L., Clemens, F. H., and Knobbe, A. J. 2019. A defect classification methodology for sewer image sets with convolutional neural networks. *Automation in Construction 104*, 281–298.

Millard, L. A. C., Kull, M., and Flach, P. A. 2014. Rate-oriented point-wise confidence bounds for roc curves. In *Machine Learning and Knowledge Discovery in*

Databases, T. Calders, F. Esposito, E. Hüllermeier, and R. Meo, Eds. Springer Berlin Heidelberg, Berlin, Heidelberg, 404–421.

Moradi, S., Zayed, T., Nasiri, F., and Golkhoo, F. 2020. Automated anomaly detection and localization in sewer inspection videos using proportional data modeling and deep learning–based text recognition. *Journal of Infrastructure Systems 26,* 3, 04020018.

Myrans, J., Everson, R., and Kapelan, Z. 2018. Automated detection of faults in sewers using cctv image sequences. *Automation in Construction 95*, 64–71.

Myrans, J., Kapelan, Z., and Everson, R. 2018a. Combining classifiers to detect faults in wastewater networks. *Water Science and Technology 77,* 9, 2184–2189.

Myrans, J., Kapelan, Z., and Everson, R. 2018b. Using automatic anomaly detection to identify faults in sewers. In *WDSA/CCWI Joint Conference Proceedings*. Vol. 1.

Ojala, T., Pietikäinen, M., and Harwood, D. 1996. A comparative study of texture measures with classification based on featured distributions. *Pattern recognition 29,* 1, 51–59.

Oquab, M., Bottou, L., Laptev, I., and Sivic, J. 2014. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1717–1724.

Pearson, K. 1901. LIII. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2,* 11, 559–572.

Ro, Y. M., Kim, M., Kang, H. K., Manjunath, B., and Kim, J. 2001. MPEG-7 homogeneous texture descriptor. *ETRI journal 23,* 2, 41–51.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. S., Berg, A. C., and Li, F. 2014. Imagenet large scale visual recognition challenge. *CoRR abs/1409.0575*.

Russo, S., Disch, A., Blumensaat, F., and Villez, K. 2020. Anomaly detection using deep autoencoders for in-situ wastewater systems monitoring data. *arXiv preprint arXiv:2002.03843*.

Salvaggio, N. 2009. *Basic photographic materials and processes*. Taylor & Francis.

Shore, J. and Johnson, R. 1980. Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. *IEEE Transactions on information theory 26,* 1, 26–37.

Sitzenfrei, R., Mair, M., Möderl, M., and Rauch, W. 2011. Cascade vulnerability for risk analysis of water infrastructure. *Water Science and Technology 64,* 9, 1885–1891.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research 15,* 1, 1929–1958.

Stanić, N., Clemens, F. H., and Langeveld, J. G. 2017. Estimation of hydraulic roughness of concrete sewer pipes by laser scanning. *Journal of Hydraulic Engineering 143,* 2, 04016079.

Stanić, N., Langeveld, J. G., and Clemens, F. H. 2014. Hazard and operability (hazop) analysis for identification of information requirements for sewer asset management. *Structure and Infrastructure Engineering 10,* 11, 1345–1356.

Sun, Y., Xue, B., Zhang, M., and Yen, G. G. 2018. An experimental study on hyperparameter optimization for stacked auto-encoders. In *2018 IEEE Congress on Evolutionary Computation (CEC)*. 1–8.

Szeliski, R. 2010. *Computer Vision: Algorithms and Applications*, 1st ed. Springer-Verlag, Berlin, Heidelberg.

TISCA programme funded by NWO-TTW. 2016-2020. Sewersense – multi-sensor condition assessment for sewer asset management.

Tscheikner-Gratl, F., Caradot, N., Cherqui, F., Leitão, J. P., Ahmadi, M., Langeveld, J. G., Le Gat, Y., Scholten, L., Roghani, B., Rodríguez, J. P., et al. 2019. Sewer asset management–state of the art and research needs. *Urban Water Journal 16,* 9, 662–675.

Unser, M. 1995. Texture classification and segmentation using wavelet frames. *IEEE Transactions on image processing 4,* 11, 1549–1560.

van der Steen, A. J., Dirksen, J., and Clemens, F. H. 2014. Visual sewer inspection: detail of coding system versus data quality? *Structure and infrastructure engineering 10,* 11, 1385–1393.

Van Rossum, G. and Drake, F. L. 2009. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA.

Wirahadikusumah, R., Abraham, D., and Iseley, T. 2001. Challenging issues in modeling deterioration of combined sewers. *Journal of infrastructure systems 7,* 2, 77–84.

Wolpert, D. H. and Macready, W. G. 1997. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation 1,* 1, 67–82.

Wu, Y., Jiang, S., Xu, Z., Zhu, S., and Cao, D. 2015. Lens distortion correction based on one chessboard pattern image. *Frontiers of Optoelectronics 8,* 3, 319–328.

Yang, M.-D. and Su, T.-C. 2008. Automated diagnosis of sewer pipe defects based on machine learning approaches. *Expert Systems with Applications 35,* 3, 1327–1337.

Zimek, A. and Schubert, E. 2017. *Outlier Detection*. Springer New York, New York, NY, 1–5.

Curriculum Vitae

Dirk Meijer (he/him) was born in 1989 in The Hague, the Netherlands. He graduated from the Oranje Nassau College in Zoetermeer in 2007 and went on to study in Delft.

He obtained a BSc degree in Applied Physics in 2014 from the Delft University of Technology (thesis: *Evaluation of Fundus Photo Registration Quality for Progression Detection of Diabetic Retinopathy*; supervisor: prof.dr.ir. Lucas J. van Vliet) with a minor in Media & Knowledge Engineering, which provided access to the Computer Science master at the same university. In 2016 he obtained a MSc degree in Computer Science at the Delft University of Technology (thesis: *Regularizing AdaBoost to Prevent Overfitting on Label Noise*; supervisor: dr. David. M. J. Tax) and became an engineer.

Dirk set out to obtain a doctorate degree at Leiden University in 2016, joining the *SewerSense* project, a cooperation between the Delft University of Technology, Leiden University, and a consortium of NWO/TTW, RIONED, STOWA, and KPUD. This thesis is the culmination of Dirk's part in the SewerSense research project.

Since 2022, Dirk is employed as a post-doctoral researcher and scientific programmer at the Human-Centered Data Analytics (HCDA) group at the Centrum voor Wiskunde en Informatica (CWI) in Amsterdam.

List of Publications

In chronological order.

- Couvert, Rosalie; **Meijer, Dirk W. J.**; Ensing, Ronald M.; Martinez, José P.; Vermeer, Koenraad A.; and van Vliet, Lucas J.
  Normalization And Registration of Series of Fundus Photos in Longitudinal Screening for Diabetic Retinopathy
  In: Journal of Investigative Ophthalmology & Visual Science, Volume 53, Issue 14, 2012.
- Adal, Kedir M.; Couvert, Rosalie; **Meijer, Dirk W. J.**; Martinez, José P.; Vermeer, Koenraad A.; and van Vliet, Lucas J.
  A Quadrature Filter Approach for Registration Accuracy Assessment of Fundus Images
  In: Proceedings of the Ophthalmic Medical Image Analysis First International Workshop (OMIA), 2014.
- **Meijer, Dirk W. J.**; and Tax, David M. J.
  Regularizing AdaBoost with Validation Sets of Increasing Size
  In: Proceeding of the 23rd IEEE International Conference on Pattern Recognition (ICPR), 2016.

◇ **Meijer, Dirk W. J.**; and Knobbe, Arno J.
Unsupervised Region of Interest Detection in Sewer Pipe Images: Outlier Detection and Dimensionality Reduction Methods (Extended Abstract)
In: Proceedings of the 26th Benelux Conference on Machine Learning (BeneLearn), 2017.

◇ **Meijer, Dirk W. J.**; Kestloo, Mitchell; and Knobbe, Arno J.
Unsupervised Anomaly Detection in Sewer Images with a PCA-based Framework
In: Proceedings of the International Conference on Pattern Recognition and Artificial Intelligence (ICPRAI), 2018.

◇ **Meijer, Dirk W. J.**; Scholten, Lisa; Clemens, Francois H. L. R.; and Knobbe, Arno J.
A Defect Classification Methodology for Sewer Image Sets with Convolutional Neural Networks
In: Automation in Construction, Volume 104, 2019.

◇ **Meijer, Dirk W. J.**; Toussaint, Guus; and Knobbe, Arno J.
Deep Subgroups: A Neural Network Architecture for Subgroup Discovery
Web published, 2020.

◇ **Meijer, Dirk W. J.**; Luimes, Rianne A.; Knobbe, Arno J.; and Bäck, Thomas H. W.
Anomaly Detection in Urban Drainage with Stereovision
In: Automation in Construction, Volume 139, 2022.

English Summary

Sewer pipes are an essential infrastructure in modern society and their proper operation is important for public health. To keep sewer pipes operational as much as possible, periodical inspections for defects are performed. Instead of repairing sewer pipes when a problem becomes critical, such inspections allow municipalities to plan maintenance. This means the disruptions of the service can be planned for by users of the pipe in question, and there is less chance that a problem slips by unnoticed.

Sewer pipe inspections are generally performed visually with the aid of a *pipe inspection gadget*, or PIG. The PIG is a remote-controlled vehicle equipped with cameras and possibly other sensors. The PIG is lowered into a manhole to inspect a stretch of pipe, after which it is returned to the surface. A trained human operator inspects the footage recorded by the cameras, often while controlling the PIG from a vehicle at ground level.

Inspection reports are made according to a European classification norm. This norm groups defects of a similar nature together and has guidelines for what constitutes ratings from 1 ("no intervention necessary") to 5 ("immediate intervention necessary"). Problematically, these guidelines consider defects in a vacuum. Take a fissure in the wall of a pipe for example, the guideline assigns a rating 1 to 5 to different ranges of fissure sizes. The actual consequences depend on many more factors, such as whether the pipe is above or below the groundwater level, the zoning district the pipe is located in, etc. As a result, operators have learned to assign ratings not according to the guidelines, but according to an intuitive assessment of severity. This, in turn, means that severity ratings can vary wildly between operators, and even between inspections by the same operator.

This makes sewer pipe inspections an attractive target for automation. While a potential improvement in terms of assessment quality and processing efficiency is generally promised by automation, in this case we would also decrease the variability which is a current problem. Besides the reasons for automating, the methods for automating are also attractive: a lot of (visual) data has been gathered over the past decades which may be used to train algorithms.

This thesis compiles the results of five years of research into the possible automation of sewer pipe inspections with the tools of machine learning and computer vision. In this thesis, three distinct, yet complementary approaches to automating sewer pipe inspections are described.

Chapter 3, *Image-Based Unsupervised Anomaly Detection*, describes an approach based on anomaly detection of the contents of the images. At this stage, the data that was available to us consisted of images from inspections performed in two Dutch municipalities. The inspection reports themselves were not available at that time, meaning it was unclear which images were showing defects and which were not. While more complete data became

available at a later date, at this stage we decided to leverage the image data that we did have.

The structure of the different images is very similar: the pipes were photographed with the same equipment, and the pipes from the same municipality were often installed in the same year, from the same manufacturer, and had seen similar use. This resulted in a delineation of two image sets, one of images of pipes made of smooth concrete, one of images of pipes made of granulate. Within either of the image sets, the images look mostly uniform, meaning that anomalies—both expected (such as pipe joints) and unexpected (such as defects)—stand out.

We applied principal component analysis to the images and extracted features from the images, to detect the most common elements in an image set. Then, when we express an image in those most common elements, we obtain a faithful reconstruction for the images that do not contain any anomalies, and a less perfect reconstruction for the images that do contain anomalies. Leveraging this reconstruction error, we compare the reconstruction to the original image to estimate how likely it is to contain an anomaly.

In addition, we trained a convolutional autoencoder, a type of artificial neural network, to perform a function similar to the principal component analysis, without enforcing a linear relation of the common elements.

The results of these experiments were promising for the images of pipes made from smooth concrete, but less so for the images of pipes made from the rougher granulate.

Chapter 4, *Convolutional Neural Network Classification*, describes an approach based on supervised classification with a convolutional neural network. Convolutional neural networks are artificial neural networks that are particularly suited to handle images, audio and video. We were provided with sewer pipe images like the ones used in chapter 3, but a much larger volume and including machine-readable classifications as assigned by human operators. A total of 2.2 million images were available and the classification data allowed us to estimate what defects should be visible in any given image. A single neural network was trained to detect the twelve most common defect types in the dataset.

The problem of sewer pipe defect detection a strongly unbalanced one: only approximately 1 % of images actually contain defects. Most of the existing literature at the time was assessing the performance of their models in terms of accuracy, the fraction of correctly classified images, both as having, or not having, a defect. On a realistic dataset, an accuracy of 99 % is then to be expected if we classify every image as not having any defects, which is clearly not the intention of defect detection. To counter this, many works rebalanced the dataset to contain about 50 % images with defects. While this is not per se a bad idea, nearly every one of them also rebalanced the test set that was used to assess the performance

of the model, making the assessed performance not at all indicative of actual, real-world performance. Many also treated false positive and false negative detections identically, while these have very different results in a realistic scenario: the former costs time, the latter might pose a public health hazard.

Many earlier works randomly divide images of pipes into training and test set, meaning that images of the same pipe at locations close to one another might end up in both training and test set. This introduces a danger of data leakage: a high performance on the test set might not necessarily mean that the defects themselves are being detected, but could rather mean that the pipe is being detected.

To have any real-world meaning, we asserted that the test set used to assess the model must be as realistic as possible, including having a realistic ratio of images with and without defects, and only containing pipes that were not also used to train the model. We also approach the problem from a more context-sensitive perspective, noting that accuracy is not a useful metric in realistic situations, and introducing metrics that can be more meaningfully interpreted by a human, as well as translate more directly to operational impact.

Chapter 5, *Stereovision and Geometry Reconstruction*, extends beyond the current sewer pipe inspection process and investigates the added value of a second camera, allowing us to reconstruct the three-dimensional geometry of the sewer pipe. Much like how human beings perceive depth only with both eyes open, a second camera allows us to estimate the positions of objects in relation to the viewpoint.

In collaboration with Eindhoven University of Technology, we photographed 26 sewer pipes in various conditions with a set of two side-by-side cameras. We built upon existing stereovision techniques and adapted them for this unique use case to reconstruct a three-dimensional point cloud of the pipe's inner surface.

A pipe surface model is constructed under the assumption that the cameras are aligned approximately along the pipe center. The model is powerful enough to capture the geometry of any of the pipes we have used, but also based on human understanding of the shape of a pipe, making it very interpretable.

The model is fit to the point cloud to estimate the original pipe geometry, without taking into account minor deviations that are visible in the pipes after years of use. This allows us to easily detect the portions of the pipe where the surface wall is deviating from the expected shape. The detected deviating portions of the surface correlate with the presence of actual defects in this small-scale experiment. The end result is an interpretable computer vision technique that can be used to assist human-guided inspections.

Nederlandse Samenvatting

Rioolbuizen vormen een essentiële infrastructuur in de moderne samenleving, het goed functioneren ervan is van belang voor de volksgezondheid. Rioolbuizen worden met regelmaat geïnspecteerd op defecten om ze zoveel mogelijk operationeel te houden. Zulke inspecties maken het mogelijk onderhoud in te plannen in plaats van reparaties uit te voeren als het probleem kritiek geworden is. Op deze manier kunnen de gebruikers van een riool rekening houden met de onderbreking; daarnaast is het hierdoor minder waarschijnlijk dat problemen langdurig onopgemerkt blijven.

Rioolinspecties worden over het algemeen visueel uitgevoerd met behulp van een "pipe inspection gadget", afgekort PIG. De PIG is een op afstand bestuurd voertuig met camera's en mogelijk andere sensoren. De PIG wordt door een rioolput naar beneden gebracht om een deel van de riool te inspecteren, waarna het weer omhoog gebracht wordt. Een speciaal getrainde inspecteur beoordeelt de camerabeelden, vaak tegelijkertijd met het besturen van de PIG vanuit een voertuig op straatniveau.

Inspectierapportages worden gedaan volgens een Europese classificatienorm. Deze norm groepeert defecten in typen en heeft richtlijnen voor wanneer gradaties van 1 ("geen interventie nodig") tot 5 ("onmiddellijke interventie noodzakelijk") aan de orde zijn. Een probleem is echter dat deze richtlijnen geen rekening houden met externe factoren. Een scheur in de rioolwand bijvoorbeeld, krijgt volgens de richtlijnen een gradatie van 1 tot 5 afhankelijk van de afmetingen van de scheur. De werkelijke gevolgen van een scheur hangen af van veel andere factoren, zoals of de riool zich boven of onder het grondwaterpeil bevindt, of de omgeving een woonwijk of industriewijk is, etc. Inspecteurs waarderen de gradatie daarom vaak niet volgens de richtlijnen, maar naar hun intuïtieve inschatting van de ernst. Dit heeft als gevolg dat de gradatie die een defect in een rapportage ontvangt veel kan verschillen tussen verschillende inspecteurs, en zelfs tussen verschillende rapportages van dezelfde inspecteur

Dit maakt rioolinspecties aantrekkelijk om te automatiseren. Automatisering belooft in het algemeen een potentiële verbetering in kwaliteit en efficiëntie; in dit geval zou het ook de problematische variabiliteit van de rapportages verminderen. Naast de redenen voor automatisering zijn de mogelijkheden voor automatisering ook aantrekkelijk: er is veel visuele data verzameld in de loop van decennia die gebruikt kan worden om algoritmen te trainen.

Dit proefschrift beschrijft het resultaat van vijf jaar onderzoek naar mogelijke automatisering van rioolinspecties met behulp van *machine learning* en *computer vision* technieken. Drie verschillende maar complementaire aanpakken van automatisering van rioolinspecties worden behandeld.

Hoofdstuk 3, *Image-Based Unsupervised Anomaly Detection*, beschrijft een aanpak met als kern het detecteren van afwijkingen in afbeeldingen. In deze fase van het onderzoek bestond de voor ons beschikbare data uit afbeeldingen van inspecties uit twee Nederlandse gemeenten. De rapportages over de inspecties waren niet beschikbaar, wat betekende dat het onduidelijk was welke afbeeldingen wel en geen defecten lieten zien. Hoewel meer data in een later stadium beschikbaar zou worden, hebben we besloten de afbeeldingen die we hadden toch te gebruiken.

De structuur van de verschillende afbeeldingen is soortgelijk: de buizen waren gefotografeerd met dezelfde apparatuur, en buizen uit een gemeente worden veelal geïnstalleerd in hetzelfde jaar, zijn afkomstig van dezelfde fabrikant, en worden gebruik onder grotendeels dezelfde omstandigheden. Dit leidde tot een tweedeling van de afbeeldingen, één set van afbeelding met buizen van glad beton, één set van afbeeldingen van buizen van ruwer granulaat. Binnen een van deze twee sets zien de afbeeldingen er grotendeels hetzelfde uit, waardoor afwijkingen–zowel verwachte (zoals aansluitingen) als onverwachte (zoals defecten)–opvallen.

We gebruiken *principal component analysis* (factoranalyse) op zowel de afbeeldingen zelf als op geëxtraheerde kenmerken uit de afbeeldingen, om gemeenschappelijke factoren te herkennen. Wanneer we een afbeeldingen uitdrukken in de meest voorkomende van deze gemeenschappelijke factoren, blijft een afbeelding zonder afwijking getrouw aan het origineel, terwijl een afbeelding met afwijkingen minder getrouw aan het origineel zal zijn. Gebruikmakend van dit feit vergelijken we een gereconstrueerde afbeelding met het origineel om de waarschijnlijkheid dat het origineel een afwijking bevat in te schatten.

Daarnaast hebben we ook een convolutionele autoencoder, een type neuraal network, getraind om een soortgelijke functie als de factoranalyse uit te voeren, zonder de beperking van factoranalyse dat de factoren een lineaire relatie beschrijven.

De resultaten van deze experimenten waren veelbelovend voor de set met afbeeldingen van glad beton, maar minder succesvol voor de set met afbeeldingen van ruwer granulaat.

Hoofdstuk 4, *Convolutional Neural Network Classification*, beschrijft een aanpak gebaseerd op *supervised learning* ("leren onder toezicht") met een convolutioneel neuraal netwerk.

Convolutionele neurale netwerken zijn neurale netwerken die bijzonder geschikt zijn om afbeeldingen, geluid, en video te classificeren. We waren voorzien van afbeeldingen van rioolbuizen zoals die in hoofdstuk 3, maar in groter volume, en ditmaal inclusief de rapportages zoals toegekend door de menselijke inspecteurs, in een formaat geschikt voor digitale verwerking. In totaal waren 2,2 miljoen afbeeldingen beschikbaar en de bijbehorende classificaties lieten ons inschatten welk type defecten zichtbaar zouden moeten

zijn in elk van deze afbeeldingen. Een enkel neuraal netwerk werd getraind om de twaalf meest voorkomende defecten te detecteren.

Het detecteren van defecten in riolen is een 'ongebalanceerd' probleem: slechts ongeveer 1 % van de afbeeldingen bevat werkelijk een defect. Het merendeel van de bestaande wetenschappelijke literatuur maakte een inschatting van de prestaties van een model door te kijken naar de *accuracy* (nauwkeurigheid), het deel van de afbeeldingen dat correct geclassificeerd is op de aanwezigheid of afwezigheid van een defect. Met een realistische dataset is het behalen van een accuracy van 99 % mogelijk door elke afbeelding te classificeren als geen defect bevattend, wat duidelijk niet de intentie van defectdetectie is. Om dit probleem te corrigeren, worden datasets veelal 'hergebalanceerd', zodat deze ongeveer 50 % afbeeldingen met defecten bevat. Hoewel dit niet per se een probleem oplevert, is het belangrijk dat deze herbalancering alleen op de trainingsset plaatsvindt zodat de inschatting van de prestaties niet beïnvloedt wordt, wat vaak niet het geval was in de literatuur. Veelal werden foutpositieve resultaten (onterechte detectie van een defect) en foutnegatieve resultaten (onterecht gebrek aan detectie van een defect) gelijk behandeld, terwijl deze zeer verschillende resultaten kunnen opleveren: een foutpositief resultaat kost extra tijd, maar een foutnegatief resultaat kan een gevaar voor de volksgezondheid opleveren.

Een groot deel van de bestaande literatuur verdeelde de afbeeldingen van riolen willekeurig in training en test set, wat kan betekenen dat afbeeldingen van dezelfde riool op nabije locaties zich zowel in de training als test set kunnen bevinden. Dit introduceert een gevaar voor *overfitting*: een goede prestatie op de test set betekent in dit geval mogelijk niet dat de defecten worden herkend, maar dat de riolen zelf worden herkend.

Wij opperden dat om enige betekenis te hebben in de echte wereld, de test set die gebruikt wordt bij het inschatting van de prestaties van het model zo realistisch mogelijk moet zijn, betekenend onder meer dat deze een realistische verhouding van afbeeldingen met en zonder defecten heeft, en dat deze geen afbeeldingen bevat van riolen die ook in de trainingsset aanwezig waren. We hebben het probleem ook meer beschouwd vanuit een context-gevoelig perspectief dan eerder werk, en geopperd dat accuracy geen nuttige maatstaf is in realistische situaties. In plaats daarvan hebben we meer betekenisvolle maten geïntroduceerd die gemakkelijker door mensen geïnterpreteerd kunnen worden, en direct vertaald kunnen worden naar operationele impact.

Hoofdstuk 5, *Stereovision and Geometry Reconstruction*, reikt voorbij het huidige inspectieproces en onderzoekt de toegevoegde waarde van een tweede camera die ons een driedimensioneel beeld van de riool laat reconstrueren. Zoals mensen diepte kunnen zien met beide ogen open laat een tweede camera ons de posities van objecten in verhouding tot het

kijkpunt inschatten.

In samenwerking met de Technische Universiteit Eindhoven hebben wij 26 riolen in verschillende staten van gebruik gefotografeerd met twee zij-aan-zij camera's. Door te bouwen op bestaande stereovisie technieken en deze uit te breiden en aan te passen voor ons unieke doeleinde, kunnen we een drie-dimensionale *point-cloud* reconstrueren van de binnenwand van een riool.

De rioolwand kan nu worden gemodeleerd met een model, onder enkel de aanname dat de camera's correct uitgelijnd zijn en ongeveer middenin de riooldoorsnede gericht staan. Het model is complex genoeg om de afmeting van alle riolen die gefotografeerd zijn te vatten, maar gebaseerd op menselijk begrip van de vorm van een riool, waardoor het resultaat goed te interpreteren is.

Het model wordt toegepast op de point cloud om de oorspronkelijk vorm van de riool in te schatten, zonder de mogelijke gebruikssporen hierin mee te nemen. Hierdoor kunnen de gebieden waar het oppervlak van de rioolwand afwijkt van de te verwachten vorm herkend worden. De gedetecteerde afwijkingen op het oppervlak correleren met de aanwezigheid van defecten in dit kleinschalige experiment. Het eindresultaat is een interpreteerbare beeldverwerkingstechniek die gebruikt kan worden om inspecteurs te assisteren.

Acknowledgements

A friend once told me that they loved the first 80% of writing their thesis but absolutely hated the last 80% and this experience seems to be universal. While the proverbial blood, sweat, and tears may not show through these pages, I assure you they have been spilled. I am proud of the work that lies before you and the works it is based on, but like all scientific works it has been a collaborative effort as well.

I owe much to Arno, a better supervisor than I could have wished for. I lucked into being his promotee by accident, which was in hindsight the best thing that could have happened. His advice for the road ahead of me was practical and insightful, and his openness to learn from me as well was a refreshing change from many hierarchies that can be observed in academia. I count him not only as a valued mentor, but as a friend as well.

I wish to thank the other co-authors of articles written during my PhD: Lisa, Francois, Rianne, Thomas, Guus, and Mitchell, for their valuable insights and for performing experiments and writing parts of the articles when I had too much on my plate.

When I think back on this time, I am extremely grateful for my closest colleagues, and their contributions to the most fun working environment I have had the pleasure to be part of. Without Lise, Rens, Marvin, Arie-Willem, Irene, Leon, Ricardo, José, Zé, Jeroen, Jan, Benjamin, Rob, Auke, Stephan, Hugo, and undoubtedly others I am forgetting (sorry!), this time would have been a lot less enjoyable.

The same goes for my colleagues in Delft, although we saw less of each other than originally planned, you have made me feel very welcome on the days I have shared your office. Thank you Danai, Juan C., Juan A., Matthijs, Adithya, Eva, Job, Elena, Bram, Franz, and again anyone I may be forgetting.

I extend gratitude to Laura and Davide at CWI, for their seemingly limitless patience at the end of this trajectory and the start of a new one.

Many thanks to my paranymphs, Lise and Frank, you are among my closest friends. Who I am today I am in part thanks to your friendship. I'm glad to have you both at my side during the defense.

Lastly and most importantly, I thank Elja, for being the co-author of my life, my inspiration, and my best friend.

Thank you all,
Dirk.