

UNIVERSITY OF MALTA
FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
DEPARTMENT OF COMMUNICATIONS AND COMPUTER ENGINEERING

A Tunnel Structural Health Monitoring Solution using Computer Vision and Data Fusion

Leanne Attard

supervised by
Prof. Carl James Debono

co-supervised by
Dr. Gianluca Valentino

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Communications and Computer Engineering

May, 2020

L-Università ta' Malta

University of Malta Library – Electronic Thesis & Dissertations (ETD) Repository

The copyright of this thesis/dissertation belongs to the author. The author's rights in respect of this work are as defined by the Copyright Act (Chapter 415) of the Laws of Malta or as modified by any successive legislation.

Users may access this full-text thesis/dissertation and can make use of the information contained in accordance with the Copyright Act provided that the author must be properly acknowledged. Further distribution or reproduction in any format is prohibited without the prior permission of the copyright holder.

I dedicate this thesis to those who helped me achieve;
my parents, my brother and Aeden.

Abstract

Tunnel structural health monitoring is predominantly done through periodic visual inspections, requiring humans to be physically present on-site, possibly exposing them to hazardous environments. Drawbacks associated with this include the subjectivity of the surveys and, most of the time, the shutting down of operations during the inspection. To mitigate these, an increasing effort was made to automate inspections using robotics to reduce human presence and computer vision techniques to detect defects along tunnel linings. While defect identification is beneficial, comprehensive monitoring to identify changes on tunnel linings can provide a more informative survey to further automate inspection and analysis.

CERN, the European Organisation for Nuclear Research has more than 50 km of tunnels which need monitoring. This raised the need for a remotely operated surveying system to monitor the structural health of the tunnels. Hence, a tunnel inspection solution to monitor for changes on tunnel linings is proposed here.

Using a robotic platform hosting a set of cameras, tunnel wall images are automatically and remotely captured. The tunnel environment poses a number of challenges, with two of these being different light conditions and reflections on metallic objects. To alleviate this, pre-processing stages were developed to correct for the uneven illumination and to localise highlights. Crack detection using deep learning techniques is employed following the pre-processing stages to identify cracks on concrete walls. A change detection process is implemented through a combination of different bi-temporal pixel-based fusion methods and decision-level fusion of change maps. The evaluation of the proposed solution is made through qualitative analysis of the resulting change maps followed by a quantitative comparison with ground-truth changes. High recall and precision values of 81% and 93% were respectively achieved. The proposed solution provides a better means of structural health monitoring where data acquisition is carried out on-site during shutdowns or short, infrequent maintenance periods and post-processed off-site.

Acknowledgements

"And, when you want something,
all the universe conspires in helping you to achieve it."
- The Alchemist by Paulo Coelho

Every challenging work needs self-efforts as well as support from those around. I would like to express my gratitude towards all those who sustained me during the completion of this thesis. I am extremely grateful to my parents for their unconditional love and support through the years. I am also indebted to my brother for his constant encouragement, his interest through the course of this research and most of all for keeping up with my anxiety throughout. I am truly grateful to Aeden, for his love and understanding, for making me feel whole again when I was in pieces, for supporting me towards completing my studies, for being there. I also owe thanks to other relatives who always cared for me even from miles away.

Earnest thanks go to my supervisors, Prof. Dr. Ing. Carl James Debono and Dr. Ing. Gianluca Valentino, from the University of Malta, who have provided me with utmost guidance and assistance throughout the course of my studies and for being generous with their time. Equal thanks go to my supervisor Dr. Mario Di Castro and my colleague Giacomo Lunghi, for their recurring support and vital guidance throughout my placement at CERN. Futhermore, I would like to thank the SMB department at CERN for the opportunity they gave me to work on such a project.

Sincere thanks go to you my neighbours Esmeralda and Joost. In you, I felt like having second parents while living in France. Gratitude also to those I got to know during these 3 years. Thank you Meng for the long chats, shopping and great meals we shared, especially the Chinese hot-pot. Thank you Franci for your care, laughs, good food and Italian conversations. Thank you Laura, for the nice chats and your help when at CERN. Thank you Luca for your support whenever I asked your help at work. Thanks to those I shared R1-009 with, over the years; Clare, Antonio, Julia. Thanks to my 'brothers' Simone and Daniele for the good moments shared with the 'family'. Thank you Francesco, Pawel, Jorge, Andrzej and the rest of my colleagues residing at one time or another in Bld. 628. You all made me feel at home while being an expat.

A heartfelt thanks to my Maltese friends who cheered me up during our travels, through their messages and our meet-ups when visiting the rock.

Last but not least, I would like to thank SITES for providing ScanTubes® camera system for a demo test in the LHC tunnel. The data collected during this test allowed me to appropriately develop and test the proposed solution.

List of Related Contributions

1. Leanne Attard, Carl James Debono, Gianluca Valentino, Mario Di Castro, Tunnel inspection using photogrammetric techniques and image processing: A review, ISPRS Journal of Photogrammetry and Remote Sensing, Volume 144, 2018, pp. 180-188
   doi: https://doi.org/10.1016/j.isprsjprs.2018.07.010.

2. L. Attard, C. J. Debono, G. Valentino and M. Di Castro, J. A. Osborne, L. Scibile, A comprehensive virtual reality system for tunnel surface documentation and structural health monitoring, 2018 IEEE International Conference on Imaging Systems and Techniques (IST), Krakow, 2018, pp. 1-6.
   doi: 10.1109/IST.2018.8577139

3. L. Attard, C. J. Debono, G. Valentino, M. Di Castro, A. Masi and L. Scibile, Automatic Crack Detection using Mask R-CNN, 2019 11th International Symposium on Image and Signal Processing and Analysis (ISPA), Dubrovnik, Croatia, 2019, pp. 152-157.
   doi: 10.1109/ISPA.2019.8868619

4. L. Attard, C. J. Debono, G. Valentino, M. Di Castro and A. Masi, VR-SHM - A structural health monitoring tool to assist crack detection using deep learning and virtual reality, Sustainable Built Environment conference (SBE), Malta, 2019

5. L. Attard, C. J. Debono, G. Valentino and M. Di Castro, Specular highlights detection using a U-Net based deep learning architecture, International Conference on Multimedia Computing, Networking and Applications (MCNA2020) [under review]

6. L. Attard, C. J. Debono, G. Valentino and M. Di Castro, Automatic crack detection in concrete infrastructure using deep learning models - a comparative analysis, Automation in Construction [under review]

7. L. Attard, C. J. Debono, G. Valentino and M. Di Castro, A machine vision solution for change detection on tunnel linings using fusion, Journal of Machine Vision and Applications [under review]

The reader is referred to Appendix E for copies of the above publications.

Contents

Abstract . . . ii
Acknowledgements . . . iii
List of Related Contributions . . . v

1 Introduction **1**
1.1 Aims and objectives . . . 2
1.2 CERN . . . 3
1.2.1 LHC tunnel scenario . . . 4
1.3 Thesis structure . . . 5

2 Literature review **6**
2.1 General tunnel inspection . . . 6
2.1.1 Tunnel wall deformation . . . 7
2.1.2 Crack and defect detection . . . 8
2.1.2.1 Crack detection using deep learning . . . 11
2.2 Data fusion . . . 13
2.3 Image fusion . . . 14
2.3.1 Levels of image fusion . . . 15
2.3.2 Image fusion categories . . . 15
2.3.2.1 Multi-view image fusion . . . 16
2.3.2.2 Multi-modal image fusion . . . 17
2.3.2.3 Multi-temporal image fusion . . . 18
2.3.3 Image fusion domains and techniques . . . 19
2.3.3.1 Spatial domain . . . 19
2.3.3.2 Pyramid-based . . . 21
2.3.3.3 Transform and wavelet-based . . . 21
2.4 Change detection . . . 21
2.4.1 Pre-processing for change detection . . . 22
2.4.1.1 Alignment corrections . . . 22
2.4.1.2 Radiometric adjustments . . . 23
2.4.1.3 Semantic segmentation . . . 23
2.4.2 Change detection techniques . . . 24
2.4.2.1 Pixel-based methods . . . 25
2.4.2.2 Object-based methods . . . 27
2.4.2.3 Anomaly detection methods for change detection . . . 28

2.4.3 Change detection in tunnel environments . . . 28
2.4.4 Common metrics for performance evaluation . . . 30
2.5 Tunnel surface visualisation . . . 31
2.5.1 3D reconstruction . . . 32
2.5.2 Virtual reality . . . 33

3 Solution overview **34**
3.1 Pipeline . . . 34
3.2 Data acquisition . . . 35
3.3 Crack detection . . . 36
3.4 Specular highlight localisation . . . 36
3.5 Change detection . . . 37
3.6 Visualisation . . . 38

4 Data acquisition **40**
4.1 Environment constraints . . . 40
4.2 Image sensors investigation . . . 42
4.3 Mobile platforms . . . 43
4.4 Preliminary camera system . . . 44
4.4.1 Camera setup . . . 44
4.4.2 Automatic image capturing . . . 45
4.4.3 Dataset . . . 46
4.5 Operational camera system . . . 47
4.5.1 Setup of the system . . . 47
4.5.2 Demo test dataset . . . 48

5 Crack detection and monitoring **51**
5.1 Semantic segmentation method . . . 52
5.1.1 U-Net model . . . 52
5.1.2 SegNet model . . . 53
5.1.3 Methodology . . . 55
5.1.3.1 Pre-processing . . . 55
5.1.3.2 Encoder architectures . . . 56
5.1.3.3 Data augmentation . . . 56
5.2 Instance segmentation method . . . 56
5.2.1 Mask R-CNN model . . . 56
5.2.2 Methodology . . . 59
5.2.2.1 Transfer learning . . . 59
5.2.2.2 Data augmentation . . . 60
5.3 Crack datasets . . . 60
5.3.1 SDNET subset . . . 60
5.3.2 LHC dataset . . . 61
5.4 Comparative analysis . . . 63
5.4.1 Quantitative results . . . 63

5.4.1.1 SDNET . . . 63
5.4.1.2 LHC . . . 70
5.4.2 Qualitative results . . . 75
5.4.2.1 SDNET . . . 75
5.4.2.2 LHC . . . 76
5.5 Class-specific object-based change detection . . . 78
5.5.1 Temporal comparison of cracks . . . 78
5.6 Contributions summary . . . 82

6 Specular Highlight Localisation 85
6.1 Specular highlight detection . . . 86
6.2 U-Net semantic segmentation . . . 87
6.3 Methodology . . . 87
6.3.1 Batch normalisation . . . 88
6.3.2 Dropout . . . 88
6.3.3 Training and optimisation . . . 89
6.3.4 Data augmentation . . . 89
6.4 Highlights datasets . . . 90
6.4.1 PURDUE set . . . 90
6.4.2 LHC set . . . 90
6.5 Experiments and results . . . 92
6.5.1 Quantitative results . . . 92
6.5.1.1 PURDUE set . . . 93
6.5.1.2 LHC set . . . 96
6.5.2 Qualitative results . . . 103
6.5.2.1 PURDUE set . . . 103
6.5.2.2 LHC set . . . 103
6.6 Contributions summary . . . 105

7 Change detection 106
7.1 Pre-processing . . . 107
7.1.1 Uneven illumination correction . . . 107
7.1.2 Specular highlights localisation . . . 109
7.2 Ideal change detection . . . 110
7.3 PBCD using bi-temporal image fusion . . . 111
7.3.1 Image difference . . . 112
7.3.2 Principal Component Analsyis (PCA) . . . 117
7.3.3 Structural Similarity Index (SSIM) . . . 121
7.4 Decision-level fusion . . . 124
7.4.1 Fusion using logical operations . . . 124
7.4.2 Fusion using PCA-weighted summation . . . 125
7.4.3 Fusion using majority voting . . . 127
7.5 Change map analysis . . . 127
7.5.1 Specular highlight filtering . . . 128

7.5.2 Morphological operations . . . 128
7.5.3 Connected components labelling . . . 129
7.5.4 Dimension filtering . . . 129
7.5.5 Binary comparison . . . 130
7.6 Performance evaluation . . . 132
7.6.1 Evaluation metrics . . . 133
7.6.2 Quantitative results . . . 134
7.6.3 Qualitative results . . . 135
7.7 Contributions summary . . . 136

8 Conclusion and Future work **141**
8.1 Conclusion . . . 141
8.2 Summary of contributions . . . 142
8.3 Recommendations for future work . . . 142

Bibliography **146**

Appendices **179**

Appendix A Examples of acquired LHC tunnel images **180**

Appendix B Crack detection **184**

Appendix C Highlight detection **188**

Appendix D Change detection **190**

Appendix E Publications **194**

List of Figures

1.1 Some of CERN's tunnels 1
1.2 CERN accelerator complex 3
1.3 Low lighting conditions which vary from one area to another 4

2.1 Invar wire and gauge used during manual surveys 7
2.2 An example of 'photogrammetric levelling' 7
2.3 Original image and crack segmentation result 9
2.4 An example of an image acquisition system 10
2.5 A sample of crack detection results from DeepCrack 13
2.6 Several applications of image fusion applications 14
2.7 On-site structural inspections in the LHC tunnel 29

3.1 Block diagram of the proposed inspection solution 35
3.2 Block diagram of the change detection module within the proposed inspection solution 38

4.1 Cross-section of the LHC tunnel 41
4.2 Camera on the arm extending from one of the wagons of the TIM . 43
4.3 Multiple cameras on the CERNbot robotic platform 43
4.4 Robust metal rig with multiple cameras on horizontal metal blocks fixed to a vertical structure on a robust base fixed on the CERNBot 45
4.5 Camera attached to a quick release plate 45
4.6 Samples of images captured by the three cameras on the vertical structure placed on the CERNBot 47
4.7 Camera rig in the provisional commercial camera system 48
4.8 Provisional commercial camera system integrated on the CERNBot 48
4.9 A sample set of images captured using the provisional commercial camera system during the demo test in the LHC tunnel 49
4.10 Orthophotos generated from $DataT_1$ and $DataT_2$ captured during the demo test 50

5.1 U-Net architecture pipeline 53
5.2 SegNet architecture pipeline 55
5.3 Mask R-CNN pipeline 59
5.4 A sample of crack markings from the SDNET subset 61
5.5 A sample of crack markings from the LHC dataset 62

5.6 A plot of the cross entropy loss during training of different models on the SDNET subset 66
5.7 A plot of the cross entropy loss during validation of different models on the SDNET subset 67
5.8 The plots of the class loss and mask loss while training Mask R-CNN on the SDNET subset 69
5.9 A plot of the cross entropy loss during training of U-Net and SegNet models with different encoder architectures, on the LHC dataset . . 72
5.10 A plot of the cross entropy loss during validation of U-Net and SegNet models with different encoder architectures, on the LHC dataset 73
5.11 The plots of the class loss and mask loss while training Mask R-CNN on the LHC dataset 74
5.12 Crack detection results using the U-Net model on the SDNET subset 75
5.13 Crack detection results using the SegNet model on the SDNET subset 76
5.14 Crack detection results using the Mask R-CNN model on the SDNET subset 77
5.15 A crack detection example from the LHC dataset 77
5.16 Crack masks 79
5.17 Crack Bounding boxes 79
5.18 Flow diagram for the crack comparison procedure 80
5.19 Crack detection example 1 83
5.20 Crack comparison result listing the status of each identified crack in example 1 83
5.21 Crack detection example 2 84
5.22 Crack comparison result listing the status of each identified crack in example 2 84

6.1 U-Net model with the proposed modifications 88
6.2 A sample of images and their corresponding original GT mask from the PURDUE dataset and the inverted mask. 91
6.3 A sample of images and their corresponding GT markings from the annotated specular highlights dataset built using the LHC dataset . 91
6.4 F-score and loss from the U-Net model with the proposed encoder architecture during training and validation on the PURDUE set . . 95
6.5 Plots of training and validation F-score for the LHC set using the U-Net model with different batch sizes 97
6.6 Plots of training and validation IoU for the LHC set using the U-Net model with different batch sizes 98
6.7 Plots of training and validation loss for the LHC set using the U-Net model with different batch sizes 99
6.8 Plots of training and validation F-score for the LHC set using the U-Net model with different encoder architectures 100
6.9 Plots of training and validation IoU for the LHC set using the U-Net model with different encoder architectures 101

6.10 Plots of training and validation loss for the LHC set using the U-Net model with different encoder architectures . . . 102
6.11 Comparison of the GT and segmentation masks on the PURDUE set 103
6.12 Specular highlights example 1 . . . 104
6.13 Specular highlights example 2 . . . 104

7.1 Light variation in the LHC tunnel . . . 107
7.2 The original reference and survey images at a particular position and the corresponding pre-processed images . . . 109
7.3 Difference images of the greyscale and pre-processed images . . . 110
7.4 Specular highlight localisation on the reference and survey images and the corresponding highlight mask . . . 111
7.5 Change Detection in an ideal-world scenario . . . 112
7.6 Difference images of the greyscale and pre-processed images with illumination changes . . . 113
7.7 Image difference using different values for the fixed threshold . . . 114
7.8 Histogram of the pixel absolute difference values . . . 114
7.9 Image difference using different automatic thresholding techniques . 117
7.10 The first 4 principal components of the stacked original images . . . 118
7.11 Histogram of normalised $PC1$ from PCA on original RGB images . 119
7.12 The principal components of the stacked pre-processed images . . . 119
7.13 Histogram of normalised PC_0 from PCA on pre-processed images . 120
7.14 Resulting CMs from PCA applied to different images . . . 121
7.15 Diagram of the SSIM measurement system . . . 121
7.16 CMs from SSIM . . . 124
7.17 Fusion of CMs using logical operators . . . 125
7.18 Diagram of CM fusion by PCA-weighted summation . . . 126
7.19 CM decision-level fusion by PCA-weighted summation . . . 126
7.20 CM decision-level fusion by majority voting . . . 127
7.21 CM analysis process . . . 127
7.22 Concept of morphological operations . . . 128
7.23 Concept of connected components labelling . . . 130
7.24 Change candidates and their difference ratios . . . 131
7.25 Change candidate patches . . . 132
7.26 Confusion matrix . . . 133
7.27 An example showing similar results for both majority voting and PCA 137
7.28 An example showing different detection results from majority voting and PCA-weighted summation . . . 138
7.29 An example showing a different simulated defect on the wall . . . 139
7.30 An example exhibiting lighting changes . . . 140

8.1 Augmentation of inspection findings on a VR model . . . 143
8.2 A screenshot of ThermoVis . . . 144
8.3 Thermal and RGB camera placed on a tripod . . . 144

8.4 A sample of the captured RGB and TIR images 145

A.1 Images captured by the three cameras on CERNBot 181
A.2 Example 1 of a sample set of images captured using the provisional commercial camera system during the demo test in the LHC at a particular location 182
A.3 Example 2 of a sample set of images captured using the provisional commercial camera system during the demo test in the LHC at a particular location 183

B.1 Example 1 of crack detection results from the SDNET subset 185
B.2 Example 2 of crack detection results from the SDNET subset 185
B.3 Example 1 of crack detection results from the LHC dataset 186
B.4 Example 2 of crack detection results from the LHC dataset 186
B.5 Example 3 of crack detection results from the LHC dataset 187

C.1 Example 1 of specular highlights detection results from the LHC dataset 189
C.2 Example 2 of specular highlights detection results from the LHC dataset 189

D.1 Example showing similar detection results from the majority voting and PCA-weighted summation methods 191
D.2 Example showing different change detection results from the majority voting and PCA-weighted summation methods, as a result of the change map analysis stage on the respective change maps 192
D.3 Example showing different change detection results from the majority voting and PCA-weighted summation methods 193

List of Tables

2.1 Comparison of image fusion levels and performance 16

4.1 Dataset summary, including data type, camera and resolution . . . 47

5.1 Different augmentation pipelines 60
5.2 Mean IoU from the Mask R-CNN model trained on the SDNET subset 65
5.3 IoU from the different models trained on the SDNET subset 68
5.4 Mean IoU from the Mask R-CNN model trained on the LHC dataset 71
5.5 IoU from the different models trained on the LHC dataset 71

6.1 Summary of results on the PURDUE dataset during the validation of the U-Net model with different encoder architectures 94
6.2 Validation results on the LHC dataset for different encoder architectures 96

7.1 Quantitative results from the change detection algorithm 135

List of Acronyms

AI Artificial Intelligence. 60

ALICE A Large Ion Collider Experiment. 3

ANN Artificial Neural Network. 27

API Application Programming Interface. 36

ASPP Atrous Spatial Pyramid Pooling. 12

ATLAS A Toroidal LHC Apparatus. 3

BN Batch Normalisation. 54, 87, 88, 93, 94, 96

CERN European Organisation for Nuclear Research. 1–4, 40

CM Change Map. 25, 27, 30, 37, 106, 109, 110, 112, 113, 116, 120, 123–125, 127–130, 134, 136, 142

CMS Compact Muon Solenoid. 4

CNN Convolutional Neural Network. 11, 12, 24, 30, 51, 56, 75, 76

CSV Comma-Separated Values. 143, 144

CT Computed Tomography. 17

CVA Change Vector Analysis. 18, 26

DCT Discrete Cosine Transform. 21

DSLR Digital Single Lens Reflex. 44

DT Discrete Transform. 21

DWT Discrete Wavelet Transform. 21

EM Expectation-Maximisation. 27

EN-SMM Engineering, Survey, Mechatronics and Measurements. 2

FCN Fully Convolutional Network. 12, 24

FN False Negative. 133

FoV Field of View. 2, 32, 42, 44, 130

FP False Positive. 133

FPN Feature Pyramid Network. 57

GT Ground-Truth. 60–63, 75–77, 90–92, 103, 185

GUI Graphical User Interface. 143

HOG Histogram of Oriented Gradients. 12

IHS Intensity Hue Saturation. 19, 20

IoU Intersection over Union. 63, 64, 68, 70, 71, 75, 76, 81, 92, 93, 96

IR Infrared. 17

KTT Kauth-Thomas Transformation. 26

LHC Large Hadron Collider. 1, 2, 4, 40–43, 45, 46

LiDAR Light detection and ranging. 17

MLP Multiple Layer Perceptron. 12

MRI Magnetic Resonance Imaging. 17

MRO Mechatronics, Robotics and Operations. 2

ND Non-Destructive. 6

OBCD Object-based Change Detection. 78, 81, 82

PBCD Pixel-based Change Detection. 37, 78, 81, 106, 111, 123, 124

PCA Principal Component Analysis. 18–20, 26, 37, 117, 118, 124, 125, 135, 136

ReLU Rectified Linear Unit. 13, 52–54, 87

ROI Region of Interest. 56, 58, 59

ROV Remotely Operated Vehicle. 2, 43

RPN Region Proposal Network. 57, 58, 64

SDK Software Development Kit. 46

SfM Structure from Motion. 30, 32, 33

SLAM Simultaneous Localisation and Mapping. 29

SMB Site and Management Buildings. 2, 4

SNR Signal-to-Noise Ratio. 18

SP segmentation prediction. 63

SPC Statistical Process Control. 119, 120

SPS Super Proton Synchrotron. 1

SSIM Structural Similarity Index. 37, 121–124

SVDD Support Vector Data Description. 11

SVM Support Vector Machine. 11, 27, 28

SWT Stationary Wavelet Transform. 21

TIM Train Inspection Monorail. 2, 40, 43, 44

TIR Thermal Infrared. 9, 17, 42, 143, 144

TLS Terrestial Laser Scanner. 7, 8, 32

TM Thematic Mapper. 20

TN True Negative. 133

TP True Positive. 133

TS Total Station. 33

TT1 Transfer Tunnel 1. 1

UQI Universal Quality Index. 121

VE Valley Emphasis. 115, 116

VR Virtual Reality. 33, 34, 39, 143

1 | Introduction

To safeguard the structural integrity of concrete tunnels, periodic inspections are essential to pre-empt further damages and accidents. Monitoring is principally done through visual observations which require people to be physically present on-site, possibly exposing them to hazards that might be present in the environment. This has led to an increase in the need for these inspections to be done through automatic platforms. Robotic operations can reduce direct human intervention and lower the time needed for inspection and operation disruption, while deployment of computer vision techniques allows more objective and faster inspection analysis.

The European Organisation for Nuclear Research (CERN) has more than 50km of tunnels hosting machinery used for various experiments in difficult environments. Examples of these tunnels, are those hosting the Large Hadron Collider (LHC), Super Proton Synchrotron (SPS) and Transfer Tunnel 1 (TT1) shown in Fig. 1.1.



(a) LHC (b) SPS (c) TT1

Figure 1.1: Some of CERN's tunnels

Systems to aid tunnel monitoring are already in place through mobile applications used to record the structural integrity of the tunnels using location-tagged images taken during observations, however these are used as a reference rather

than to automate inspections. A vision-based system that monitors changes on the tunnel wall linings was further developed in an earlier project [1, 2]. It uses images from a single camera placed on a Train Inspection Monorail (TIM) and thus has a limited Field of View (FoV).

This work forms part of the strategy at CERN to develop an automated tunnel structure health monitoring solution. It is part of a collaborative project between the University of Malta, the Mechatronics, Robotics and Operations (MRO) section within the Engineering, Survey, Mechatronics and Measurements (EN-SMM) group and Site and Management Buildings (SMB) Department at CERN. This work builds on [1] while focusing on the implementation of novel techniques for remote and automated tunnel monitoring using computer vision and data fusion.

1.1 Aims and objectives

This work aims to contribute to the field of tunnel inspection by providing a remotely operated comprehensive framework to monitor the CERN LHC tunnel which with a few modifications might be also deployed in other infrastructures. The main objective is to automate the inspection process. First, the data acquisition needs to be automated. Hence, this works aims to design a multiple image sensor set up to obtain data for tunnel inspection and to automate the capturing of data from different sensors placed on a Remotely Operated Vehicle (ROV).

Following this, to automate the structural inspection process in general, this work mainly seeks to:

- automate crack detection;
- use computer vision techniques to implement different bi-temporal and decision-level image fusion for change detection;

1.2 CERN

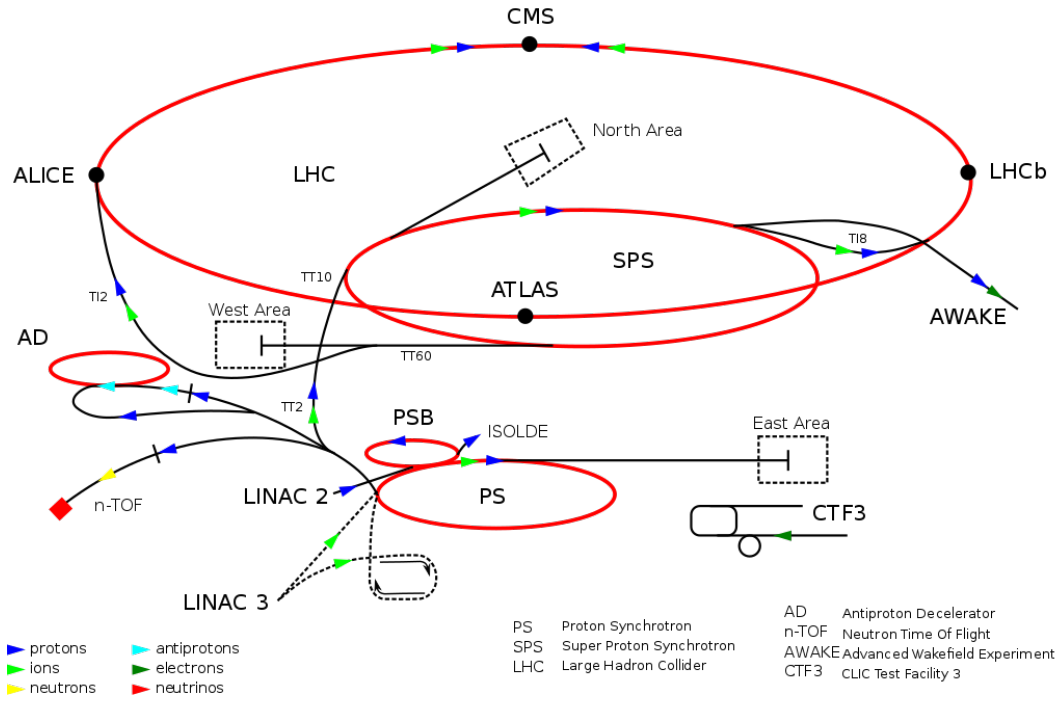


Figure 1.2: CERN accelerator complex [3]

CERN is a European research organisation located in the suburbs of the Swiss city of Geneva, operating the largest particle physics laboratory in the world. At CERN, scientists, physicists and engineers from different countries (including Malta) collaborate on various projects, rendering it a very multicultural and diverse research environment. It was established in 1954 and it has been continuously evolving since then. CERN is at the forefront of scientific research with very important discoveries taking place inside the experimental areas within its premises. Technological developments necessary for the construction and operation of CERN's particle accelerators and detectors have resulted in several spin-offs within the fields of engineering, computer science and medicine.

CERN uses large, complex instruments and machinery to study the basic constituents of matter and the related fundamental forces. As a result, a number of facilities hosting different experiments: A Toroidal LHC Apparatus (ATLAS), A

Large Ion Collider Experiment (ALICE) and Compact Muon Solenoid (CMS) were constructed within the CERN accelerator complex as illustrated in Fig. 1.2. The main machines used at CERN are purpose-built particle accelerators and detectors. One of CERN's accelerators, the LHC was built in order to answer open fundamental questions in particle physics, in particular those concerning the Standard Model and the Higgs Boson particle. The 27 km long tunnel hosting the LHC lies at around 100 m below the ground, with most of it being located in France.

1.2.1 LHC tunnel scenario

The LHC tunnel is circular in shape however, over its large distance, the curve is practically negligible for the field of view considered, as shown in Fig. 1.1(a) and Fig. 1.3. This tunnel requires regular monitoring whereby measurements of radiation, oxygen presence, temperature and humidity are conducted. Furthermore, by sending personnel on site, infrastructure inspection is carried out by visually checking the area for any structural changes and making the necessary sketches and measurements to later consult the SMB Department taking care of the facility. Such an operation entails a considerable amount of time, which is also limited by the tunnel access time as well as personnel to physically enter the tunnel, with risks of radiation present, further limiting them only to certain areas.



Figure 1.3: Low lighting conditions which vary from one area to another

To a certain degree, the LHC tunnel presents a harsh working environment. It has low lighting conditions which vary from one area to another as shown in Fig. 1.3 as well as over time. Due to the limited free space available, any machinery used for inspection surveys should be small in dimensions. The non-uniform environment, comprising cables, wall racks, pipes and the accelerator itself also present difficulties. The amount of dust present in the LHC tunnel should also be taken into consideration when placing devices in the tunnel, as they can be affected.

Such scenario conditions raise the need of a structural health monitoring solution to remotely collect data from the tunnel and perform objective inspections.

1.3 Thesis structure

The remainder of this thesis is structured as follows. Chapter 2 reviews previous works in the related fields. An overview of the proposed solution together with an introduction to its different modules are presented in Chapter 3. The data acquisition module is described in Chapter 4. Crack detection using deep learning techniques is discussed in Chapter 5. Specular highlights localisation using a deep-learning based segmentation approach, as a pre-processing stage before image comparison is described in Chapter 6. The implemented change detection techniques using computer vision and data fusion techniques are explained in Chapter 7. A summary description of the proposed solution highlighting the contributions to the state of the art and suggestions for future work conclude the thesis in Chapter 8.

2 | Literature review

2.1 General tunnel inspection

Aging infrastructures may have issues in structural integrity due to poor maintenance, construction defects, unexpected overloading and natural phenomena. Consequently, to maintain compliance and safety in concrete tunnels, regular inspections are necessary. To avoid negative effects, Non-Destructive (ND) approaches such as strength-based, sonic and ultrasonic, radar, thermographic, electrical and endoscopic are commonly adopted as discussed in [4, 5]. However, inspections are predominantly performed through periodic visual observations, looking for lining defects such as cracks, spalls and water deposition to locate possible changes between one survey and another. To make such observations, inspectors are required to be physically on-site. Associated with this, there are several disadvantages such as the human presence in hazardous environments and the financial cost involved to train and hire people to do the surveys. In addition, these inspections require considerable time to perform, leading to longer operation down-times and thus higher monetary losses. In addition, the outcome is subjective, leading to possible inaccuracy such as missing or false detections.

For these reasons, significant attention has been given to the field of automated inspections of tunnel structures as recorded in [4] and [6]. Such solutions were proposed to increase personnel safety and save time with remotely operated fast data acquisition, identification and documentation of tunnel lining defects. The extensive review in [7] presents previous works made within the different fields of



Figure 2.1: Invar wire and gauge used during manual surveys [11]

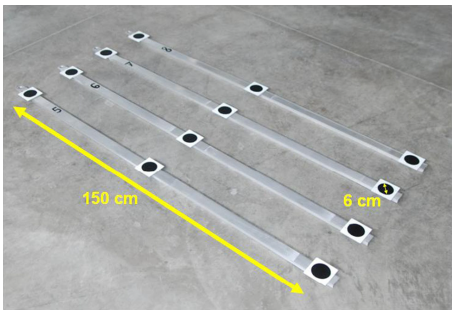


Figure 2.2: An example of 'photogrammetric levelling' as used in [11]

the image-based tunnel inspection spectrum, such as for the monitoring of tunnel profiles, crack and leakage localisation and tunnel surface documentation.

2.1.1 Tunnel wall deformation

The structural condition of a tunnel can be analysed through measuring and monitoring of the deformation of its cross-section such that proactive maintenance can be conducted in time. To measure tunnel profiles, several methods include the use of a mechanical gauge such as a tape extensometer as shown in Fig. 2.1, Terrestial Laser Scanner (TLS) [8–10] and geodetic instruments. Physical indicators as proposed in [11], and shown in Fig. 2.2, may be used to measure the tunnel profile. Deformations in the tunnels' cross-section are found by measuring the target coordinates on the images captured along the wall.

Other works, instead of using physical targets, project a laser light to create virtual targets. In [12], laser pointers are used to outline the tunnel surface and an image of the resulting profile is captured. 3D tunnel clearance inspection using

the optical triangulation principle through structured light projectors and multiple cameras was proposed in [13]. A laser emitter and a set of cameras were similarly used in [14] to generate a perpendicular plane to the horizontal tunnel axis. Later, photogrammetric methods and geometry equations are used to obtain features of the tunnel profile. Commercial systems, such as [15], were also proposed to replace conventional methods to monitor tunnel profiles. Despite these efforts to automate tunnel clearance measurement using image processing, both commercial projects and literature on this are lacking and the most common approach is still TLS. This is mostly because of its large scale tunnel scan capacity and simultaneous 3D model generation and work progress monitoring.

2.1.2 Crack and defect detection

Cracks are the initial signs of infrastructure deterioration, thus if detected at an early stage, larger damages can be preempted. Cracks may develop due to aging, topographic changes, downpours, recurring weight loading, poor repair and expansion/contraction variations of concrete due to temperature changes. On-site visual inspection, physical measurements and manual sketches are generally used to locate cracks in concrete structures. Such an approach is dependent on the surveyors' experience, leading to report subjectivity. Hence, considerable effort has been made to objectively identify and evaluate cracks' status using image processing and pattern recognition techniques. Literature specifically dealing with the detection of cracks in tunnels is limited. However, as reviewed in [16, 17], various image processing-based crack identification methods were proposed in other infrastructures including bridge decks, pavements and roads.

Since the background is usually brighter than crack areas, thresholding techniques can be used to segment the potential crack regions in an image. In [18], images of tunnel linings are captured and cracks with a larger variation along the line edges, are chosen. Detected edges joined to others are extracted using hysteresis thresholding. To locate cracks in subway tunnels, a wireless sensor network was



Figure 2.3: Wall image and the result of crack segmentation from [13]

built in [13]. A threshold segmentation using the Otsu method [19] is made. After, crack properties including length, width and area are calculated from the segmented images such as that in Fig. 2.3 and compared against different thresholds to locate the definite crack areas.

Cracks usually occupy a small region within images and the variance with the background is affected by other objects on the surface including racks, cables and pipes thus, it is difficult to differentiate them from the background. To mitigate such issues, a block binarisation was proposed in [20]. First, the image contrast is improved and the noise is reduced through filtering. Then, to detect cracks, segmentation via local binarisation using a threshold set to the mean intensity of a square neighbourhood of pixels is made.

The rig of line scan cameras in Fig. 2.4 was proposed to capture images and identify cracks in [21]. The centre pixel of an image region is identified as a crack seed if the overall gray level difference of the area is below a pre-defined threshold value. By recognising the line connecting these seeds, a crack is located.

In [22], Thermal Infrared (TIR) is used to detect cracks on a tunnel lining. Pre-processing is applied to each image in the frequency domain. The pre-processed image is then split up into regions such that the directionality of the texture is calculated and crack regions are identified through a thresholding stage.

Crack identification in different concrete structures using a threshold-based approach include [23–29]. While this method is relatively straightforward and computationally inexpensive, its accuracy merely depends on the preset threshold value, causing difficulty when the sizes of cracks vary considerably.

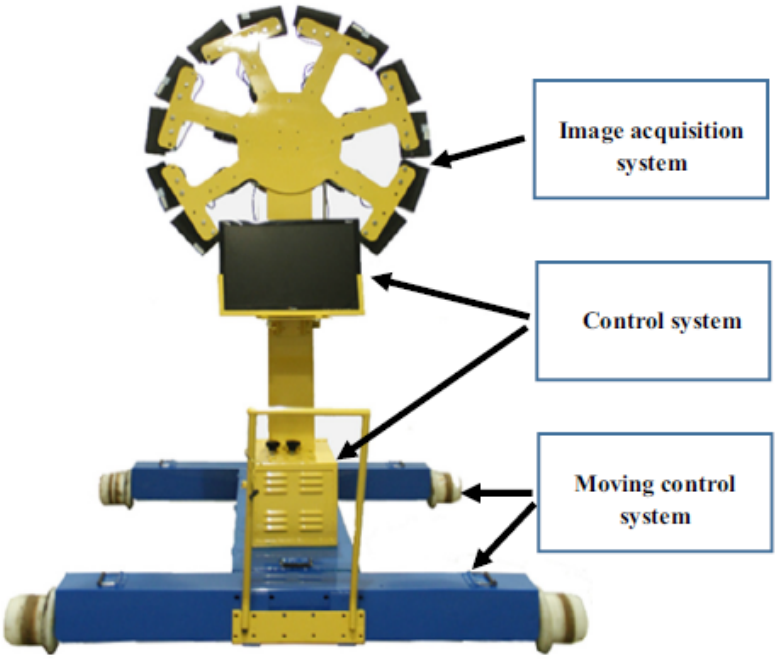


Figure 2.4: The image acquisition system used in [21]

Visual changes in the texture of a surface often indicate defects or flaws in it. Using a Wigner model, [30] proposes an algorithm to identify cracks in backgrounds with complex textures. In works such as [31, 32] a rotation invariant Gabor Filter is used to conduct texture analysis for crack detection at the pixel level, regardless of the crack's direction.

Due to their contrast with the surroundings, salient regions are visually more conspicuous however, works using saliency for crack identification, such as [33] are very limited in number. In [34], complex coefficient maps are generated from a 2D continuous wavelet transform and wavelet coefficients maximal values are found for crack detection. However, such approaches cannot handle examples with cracks lacking continuity or having a high curvature due to the anisotropic characteristic of wavelets.

Challenges due to the environment and characteristics of concrete cracks, make it difficult to apply rule-based methods that are capable of effectively extracting

generalised features. These methods often rely on manually fine-tuned parameters which do not cater for the complex conditions often exhibited by concrete surfaces. A more adaptive solution relies on the use of pattern recognition and machine learning algorithms. In [35], a thresholding stage is used to identify crack areas which are then analysed through features such as pixel number, average grey level and standard deviation of the shape distance histogram. Such features are fed as inputs to various machine learning models for a crack vs. non-crack classification of the candidates. In [36], Support Vector Data Description (SVDD) was adopted to identify cracks on concrete surfaces. Colour images are first converted to grayscale and segmented through a threshold followed by a morphological closing operation. To identify cracks, properties including packing density, eccentricity and circularity are used to build a feature vector which is input into a trained SVDD.

For automatic identification and characterisation of cracks in pavements, [37] used a combination of unsupervised learning (clustering) followed by supervised learning (classification). A fuzzy logic-based algorithm was introduced in [38] to locate cracks in pavements. In [39], AdaBoost was used to form textural descriptors while CrackForest [40], adopts a random structured forests descriptor to characterise cracks. In [41], a comprehensive review of defect detection on pavements identified Support Vector Machine (SVM) as the most commonly used supervised learning approach for the detection of road cracks.

2.1.2.1 Crack detection using deep learning

Although the performance of these methods is high, it is dependent on the extracted features. Due to surface complexities, it is difficult to find features applicable for diverse structural scenarios. Hence, deep learning algorithms have been recently used to overcome such variability limitations.

In [42, 43], vision-based methods using a deep Convolutional Neural Network (CNN) architecture was proposed to detect concrete cracks. However, these do not consider the pixel level and can only identify cracks at patch level. Using local

patch information, [44] suggests a CNN to classify an individual pixel as part of a crack or not. Despite this, the spatial relations among pixels are still ignored and the crack width is overestimated. Similarly, [45] uses a CNN to predict the class for every pixel in an image, however, at the pre-processing stage, manually designed feature extractors are still needed, thus the CNN is only used as a classifier. A CNN-based defect detector was proposed in [46]. Image properties including edges, texture, entropy, frequency and Histogram of Oriented Gradients (HOG) are used to construct input high-level features to a Multiple Layer Perceptron (MLP), trained to detect tunnel lining defects. In [47], a modified version of AlexNet is used as a classifier and a sliding window search is made to locate cracks in an image. Using a CNN and taking the advantage of atrous convolution, Atrous Spatial Pyramid Pooling (ASPP) module together with a depthwise separable convolution, an end-to-end crack detection model is proposed in [48].

In [49], defects on a subway tunnel are found using semantic segmentation through features extracted by Fully Convolutional Network (FCN). Using multiple forward inference and backward learning loops, separate FCN models are trained for crack and leakage detection. An encoder-decoder FCN network with the VGG16-based encoder is trained end-to-end on a subset of annotated crack-labeled images for semantic segmentation to detect cracks on concrete in [50].

The U-Net [51] model was adopted to detect the concrete cracks in [52]. The focal loss function is selected as the evaluation function and the Adam optimiser is used to train the network on a small set of images under various conditions such as different illumination, complex backgrounds and varying crack widths. In [53], a cost function based on a distance transform is introduced to assign pixel-level weight according to the minimal distance to the ground-truth segmentation in order to train a U-Net based model for automatic crack detection, achieving a high pixel level segmentation accuracy. The pixel-level surface crack detection proposed in [54] is also based on U-Net with a slight modification to avoid shrinking for all the convolutional layers through zero padding.

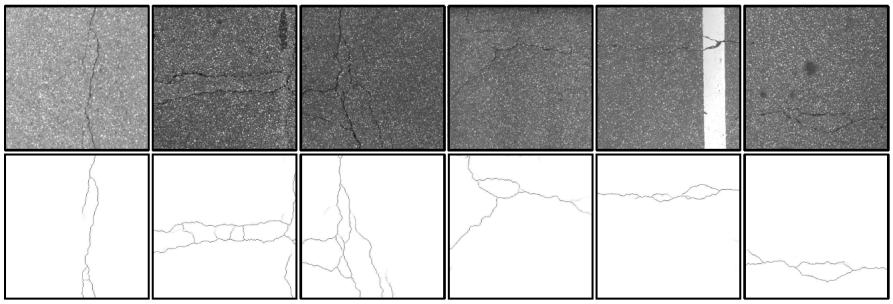


Figure 2.5: A sample of crack detection results from DeepCrack [55]

To detect cracks on bridges, [56] proposes a model based on SegNet [57] with a few changes to the original architecture, including a different input size, a batch normalisation layer between the convolutional layers and a Rectified Linear Unit (ReLU) activation function. To segment crack images such as those in Fig. 2.5, DeepCrack [55], uses a skip-layer fusion to connect the encoder and decoder paths in the original SegNet model in order to utilise both continuous and sparse feature maps at each scale. Using the cross-entropy loss, a one-channel prediction map showing the pixel probability of belonging to the crack, is generated.

2.2 Data fusion

In some instances, neither a single method nor an individual sensor suffice to fully examine objects under inspection. Multiple techniques should be applied such that their combination ultimately benefits from their respective advantages achieving increased reliability, lower detection error rate, higher redundancy and improved identification. Data fusion has been used in various fields, such as: battlefield surveillance, remote sensing [58, 59], control of autonomous vehicles [60, 61], biometrics [62–64], wireless sensor networks [65–67] and robotics [68, 69].

Applications for data fusion are so widespread that no common architecture can be used across all fields. The type of fusion architecture has a vital role in the efficiency of the processed information and the significance of the decision made at the output level. Considering this, significant research concerning fusion architectures

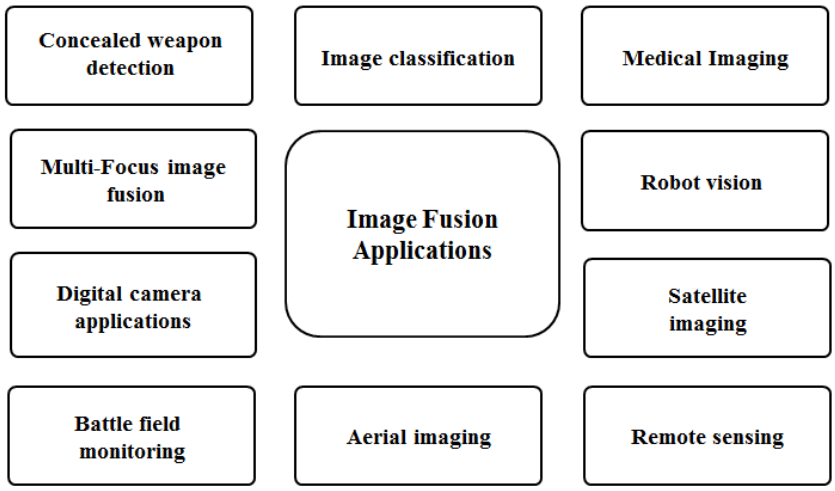


Figure 2.6: Several applications of image fusion applications [77]

was made, as discussed in [70, 71]. The first architecture was proposed in [72]. A general data fusion structure based on multi-sensor integration was then presented in [73]. The fusion centre processes the data collected at sensor level hierarchically and sequentially. A hierarchical architecture and model for data combination was proposed in [74]. The architecture introduced in [75, 76] consisting of four levels: logical robot, functional, control and decision levels implements modules in real-time systems. The architecture selection is a trade-off between different features; computing resources, desired accuracy, sensors capability, communication bandwidth and available budget.

2.3 Image fusion

Image fusion integrates multiple images in order to increase the visual interpretation both for humans as well as in machine vision. Image fusion is widely used in the areas of medical diagnosis, military, satellite imaging, object detection and recognition, robotic vision, surveillance and other fields as illustrated in Fig. 2.6.

2.3.1 Levels of image fusion

Image fusion can be categorised into low, medium and high levels while other literature [78] may refer to pixel, feature and decision levels. Regardless of the fusion level, the main aim is to maintain all valid information from multiple images such that the resultant image contains more accurate and complete information than the individual images. At the same time, the fusion should not introduce artifacts that interfere with subsequent analysis.

At the lowest, pixel-level, the unprocessed outputs from different sensors are mixed in the signal domain to generate a fused signal. The fused result, a single gray or chromatic image, is created by fusing individual pixels, usually after some form of processing is applied to the source images. Feature-level fusion extracts salient information such as edges, lines, corners and texture parameters from independent images and merges them into one/multiple feature map/s which are used with/instead of the original data for further processing. Such a fusion type is commonly used during pre-processing for image segmentation or change detection. Decision-level fusion merges interpretations of different images obtained by multiple algorithms of local decision makers to yield a final decision. When the results are expressed as probabilities, fusion is referred to as 'soft fusion' while, if decisions are used, the term 'hard fusion' applies. Table 2.1 portrays the different levels of image fusion including a comparison of their individual characteristics.

2.3.2 Image fusion categories

These include:

- multi-view - fusing images from the same modality from different viewpoints;
- multi-modal - fusing images coming from different sensors;
- multi-temporal - fusing images taken at different times to identify changes among them or to synthesise images of objects;

Table 2.1: Comparison of image fusion levels and performance (adapted from [79])

Characteristic	**Pixel-level**	**Feature-level**	**Decision-level**
information content	highest	medium	lowest
loss of information	lowest	medium	highest
fault tolerance	worst	medium	best
noise immunity	worst	medium	best
sensor dependancy	highest	medium	lowest
merging difficulty	hardest	medium	easiest
pre-processing	lowest	medium	highest
classification performance	best	medium	worst

- multi-focus - fusing images of the same viewpoint at different focal lengths;
- multi-frame super-resolution - fusing two or more same-scene, same-modality images which are blurred and/or noisy to produce a deblurred and/or de-noised image.

In every category, if fusion is conducted at the pixel level, the first stage involves image registration to bring the input images in spatial alignment to each other. Registration applications can be classified by the image information used. Methods which use the whole image's data are referred to as area-based while those using only particular pixels within the image are called feature-based methods.

2.3.2.1 Multi-view image fusion

In multi-view fusion, images of the same scene captured from several viewpoints by the same sensor or a set of sensors having similar characteristics, are fused to supply complementary information from different views. This is commonly used to attain a higher resolution image or to recover a scene 3D representation. A multi-view dense matching algorithm for high-resolution aerial images based on a graph network was proposed in [80]. Based on the generated graph, point clouds of base views are constructed by triangulating all connected nodes, followed by a fusion process using the mean reprojection error as a priority measure. A probabilistic

algorithm for multi-view reconstruction from the fusion of calibrated images was presented in [81]. The algorithm is based on multi-resolution volumetric range image integration. The method introduced in [82], merges a set of depth maps into a single point cloud so that the redundant points are fused together by assigning a higher weight to specific measurements. The original depth maps are then re-registered to the fused point cloud to generate the refined extrinsic parameters.

2.3.2.2 Multi-modal image fusion

This type of fusion combines images captured by different types of sensors to provide a more comprehensive and informative view of the task at hand. Images of different types, such as visible, Infrared (IR), Computed Tomography (CT), and Magnetic Resonance Imaging (MRI), are good source images for fusion. A review of existing approaches and results on multi-modal data fusion from different disciplines was presented in [83]. An overview of the main challenges in this type of fusion is given in [84].

The applications using this type of fusion vary among remote sensing, medical imaging, multimedia and inspection. Multi-spectral, hyperspectral, radar and Light detection and ranging (LiDAR) images can be available for the same geographical region. In [85], a review of the approaches used to combine this data to improve the classification of materials, was presented. New approaches and challenges of audiovisual data fusion for speech recognition were presented in [86]. TIR images can differentiate targets from their backgrounds through their radiation difference. This functions in all-weather and all-day/night conditions. On the other hand, visible images provide colour and texture details with high spatial resolution which are consistent with the human visual system. Hence, by fusing such image types, thermal radiation information and detailed texture information can be combined. A survey of works employing this type of fusion was published in [87].

2.3.2.3 Multi-temporal image fusion

This type of fusion merges images of the same view, captured using the same modality at different times. Generally, it is used for the detection of the medical and environmental changes or to synthesise images of items which were not photographed at a specific time. A number of methods can be applied to statistically and numerically process co-registered temporal images to generate new data. General temporal image fusion techniques include arithmetic operations involving addition/subtraction, rationing/multiplication or transformed-based methods such as Principal Component Analysis (PCA).

A common approach for multi-temporal optical image fusion is that of image comparison which mainly relies on the difference operator. This is because the noise model in optical images is additive and the natural classes have a Gaussian distribution. Thus, the difference operator results to be the most effective one. Subtraction of same-scene images serves to enhance their differences so that even minor changes become detectable and can be evaluated correctly. In remote sensing, several difference operators such as univariate image differencing, vegetation index differencing and Change Vector Analysis (CVA) are used to monitor for environmental changes as discussed in [88]. In medical imaging, a typical example of temporal fusion using subtraction, is subtractive angiography where the base image captured before applying a contrast agent is subtracted from the image of the same scene once the substance is applied, as explained in [89]. Similarly, temporal images may be added together or averaged in order to increase the contrast of interesting structures and suppress random zero-mean noise. This technique may be useful in gamma imaging, where the Signal-to-Noise Ratio (SNR) of individual images is usually low.

2.3.3 Image fusion domains and techniques

Image fusion methods can be further classified by the type of domain in which they operate. The spatial domain methods deal with the manipulation of pixel values of source images. Intensity Hue Saturation (IHS) [90] and PCA [91] are commonly used in this domain. In frequency domain fusion approaches, the image is first transferred to the frequency domain, then the fusion operations are conducted and finally the inverse transform is applied to get the final image. Pyramid-based transforms and Discrete transforms are commonly used. Reviews of the different image fusion techniques and applications can be found in [92–96].

2.3.3.1 Spatial domain

The first evolution of image fusion research performed basic pixel by pixel related mathematical operations like summation, difference and mean. Average fusion estimates the mean of the intensity of the input images on a pixel-by-pixel basis. The technique assumes very accurate spatial and radiometric alignment. For each pixel in every image, its block mean is computed. The corresponding pixel in the combined image is selected by taking the pixel with the maximum block mean amongst all the corresponding pixels in the images. This fusion technique lowers the resultant image quality by bringing in noise into the combined image and tends to reduce the contrast also. In order to improve the fusion reliability, [97] uses a weighted average technique where varying weights are designated to all source images. The fused image is obtained through the weighted addition of all corresponding pixels. The select-maximum/minimum technique chooses the maximum and minimum pixel values from corresponding images and generates the fused image by averaging the minimum and maximum values of all corresponding pixels in all the images.

PCA transforms a number of correlated variables into several uncorrelated ones. For image fusion, PCA is applied to create a weighted sum of the source images.

The weights for each source image are obtained from the normalised eigenvector of the covariance matrices of each source image. This approach is commonly used in multi-spectral imaging such as in [98, 99], where the PCA method generates uncorrelated images and replaces the first component with the panchromatic band, which has a higher spatial resolution than the multi-spectral images. After, the inverse PCA transformation is used to get the image in the RGB colour space. Image fusion using PCA and wavelets can also be used as in [100]. Useful information may not be completely represented by a single pixel but also present in the size, shape and edges of the image content hence, [101] suggests a PCA region-based fusion using a 3×3 kernel of the input images. The PCA method is simple, computationally efficient and results in high spatial quality. On the other hand, it may introduce some colour distortion and spectral degradation.

IHS is a common approach for fusing single band, pan, high and low spatial resolution, multispectral remote sensing images such as in [102, 103]. The R, G and B bands of the multispectral image are transformed into IHS components, using the pan image to replace the intensity component and the inverse transformation is then performed to obtain a high spatial resolution multi-spectral image. In [104], IHS is used to integrate radar with Landsat Thematic Mapper (TM), airborne geophysical and thematic data. The IHS method is simple, computationally efficient and has high sharpening ability. Unfortunately, it only processes three multi-spectral bands and may generate some colour distortion.

Multiplication is not widely used as an image fusion operator, however, an important fusion application which uses multiplication is in Brovey Transform. As an example, [105] proposes a remote sensing image fusion approach based on a modified version of Brovey transform and wavelets. The approach is relatively simple to implement, computationally efficient and produces higher contrast images, however it can generate some colour distortion.

2.3.3.2 Pyramid-based

Another fusion approach is to construct a pyramid transform, comprising multiple images at a set of scales which together represent the original image. The fused image is obtained by merging the pyramids at the respective levels and then taking the inverse pyramid transform. The Gaussian pyramid [106], Laplacian pyramid [107] and ratio of low pass pyramid [108] are different types of pyramids used in this technique. Pyramid decomposition-based methods generate very similar outputs while the number of decomposition levels affects the fusion result.

2.3.3.3 Transform and wavelet-based

Whereas pyramid-based fusion methods use filters, Discrete Transform (DT) methods use transforms. Although different transforms may be used, a common pipeline is followed. If the input images are in colour, their RGB channels are first separated and the specific transform technique is then applied. Next, the mean of corresponding pixels is computed to get the fused transform components. The inverse transform is then applied to convert the transform components into an image. In the case of colour images, the separated R, G and B planes are finally combined. A review on DT image fusion is found in [109].

Discrete Wavelet Transform (DWT) is used to fuse medical images in [110]. The Kekre's Wavelet Transform was introduced in [111] and a comparison of different techniques using this transform for image fusion was presented in [112]. Other transforms such as Discrete Cosine Transform (DCT) [113], Stationary Wavelet Transform (SWT) [114] and a combination of DCT and Stationary Transform [115] are also recorded in image fusion literature.

2.4 Change detection

This is the identification of variations in an object state by monitoring it at different times. It quantifies temporal effects using multi-temporal datasets. Timely and

correct change detection provides the basis for better understanding the evolution of an environment while identifying the relationships, temporal effects and interactions of objects within a scene. Identifying changed areas in same-scene images taken at different times is essential for varying applications in multiple disciplines. A vast amount of related literature is found in [116–118] amongst others. Important change detection applications include video surveillance [119–121], remote sensing [122–124], medicine [125–127], underwater sensing [128–130], civil infrastructure [131–133] and intelligent transportation and traffic systems [134–136].

A change may be caused by a number of factors, including appearance or disappearance, relative motion or shape changes of objects. Furthermore, images of static objects can change in brightness and/or colour. In general, change detection answers some of the fundamental questions such as how fast changes are taking place, their size, shape and also the trend at which they are occurring. However, during this detection process, various challenges exist, limiting its accuracy. The absence of a reference background, differences in lighting conditions and varying viewpoints make the multi-temporal comparison difficult. Moreover, the lack of a priori information about the type, shape and size of changed areas make identification challenging.

2.4.1 Pre-processing for change detection

The aim of change detection is to simultaneously identify significant changes and reject unimportant ones. Apparent intensity changes resulting from camera motion and different lighting should be ignored. Hence, pre-processing involving geometric, radiometric adjustments and semantic segmentation is generally required.

2.4.1.1 Alignment corrections

Proper alignment of images from different viewpoints and at different times is fundamental for change detection. This is commonly referred to as image registration and is the process of spatially aligning multiple same-scene photos which are either

taken from different angles using multiple sensors or at a different time. Detailed surveys on image registration methods can be found in [137–139]. The possibility of localised registration errors causing false changes, should be carefully catered for, consequently, [140–142] study how change detection is affected by registration errors.

2.4.1.2 Radiometric adjustments

Intensity changes caused by variations in the strength or location of light sources within a scene should be compensated for to prevent their classification as changes. Several techniques for radiometric adjustments exist, including: intensity normalisation, homomorphic filtering, illumination modelling and linear transformations of intensity. Independent on the field in which change detection is applied, this pre-processing step should be considered in order to achieve a good quality change detection result. A discussion of the various image pre-processing techniques that can be applied, is found in [116].

2.4.1.3 Semantic segmentation

This is the process of segmenting an image by specifying a class for each pixel. Semantic segmentation plays a very important role in scene understanding in various computer vision applications in which each visual information has to be associated with an entity while considering the spatial information. In change detection applications, semantic segmentation can be applied to delimit nuisance regions that might otherwise be falsely identified as a change such as specular highlights occurring because of electronic flash units.

There are different types of segmentation approaches such as those using thresholding, edge detection and clustering. While these are relatively easy to implement and incur low computational cost, they have various limitations. When there is no significant grayscale of colour difference, it is very difficult to get accurate segments with region-based segmentation using thresholding. Edge-based approaches

are not suitable when a large amount of edges are present in the image or if there is low contrast between objects. Whilst they generate excellent bounded regions, clustering-based approaches incur an expensive computation time and are not suitable for clustering non-convex scenarios when using distance based algorithms such as k-means clustering.

With the introduction of CNN and deep learning, semantic segmentation advanced rapidly in the last few years. Initial approaches involved patch classification to separately classify each pixel into classes using a patch around it. To overcome the fixed size constraint of fully connected layers, FCN was proposed in [143]. The latter popularised CNN architectures for dense predictions without fully connected layers, allowed segmentation maps to be generated for images irrelevant of their size. Subsequent approaches on semantic segmentation used this paradigm.

In addition to fully connected layers, another issue with using CNNs for semantic segmentation is pooling layers. While the latter increase the field of view and aggregate the context, they discard the 'where' information, which is required by semantic segmentation. There are two main architectures to tackle this issue: encoder-decoder and atrous convolutions. Popular architectures using the encoder-decoder structure include U-Net [51], SegNet [57] and RefineNet [144]. Works based on dilated/atrous convolutions for semantic segmentation include [145], DeepLabV2 [146], DeepLabV3 [147] and Pyramid Scene Parsing Network [148].

2.4.2 Change detection techniques

Change detection techniques can be categorised by the unit of image analysis. Pixel-based approaches use pixel values as the fundamental unit of analysis while object-based methods use segmentation to extract regions on which analysis is then made. Below, both types of change detection are discussed in terms of the different approaches taken, including image algebra, transformation and classification.

2.4.2.1 Pixel-based methods

In the simplest approach, referred to as image differencing, two images $I(x, y, t_1)$ and $I(x, y, t_2)$ of the same scene taken at times t_1 and t_2 respectively, are subtracted pixel-wise. Following subtraction, the magnitude of the difference value is checked against a threshold. Pixels with a difference higher than the threshold are noted as 'change' and set to 1, otherwise they are noted as 'no change' and set to 0, creating a Change Map (CM). Due to its simplicity and computationally inexpensive nature, this approach is most often adopted in motion detection such as in [149]. However, it requires exact image registration and is highly dependent on a threshold.

A similar method that uses the same pixel by pixel logic, but calculating a ratio instead, is commonly referred to as image ratioing. Pixel intensity values of one image at t_1 are divided by the corresponding pixel values at t_2. Unchanged pixels will have a ratio equal to or near 1. This method also requires exact registration as it is pixel-based as well as a comparison against a ratio threshold. Also, cases requiring division by zero need to be handled. On the other hand, a vital advantage of this method is that it minimises the variations in illumination such as shadow. A comparative study of the image differencing and image ratioing methods as used in remote sensing is found in [150].

Image regression can also be used for change detection. This approach establishes a relationship between temporal images through a regression function. Image at time t_2 can be expressed as a linear function of image at t_1:

$$I(x, y, t_2) = \alpha I(x, y, t_1) + \beta \tag{2.1}$$

where α and β are error constants. Under this assumption, one can adjust $I(x, y, t_1)$ to match the radiometric conditions of image $I(x, y, t_2)$ using least-squares regression and then subtract the regressed image from $I(x, y, t_1)$. Image regression reduces the impact of sensor and environmental differences, however it requires exact image registration and does not provide a change matrix.

CVA uses multiple image bands/channels to detect change and is often used in remote sensing. Pixel values are considered to be the vectors of the spectral values and the change vector can be computed by subtracting the vectors at various dates. The type of change can be determined from the direction of the change vector while the length corresponds to the magnitude of the change. A review on CVA algorithms for change detection was published in [151].

In contrast to the previous techniques, the following works use a transformation approach rather than linear equations. PCA [152] reduces the data redundancy by transforming multivariate data to new components with the assumption that change regions have a low correlation. Similar to PCA, other methods such as Kauth-Thomas Transformation (KTT) [153] and Gram Schmidt (GS) [154] emphasise different information in derived components however they cannot provide detailed change matrices and require the selection of thresholds to identify changed areas. Another disadvantage is the difficulty in interpreting and labelling the change information on the transformed images.

Another category of change detection techniques comprises post-classification methods [155, 156] where multi-temporal images are classified into thematic maps using both supervised and unsupervised classification. A pixel by pixel comparison of the classified images is then applied to measure the changes. A limitation of this method is that the accuracy of the final image depends entirely on the classification accuracy of the individual images. When using supervised classification methods, accurate, complete and high quality labelled training datasets are inevitable to produce accurate classification. However, acquiring such data is often difficult and time consuming. Unsupervised classification on the other hand, encounters problems in selecting the number of clusters.

Another type of classification-based approach uses probabilistic mixture models. Such an approach smoothly classifies pixels into mixture components corresponding to various generative models of change including parametric object/camera motion and illumination amongst others. This technique is best illustrated by [157].

Here, the algorithm uses the optical flow field between two images and applies the Expectation-Maximisation (EM) [158] algorithm to assign each vector in the flow field to the possible classes.

Object texture provides structural information on the items and their local neighbourhood relationships. In [159], textural values comparison is used to measure changes using a gray level co-occurrence matrix. Instead of per-pixel comparison, initially the image is split into smaller regions on which the texture is then computed and later compared at window level.

The use of machine learning for change detection is continuously increasing. One approach is to use an Artificial Neural Network (ANN) algorithm which builds networks between input images and the changes represented by the output nodes. The CM is then obtained by applying the trained network to the main dataset as in [160, 161]. Other works such as [162, 163] use SVMs as a binary classifier on stacked multi-temporal images to categorise change vs no-change. The algorithm learns from training data and finds threshold values from the spectral features automatically for classifying change from no-change. The decision tree, a non-parametric classification algorithm can also be used for change identification as in [164]. It builds a hierarchical structure where every node is used to test multiple attribute values, the test outcome is represented by each branch and tree leaves stand for the classes and their distribution. The node classification rules depend on the attribute value analysis. Genetic programming [165] and random forest [166] are amongst other machine learning algorithms that are applied for change identification.

2.4.2.2 Object-based methods

Rather than using pixels, some methods use objects extracted from the images. Object-based algorithms segment images into objects using thresholds or by extracting features before comparing the different regions. Image-object methods focus on direct image-object comparisons of geometrical properties (width, area,

compactness etc.), spectral information or connectivity analysis such as in [167]. Class-object approaches for change detection extract objects and assign them to specific classes. These independently classified objects from multi-temporal images are then compared. The classification algorithms incorporate both texture and spectral information. Such techniques include decision-tree, maximum likelihood, nearest neighbour and fuzzy logic [168].

2.4.2.3 Anomaly detection methods for change detection

Change detection can also be thought of as an anomaly detection task whereby a data instance is found to be different with respect to others in the dataset, with the deviating data being the change occurring. Anomaly detection has been well recorded in literature and various techniques were considered. Anomalies are rare under most conditions. Thus, even though training data may be available, often only very few anomalies exist among huge data points sets. Classification methods such as SVM or Random Forest will classify almost all data as normal. Generally, the class imbalance is catered for through an ensemble built by resampling data several times. If the data points are autocorrelated with each other, then simple classifiers are not adequate. In such situations or when no training data is available, unsupervised anomaly detection techniques are used. These include clustering [169], One-Class SVM, Isolation Forest [170] and Local Outlier Factor algorithm [171] among others.

2.4.3 Change detection in tunnel environments

Further to identifying defects, analysing their evolution is more advantageous as it manifests the tunnel health condition and deterioration. Such changes are usually observed by human inspectors who traverse a tunnel looking for any variations arising since an earlier survey, through on-site physical measurements as shown in Fig. 2.7. This is an expensive and subjective process and since some tunnels may present precarious working conditions, automating this process is advantageous.



Figure 2.7: On-site structural inspections in the LHC tunnel

Works dealing with tunnel wall change detection are still low in number, possibly due to the challenges in this field including the lack of light, contrast and image features, characterising images captured in tunnels. An essential prerequisite for change detection via image comparison is accurate image registration. To register remote sensing images, GPS is usually utilised but this is not available in tunnels. Instead, Simultaneous Localisation and Mapping (SLAM) may be used for image registration and to aid navigation as in [172] or to assist 3D model generation. Despite such issues the following tunnel change detection systems were proposed.

A railway tunnel change detection system using a set of cameras with overlapping viewing angles placed on a rail trolley is described in [173]. Images are aligned and filtered using normalised cross-correlation to identify the differences between two images. The method takes into account the neighbouring pixels and corrects lighting variations via mean-based intensity normalisation, making it more robust, however, the presented theory details are insufficient .

A tunnel lining change detection system using a camera on a monorail inspection train, was proposed in [1]. Position offsets between different survey images are corrected for using the mosaic-based method presented in [174]. Following this, a

hybrid change detection method using image subtraction, binary pixel comparison and optical flow is applied. Then, the 'actual change' regions are noted while 'false changes' caused by misalignment, parallax and shadows are ignored, by using a combined-weight model. Despite the good results obtained, this system monitors only a limited section of the wall.

The automated system presented in [175] uses five synchronised cameras with electronic flash units. Photos are then registered to a 3D model generated through Structure from Motion (SfM) [176] techniques. By establishing a distance function between an inspection image and its corresponding one in a previous image set, a CM is estimated. Using SfM information, a geometric expression mapping image locations to corresponding 3D points is formed. On-surface and off-surface two-dimensional SIFT features are distinguished by the distance of their 3D points from the nearest point on the constructed surface. The image pixels are divided into groups of similar colour and textures through mean shift segmentation. A pixel group's inliers and outliers vote towards its overall classification. At points that are considered to be less reliable or off-surface, the prior has a lower weight, reducing the false detection of changes implied by pipes and/or other items.

In [177], overlapping images along the tunnel cross-section are captured by an autonomous system of a camera and polarised lighting moved along a monorail. Panoramas of the surface are built using SfM and neighbouring reconstructed sets are temporally registered via Procrustes alignment [178] on a confident feature correspondence set. The input pair is classified as changed or unchanged using a CNN architecture.

2.4.4 Common metrics for performance evaluation

Change detection performance can be qualitatively and quantitatively evaluated, based on the requirements of the particular application. Furthmore, components of change detection methods can also be individually evaluated.

During qualitative/visual evaluation, the most reliable approach is to superim-

pose the change mask on either image such as using a semi-transparent overlay, with different colours for different change types. Another practice is to display a short video file of a registered pair of images, played in fast successive intervals of around a second. When there is no change, a static image is perceived while flickering appears when changes are present.

Quantitative evaluation can be more challenging, mainly due to the difficulty of obtaining a valid ground-truth. Generally, an expert human observer is the most suitable source for ground-truth data, however, over time the same person may also give different interpretations for the same data. Considering this, an algorithm may need to crate a ground-truth from a number of conflicting observers. An improved approach that does not merely depend on a single observer is to use a majority rule. Once the ground-truth is obtained, there are various standard methods that can be used to compare the ground-truth to a candidate binary change mask.

For quantitative performance analysis, the following metrics are generally calculated:

$$\text{Recall} = \text{True Positive Rate (TPR)} = \frac{TP}{TP + FN} \tag{2.2}$$

$$\text{Precision} = \text{Positive Detection Rate (PDR)} = \frac{TP}{TP + FP} \tag{2.3}$$

$$\text{False Positive Rate (FPR)} = \frac{FP}{FP + TN} \tag{2.4}$$

$$\text{F1-score} = 2 \times \frac{PDR \times TPR}{PDR + TPR} \tag{2.5}$$

where TP is the number of objects/pixels correctly identified as changed, TN is the number of correctly identified no-change objects/pixels, FP is the number of false change detections and FN is the number of misses/false negatives.

2.5 Tunnel surface visualisation

In this regard, visualisation, is a way of organising vast sets of images to build a site-plan of the tunnel lining to improve the inspection process. By applying

vision-based techniques on the generated models, technical condition evaluation can be made remotely, reducing the human presence in the tunnels.

Conducting inspections based on photogrammetric methods, generate large amounts of image data which need to be well organised. Image mosaicing is typically applied to stitch individual photos to build a larger image. A larger surface FoV improves the identification of minor defects including fine cracks, that might be missed in the single image context.

The following works use image mosaicing in tunnel environments. To build tunnel surface panoramic images, a set of line scan cameras was built in [179]. Images are stitched together by extracting and matching feature points based on the colour and textured differences. A modified rail carriage hosting multiple line scan cameras and laser lights was deployed in [35] to acquire images which are later stitched into a mosaic assuming a horse-shoe geometry. Rather than geometry, [180–182] use SfM information to spatially register the images before stitching.

2.5.1 3D reconstruction

A tunnel 3D model gives comprehensive visual and geometric information of its linings that can help surface documentation. Furthermore, it can be used by surveyors to better contextualise the defects located during on-site visits and enables visual validation of such defects with respect to their neighbouring regions. Since it provides ample data to reconstruct the actual tunnel geometry, TLS is the most used approach taken to inspect tunnel surfaces. A commercial solution which takes this approach is [183] and a review of works on laser scanning can be found in [184]. Laser-based 3D models lack image data which can be more beneficial for inspection. In contrast, photogrammetric methods require relatively cheaper and smaller equipment while also generating image and texture data.

Active and passive image sensor fusion was proposed in [185] for high resolution and dense surface reconstruction. In [186], a 3D surface model is generated from overlapping images covering the full tunnel profile. Geo-referencing and alignment

is made using targets signalled by a laser pointer. Later, a transformation between the known 3D Total Station (TS) signalised points and their local 3D coordinates obtained via local bundle adjustment, is used for the global images orientation. Dense stereo matching is made and the disparities are then used to project the generated 3D texture on a surface mesh.

While works such as [182] assume a cylindrical shape, to deal with varying tunnel geometries, the following use SfM techniques. In [187], the 3D geometry of the tunnel is recovered by fitting quadratic surfaces to the generated point cloud locally. A wire-frame 3D surface model is reconstructed and textured by image data. The approach in [188] uses SfM to create a dense point cloud which is used to generate a 3D mesh frame. The same images used by SfM are later used to texture this mesh. In [189], images gathered by a stereo camera pair are used to create high fidelity models of crack areas which are later used to analyse tunnel wall cracks. In [190, 191], 3D scene reconstruction is employed to detect cracks in structures.

2.5.2 Virtual reality

Virtual Reality (VR) makes use of image processing, computer graphics and multimedia technology to build an interactive computer simulation, sensing the movements of a user and subsequently replaces sensory feedback information so the user can experience a sense of immersion in the simulation (virtual environment). As discussed in [192], VR applications include those related to entertainment, design, health-care, engineering and education. VR technology has also been applied within different areas in civil engineering [193], including design, planning, construction progress demonstration and monitoring/inspection. In [194], two VR-based prototype solutions for infrastructure maintenance planning were proposed to aid regular monitoring of both interior/exterior wall maintenance. An inspection and reporting system using 3D modeling techniques, VR and multimedia was developed for tunnels in [195].

3 | Solution overview

This thesis aims to advance the state of the art by contributing to the field of robotics, vision and inspection; proposing a vision-based comprehensive system to aid structural health monitoring and to provide a better means of tunnel surface documentation. A mobile robotic platform is equipped with sensors to acquire data in a tunnel environment. The images are then processed to generate useful inspection information through subsequent image processing and deep learning techniques. Such data include results from pre-processing steps involving radiometric adjustments and specular highlight localisation as well as crack detection and monitoring. Image fusion is employed at different stages to achieve change detection.

3.1 Pipeline

This chapter gives an overview of the proposed solution shown in Fig. 3.1, focusing on the highlighted area. Section 3.2 describes the data acquisition, while crack detection and monitoring are explained in Section 3.3. An overview of specular highlight localisation as a pre-processing step is given in Section 3.4. Change detection is then explained in Section 3.5. The possibility of visualisation using 3D models and VR to aid tunnel surface documentation is mentioned in Section 3.6.

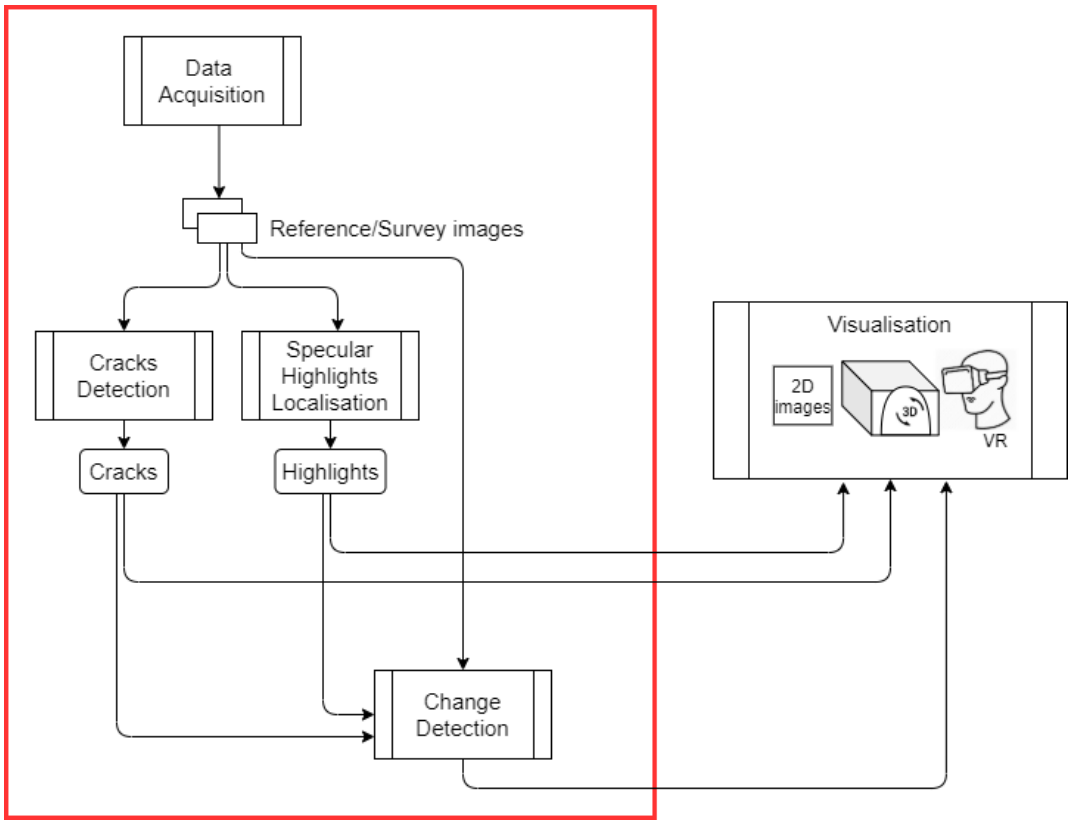


Figure 3.1: Block diagram of the proposed inspection solution

3.2 Data acquisition

The design of the data acquisition system is generally dependent on the constraints present in the particular scenario. These include space limitations, available time and environmental conditions. More important, the choice of the sensors utilised depends on the data that is required; its quality, resolution and accuracy.

After an investigation in the possible sensors that can be used for tunnel inspection, visible light cameras were found to be the best option. Visible light cameras provide the ability of inspection at a distance, in contrast to the close proximity required by other sensors such as ultrasonic ones. Furthermore, visible light cameras are compact and multiple sensors can be hosted in a relatively small space. In general, visible light cameras are less expensive and require trivial training to operate them.

The primary purpose of the developed health monitoring solution is to automate inspection while reducing the on-site human presence, thus automatic image capturing was implemented. Using the cameras' Application Programming Interface (API), images are captured remotely by sending commands from a client application to a server program hosted on the robotic platform.

Further on, during continuous market research on available camera systems used for inspection, a camera system that was primarily designed to inspect cylindrical infrastructures such as tunnels, was identified. A demo test of this system in the LHC tunnel was carried out and the generated images were used to implement and test subsequent modules of the developed solution.

3.3 Crack detection

To mitigate the disadvantages of manual inspection, multiple research works have proposed automatic crack detection methods as a partial replacement of manual inspections. In this research, a crack detection module is fitted within the solution to automate part of the defect detection process. In this solution, state of the art semantic segmentation models and an instance segmentation model are used to compare their effectiveness at detecting cracks in an image.

Rather than merely classifying an image as containing a crack or not, such models also generate the predicted mask for each target which is useful for further processing. While the detection of cracks is essential, monitoring their evolution can be even more beneficial. A temporal comparison of the detected cracks is also proposed to identify new cracks or any changes occurring in existing ones.

3.4 Specular highlight localisation

Due to low lighting conditions, electronic flash units are added to the camera system during image acquisition. These can cause reflections, resulting in specular higlights

in the images. Such highlights are not constant neither in time nor in place, leading to false detections when monitoring for changes. Therefore, a specular highlight localisation module is used as a pre-processing stage to identify these highlights in the reference and survey images to generate binary masks. Such highlight masks are later fused with CMs to mask out these false change candidates.

3.5 Change detection

Timely and accurate monitoring for changes in an infrastructure provides the foundation for better understanding the evolution of structural degradation. In addition to new faults, the evolution of already existing ones including, higher level of corrosion appearing on the wall and an increase in the length, depth or width of a crack or spall are also important to identify. Hence, the change detection module illustrated in Fig. 3.2 is included within the proposed solution.

In order to properly compare temporal images for change detection, accurate image registration is essential. In this scenario, the images are registered using location information from the encoder wheel attached to the robot together with a position reference on the lining of the tunnel that is used as the starting point.

To identify changes between reference and survey images, bi-temporal image fusion is implemented using different Pixel-based Change Detection (PBCD) techniques including image differencing, PCA and Structural Similarity Index (SSIM). Each of these methods generates a binary CM indicating the presence of change in each pixel.

The developed solution combines the benefit of each of these different change detection methods by using decision-level fusion. Merging of the resulting PBCD CMs is investigated through the implementation of various fusion strategies to empirically find the optimal one. Furthermore, specular highlight localisation masks are combined to the final fused CM to minimise false change candidates.

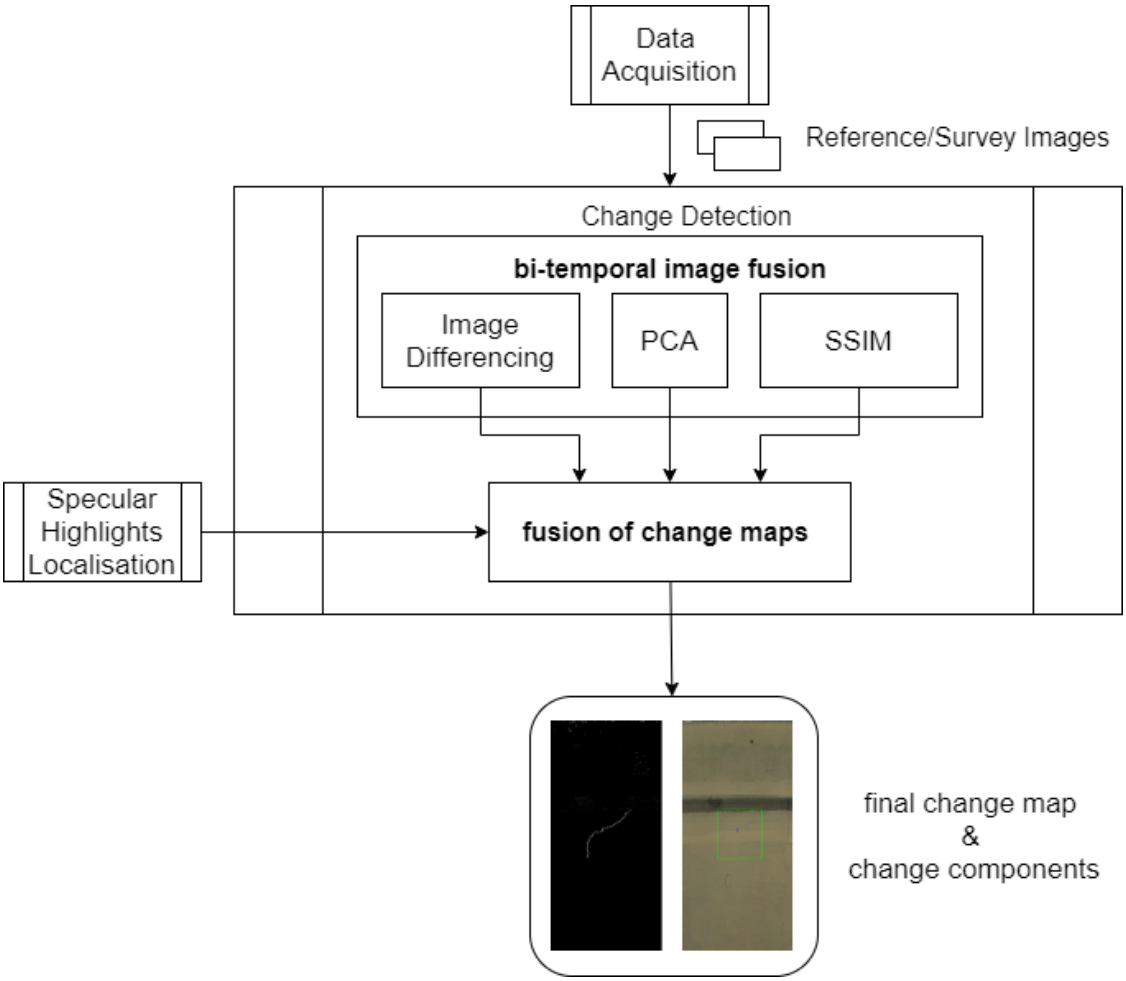


Figure 3.2: Block diagram of the change detection module within the proposed inspection solution

3.6 Visualisation

The research contributions in this thesis concentrate on the part enclosed with a red box in Fig. 3.1, however the use of different visualisation methods as a means to aid structural health documentation was also investigated. Outputs from the crack detection and change detection modules can be overlaid on images captured by the data acquisition module to provide a better means of visualisation of the automatic inspection outcome. Such inspection data can then be directly analysed on the tunnel images, providing a better context for the findings.

Furthermore, by organising image datasets to create a layout plan of the tunnel linings, surface documentation can be done using a 3D model. Using the images captured by the data acquisition module, 3D models can be reconstructed. Moreover, the output of the crack detection module can be used to re-texture the reference 3D models to include also the identified cracks. Such tunnel wall models provide comprehensive visual and geometric images of its environments, aiding inspectors to better contextualise the location of damages found during observations. In addition, 3D information enables visual validation of defects with respect to the areas around them.

In addition to this, with the introduction of VR, easier and more contextualised technical condition evaluation can be conducted offline and analysed further using a VR headset to observe the reconstructed 3D models. A game engine can be used to generate the virtual model and refine it by changing the scale, adding lights and other modifications through a user interface. In turn, the VR headset together with a handheld controller can then be used to view the VR model and navigate through the scene.

4 | Data acquisition

This chapter describes the data acquisition module implemented within the tunnel health monitoring solution. In this scenario, the CERN LHC tunnel is considered. As will be explained in Section 4.1, the latter posed different environment constraints when choosing the acquisition method. Considering these constraints, an investigation into the possible sensors used for infrastructure monitoring was done and the outcome is discussed in Section 4.2. At CERN, there are currently two mobile platforms that can be used for the LHC inspection system: TIM [196] and CERNbot [197] which are mentioned in Section 4.3. The designed camera system is described in Section 4.4, where details of the camera setup, automatic image capturing and the acquired dataset, are given. Section 4.5 presents the operational camera system and the dataset generated during the demo test.

4.1 Environment constraints

The LHC tunnel structure consists of eight straight sections connected by eight arcs. Due to its large scale it imposes a further restraint on the data capturing method to be used, not to be very time consuming. The main tunnel cross-section has an internal diameter of around 3.76 m. As shown in Fig. 4.1, this cross-section is divided into two parts:

- inner side - which is set aside for handling equipment and for the passage of personnel (around 1.4m passageway available for inspection equipment);

- outer side - where the machine components are installed (not available for inspection equipment)

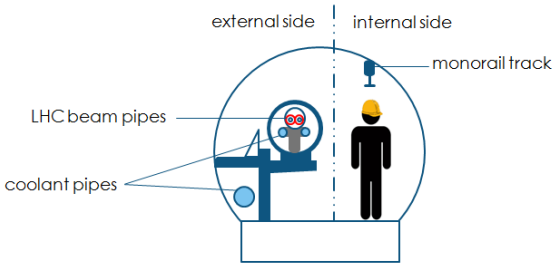


Figure 4.1: Cross-section of the LHC tunnel, the inner side is the passageway available for personnel, robots etc.

As the primary priority, the developed system needs to scan the inner side of the tunnel, which only includes the wall (including service equipment such as extinguishers, pipes, cables etc). Consequently, the equipment to be used for data capturing should fit the width of the inner side.

The LHC tunnel has a non-uniform environment made up of pipes, machinery, cables and dust, amongst other conditions present within its structure. These pose various constraints on the choice of the acquisition setup. In addition to this, the tunnel has a limited amount of ambient light and thus some extra lighting of adequate brightness might be required. Moreover, the amount of light varies from one part of the tunnel to another as well as over time as some parts may not be fully lit.

Furthermore, the time windows available for data acquisition in the LHC tunnel are very limited. Although robots and other equipment can be remotely operated to access the tunnel even when this is not possible to personnel, such availability is still restricted to certain periods when the accelerator beam is turned off. In addition, during data acquisition missions, some personnel intervention may be required, such as setting up the equipment, imposing a further time constraint.

4.2 Image sensors investigation

removed detailed sections on other sensors

An investigation into the possible use of different techniques for inspection such as sonic, ultrasonic, electrical and strength-based, was made. Although these methods can offer valuable and detailed information on the tunnel wall, they should be used in contact or close proximity to the surface. Such an approach is not feasible in a large scale scenario such as the 27 km long LHC tunnel.

Thus, to keep up with time and space constraints, the inspection system must be easy to set up and small in size while the method used to capture the data should be time efficient. Taking these into consideration, an investigation into the different possible sensors to use, was conducted. These include monocular, colour-depth (RGB-D) and TIR cameras. Each sensor was analysed both individually and in combination with others.

Following that, a set of images were captured using a FLIR A300 TIR camera and an Intel RealSense RGB-D camera in order to investigate their usefulness for a change monitoring system. Compared to the DSLR cameras, both the thermal and RGB-D cameras provided further information that was otherwise hidden in the visible spectrum. These include water or moisture presence in thermal images and further information in the third dimension for cracks and other defects in the depth images. Whilst producing supplementary data, the latter sensors usually have a small FoV and images from multiple such sensors are difficult to register and stitch without any human-provided input such as fiducial points or markers. Thermal cameras cannot be used to properly identify cracks that are flat. Moreover, active thermography is usually suggested for inspection tasks and this requires a further heat source such as a powerful lamp making an inspection system larger and with a high power consumption possibly requiring an added power supply. Depth information can be useful to help with the generation of 3D models as well as to give information on the size/shape of known defects however the primary identification of such defects is not possible when their depth is shallow.

Hence, for the primary purpose of crack identification and change monitoring, visual inspection through images captured using multiple visible light cameras was opted for. Thermal and depth cameras can be used for a deeper inspection of specific areas with already known defects.

4.3 Mobile platforms

TIM is a remotely operated modular inspection train moving on an overhead track installed on the LHC tunnel ceiling. For image capturing, a camera can be fixed on a robotic arm, which extends downwards from one of the wagons as shown in Fig. 4.2. In the future, other robotic arms, even customised for an inspection system, can be attached to the TIM. CERNbot is an in-house developed ROV on which different devices, such as sensors and robotic arms, can be placed to conduct different interventions. For image capturing, it is equipped with a metal structure on which multiple sensors can be placed as shown in Fig. 4.3.



Figure 4.2: Camera on the arm extending from one of the wagons of the TIM

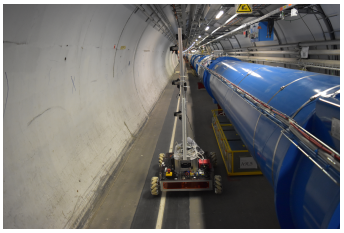


Figure 4.3: Multiple cameras on the CERNbot robotic platform

4.4 Preliminary camera system

Either Digital Single Lens Reflex (DSLR) or mirror-less cameras can be used with both mentioned mobile platforms however the mirror-less type was preferred due to its compactness and light weight. Only a single camera can be installed on the current robotic arm attached to the TIM while multiple ones can be placed on the CERNbot. Using multiple cameras, overlapping images can be captured and stitched, allowing a larger area of the tunnel wall to be observed. Consequently, CERNbot was selected as the mobile platform for the preliminary system.

4.4.1 Camera setup

Three mirror-less cameras facing the tunnel wall were used to gather overlapping images of the inner wall, for a possible larger FoV via image stitching. Each one of them was a Nikon 1 V3 mirror-less camera. Due to the close distance to the wall, a wide angle lens of 6.7-13 mm was used to provide a sufficient FoV.

Using the given camera's sensor dimensions, the lens focal length and the distance from the wall, the image overlap or the spacing between the cameras is estimated. This is done by roughly assuming that the sensor lies at the centre of the camera and that the cameras themselves are approximately aligned with each other. The angle of view (AoV) in an image can be calculated by using the camera sensor dimensions and the distance from the wall:

$$AoV = \alpha = 2 \times \arctan \frac{dim_s}{(2 \times f)} \tag{4.1}$$

where dim_s is the sensor dimension and f is the focal length of the lens. Using the AoV and d, the distance from the wall, the FoV covered by a single image can be approximated by Eq. 4.2, where α is the angle of view.

$$FoV = 2 \times R = 2 \times d \tan \frac{\alpha}{2} \tag{4.2}$$

removed section with screenshot of application

A more robust rig with multiple cameras on a vertical structure with horizontal metal blocks as shown in Fig. 4.4 was built to replace the previous metal structure in Fig. 4.3. These horizontal blocks are adjustable such that the cameras can be placed at varying distances from the wall, possibly forming an arch structure. Furthermore, the cameras themselves are attached to quick release plates which are in turn fixed to adjustable holders such that the sensors' orientation can be adapted to the optimal capturing one as shown in Fig. 4.5.



(a) Robust metal rig (b) Three-Camera setup

Figure 4.4: Robust metal rig with multiple cameras on horizontal metal blocks fixed to a vertical structure on a robust base fixed on the CERNBot



Figure 4.5: Camera attached to a quick release plate

4.4.2 Automatic image capturing

In general, closure of infrastructures to conduct inspection on roads, bridges and tunnels, should be kept at a minimum not to disrupt traffic flow and normal operation. In this scenario, since the access to the LHC tunnel is restricted to the

technical stops, any sensor data can only be acquired during such time windows. Hence, the time to capture the images should be kept at a minimum.

To allow this, images are captured automatically while the robotic platform is moving. This is possible through the software interface developed using the Nikon camera Software Development Kit (SDK) [198]. This camera interface can capture both images and videos and save them to the SD card and/or the host computer according to the previously defined configuration parameters. Each camera has its own thread such that three threads are running simultaneously and once a capture command is sent to the server, each camera takes a photo with only a small latency due to synchronisation being done at the software level. Similarly, when videos are required, start and stop commands sent to the server initiate and terminate the recording on each of the three cameras. Further improvement may involve hardware-level synchronisation.

4.4.3 Dataset

Due to the limited time periods in which the LHC tunnel can be accessed as well as the similarities of the different sections, the datasets generated are limited to certain small sections representative of the rest of the tunnel. Using the setup described in the previous section, two types of datasets were generated; image and video. A summary of the properties for each is given in Table 4.1. A few samples of the images captured by the three mirror-less cameras while the CERNBot is moving can be observed in Fig. 4.6. The use of video enabled the dataset generation to be done faster as it does not require the robotic platform to be moved slowly to avoid photo blurring. Furthermore, video frames for 3D reconstruction can provide more complete models due to a large overlap between the frames.

Table 4.1: Dataset summary, including data type, camera and resolution

Dataset type	**Camera used**	**Resolution**	**FPS**
Images	Nikon 1 V3	2607 x 1744	N/A
Video	Nikon 1 V3	1920 x 1080	59



(a) Camera 1 (b) Camera 2 (c) Camera 3

Figure 4.6: Samples of images captured by the three cameras on the vertical structure placed on the CERNBot

4.5 Operational camera system

Market research on camera systems used for inspection surveys was continuously conducted throughout the project. Considering various systems suggested by different consultants, a demo test with the camera rig [199] that is shown in Fig. 4.7 was requested.

4.5.1 Setup of the system

The system is composed of twelve (5MP) cameras with adapted lenses, two electronic flash units, an encoder wheel, two batteries and a small computer unit with software for camera synchronisation. For the demo test, the camera rig together with the PC unit and batteries were placed on the CERNBot base as shown in Fig. 4.8(a) while an encoder wheel was attached to one side of the CERNBot as shown in Fig. 4.8(b).



Figure 4.7: Camera rig in the provisional commercial camera system



(a) Integrated system



(b) Encoder wheel

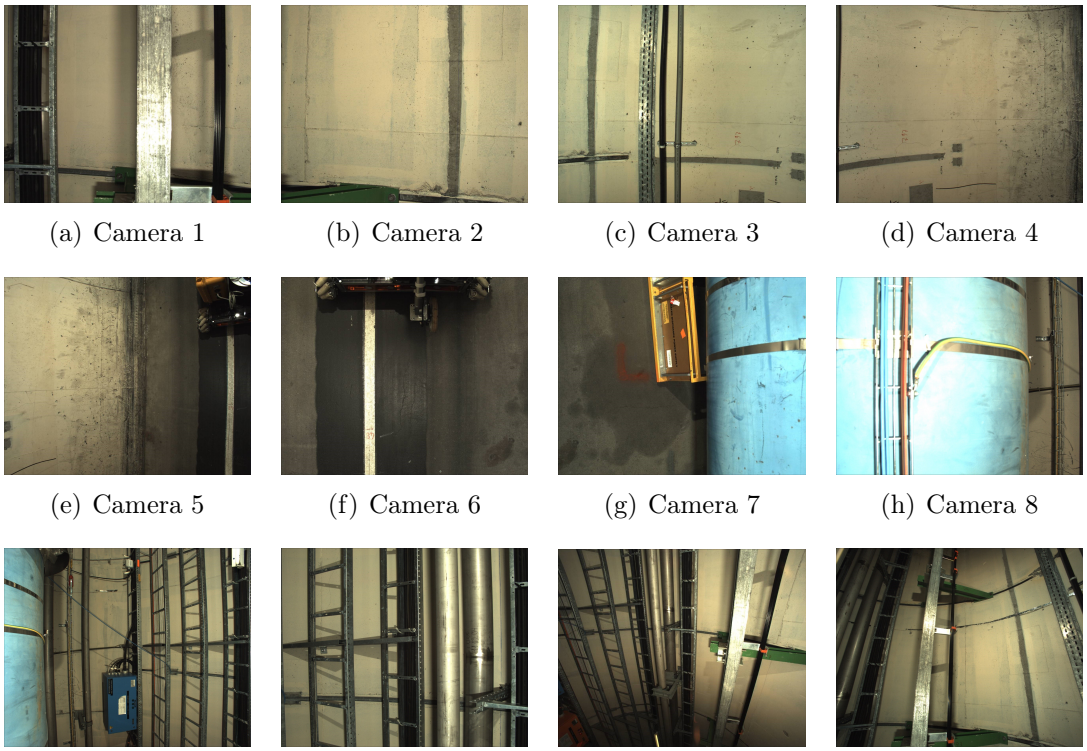
Figure 4.8: Provisional commercial camera system integrated on the CERNBot

4.5.2 Demo test dataset

A system demo was performed over a short section of around 60m of the LHC tunnel. For this demo, the adaptation of the system [199] on the CERN robot was done together with a representative from the external company. The latter then provided, on-site data capture and a set of raw data during the test and after, generated 3D models and orthophotos.

The CERNbot was placed such that the camera head was at around 1.5m from the wall. It was then driven at a speed of around 0.2m/s along a tunnel section in one direction while capturing images from the synchronised camera set. This

image set is referred to as $DataT_1$. Changes were then simulated by marking crack-like defects on the wall and inserting/moving objects in the scene. The CERNbot was again driven for the same distance and at the same speed in the same section capturing the dataset $DataT_2$. A sample set of images from the camera head at a single location is displayed in Fig. 4.9.



(a) Camera 1 (b) Camera 2 (c) Camera 3 (d) Camera 4

(e) Camera 5 (f) Camera 6 (g) Camera 7 (h) Camera 8

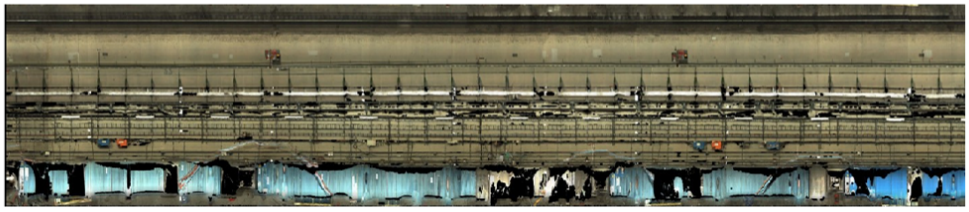
(i) Camera 9 (j) Camera 10 (k) Camera 11 (l) Camera 12

Figure 4.9: A sample set of images captured using the provisional commercial camera system during the demo test in the LHC tunnel

The 3D models generated using the synchronised camera images captured during the demo test were then unwrapped into orthophotos. Using the location information from the encoder wheel, orthophotos could be easily registered with high precision as shown in Fig. 4.10. These generated orthophotos were segmented into smaller images. Each orthophoto was segmented in ten parts along its height and each of the image crops covers 0.5m of the tunnel length. Such images were used for training and testing of the models and algorithms of the developed system. In the next chapters, this dataset is referred to as the LHC Dataset.



(a) Orthophoto from $DataT_1$



(b) Orthophoto from $DataT_2$

Figure 4.10: Orthophotos generated from $DataT_1$ and $DataT_2$ captured during the demo test

5 | Crack detection and monitoring

To improve on manual surveys, various works have used a number of image processing techniques to automate crack detection. Whilst they are robust in various scenarios, these methods use shallow representations and conditions that may not overcome the intrinsic challenges related to crack images. These include difficult crack topology, surface texture variation, crack inhomogeneity, background complexity and resemblance of objects of similar shape/texture to cracks such as joints. To cater for these, deep learning methods have been recently used, allowing better abstractions and generalisation without extracting fixed features. Using CNNs, dramatic advances in cutting-edge solutions for fundamental tasks such as object detection, were made. Here, the semantic segmentation models of SegNet [57], U-Net [51] and Mask R-CNN [200] as an instance segmentation model, are used to compare their effectiveness at detecting cracks in an image. Using these models, a mask is predicted for the crack targets, which is useful for further analysis.

The rest of this chapter is structured as follows. Crack detection using semantic segmentation is explained in Section 5.1, where an introduction on the U-Net and SegNet models is given and the applied methodology is described. In Section 5.2, crack detection using instance segmentation through the Mask R-CNN model is discussed, giving related background information and details of the methodology used. The datasets utilised for training and testing of the models are presented in Section 5.3. A comparative analysis of the models used for crack detection is made in Section 5.4, where quantitative and qualitative results are discussed.

5.1 Semantic segmentation method

Semantic segmentation involves the classification of each image pixel as part of a particular object class in order to understand the image at pixel level. In this chapter, semantic segmentation is used to segment images of walls in order to detect cracks using the U-Net and SegNet models. Below, is an explanation of the two models, including background information on their architectures and the methodology applied.

5.1.1 U-Net model

The U-Net model proposed in [51] also has an encoder-decoder architecture. In a classical autoencoder architecture, the size of the input information is initially reduced, along with the following layers. Later, linear feature representation is learned in the decoder section, and the size gradually increases. At the end of the architecture, the output size is equal to the input size. While this architecture is ideal to preserve the output size, it compresses the input linearly, resulting in a bottleneck in which all features cannot be transmitted. In contrast, U-Net performs deconvolution on the decoder side (i.e. in the second half) and can overcome this bottleneck problem, which results in the loss of features through connections from the encoder side of the architecture.

U-Net consists of multiple convolutional layers arranged in a top-down and bottom-up manner in different paths creating a U-shaped network.

Encoder The first path is referred to as the encoder or contracting path. It is made up of convolutional and max-pooling layers and is used to extract features while capturing the context in an image. As shown in Fig. 5.1, U-Net's pipeline involves the recurrent application of two 3×3 unpadded convolutions. Every convolution is succeeded by a ReLU and a 2×2 max-pooling operation using a stride of 2 for downsampling.

Decoder The second part of the U-Net network is referred to as the decoder or expansion path. It uses transposed convolutions to enable precise localisation. In this path, at each step, the feature map is upsampled and then a 2×2 convolution is applied, reducing the number of channels by a factor of two. After, a concatenation of the generated feature maps with the corresponding ones from the contracting path is made.

Next, two successive 3×3 convolutional layers each followed by a ReLU are applied. To map each feature vector to the specific classes, a 1×1 convolution is used at the final layer. In total, the model has four levels in each of its two paths with a bridge connection in between.

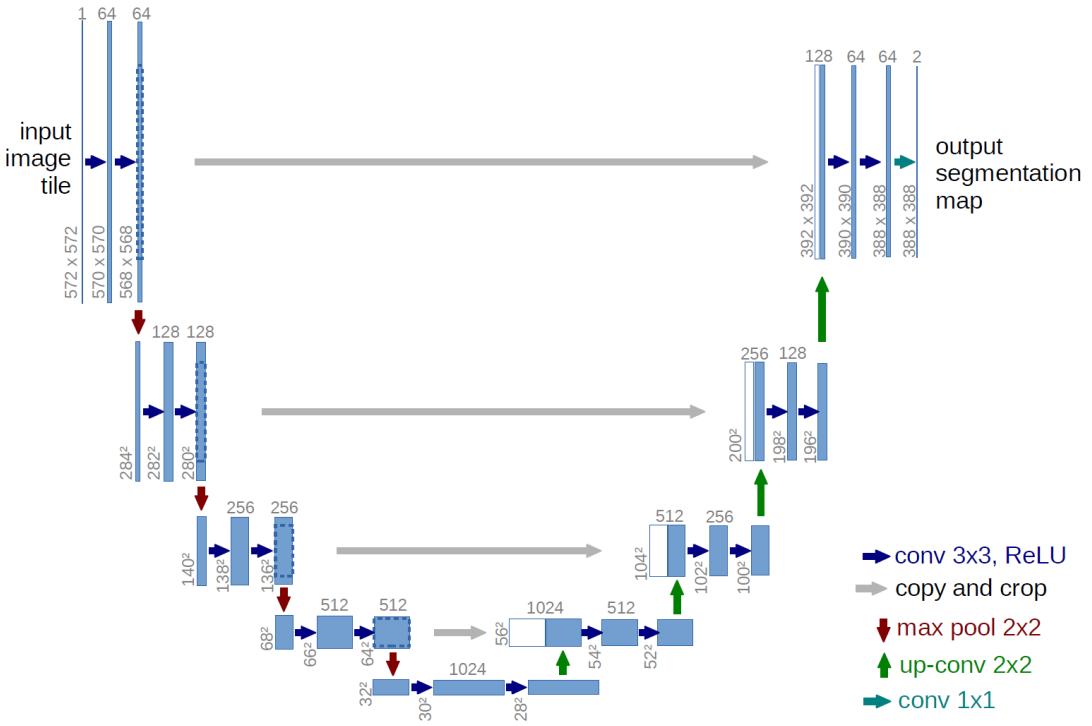


Figure 5.1: U-Net architecture pipeline from [51]

5.1.2 SegNet model

The architecture consists of a sequence of non-linear processing layers (encoders) and a corresponding set of decoders followed by a pixelwise classification layer as

illustrated in Fig. 5.2. The encoder network consists of 13 convolutional layers (VGG16 network's first 13 convolutional layers). In SegNet, the fully connected layers are removed to keep higher resolution feature maps at the deepest encoder layer and at the same time reduce the number of parameters. For every encoder layer, there is a decoder layer. At the end, class probabilities for each pixel are generated by a multi-class soft-max classifier.

Encoder Each block in the encoder network has a filter bank which generates feature maps using convolutions. Batch Normalisation (BN) and an element-wise ReLU are then applied consecutively. A max-pooling operation with a 2×2 non-overlapping window and a stride of 2 is performed and the result is sub-sampled by a factor of 2. Translation invariance over minor spatial shifts in the input image is achieved through max-pooling. Sub-sampling allows a larger spatial context for every feature map pixel. Using multiple layers of max-pooling and sub-sampling has the drawback of a loss in the spatial resolution of the feature maps. To cater for this, SegNet captures and stores the encoder feature maps' boundary information before applying sub-sampling, using max-pooling indices. For every encoder feature map, the points of the highest feature value in each pooling window, are kept. This has the important advantages of retaining high frequency details in the segmented images and also reducing the total number of trainable parameters in the decoders. Furthermore, due to this, SegNet requires less memory than U-Net which instead of using pooling indices, it transfers entire feature maps from the encoder to the decoder.

Decoder For each encoder block, the corresponding decoder block upsamples its input feature maps using the recorded max-pooling indices, producing sparse feature maps. Following this, a decoder filter bank produces dense feature maps on which BN is then applied. The high dimensional feature representation at the output of the final decoder is fed to a trainable soft-max classifier which classifies each pixel independently. Its output is a K-channel image of probabilities for

K classes. The predicted segmentation corresponds to the class with maximum probability at each pixel.

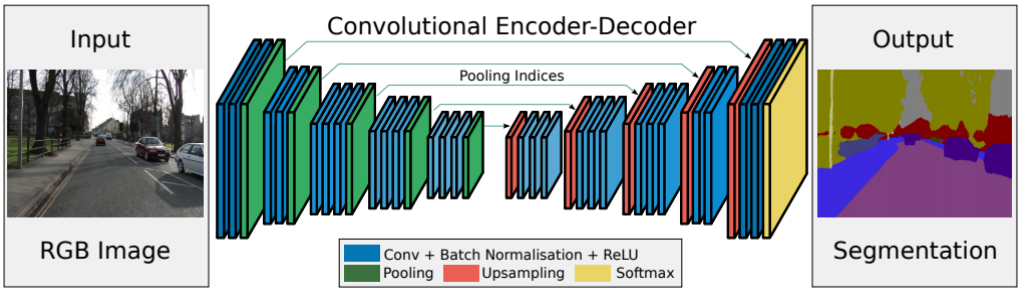


Figure 5.2: SegNet architecture pipeline from [57]

5.1.3 Methodology

To train both the SegNet and U-Net models for crack detection, the Keras deep learning framework was utilised. The code in [201] was used as a basis and then, various modifications were made to adapt the implementation to our scenario.

Although the SegNet and U-Net model architectures are different, the same common training pipeline is used. First, the training and validation datasets are verified, checking that each image has its corresponding mask image. Every image-mask pair is pre-processed to the expected format at the input. The model is then created followed by training and validation on the respective subsets.

5.1.3.1 Pre-processing

The input image is first resized to the dimensions required by the network at its input. Such dimensions are configurable and set empirically. For faster convergence during training, the image mean is subtracted across every individual pixel in the image. Such a pre-processing step has the geometric interpretation of centering the cloud of data around the origin along every dimension. Since image pixel values are all within the 0-255 range, normalisation is implicit. The sample mean computed

on a large training set of the ImageNet dataset [202] is used and the values of 123.68, 116.779 and 103.939 are subtracted from the R, G and B channels respectively.

5.1.3.2 Encoder architectures

Both SegNet and U-Net can be used in their original architecture format or with other known architectures for the encoder part. In this work, Vanilla CNN, VGG16 [203] and ResNet-50 [204] based encoders are used and a comparative analysis of the trained models is made at the end of this chapter.

5.1.3.3 Data augmentation

To learn the desired invariant features and have robustness properties, a deep learning model requires training on a large amount of data. When only a few samples are available, data augmentation is applied to expand the variability of the available data. Here, smooth deformations of the existing image samples are generated through vertical and horizontal flips, vertical and horizontal displacements in the range [-20% , 20%] and rotations in the range [-45°, 45°].

5.2 Instance segmentation method

Semantic segmentation can locate objects in an image, separate them from the background and cluster them based on their class. Instance segmentation further detects each individual item within a group of objects, identifying the boundaries for each of them. The Mask R-CNN model is an example of this.

5.2.1 Mask R-CNN model

The initial deep learning object detection methods were based on the Region-based CNN (R-CNN) approach [205]. After, Fast R-CNN [206] was proposed to extend R-CNN addressing multiple Region of Interest (ROI)s on feature maps using RoI-Pooling. After that, Faster R-CNN [207] replaced the slow selective search

algorithm by a Region Proposal Network (RPN). For pixel level segmentation, Mask R-CNN [200] was later introduced.

Mask R-CNN is a two stage framework. The first stage scans the image and generates proposals of areas that are likely to contain an object. The second stage classifies the proposals and generates bounding boxes and masks. This part is referred to as the decoder or expansion path. It uses transposed convolutions to enable precise localisation.

Backbone The backbone architecture of Mask R-CNN identifies low level features by the early layers while the later layers successively detect features at an upper level. Mask R-CNN improves on this base architecture by using a Feature Pyramid Network (FPN). High level features are propagated to lower layers, such that features at each level have access to both their upper and lower level features. A ResNet [208] architecture with a FPN backbone is used in this Mask R-CNN implementation.

RPN First a RPN moves a sliding window over the backbone feature maps using anchors distributed over the image to identify whether or not there is an object, per location per anchor box. These anchors are boxes distributed over the image area. Generally, there are about 200K anchors of different sizes and aspect ratios and they overlap to cover as much of the image as possible. The RPN scans over the backbone feature map rather than over the raw image directly, thus reuses the extracted features efficiently and avoidS duplicate calculations.

The RPN generates two outputs for each anchor. The class specifies the foreground or background implying the possible presence or not of an object. A foreground (positive) anchor might not be centred perfectly over the object, so the RPN further estimates a delta (change in x, y, width, height) to refine the anchor box to fit the object better. Using the RPN predictions, the top anchors that are likely to contain objects are selected and their location and size are refined.

ROI classifier and bounding box regressor This stage runs on the regions of interest (ROIs) proposed by the RPN. Similar to the previous stage, it generates two outputs for each region of interest. This network is deeper and has the capacity to classify regions to specific classes, so the class output in this case gives a particular class rather than distinguising between foreground and background. The purpose of bounding box refinement, is to further refine the location and size of the bounding box to encapsulate the object.

ROI-Align Classifiers typically require a fixed input size, however, due to the bounding box refinement step in the RPN, the ROI boxes can have different sizes. To solve this issue, ROI-pooling is used to crop a part of a feature map and resize it to a fixed size. To improve on the segmentation accuracy, Mask R-CNN uses ROI-Align which samples the feature map at different points and apply a bilinear interpolation. Unlike the ROI-pooling, ROI-Align does not adjust the input proposal from RPN to fit the feature map. Instead, it splits the object proposal into a specific number of bins. Multiple points are sampled from every bin and their values are determined via bilinear interpolation.

Segmentation masks The final part of Mask R-CNN involves a mask branch. This is a convolutional network that takes the positive regions selected by the ROI classifier and generates masks for them.

When compared to Faster R-CNN, Mask R-CNN has only a minor overhead. It has been employed to detect varying classes such as cars, animals, pedestrians, traffic signs, buildings and nucleus segmentation in medical imaging. Here, Mask R-CNN is used to locate cracks in concrete. The model's pipeline is illustrated in Fig. 5.3.

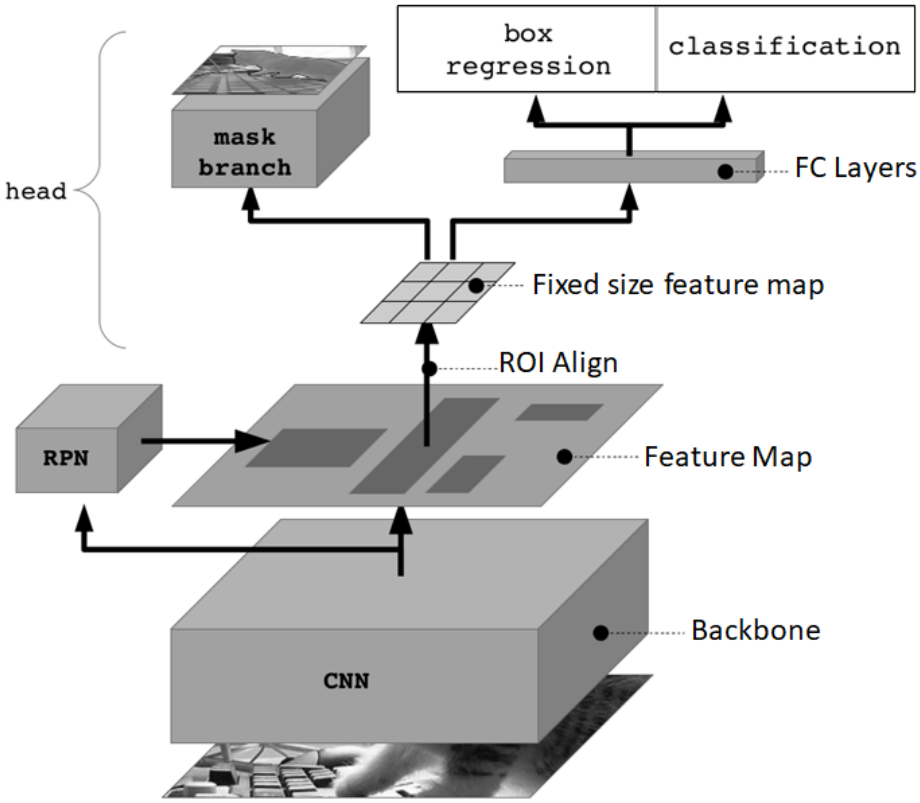


Figure 5.3: Mask R-CNN pipeline from [209]

5.2.2 Methodology

The Mask R-CNN implementation in this system is based on that released by Matterport [210]. It uses the Keras and Tensorflow libraries.

5.2.2.1 Transfer learning

Since only relatively small datasets were available, instead of training the network from scratch, a transfer learning methodology was used. The model is initialised with weights pre-trained on the COCO [211] and ImageNet [202] datasets. Modifying several hyperparameters such as learning momentum, learning rate and train ROIs per image, fine-tuned the network to adapt it to the crack data.

Table 5.1: Different augmentation pipelines

Pipeline	Functions
1	vertical, horizontal flips
2	vertical, horizontal flips rotation, blur, brightness
3	vertical, horizontal flips rotation, blur, brightness contrast normalisation, crop

5.2.2.2 Data augmentation

Additionally, to improve on the lack of training data, an augmentation pipeline was used to train the Mask R-CNN model. Experimentation with several transformations for augmentation included different rotations, vertical and horizontal flips, blurring through a Gaussian kernel and changes in the brightness. To examine the benefits of using data augmentation, different pipelines were assembled using a number of functions from the *imgaug* library [212] and tested by training with the respective pipelines. A short account of each pipeline is given in Table 5.1.

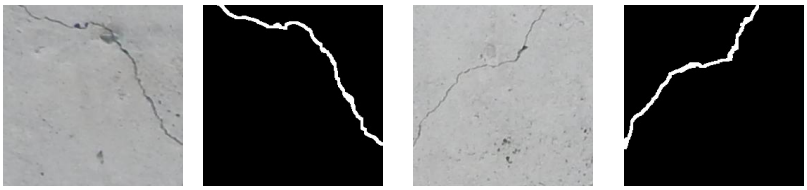
5.3 Crack datasets

The U-Net, SegNet and Mask R-CNN models were trained using crack images from two datasets. One was built from a subset of the SDNET dataset [213] and the other from images captured in the LHC tunnel as described in Section 4.5.2.

5.3.1 SDNET subset

The SDNET dataset is a benchmark image set for Artificial Intelligence (AI) crack detection algorithms. It provides only the crack vs non-crack classification, rather than Ground-Truth (GT) masks as required by U-Net, SegNet and Mask R-CNN networks. Thus, a mask dataset was built using 200 256×256 images from the whole SDNET set. Using the PixelAnnotationTool the crack masks were generated.

Examples from the built dataset are displayed in Fig. 5.4. The 80/20 rule was used to split the samples in 128 for training, 32 for validation and 40 for testing.

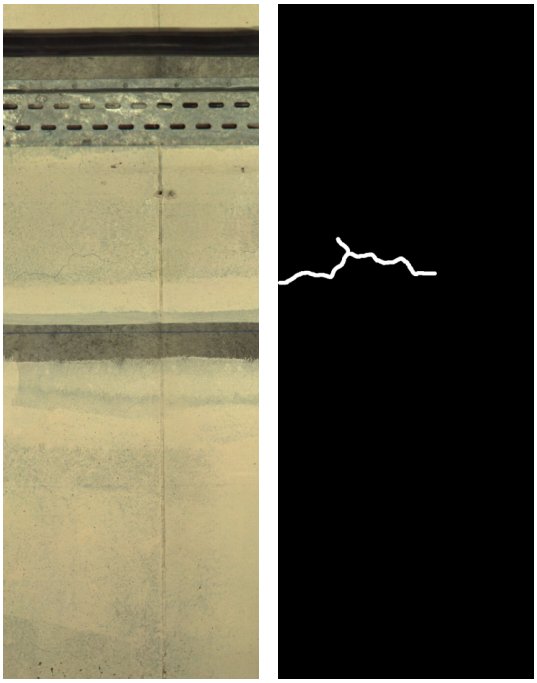


(a) crack image (b) GT of (a) (c) crack image (d) GT of (c)

Figure 5.4: A sample of images and their corresponding GT crack markings from the annotated crack dataset built using a subset of the SDNET dataset

5.3.2 LHC dataset

A subset of images showing the wall areas, was chosen from the dataset mentioned in Section 4.5.2. Then, the cracks in each of these images, were manually marked using the same tool [214] to generate the mask annotations. The images in the generated mask dataset have a resolution of 1885×711, two samples of which are displayed in Fig. 5.5. The 80/20 rule was used to split the samples into 110 for training, 28 for validation and 34 for testing.



(a) crack image (b) GT of (a)



(c) crack image (d) GT of (c)

Figure 5.5: A sample of images and their corresponding GT crack markings from the annotated crack dataset built using the LHC dataset

5.4 Comparative analysis

Experiments using different configurations of the three models were conducted to define the optimal one by analysing the resulting values of different evaluation metrics. In class imbalanced scenarios, pixel accuracy can easily give a false impression of good performance.

Thus, more reliable metrics, namely the training and validation loss and Intersection over Union (IoU), were used. By monitoring the loss, different configurations could be analysed to empirically find the optimal one, avoiding underfitting or overfitting issues. The IoU measures how good the segmentation prediction (SP) matches the corresponding GT annotation by dividing their intersection by their union:

replaced illustration by equation

$$IoU = \frac{intersection}{union} = \frac{GT \cap SP}{GT \cup SP} \tag{5.1}$$

5.4.1 Quantitative results

Crack images from both the SDNET subset and the LHC dataset were used to train the U-Net, SegNet and Mask R-CNN models using different configurations and hyperparameters. To evaluate their individual and relative performance, the loss and resulting IoU of each model were analysed as discussed below.

5.4.1.1 SDNET

When considering the U-Net and SegNet models, each of them was trained for 200 epochs using the SDNET subset, however the models' loss levelled off even before 100 epochs as can be observed in Fig. 5.6. When comparing these curves with the validation ones displayed in Fig. 5.7, one can observe that the latter are not as consistent. In general U-Net performed better and had a more consistent decaying loss when using it with Vanilla and VGG16 encoder architectures. During training, the IoU value was also monitored. This had a fairly consistently increasing

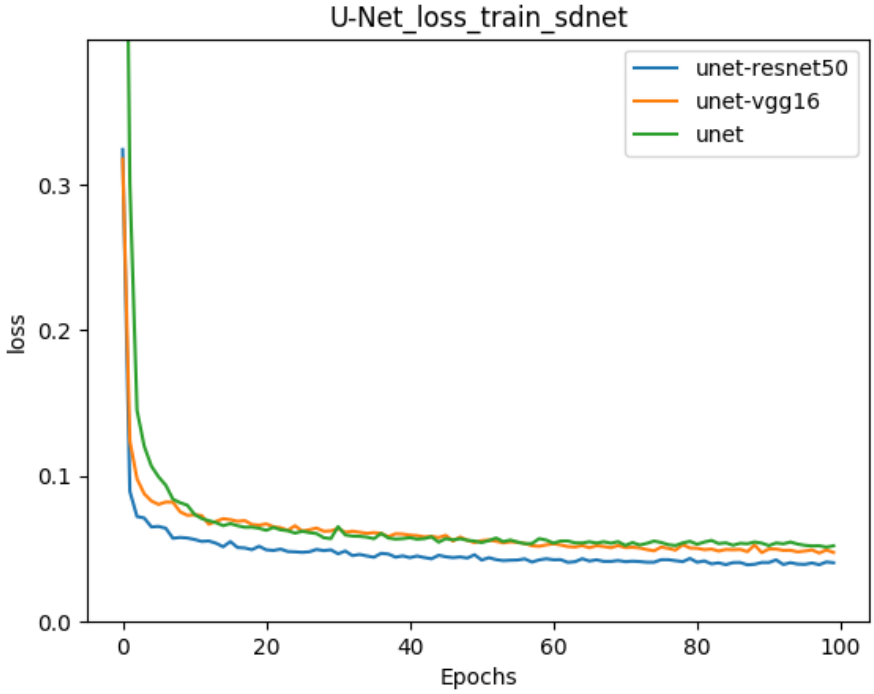
behaviour for any of the trained models, however during validation, the U-Net model performed better implying improved generalisation. Furthermore, testing the models on the testing dataset confirmed that for the SDNET subset, the U-Net model with a VGG-based encoder had the best segmentation performance with the highest mean IoU value of 0.73 as recorded in Table 5.3.

The Mask R-CNN model was initialised with weights pre-trained on the ImageNet and COCO datasets for the ResNet-50 and RestNet-101 backbones respectively. Upon training the model with these two backbones and using different hyperparameters, it was noted that ResNet-101 pre-trained on the COCO dataset performed marginally better. During training of the Mask R-CNN model with a ResNet-101 backbone, the classification and mask losses were monitored to identify the number of epochs at which the model had a high probability of giving the best performance. The class loss is the RPN anchor classifier loss and it reflects the confidence at which the model predicts the class labels. The mask loss is the output of a cross entropy loss function applied to the mask branch of the network and it penalises wrong per-pixel binary classifications.

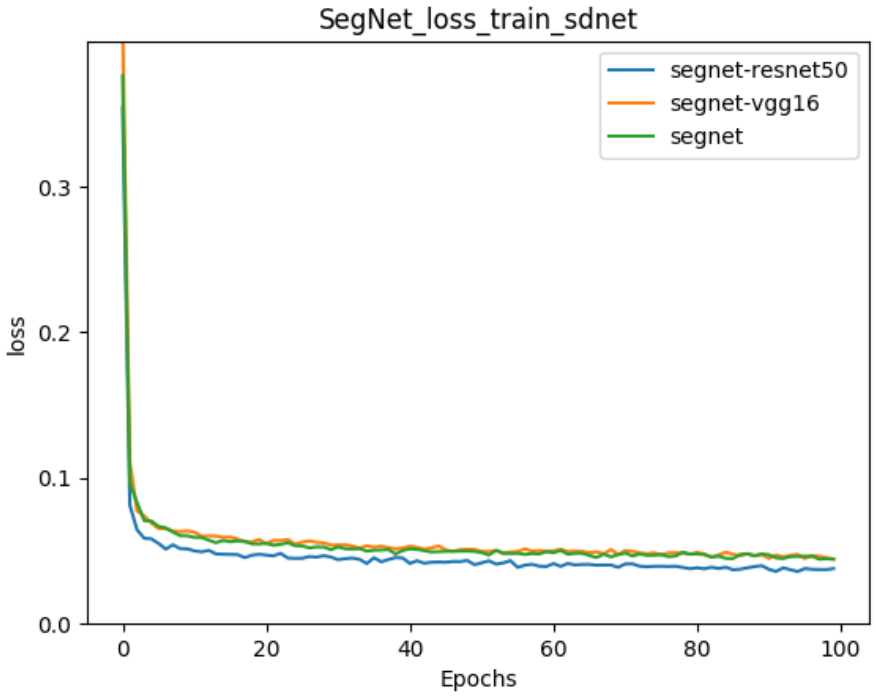
Different training schedules were used to train the model, including training solely the heads of the network, training all the layers of the network and a combination of both, with the latter outperforming the others. The plots shown in Fig. 5.8, show the losses when the heads were trained for 50 epochs followed by training all the layers for another 250 epochs using a permanent learning rate of 0.001. As observed here, training further than 200 epochs did not do any major improvements to the network. To confirm this, predictions on the testing subset were done with the trained model at 200, 250 and 300 epochs, obtaining the highest IoU value at 200 epochs as observed in Table 5.2.

Table 5.2: Mean IoU from the Mask R-CNN model trained for different number of epochs on the SDNET subset

Number of Epochs	**Mean IoU**
200	**0.68**
250	0.66
300	0.67

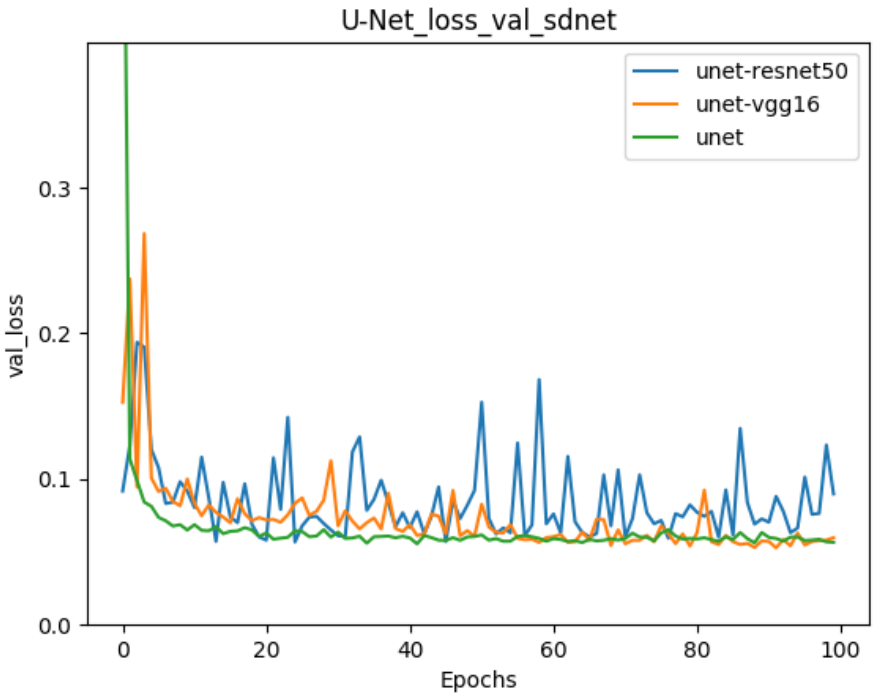


(a) U-Net

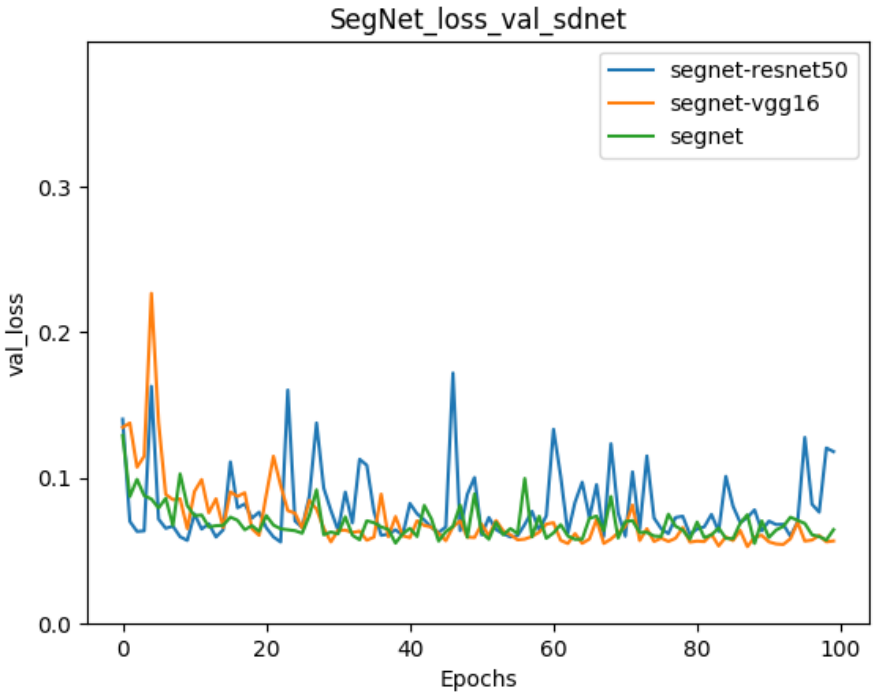


(b) SegNet

Figure 5.6: A plot of the cross entropy loss during training of U-Net and SegNet models with different encoder architectures, on the SDNET subset



(a) U-Net



(b) SegNet

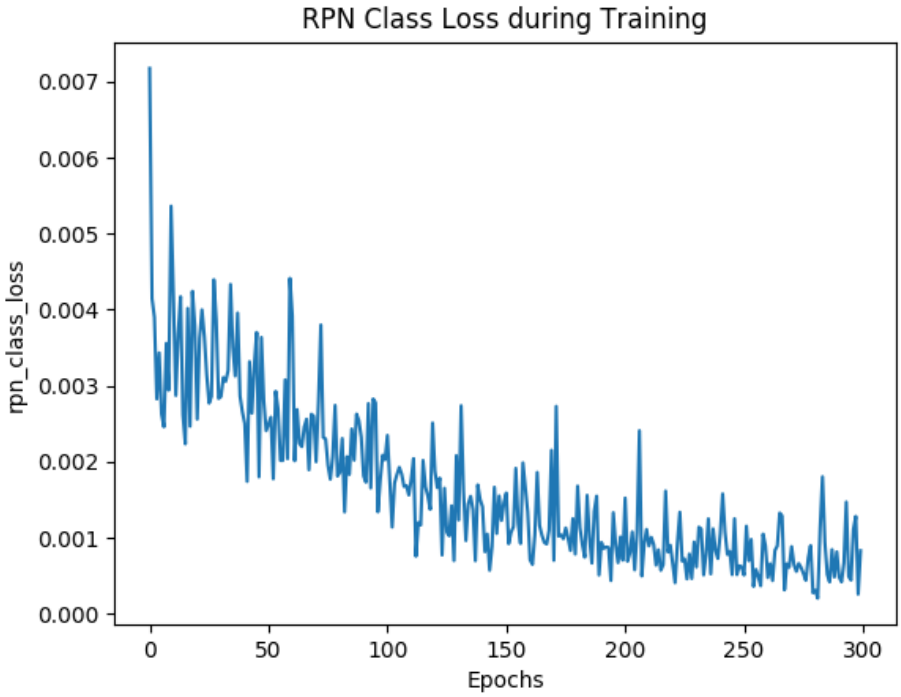
Figure 5.7: A plot of the cross entropy loss during validation of U-Net and SegNet models with different encoder architectures, on the SDNET subset

Experiments of training with the different augmentation pipelines listed in Table 5.1 were done, and the optimal results were obtained when using Pipeline 3. Hence, using the Mask R-CNN model with a ResNet-101 backbone, trained for 200 epochs with a fixed learning rate and a data augmentation pipeline involving horizontal and vertical flipping, rotation, brightness, blur, contrast normalisation and cropping resulted in the optimal configuration to generate the highest mean IoU value of 0.68.

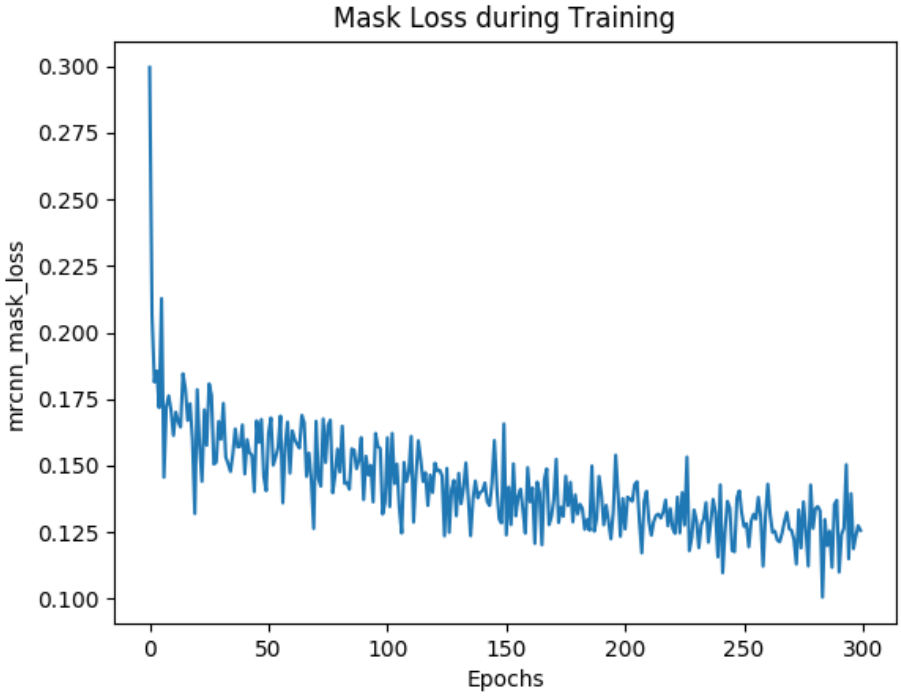
When comparing all the trained networks, Table 5.3 shows that, for the SDNET subset Dataset, the U-Net with a VGG16-based encoder generated the highest mean IoU with a value of 0.73.

Table 5.3: IoU from the different models trained on the SDNET subset

Model	**Mean IoU**
U-Net	0.70
U-Net with VGG16	**0.73**
U-Net with ResNet-50	0.56
SegNet	0.54
SegNet with VGG16	0.68
SegNet with ResNet-50	0.53
Mask R-CNN with ResNet-101	0.68



(a) Class Loss



(b) Mask Loss

Figure 5.8: The plots of the class loss and mask loss while training Mask R-CNN on the SDNET subset

5.4.1.2 LHC

Similar to the training procedure for the SDNET dataset, each of the U-Net and SegNet models was trained for 200 epochs however, the models' loss levelled off even before 100 epochs as can be observed in Fig. 5.9. Also, when comparing the training curves with the validation ones displayed in Fig. 5.10, the latter were not as consistent. One can observe that the latter had lower values for the U-Net model.

Similar to the previous dataset, when monitoring the IoU, a fairly consistently increasing behaviour for any of the trained models was observed. In contrast, during validation, the U-Net model performed better implying better generalisation. Furthermore, testing the models on the testing subset confirmed that for the LHC dataset, the U-Net model with a ResNet-based encoder had the best segmentation performance with the highest mean IoU value of 0.72 as recorded in Table 5.5.

Again, for this dataset, the Mask R-CNN model was also initialised with the ImageNet and COCO pre-trained weights for the ResNet-50 and RestNet-101 backbones respectively. Upon training the model using different configurations, the one with a ResNet-101 backbone pre-trained on the COCO dataset performed marginally better overall.

Hence, the Mask R-CNN model with a ResNet-101 backbone architecture was trained with different hyperparameters, and the classification loss and the mask loss were monitored in order to identify the number of epochs at which the model had a high probability of giving the best performance.

To train this dataset, the same training schedules utilised with the SDNET subset, were adopted. The plots displayed in Fig. 5.11, show the losses when the heads of the network were trained for 50 epochs followed by training all the layers for another 250 epochs using a fixed learning rate of 0.001. As noted here, training further than 200 epochs did not result in any major improvements to the network. To confirm this, predictions on the testing subset were done with the trained model at 200, 225 and 250 epochs, obtaining the highest IoU value at 200

epochs as observed in Table 5.4.

Furthermore, training with the different augmentation pipelines listed in Table 5.1 was made. The optimal results were attained using only horizontal and vertical flipping. Hence, using the Mask R-CNN model with a ResNet-101 backbone, trained for 200 epochs with a fixed learning rate and a data augmentation pipeline involving flipping resulted in the optimal configuration to generate the highest mean IoU with a value of 0.57.

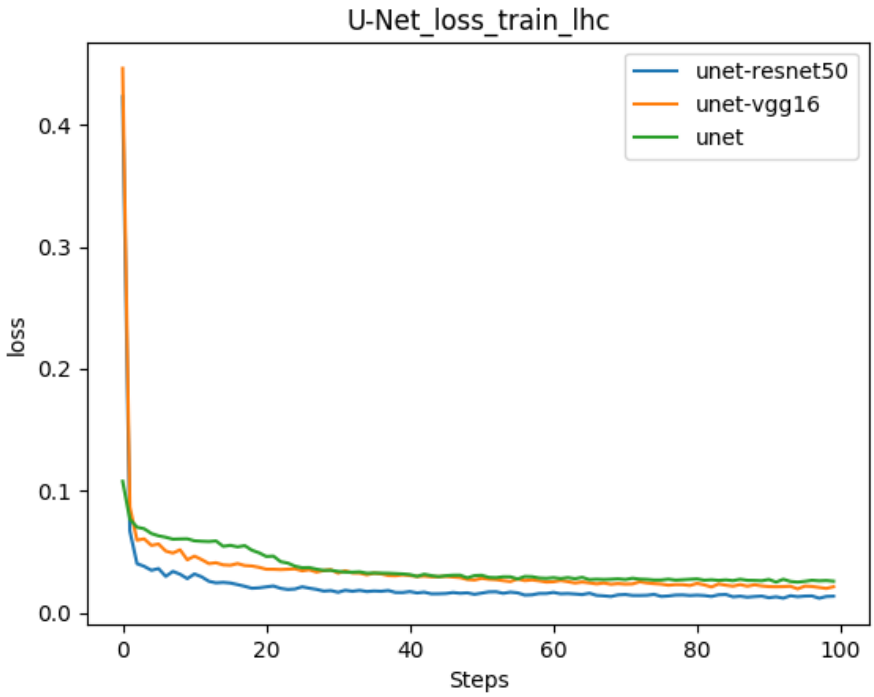
Table 5.4: Mean IoU from the Mask R-CNN model trained for different number of epochs on the LHC dataset

Number of Epochs	**Mean IoU**
200	**0.57**
225	0.55
250	0.54

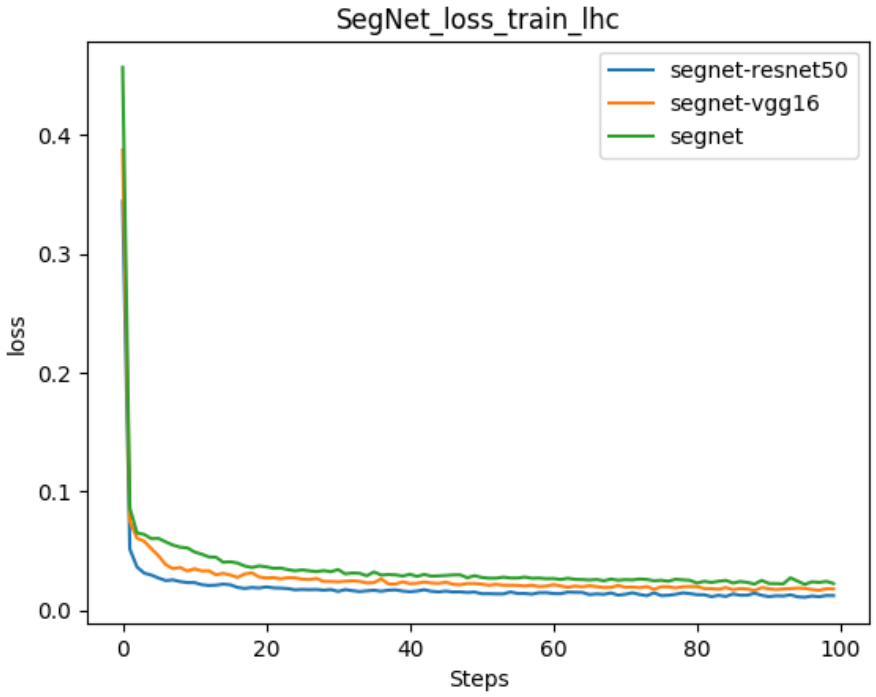
Table 5.5: IoU from the U-Net, SegNet and Mask R-CNN models trained on the dataset built from images in the LHC Tunnel

Model	**Mean IoU**
U-Net	0.70
U-Net with VGG16	0.61
U-Net with ResNet-50	**0.72**
SegNet	0.63
SegNet with VGG16	0.61
SegNet with ResNet-50	**0.72**
Mask R-CNN with ResNet-101	0.57

When comparing all the trained networks, Table 5.5 shows that, for the LHC dataset, both the U-Net and SegNet with a ResNet-50-based encoder generated the highest Mean IoU with a value of 0.72.

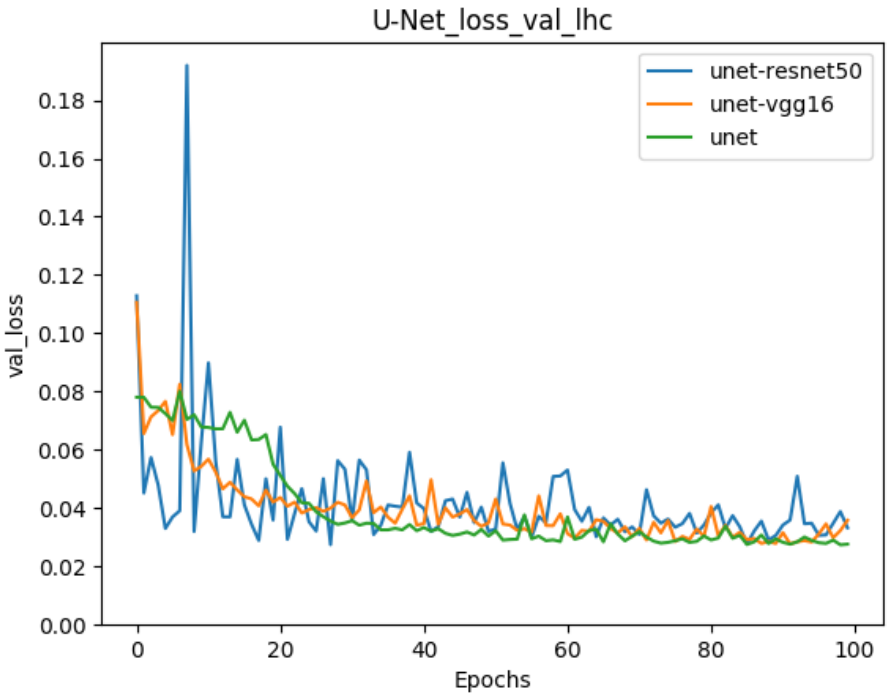


(a) U-Net

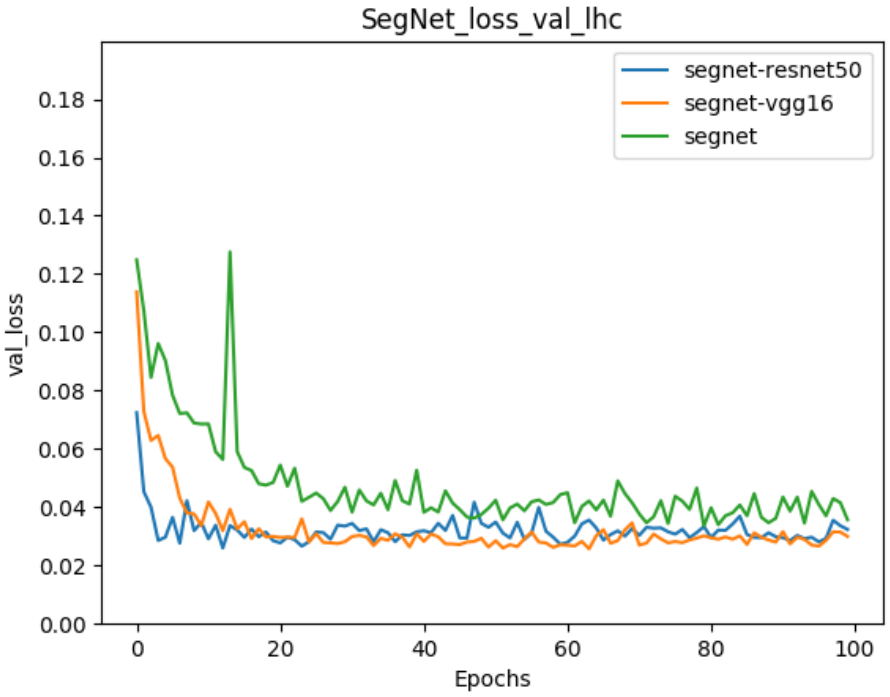


(b) SegNet

Figure 5.9: A plot of the cross entropy loss during training of U-Net and SegNet models with different encoder architectures, on the LHC dataset

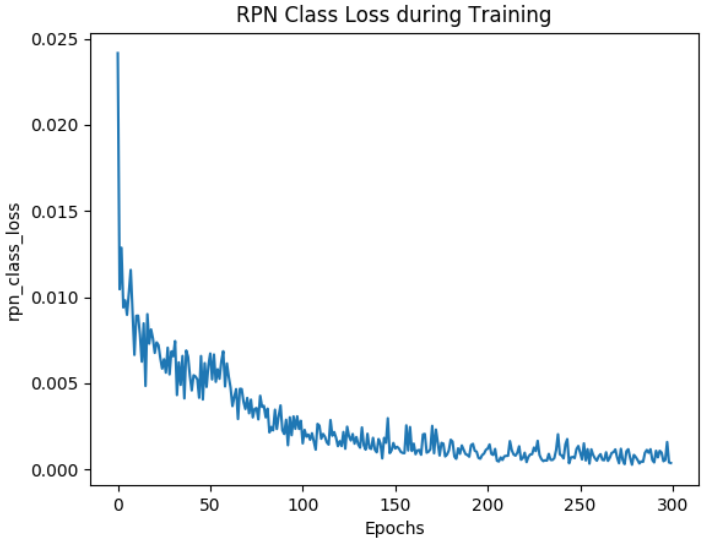


(a) U-Net

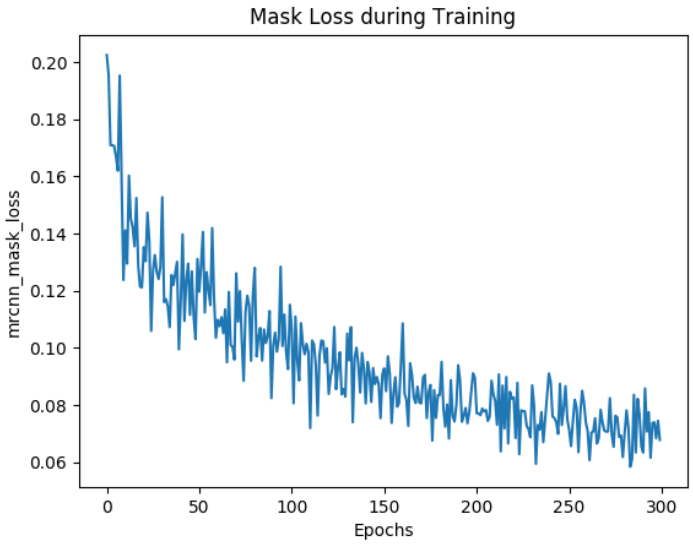


(b) SegNet

Figure 5.10: A plot of the cross entropy loss during validation of U-Net and SegNet models with different encoder architectures, on the LHC dataset



(a) Class Loss



(b) Mask Loss

Figure 5.11: The plots of the class loss and mask loss while training Mask R-CNN on the LHC dataset

5.4.2 Qualitative results

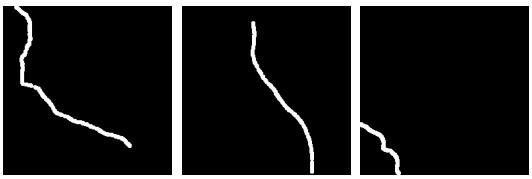
A further qualitative interpretation of the results from training the different networks on both datasets was done. In this section, a sample of these is presented, while further examples can be referred to in Appendix B.

5.4.2.1 SDNET

From the quantitative results, the semantic segmentation method using the U-Net model with a VGG16 encoder network resulted in the highest IoU. Moreover, when comparing the sample results in Fig. 5.12 and Fig. 5.13, the U-Net model's performance was in general better, with the U-Net model with a VGG16 based encoder showing the segmentation results closest to the corresponding GT mask. The trained Mask R-CNN model also generated segmentation maps very close to the GT, for the same sample images as shown in Fig. 5.14. However, drawbacks of the Mask R-CNN include a larger architecture and longer training time.



(a) Crack images



(b) GT masks of (a)



(c) U-Net with Vanilla CNN encoder

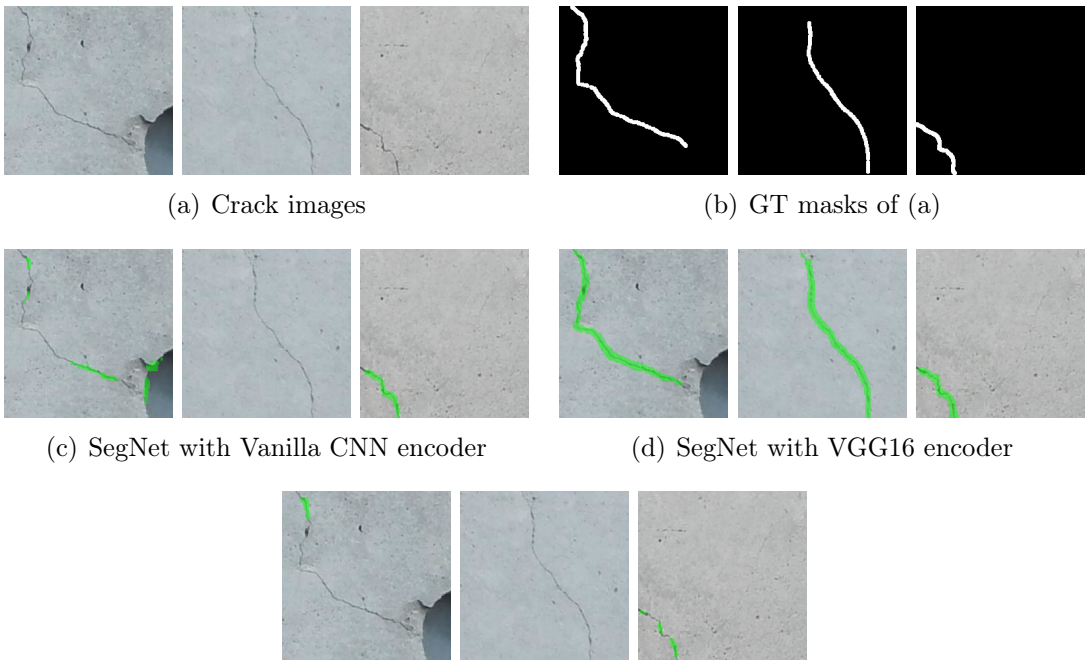


(d) U-Net with VGG16 encoder



(e) U-Net with ResNet-50 encoder

Figure 5.12: Crack detection results using the U-Net model on the SDNET subset



(a) Crack images

(b) GT masks of (a)

(c) SegNet with Vanilla CNN encoder

(d) SegNet with VGG16 encoder

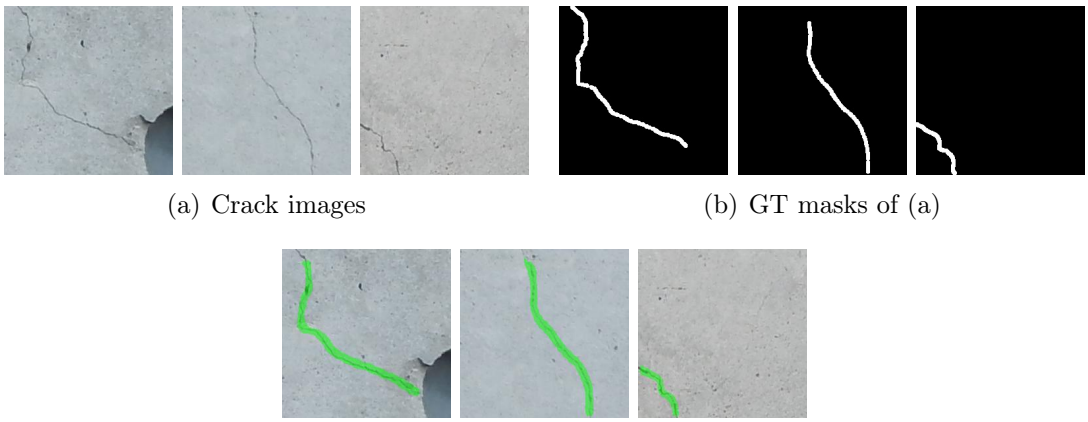
(e) SegNet with ResNet-50 encoder

Figure 5.13: Crack detection results using the SegNet model on the SDNET subset

5.4.2.2 LHC

The quantitative results from the LHC dataset presented in Section 5.4.1.2, imply that the semantic segmentation models with a ResNet-50 encoder network both resulted in the highest IoU. This is also observed in the following sample image in Fig. 5.15 where the U-Net and SegNet model's performance outcome was better than that of Mask R-CNN, with the segmentation maps being very close to the corresponding GT of each image.

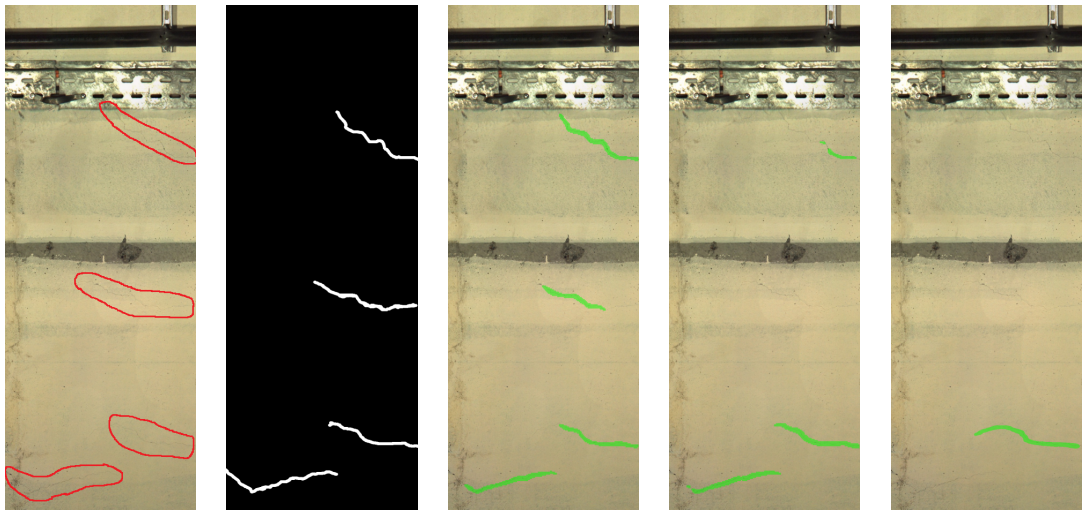
Considering further images from the LHC dataset such as those in Appendix B, the U-Net with a ResNet-50 encoder generated the best results in general. Hence, this trained model was used in the final implementation of the crack detection module of the developed monitoring solution.



(a) Crack images (b) GT masks of (a)

(c) Mask R-CNN

Figure 5.14: Crack detection results using the Mask R-CNN model on the SDNET subset



(a) Crack image (b) GT (c) U-Net (d) SegNet (e) Mask R-CNN

Figure 5.15: Original crack image example from the LHC dataset and its corresponding GT mask together with the resulting crack detection results using Mask R-CNN with ResNet-101 backbone and both U-Net and SegNet with a ResNet-50 encoder

When considering both datasets, U-Net achieved better overall results when compared to the other models. This is mainly attributed to a smaller and simpler architecture. U-Net has a lower number of model parameters and therefore well suited the relatively small datasets that were available for training.

5.5 Class-specific object-based change detection

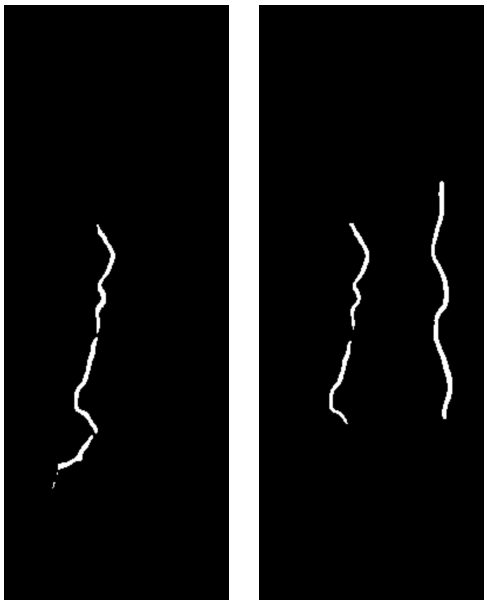
While the detection of cracks is important, monitoring their evolution can be even more beneficial. By applying change detection techniques to images from different times, temporal comparison of the detected cracks can be made to find any changes occurring due to new changes or an evolution in existing ones. One approach of change detection is Object-based Change Detection (OBCD). Rather than using individual pixels as in PBCD, this uses image objects.

OBCD methods usually involve a two-step process: object extraction and object correspondence. The former usually involves a combination of segmentation and connectivity analysis and is applied to N temporal images, such that the outcome is N sets of objects. Secondly, the objects in one set are compared to those in the other sets, using attributes like the area, perimeter and centroid.

5.5.1 Temporal comparison of cracks

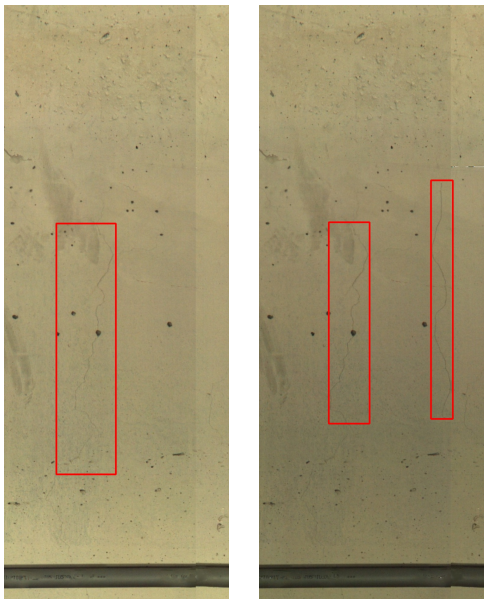
In this bi-temporal scenario, a class-specific OBCD approach is used. Cracks in the reference and survey images are extracted as 'objects' using the crack detection deep learning model and then compared using geometrical properties.

First, crack masks corresponding to the reference and survey images such as the ones shown in Fig. 5.16, are retrieved. Morphological closing is then applied to fill in any missing parts of the crack. Connectivity analysis is then made to extract the different crack bounding boxes separately as displayed in Fig. 5.17.



(a) reference (b) survey

Figure 5.16: Crack masks corresponding to (a) reference and (b) survey image



(a) reference (b) survey

Figure 5.17: Crack bounding boxes corresponding to cracks in the (a) reference and (b) survey image

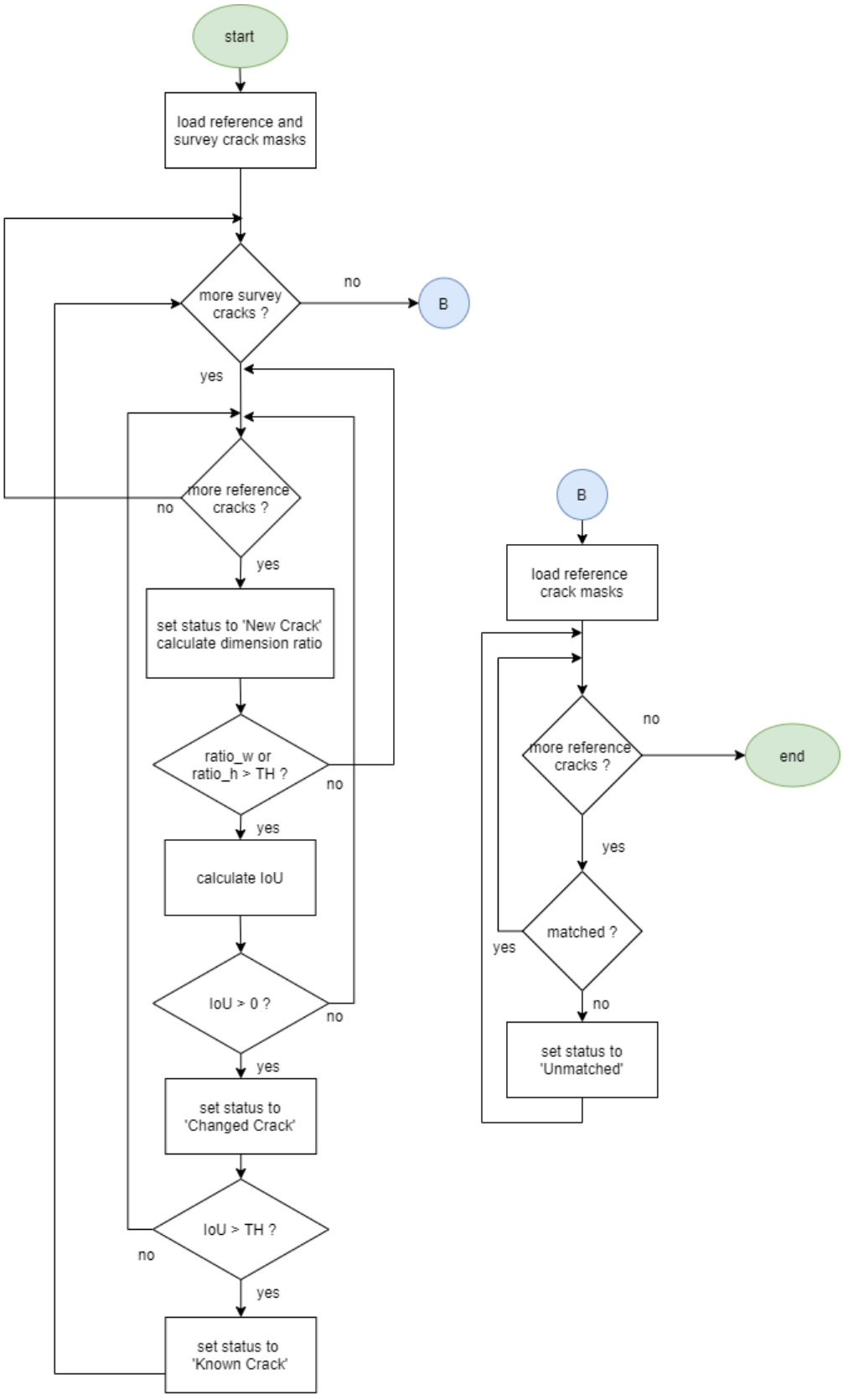


Figure 5.18: Flow diagram for the crack comparison procedure

Comparison of 'object' properties is then made using the location, width and height of the crack bounding boxes as depicted in the flow diagram in Fig. 5.18. For each crack in the survey image, the width and height ratios with respect to every crack in the reference image are calculated. If the width and/or height ratios show that the cracks are similar in dimensions, the IoU with respect to the same crack in the reference image is calculated. If this is above 0 then a match is implied. If the IoU is higher than a pre-defined threshold, a 'known crack' status is assigned to the crack. If IoU is greater than 0 but less than the threshold, the crack is marked as a 'changed crack'. If none of the conditions are satisfied for any of the reference cracks, the crack is assigned a 'new crack' status.

Once all the cracks in the survey images are assigned a status, the unmatched cracks in the reference images are recorded. This is done to cater for cases where the crack detection module detected a crack in the reference image but did not detect the same crack in the survey image, thus this should not be considered as a change.

Considering the scenario in Fig. 5.17, the survey image contains a crack which was also detected in the reference image and a new one, thus as shown in Fig. 5.20, crack [0] is marked as a 'new crack' while the other is a 'known crack'.

In another scenario displayed in Fig. 5.21, the survey image contains a new crack in the same area in which a crack was already identified in the previously captured reference image. Furthermore, the common cracks were detected differently in the reference and survey images, thus as shown in Fig. 5.22, crack [0] is marked as a 'new crack' while the other is a 'changed crack'.

Such an approach to change detection is beneficial for systems in which exact image registration is not possible. Using this OBCD method, crack monitoring can still be done to identify changes in cracks over time. During this research, registered images were later readily available, thus, for this solution, a general change detection using PBCD techniques was later implemented as will be discussed in Chapter 7.

5.6 Contributions summary

The contributions from the work described in this chapter include:

- using deep learning models to automate crack detection;
- monitoring of cracks (new and/or changed) over time using image processing and OBCD techniques;
- building groundtruth datasets of masks on images from a subset of a standard crack image set as well as a set of images captured in the LHC tunnel.

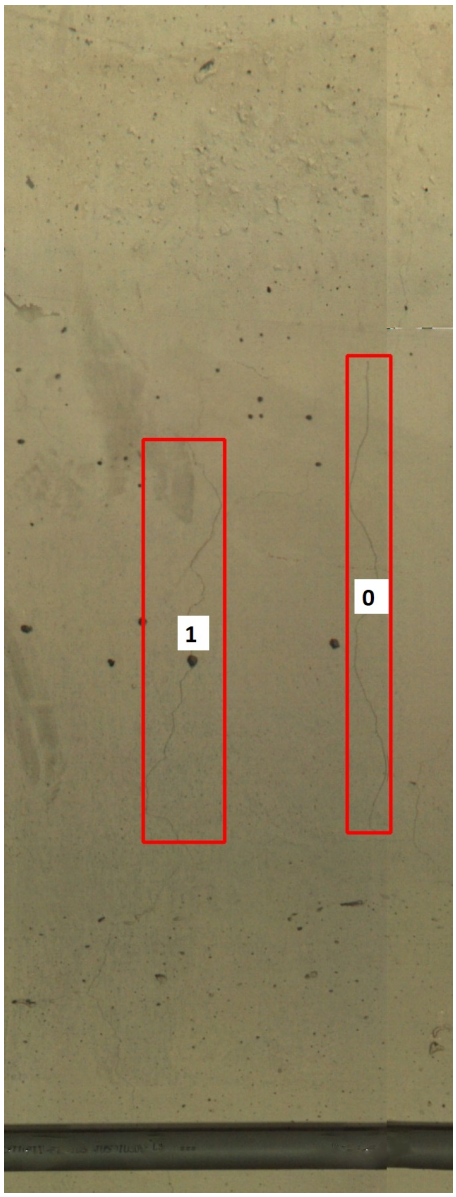
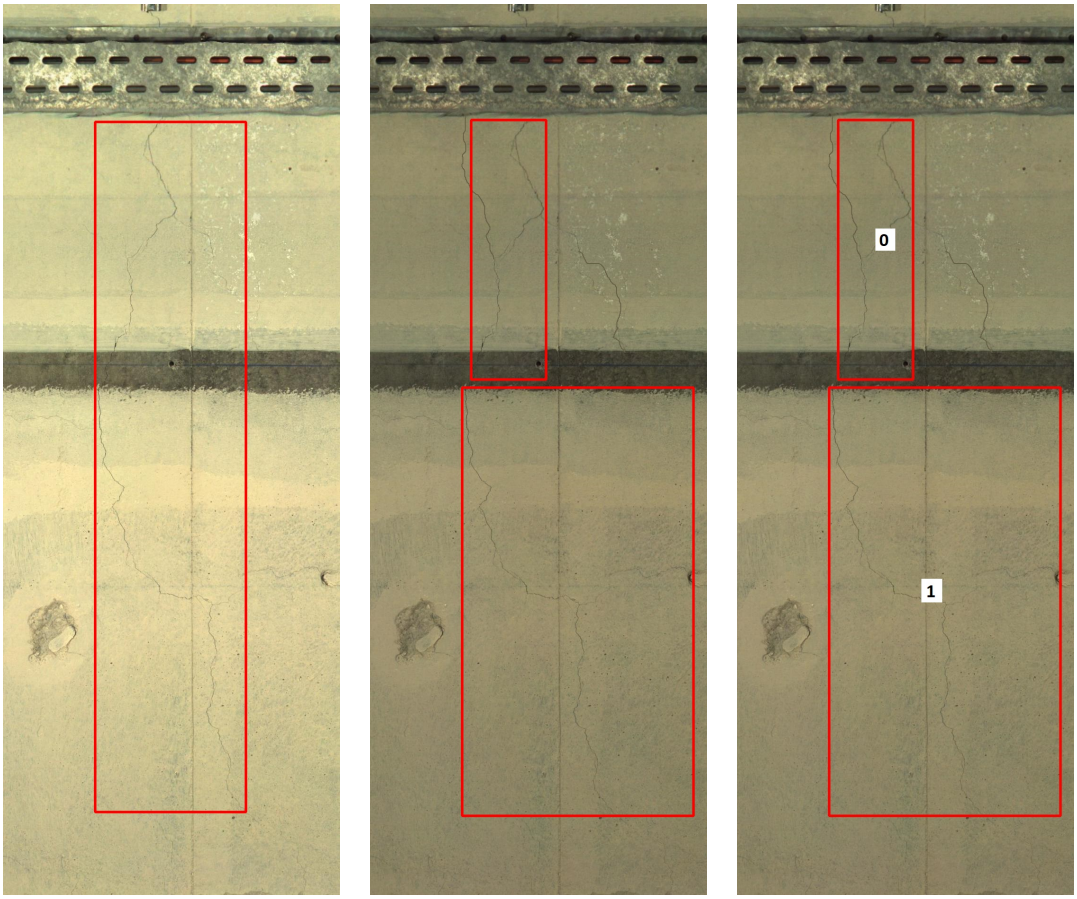


Figure 5.19: Crack detection example 1

```
Info[CH-CR002] ref 0, survey 0 IoU=0, width ratio=0.37, height ratio=0.95
Info[CH-CR002] ref 0, survey 1 IoU=0.55, width ratio=0.687831, height ratio=0.80
Info[CH-CR001] 0: newCrack
Info[CH-CR001] 1: knownCrack
```

Figure 5.20: Crack comparison result listing the status of each identified crack in example 1



(a) reference (b) survey (c) comparison

Figure 5.21: Crack detection example 2, displaying bounding boxes corresponding to cracks in the (a) reference and (b) survey image and (c) the comparison result

```
ref 0, survey 0 IoU=0.19, width ratio=0.50, height ratio=0.38
ref 0, survey 1 IoU=0.46, width ratio=0.65, height ratio=0.62
Results:
[0]: new crack
[1]: changed crack
```

Figure 5.22: Crack comparison result listing the status of each identified crack in example 2

6 | Specular Highlight Localisation

Specular highlights occur from an electronic flash unit or continuous light falling on a reflective surface. These are ubiquitous in the physical world however, they negatively impact the performance of segmentation, detection or matching algorithms in computer vision applications. When light falls on a boundary between two different media, it immediately reflects back to the medium it came from. The visual appearance of such specular reflections is known as a specular highlight. Applications involving visual recognition, object tracking, stereo reconstruction or change detection, require consistent object appearance, and hence the need to identify such highlights and/or correct them.

When electronic flash units are used during image acquisition, they can cause specular highlights which are not constant in each image. During image acquisition in the LHC tunnel, the two electronic flash units on the camera system described in Section 4.5, caused reflections on metal racks on the wall and the beamline, resulting in specular highlights in the images. Such highlights are not constant neither in time nor in place, leading to false detections when comparing images to identify changes. Hence, a specular detection module was developed as a preprocessing stage to the change detection algorithm in order to localise these areas.

The rest of this chapter is structured as follows. An introduction to specular highlight detection is made in Section 6.1, where related previous works are also mentioned. The semantic segmentation method used to localise the highlight areas is introduced in Section 6.2. Methodology details are presented in Section 6.3 while the datasets used for training and testing the models are described in Section 6.4.

Experiments of varying configurations of the U-Net model with different encoder architectures are explained in Section 6.5, where quantitative and qualitative results on the used datasets, are discussed.

6.1 Specular highlight detection

To detect these highlights, there are different types of segmentation approaches that can be adopted, including those involving thresholding, edge detection and clustering. An analytical survey in [215] discusses different methods for specular highlight detection.

A common approach is that of intensity thresholding, using either a fixed or an adaptive value that is set automatically. This works best when the threshold context is darker and the image dynamic range is well distributed. When applied to specular-free images this method might produce false positives such as white objects while images with specularities do not necessarily have a peak at the end of the histogram. Another general issue of this approach is the over/under estimation of highlight areas. In some works, instead of setting a global threshold, an adaptive threshold from different image channels is used to isolate highlight areas. In addition, other works use varying colour spaces where the threshold on one or more channels is estimated dynamically using information from the histogram of the same channel and/or brightness calculated using intensity values. Dimensionality reduction and optimisation algorithms can also be used to isolate specular reflections by mapping the colour distribution between images of an object under different illumination conditions. While these methods are relatively simple to implement and incur low computational cost, they have various limitations as mentioned above.

To mitigate such issues, machine learning has also been used to detect specular highlights. In [216], a perceptron neural network is implemented to classify specular regions. To detect highlights in endoscopic images, a deep learning approach based

on the SegNet architecture is used in [217]. The SegNet architecture is trained on pairs of images and dense per-pixel labels. For reflection segmentation, the labels specify whether a pixel is part of a reflection or not.

To detect specular highlights on objects in images captured in the LHC tunnel, semantic segmentation is used. In particular, the U-Net [51] model is used as the basis architecture, to which, some modifications were done. The following sections describe the methodology used and experiments done with different backbone architectures for the same model.

6.2 U-Net semantic segmentation

The original U-Net architecture described in Section 5.1 is used. As illustrated in Fig. 6.1, a few modifications to this baseline model were made. These include a reduction in the model size, the introduction of BN and usage of dropout, as explained in [218].

U-Net combines the location information from the downsampling path with the contextual information in the upsampling path to finally obtain information combining localisation and context, which is necessary to predict a good segmentation map as required by specular highlights localisation. Compared to other image processing methods such as thresholding, this method generalises better and does not depend on any predefined values. Considering the limited number of available training data samples, U-Net was a natural choice for the base architecture as it requires relatively small amounts of training samples.

6.3 Methodology

The modified architecture in Fig. 6.1 consists of three convolutional blocks for each of the downsampling and upsampling paths. Each block contains two 3×3 convolutional layers each followed by a ReLU. A 2×2 max-pooling layer follows

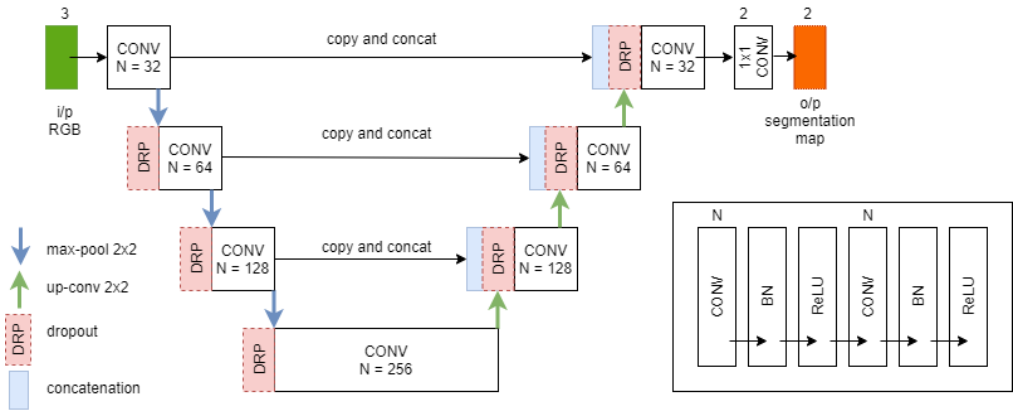


Figure 6.1: U-Net model with the proposed modifications

each convolutional block. In this path, the number of channels increases from the input three-channel image, $N = 3$ to $N = 32$ for the first block up to $N = 256$. In the upsampling phase, convolutional blocks are correspondingly symmetric to those in the downsampling path, decreasing the number of channels from $N = 256$ to $N = 32$. As opposed to the architecture in [51], a BN layer is inserted after each convolutional layer. Furthermore, experiments with dropout at different locations within the architecture were made.

6.3.1 Batch normalisation

A BN layer is added after each 3×3 convolution in the convolutional block. This normalises activations in a network, across the mini-batch during training and restricts them to have a zero mean and unit variance reducing the internal covariate shift. BN is added to the modified architecture to speed up the convergence during training and to apply an indirect regularisation term to avoid overfitting.

6.3.2 Dropout

During training, neurons develop co-dependency amongst each other, which restrains the individual power of each neuron and leads to overfitting of training data. To mitigate this, dropout is generally used, providing implicit data augmen-

tation. When using dropout, at each training stage, individual nodes are dropped with a pre-defined probability p, indirectly reducing the network size. The dropout step does not change the volume size of the output as it has no trainable parameters. Different configurations with no dropout or dropout $p = 0.2$ after each level or at the end, were tested.

6.3.3 Training and optimisation

During training, the Adadelta optimiser [219] was used. This optimiser adapts the learning rate based on a moving window of gradient updates, rather than accumulating all past gradients. In this way, Adadelta continues learning even after many updates have been made. The optimiser was initialised with a learning rate of 1.0 and a decay factor of 0.95. The weights of the network layers were initialised using the Xavier uniform initialiser which draws samples from a uniform distribution within $[-limit, limit]$, defined by:

$$limit = \sqrt{\frac{6}{(fan_{in} + fan_{out})}} \tag{6.1}$$

where fan_{in} and fan_{out} are the number of input and output units in the weight tensor.

6.3.4 Data augmentation

The successful implementation of deep learning models requires a large amount of varied training data. To enable the network to learn the desired invariance and robustness properties, a large number of training data is essential otherwise data augmentation is essential. Here, smooth deformations of the existing image samples through vertical and horizontal flipping, vertical and horizontal displacements in the range [-20%, 20%] and rotations in the range [-45°, 45°] are generated.

6.4 Highlights datasets

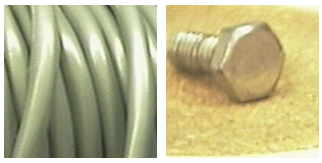
To train and test the network, two different datasets were used. In the following sections, these are referred to as the PURDUE set and LHC set. The former is a publicly available dataset with highlights on different objects while the latter was built from images captured in the LHC tunnel.

6.4.1 PURDUE set

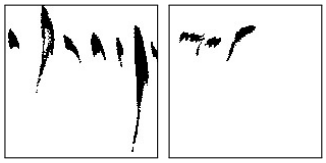
The generic specular highlights dataset PURDUE RVL SPEC-DB [220] was used. This dataset contains 300 images with specular highlights under three different conditions, namely ambient, directed and diffused. The images in this dataset have a resolution of 150×150 and consist of objects of different sizes, colours and materials. The dataset contains GT segmentation images corresponding to 200 of these images. The 80/20 rule was used to divide the data in 128 images for training, 32 for validation and 40 for testing. Fig. 6.2 presents two samples from this dataset. As observed in Fig. 6.2(b), the masks provided by [220] contain pixels with white representing background and black representing the highlights. However, to train the proposed segmentation network for n classes, pixels should have values $(1, 2, ...n)$ to represent the segmentation classes while the background is represented by '0'. Hence, the original masks were inverted, with '0' assigned to the background and '1' to highlight areas as shown in Fig. 6.2(c).

6.4.2 LHC set

A subset of images, was chosen from the dataset mentioned in Section 4.5.2. These images have a resolution of 1885×711. To generate the masks, the specular highlights in each of these images were manually marked using an annotation application [214]. Fig. 6.3 shows two samples from the generated mask dataset. Similar to the previous dataset, '0' was assigned to the background and '1' to highlight



(a) original images



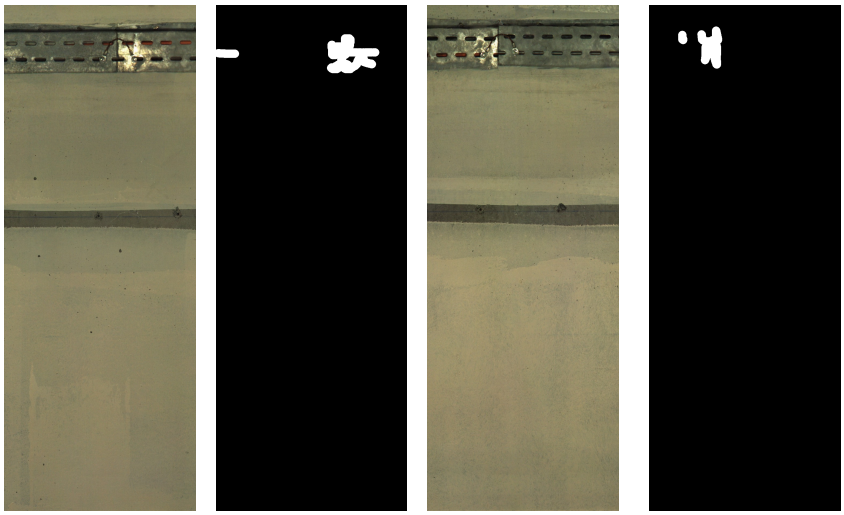
(b) GT



(c) inverted GT

Figure 6.2: A sample of images and their corresponding original GT mask from the PURDUE dataset [220] and the inverted mask

areas. The 80/20 rule was used to divide the data in 76 images for training and 18 for validation. The remaining 24 images were used for testing.



(a) scene A (b) GT of (a) (c) scene B (d) GT of (c)

Figure 6.3: A sample of images and their corresponding GT markings from the annotated specular highlights dataset built using the LHC dataset

6.5 Experiments and results

The U-Net Keras implementation in [201] was used. The previously discussed modifications to the original architecture were then made on this. Furthermore, the model was also trained with other encoder architectures, including those based on VGG16 and ResNet-50.

6.5.1 Quantitative results

Different configurations of the U-Net model with the proposed encoder modifications were trained and analysed through different evaluation metrics. The model loss, IoU, and F-score were monitored during both training and validation to empirically define the optimal configuration.

The F-score multiplies the intersection area of the predicted segmentation and GT by two and divides it by the total number of pixels in both images as defined by:

$$F - score = 2 \times \frac{intersection}{union} = \frac{GT \cap SP}{GT + SP} \tag{6.2}$$

The IoU, on the other hand, divides the intersection area by the union of the predicted segmentation and GT. Although as shown in Eq. 5.1 and 6.2, these metrics are very similar, in general, the IoU metric tends to inflict a higher penalty on single instances of bad classification than the F-score, quantitatively. The IoU metric tends to have a 'squaring' effect on the errors relative to the F-score. Hence, the F-score tends to describe the average performance, while the IoU score implies the worst case performance. By studying the loss, different configurations were analysed to find the optimal one, avoiding any underfitting or overfitting issues.

To analyse the segmentation performance, two IoU-based metrics were used; mean IoU and frequency-weighted IoU defined by:

$$mean\ IoU = \frac{1}{k}.\sum_{i=1}^{k} \frac{n_{ii}}{t_i - n_{ii} + \sum_{j=1}^{k} n_{ji}} \in [0,1] \tag{6.3}$$

$$frequency - weighted\ IoU = (\sum_{i=1}^{k} t_i)^{-1}.\sum_{i=1}^{k} t_i.\frac{n_{ii}}{t_i - n_{ii} + \sum_{j=1}^{k} n_{ji}} \in [0,1] \tag{6.4}$$

where k is the number of classes and n_{ji} with $i,\ j \in 1, 2, ..., k$ is the number of pixels which belong to class i but were labelled as class j. While, the ground-truth total number of pixels for class i is given by:

$$t_i = \sum_{i=1}^{k} n_{ji} \tag{6.5}$$

The mean IoU calculates the mean of all class results. The frequency-weighted IoU is an extension of the previous metric used to better interpret results in class imbalance scenarios. If a class dominates most part of the images in a dataset such as the background, it needs to be weighed down compared to other classes. Thus instead of taking the mean of all the class results, a weighted mean is taken based on the frequency of the class region in the dataset.

6.5.1.1 PURDUE set

For the PURDUE dataset, first the U-Net model with the proposed encoder architecture was trained with a batch size of 1. It was observed that the F-score kept increasing during both training and validation. As for the loss, the curve seemed to behave differently for training and validation, where the validation loss was constantly oscillating at higher values than during training.

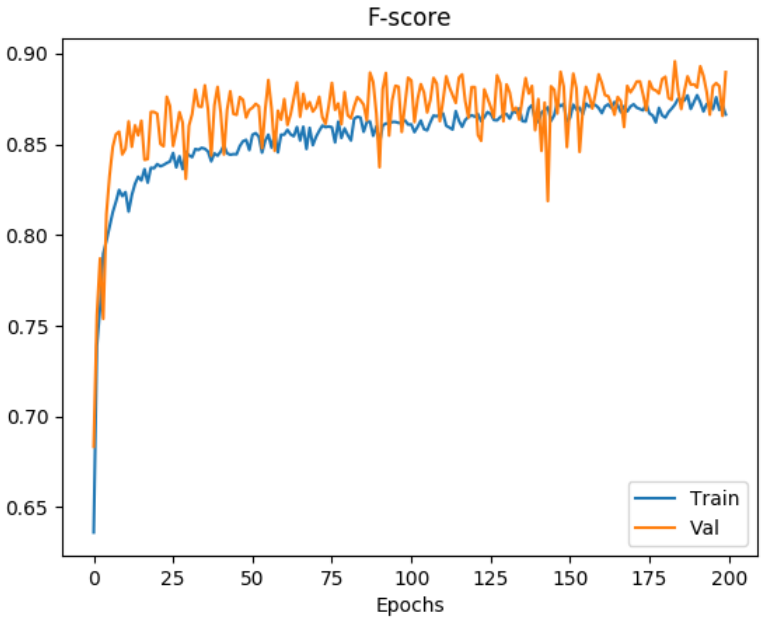
The same model was then trained using a batch size of 20 and inserting a BN layer after each convolutional layer. As shown in Fig. 6.4, the performance of the network improved, with the training and validation curves for both the Cross-Entropy loss and F-score being very close to each other implying no overfitting.

In addition to this, as depicted in Fig. 6.4, the validation loss did not improve after 120 epochs. Considering this, in the experiments that followed, the network was trained for 120 epochs. Following this, using a batch size of 20 with BN, experiments with the use of dropout within the network were made. The optimal configuration was empirically found to be a dropout after each stage as shown in Fig. 6.1. Considering the small amount of available images, a data augmentation pipeline using the transformations described in Section 6.3.4 was used.

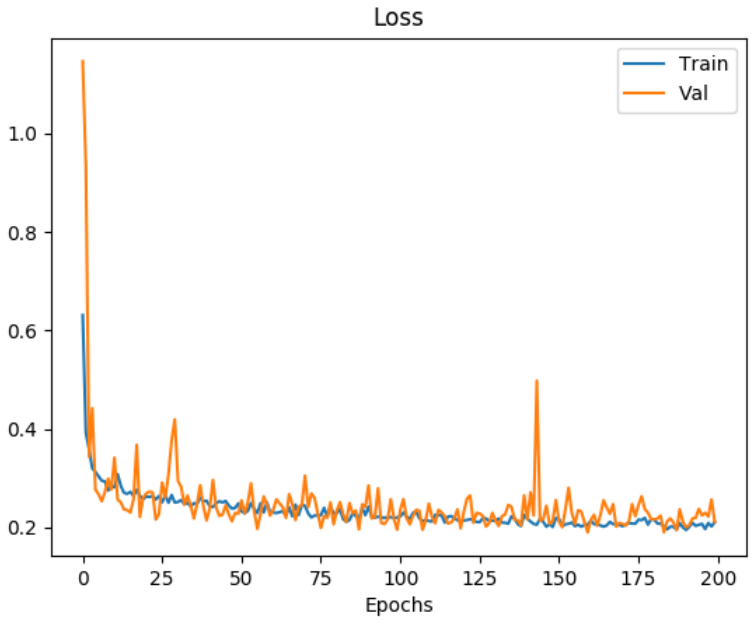
Table 6.1: Summary of results on the PURDUE dataset during the validation of the U-Net model with different encoder architectures

Encoder architecture	**Frequency-weighted IoU**	**Mean IoU**
ResNet	0.80	0.70
VGG	0.77	0.68
MobileNet	0.81	0.70
Proposed	**0.83**	**0.75**

In addition, the U-Net model was also trained using different architectures for the encoder including VGG16 and ResNet-50. Metric results in Table 6.1 show that, for the relatively small dataset available, the proposed encoder network with a smaller and simpler architecture, could achieve better overall results, achieving a highest frequency weighted and mean IoU of 0.83 and 0.75 respectively.



(a) F-score for batch size = 20, using BN



(b) Loss for batch size = 20, using BN

Figure 6.4: F-score and loss from the U-Net model with the proposed encoder architecture during training and validation on the PURDUE set

6.5.1.2 LHC set

Similarly, several experiments were conducted to train and test the model on the LHC set. These include varying configurations of the proposed encoder architecture, with and without BN, different batch sizes and dropout at different stages within the network. In addition, the U-Net model was also trained with different encoder architectures, to compare the performance with the proposed architecture.

The model was trained with a batch size of 1 and later with a batch size of 20 with BN. For both experiments, different configurations with no dropout or dropout $p = 0.2$ after each level or at the end, were tested. The optimal configuration was empirically found to be a dropout after each stage as shown in Fig. 6.1. A better general performance is observed when using a batch size of 1, where during both training and validation, the values of F-score in Fig. 6.5 and IoU in Fig. 6.6 are higher, while the loss in Fig. 6.7 is lower than when using a batch size of 20.

The U-Net model was trained using different architectures for the encoder including VGG16 and ResNet-50. In general, the training and validation curves for F-score in Fig. 6.8, IoU in Fig. 6.9 and loss in Fig. 6.10 imply that the U-Net with a VGG16 encoder architecture performed better. From these plots, one can also observe that the network did not exhibit any significant improvement after 75 epochs. Thus, the checkpoint at 75 epochs was used to test the models on the 'test' subset. From the results in Table 6.2, the proposed modified architecture with a smaller and simpler architecture than VGG16 or ResNet-50, achieved better overall results when tested on a subset of new images, achieving the highest frequency weighted and mean IoU values of 0.98 and 0.80 respectively.

Table 6.2: Validation results on the LHC dataset for different encoder architectures

Encoder architecture	**Frequency-weighted IoU**	**Mean IoU**
ResNet	0.97	0.73
VGG	0.98	0.76
Proposed	**0.98**	**0.80**

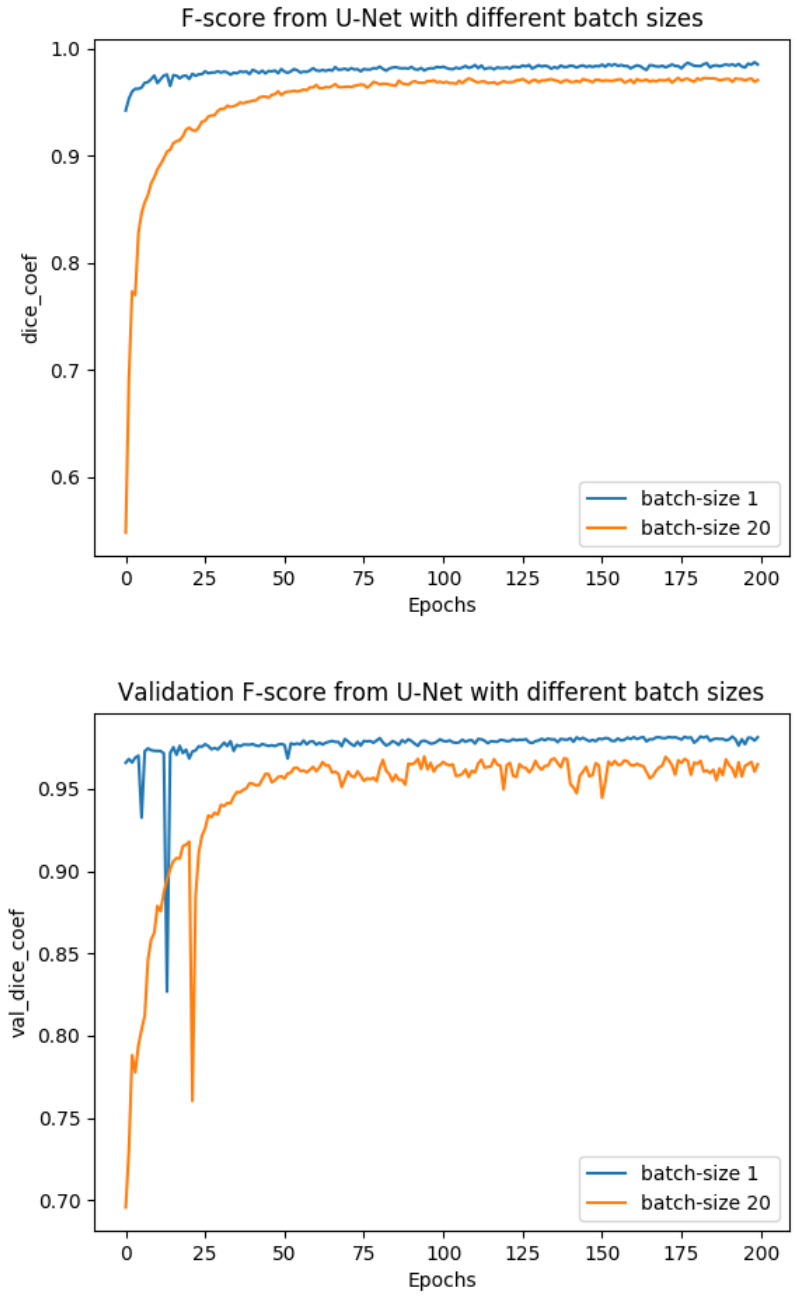


Figure 6.5: Plots of training and validation F-score for the LHC set using the U-Net model with different batch sizes

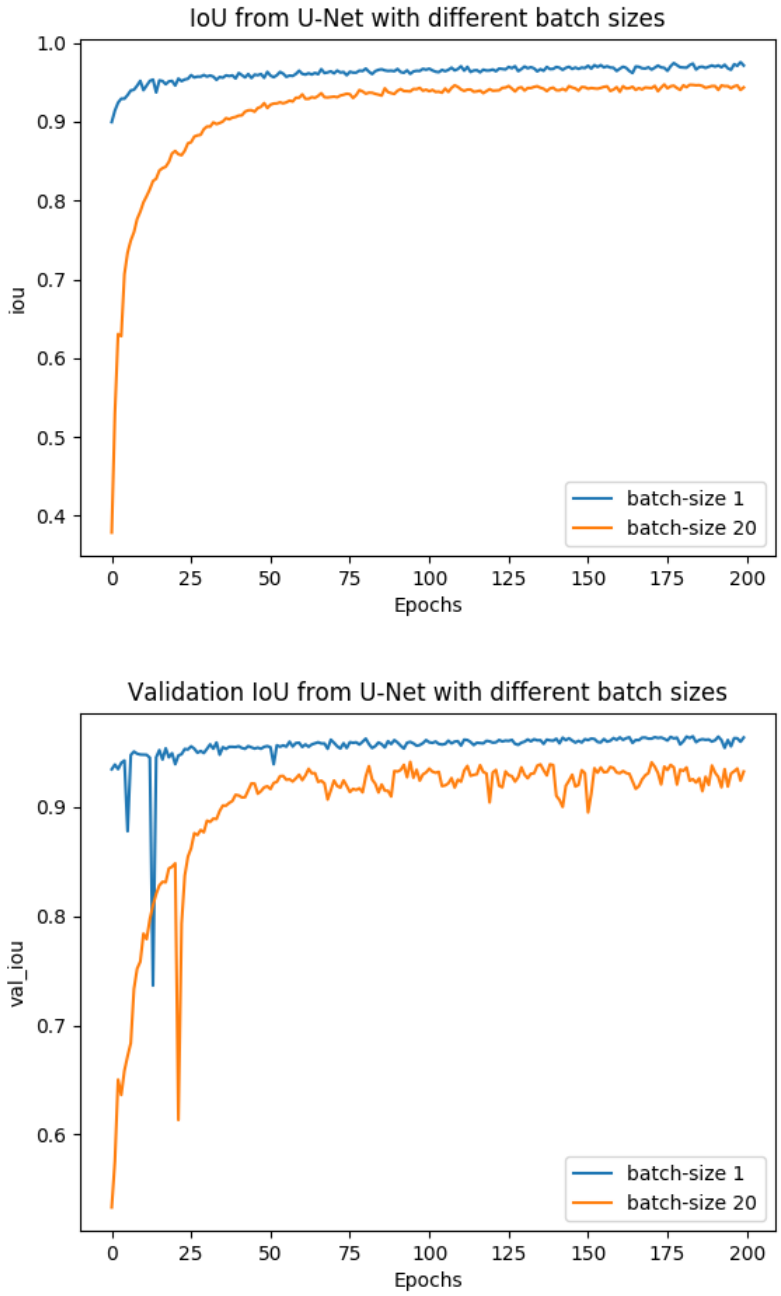


Figure 6.6: Plots of training and validation IoU for the LHC set using the U-Net model with different batch sizes

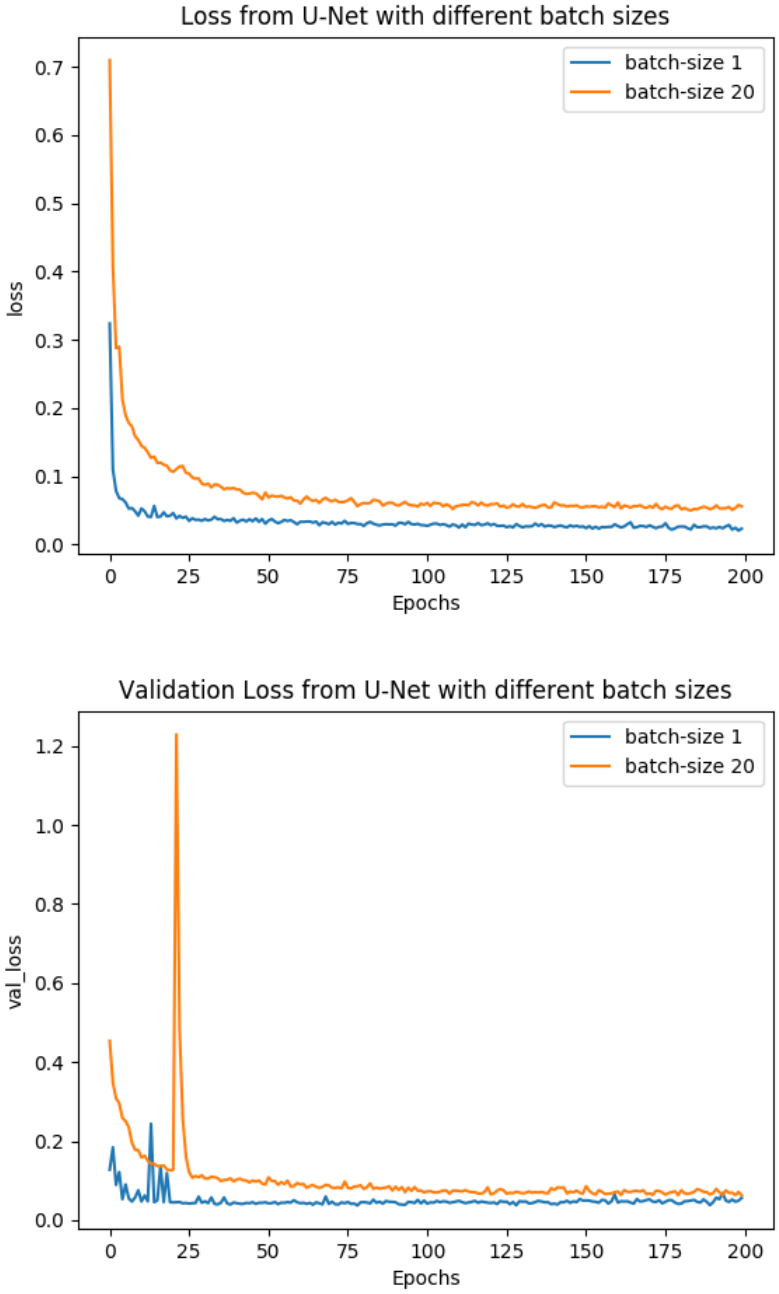


Figure 6.7: Plots of training and validation loss for the LHC set using the U-Net model with different batch sizes

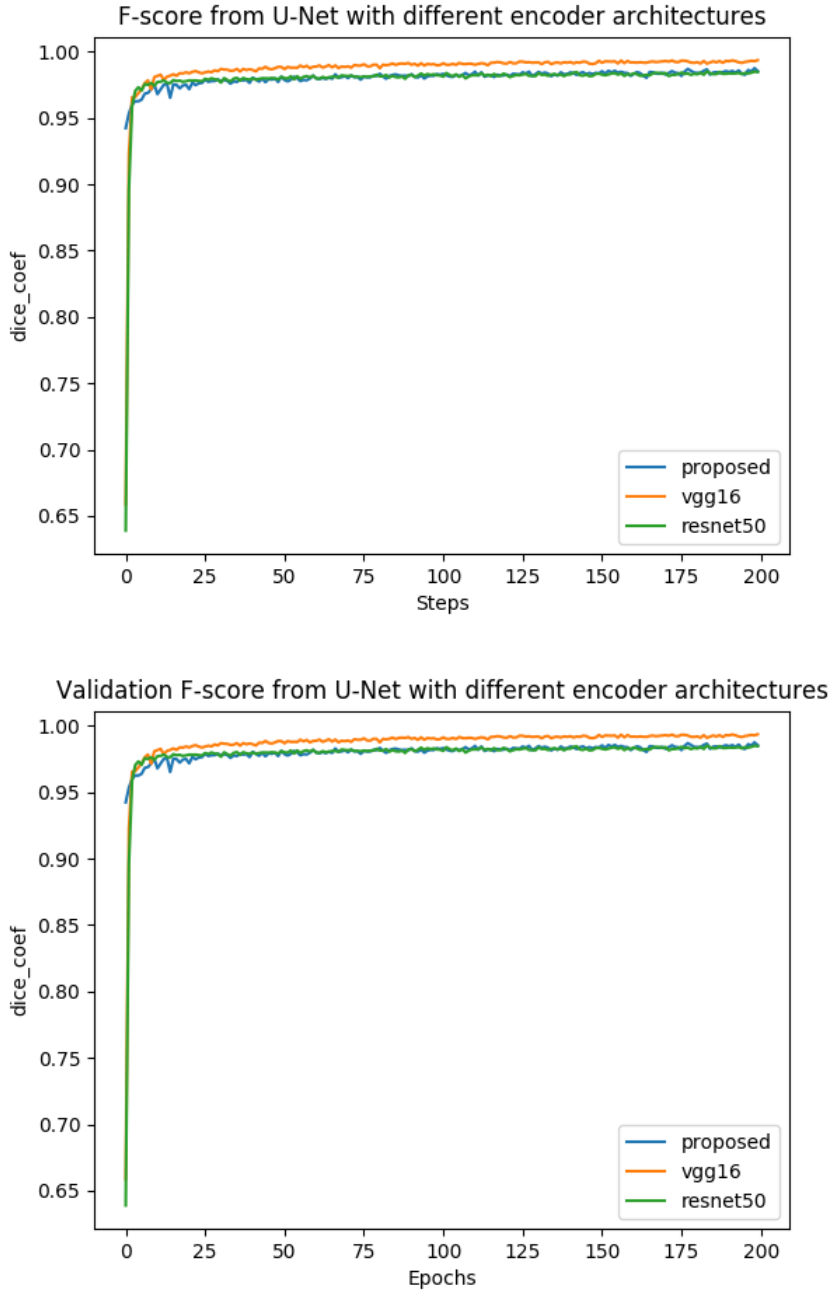


Figure 6.8: Plots of training and validation F-score for the LHC set using the U-Net model with different encoder architectures

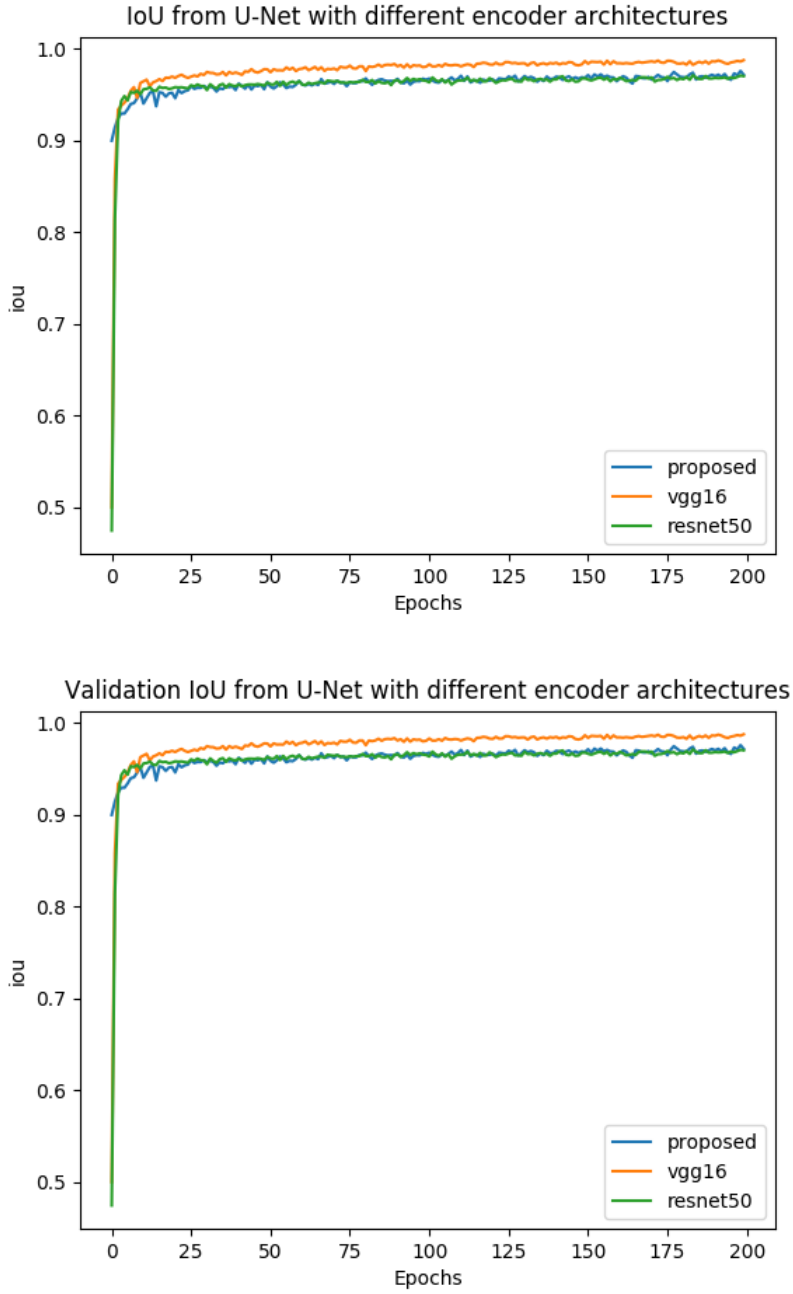


Figure 6.9: Plots of training and validation IoU for the LHC set using the U-Net model with different encoder architectures

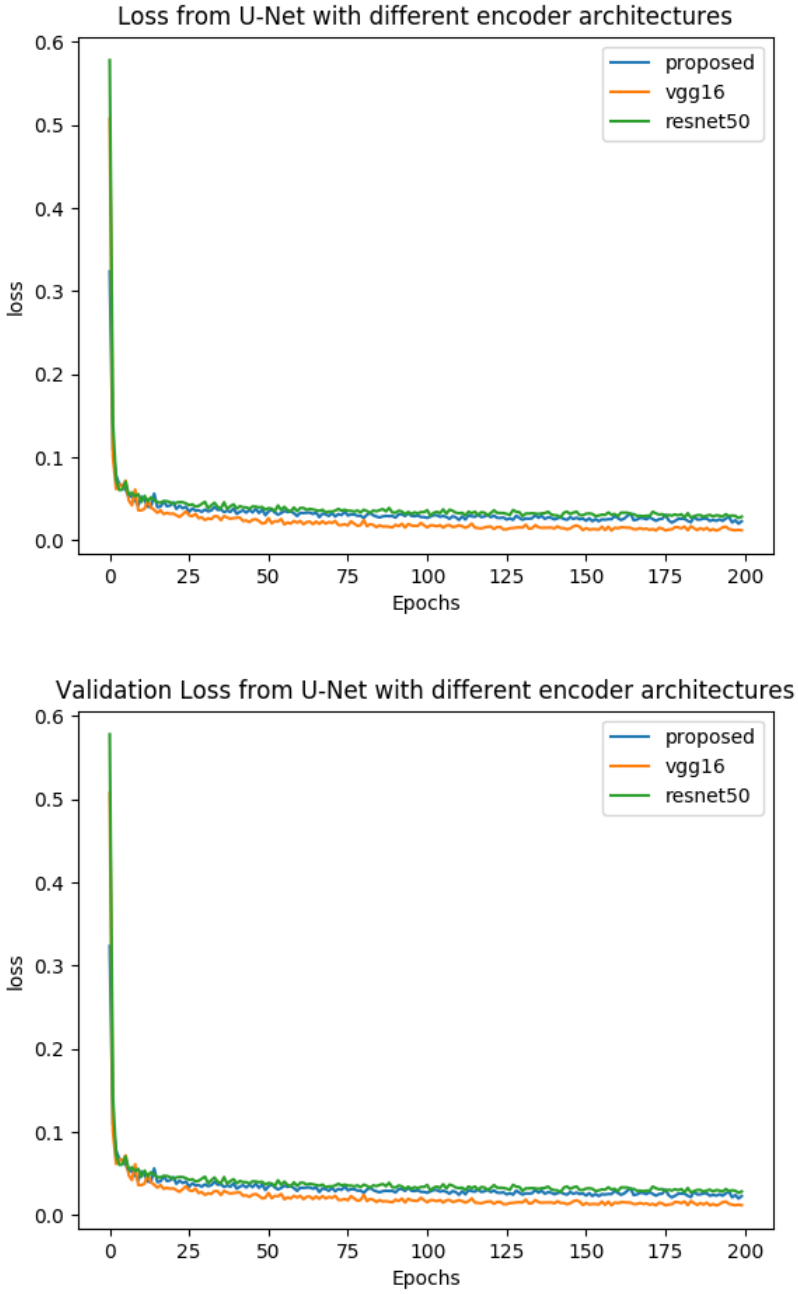


Figure 6.10: Plots of training and validation loss for the LHC set using the U-Net model with different encoder architectures

6.5.2 Qualitative results

6.5.2.1 PURDUE set

As observed in the test images in Fig. 6.11, when comparing the segmentation results with the GT masks, the proposed architecture identified the specular highlights very well. Compared to other larger encoder architectures, such as VGG16 and ResNet-50, in general, the proposed architecture gives less false positive areas.



(a) Original images



(b) GT masks



(c) U-Net with proposed encoder



(d) U-Net with VGG16 encoder

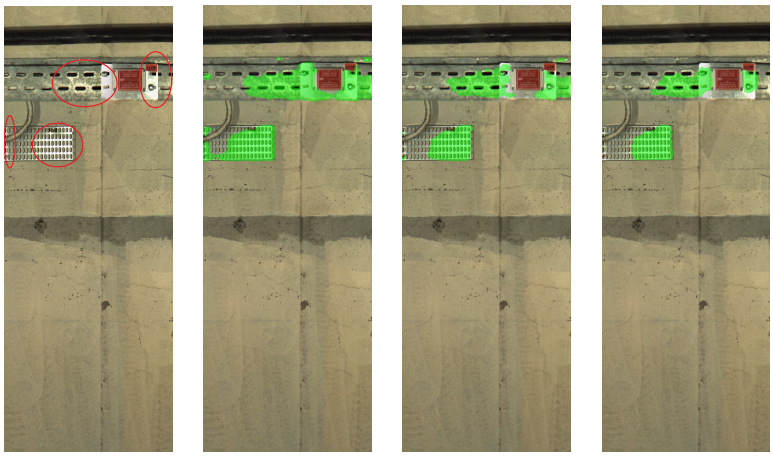


(e) U-Net with ResNet-50 encoder

Figure 6.11: Comparison of the GT (b) and segmentation masks on the PURDUE set, from U-Net with (c) the proposed architecture and (d)-(e) other architectures for the encoder

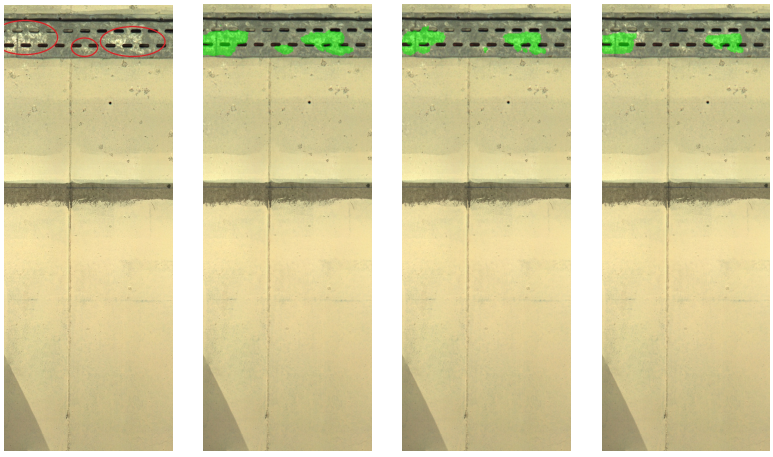
6.5.2.2 LHC set

Similarly, as observed in the test images in Fig. 6.12 and 6.13, the U-Net model with the proposed encoder architecture generated segmentation maps very similar to the GT ones. Compared to the larger architectures, VGG16 and ResNet-50, the proposed encoder, produced less false positive areas.



(a) GT (b) proposed (c) VGG16 (d) ResNet-50

Figure 6.12: Example 1 of a comparison of the highlight detection results using U-Net with (b) the proposed modified architecture and (c)-(d) other architectures



(a) GT (b) proposed (c) VGG16 (d) ResNet-50

Figure 6.13: Example 2 of a comparison of the highlight detection results using U-Net with (b) the proposed modified architecture and (c)-(d) other architectures

Summarising the quantitative and qualitative results discussed in this chapter and other sample results presented in Appendix C, the U-Net with the proposed encoder architecture with a dropout of 0.2 at every stage and a batch size of 1, generated the best results in general. Hence, this trained model was used in the final implementation of the specular highlight localisation of the developed monitoring solution.

6.6 Contributions summary

The work described in this chapter makes several contributions including the:

- proposal of a modified U-Net-based architecture for the purpose of specular highlight localisation
- generation of a groundtruth dataset of masks on image captured in the LHC tunnel;
- detection of specular highlights as a pre-processing step to subsequent inspection stages.

7 | Change detection

The majority of recorded works related to automatic tunnel inspection generally identify cracks and other deformities along the tunnel linings. Some of them, further classify the defects in terms of their type and/or severity. Whilst defect identification is beneficial to automate inspection, regular monitoring of tunnel linings can provide a more useful and informative survey to further automate inspections and analyse the structural health over time.

Detecting regions of change in multiple images captured at the same place but at different times is of widespread interest, providing a fundamental analysis tool in fields such as traffic and pedestrian surveillance, medical diagnosis, remote sensing and civil infrastructure. Thus, a considerable amount of research in change detection has been carried out however, works on change detection specific to tunnel environments are still lacking. Taking the above into consideration, in addition to crack detection, the developed tunnel structural health monitoring solution, includes also a change detection module. Using different data fusion techniques, it offers a means to automatically monitor for changes on tunnel wall lining.

The rest of this chapter is organised as follows. Section 7.1 introduces the need of pre-processing and explains the related stages. The ideal change detection scenario is presented in Section 7.2. Section 7.3 reports the implemented PBCD methods. The implementation of decision-level fusion to merge the different CMs and specular highlight masks is described in detail in Section 7.4. The CM analysis applied to obtain the final change components is explained in Section 7.5. Finally a performance evaluation of the change detection algorithm is made in Section 7.6.

7.1 Pre-processing

Whilst making sure that all actual changes are not missed, it is also beneficial if false detections are kept at a minimum. In civil infrastructure, changes in images can be due to new cracks, spalls or other defects as well as due to the evolution of already existing defects. Unfortunately, computer vision solutions have to face other possible sources of changes that include different lighting sources, uneven illumination, image noise and registration errors in some areas of the image. Such regions should be identified as a nuisance to prevent them from being propagated in a change detection pipeline giving rise to false changes. To cater for the above, images are first pre-processed to correct for uneven illumination and to localise specular highlights to prevent them from reducing the precision of the change detection system.

7.1.1 Uneven illumination correction

Lighting falling on certain areas only, varying light colour, shadows set from different light source directions and vignetting by the camera lens cause non-uniformity in images as shown in Fig. 7.1. Furthermore, these conditions lead to a varying appearance of scene objects and imply nuisance changes when comparing images. This raises the need for illumination correction.



Figure 7.1: Light variation in the LHC tunnel

To correct uneven illumination, there are two approaches. Prospective correction involves capturing additional images; namely a dark image of the scene background with no light or a bright image of the scene background with light but without objects that can be later used to correct the required image. Retrospective correction uses an estimate of the background from the image to be corrected itself. In this scenario, the prospective approach cannot be used since it is not practical to take a sample bright/dark image at each capture position along the tunnel, hence a retrospective approach was adopted.

Uneven illumination can be represented by the multiplication of the 'perfect image' and a 'shading function' as described by:

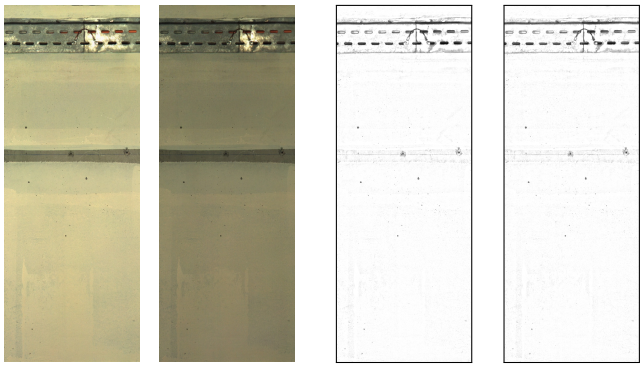
$$I(x, y) = I_{ideal}(x, y) \times S(x, y) \tag{7.1}$$

where $I(x, y)$ represents the image having uneven illumination, $I_{ideal}(x, y)$ is the 'perfect image' and $S(x, y)$ defines the shading function.

Unnaturally darker image areas can be caused by lighting not reaching certain areas or camera lenses causing a decreased brightness from the centre of the image to its ends, known as vignetting. Shadow consists of low frequency content that can be represented by low-pass filtering the original image and then isolated to get the 'perfect image'. Here, the shading algorithm proposed in [174] is applied to adjust the uneven illumination. First, the image is low-pass filtered by a large kernel median filter. Then, using Eq. 7.1, the shading corrected image is attained through a division of the original image by the low-pass filtered image element-wise.

As illustrated in Fig. 7.3(a), subtracting the original images in Fig. 7.2 (a) and (b) after converting them to greyscale generates a difference image full of 'white areas' implying change, however this is only due to the uneven illumination. On the other hand, when the images are pre-processed to correct the uneven illumination, as displayed in Fig. 7.2 (c) and (d), their difference image does not have any 'white areas' due to the change in lighting as shown in Fig. 7.3(b).

This method simultaneously discards the uneven illumination and ameliorates



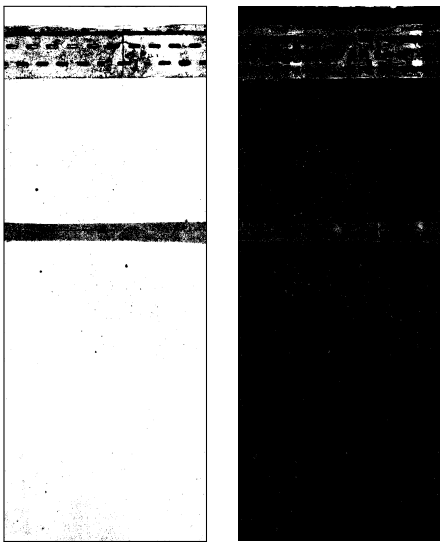
(a) R_{orig} (b) S_{orig} (c) R_{corr} (d) S_{corr}

Figure 7.2: The original (a) reference and (b) survey images at a particular position and the corresponding (c-d) illumination corrected images

the contrast in the image such that wall 'features' are better distinguished from the 'white' background. The computation time required is significant due to the filtering stage and the division operation, yet, the processing time of the latter may be decreased through parallel computing. Hence, this is an effective pre-processing method to prepare more uniform images for subsequent change detection methods.

7.1.2 Specular highlights localisation

The two electronic flash units on the camera system described in Section 4.5 caused reflections on metal racks on the wall and the beamline, resulting in specular highlights in the images. Such highlights are not constant neither in time nor in place, leading to false detections when subtracting images to identify changes as shown in Fig. 7.3. Therefore, the specular highlight detection module described in Chapter 6 is used as a pre-processing stage to localise these highlights in the reference and survey images as displayed respectively in Fig. 7.4 (a) and (b). Morphological operations and connectivity analysis are applied to the highlight segmentation results to generate bounding boxes in highlight areas both in the reference and survey images as illustrated in Fig. 7.4(c). Such highlight masks are later fused with CMs to mask out these false change candidates.



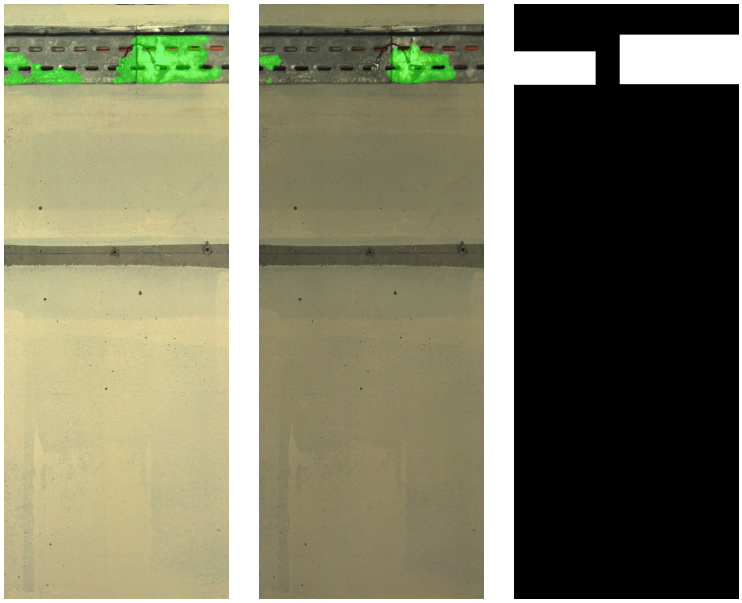
(a) original (b) pre-processed

Figure 7.3: Difference images of the (a) original greyscale (b) pre-processed reference and survey images at a particular position

7.2 Ideal change detection

In an ideal scenario in which images are taken from a fixed camera, the change detection process can be merely a subtraction between the temporal images. Since each pixel position corresponds to both images being compared, the difference in intensity identifies a change occurring in any of the images. When the bi-temporal images in Fig. 7.5 are subtracted from each other, the difference magnitude should be non-zero only at the 'crack' area bound by a red rectangle in Fig. 7.5(d).

However, in a real-world context, the CM contains some minor areas having a greyish tone, implying 'change areas', generated from image noise and even after the pre-processing stage, changes in illumination might render a few 'false changes'. These can be reduced by setting a pre-defined minimum threshold after subtraction however, no single change detection technique is able to generate the ideal CM. Hence, various change detection techniques are investigated in the following sections.



(a) reference (b) survey (c) mask

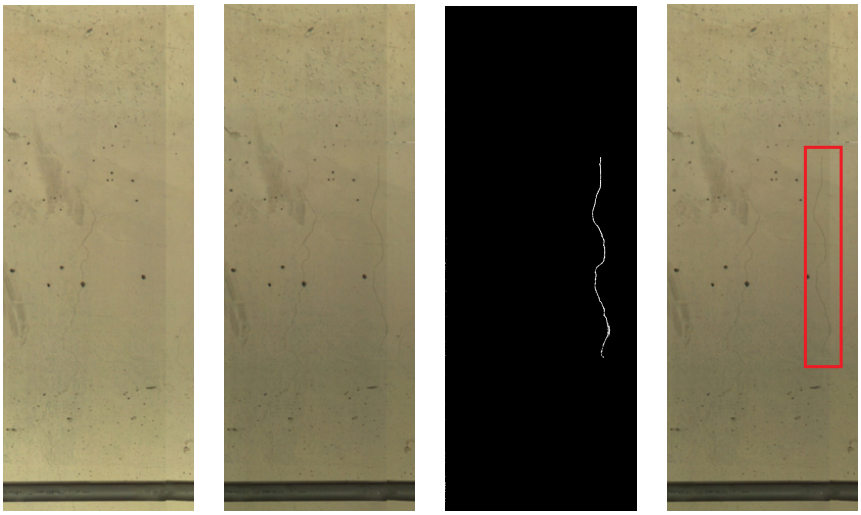
Figure 7.4: Specular highlight localisation on the (a) reference image and (b) survey image and the corresponding highlight mask (c)

7.3 PBCD using bi-temporal image fusion

Image fusion is the process of merging two or more images to produce a single composite image to reveal more inclusive information for further analysis. Multi-temporal fusion combines data from same-scene images, acquired at different times. Hence, this approach can be used to identify scene changes by comparing images. In this scenario, bi-temporal image fusion is applied between two temporal images; the reference and the survey images.

PBCD methods require exact image registration. In this scenario, this is implicitly done using the location information from the encoder wheel attached to the robotic platform, thus the reference and survey images are assumed to be captured at the same location. Consequently, pixel by pixel techniques can be applied.

Bi-temporal fusion using algebraic and transform-based methods is discussed next. These are generally simple to implement and fast to execute however, the detection depends on the registration accuracy of the images being compared.



(a) reference (b) survey (c) ideal CM (d) changed area

Figure 7.5: Change detection between the (a) reference and (b) survey images in an ideal-world scenario, generating the (c) ideal CM and (d) the corresponding bounding box

7.3.1 Image difference

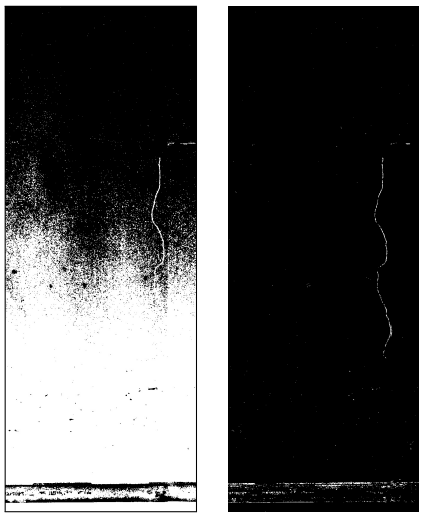
In this method, two images of the same scene taken at separate times t_1 and t_2 are subtracted pixel-wise. Following the image subtraction, the magnitude of the difference value is checked against a threshold condition. Pixels whose difference magnitude is higher than the pre-defined threshold are classified as 'change', otherwise noted as 'no change'. The CM is generated using:

$$\begin{aligned} Diff(x,y) &= |I(x,y,t_1) - I(x,y,t_2)| \\ CM_D(x,y) &= \begin{cases} 1 & \text{if } Diff(x,y) \geq T \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{7.2}$$

where $I(x,y,t_1)$ is the image at time t_1, $I(x,y,t_2)$ is the image at time t_2 and T is the threshold on the difference magnitude.

Due to its simplicity and low computation, it is the most common image comparison approach for change detection. The detection accuracy of this method,

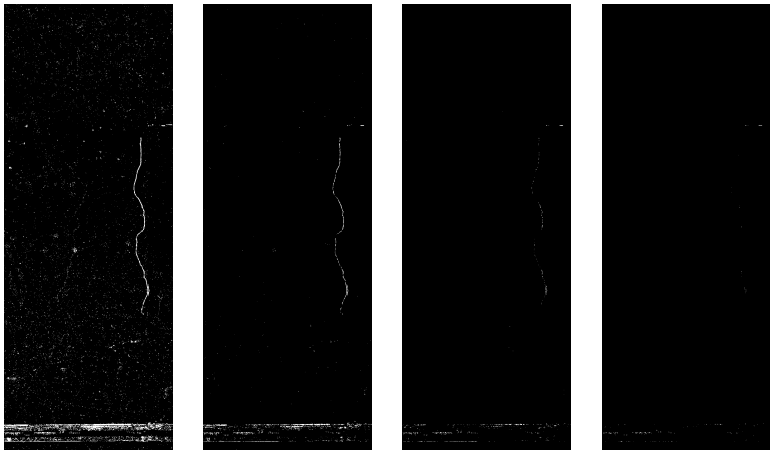
highly depends on the pre-defined threshold. Considering the images in Fig. 7.5, image differencing with a fixed threshold is applied to the converted greyscale images and the pre-processed images. As shown in Fig. 7.6, the resulting difference image from the grasycale images shows the crack pixels as change, however, it also has alot of false change pixels due to the different lighting conditions. On the other hand, the difference image from the pre-processed images, eliminated the pixels undergoing a light change while keeping the actual crack change. Thus, the pre-processed images corrected for uneven illumination are used in this method.



(a) original (b) pre-processed

Figure 7.6: Difference images of the (a) original greyscale (b) pre-processed reference and survey images with illumination changes

The outcome of modifying the threshold T is investigated. The CMs in Fig. 7.7 reveal that as T increases, the number of 'change' pixels decreases, improving noise suppression. However, the 'valid change' pixels are lost when $T \geq 30$.



(a) $T = 10$ (b) $T = 20$ (c) $T = 30$ (d) $T = 40$

Figure 7.7: Image difference using different values for the fixed threshold

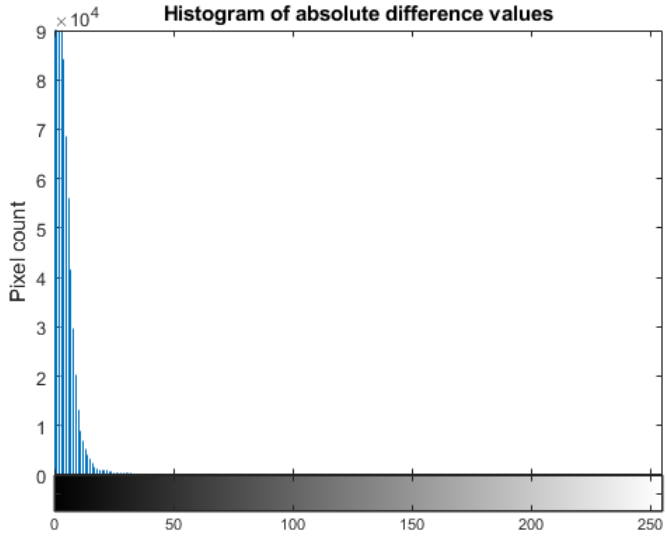


Figure 7.8: Histogram of the pixel absolute difference values

A fixed threshold value cannot however satisfy all scenarios, thus a better approach is to set the threshold automatically. Several automatic thresholding approaches such as Otsu method [19] produce adequate outcomes for images having a bimodal distribution, assuming an ideal threshold value at the valley between the two peaks of the histogram. However, most automated selection methods break if the histogram is unimodal as observed in Fig. 7.8.

If the number of pixels with value i is n_i and n is the total number of pixels in an image then, the probability of occurrence of value i is given by:

$$p_i = \frac{n_i}{n} \tag{7.3}$$

During thresholding, the image pixels are categorised into two classes $C_1 = [0, 1, \ldots, th]$ and $C_2 = [th + 1, th + 2, \ldots, L - 1]$, where th is the threshold value and L states the number of grey levels. The class probabilities are given by $\omega_1(th)$ and $\omega_2(th)$ while the mean values of the two classes are given by $\mu_1(th)$ and $\mu_2(th)$.

$$\begin{aligned} \omega_1(th) &= \sum_{i=0}^{th} p_i \\ \omega_2(th) &= \sum_{i=th+1}^{L-1} p_i \end{aligned} \tag{7.4}$$

$$\begin{aligned} \mu_1(th) &= \sum_{i=0}^{th} \frac{ip_i}{\omega_1(th)} \\ \mu_2(th) &= \sum_{i=th+1}^{L-1} \frac{ip_i}{\omega_2(th)} \end{aligned} \tag{7.5}$$

In unimodal distributions, the ideal threshold generally lies at the edge of the peak of the histogram. Considering this observation, the Valley Emphasis (VE) method was proposed in [221]. This selects a threshold value with a low occurrence probability p_{th} which also maximises the inter-class variance. Using a weight $W(th)$ that is indirectly proportional to p_{th}:

$$W(t) = 1 - p_{th} \tag{7.6}$$

the best threshold is selected by maximising the function below:

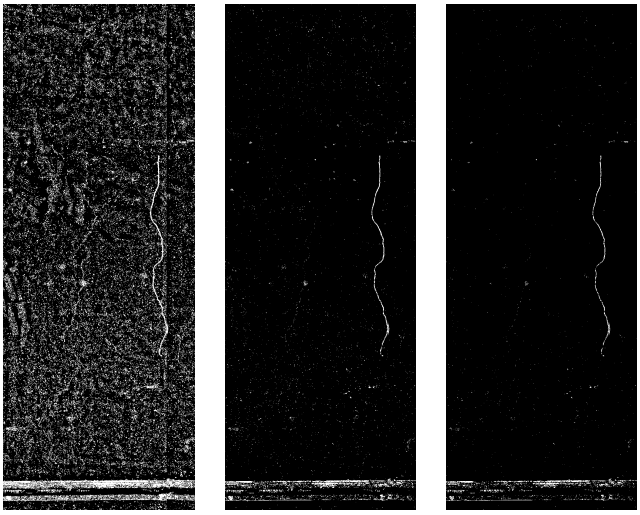
$$th^* = \underset{0 \leq th < L}{\operatorname{argmax}} \{W(th)(\omega_1(th)\mu_1^2(th) + \omega_2(th)\mu_2^2(th))\} \tag{7.7}$$

The weighting term in Eq. 7.6 might be insufficient in scenarios where the variance of one class varies considerably from the other and hence fails to find a correct value for the threshold. Thus, [222] proposed the inclusion of neighbouring values around the threshold to ameliorate the weighting effect. This was later improved by [223], which adopted a weighting scheme that includes neighbourhood information in the objective function such that for every possible value th, a new weighting term is given by:

$$W(th, \sigma) = 1 - \sum_x p_x e^{\frac{(x-th)^2}{2\sigma^2}} \tag{7.8}$$

A Gaussian window with a standard deviation σ is utilised to ensure that threshold locations which are further away from the candidate threshold get a smaller weight than those nearer. This weight term is more significant than that in Eq. 7.6 and the smoothing achieved by the Gaussian window ensures that the modified weight calculation is less vulnerable to noise. Here, σ is empirically set to 5.

The above automatic thresholding techniques were tested on various image pairs. From the resulting CMs in Fig. 7.9 corresponding to the same image pair in Fig. 7.5, it is observed that the Gaussian VE method produced the best CM, reducing the 'noise changes' while retaining the 'crack change'.



(a) VE (b) Neighbour VE (c) Gaussian VE

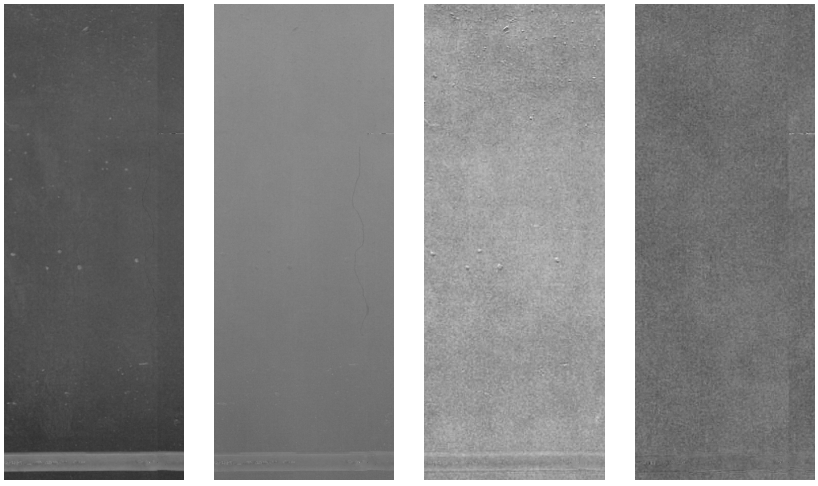
Figure 7.9: Image difference using different automatic thresholding techniques

7.3.2 Principal Component Analsyis (PCA)

PCA is a transformation-based approach to change detection which is often used with remote sensing data. This mathematical technique reduces the dimensionality of a dataset while maintaining the variances. There are mainly two ways in which PCA can be used to detect changes in images. Independent data transformation analysis applies PCA on each of the temporal images separately. The derived principal components are then analysed by applying other change detection techniques such as image differencing and regression. On the other hand, merged data transformation analysis stacks N temporal images of p channels each, fuses them into a single $N \times p$-channel image and applies PCA on the latter.

In this bi-temporal scenario, the merged data approach is used and the reference and survey images are stacked on each other. The method was investigated in terms of two input types, original colour images and pre-processed images ie. illumination corrected images.

When the original colour (RGB, $p = 3$) images ($N = 2$) were used, the stacked images were merged into a 6-channel image. It was observed that the first component (PC_0) corresponding to the highest eigenvalues, contained most of the infor-



(a) PC_0 (b) PC_1 (c) PC_2 (d) PC_3

Figure 7.10: The first 4 principal components of the stacked original images

mation from both images. PC_1 represented the difference between temporal images while later components contained noise information as seen in Fig. 7.10.

Experimental results on different image samples showed that PCA is scene-dependent, thus change detection results between different dates are often difficult to interpret using a fixed condition, implying the need to determine scenario-dependant thresholds. In this case, when considering PC_1, the 'crack change' has a low value (black), the 'pipe reflections change' has a higher value (white) and the rest of the wall has a medium value (grey). This implies that the histogram contains changes at both of its tails, thus a double threshold is required. However as observed in Fig. 7.11, the histogram shape is not clearly defined at its tails, making it difficult to find an adaptive threshold pair.

On the other hand, when the pre-processed images ($p = 1, N = 2$) were used, the stacked images were merged into a 2-channel image. Here, it was observed that the first component PC_0 represents the difference between temporal images while PC_1 contains most of the information from both images as seen in Fig. 7.12.

In this case, when considering PC_0, the 'crack change' has a high value (white), the 'pipe reflections change' has a low value (black) and the rest of the wall has a medium value (grey). This again, implies that the histogram contains changes

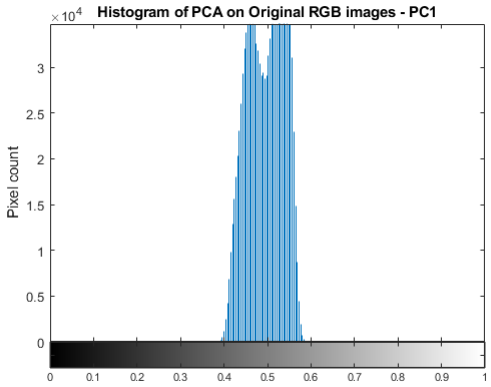
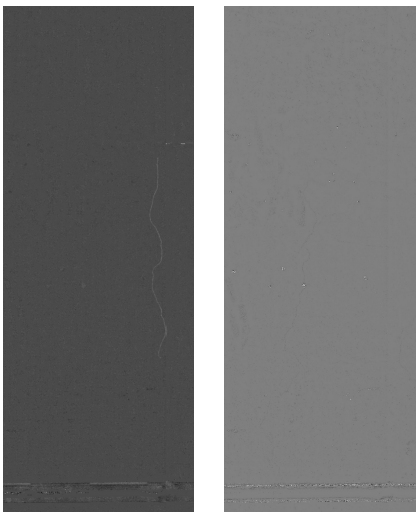


Figure 7.11: Histogram of normalised $PC1$ from PCA on original RGB images



(a) PC_0 (b) PC_1

Figure 7.12: The principal components of the stacked pre-processed images

at both of its tails. In this case, however, as observed in Fig. 7.13, the histogram shape follows a Gaussian trend.

To automatically find a threshold pair, the Statistical Process Control (SPC) principle [224] was used to set up the control limits to distinguish between 'change' and 'no-change' pixels. This involves binarising an image with a range of pixel values away from the mean pixel level where the range is controlled by an input control factor.

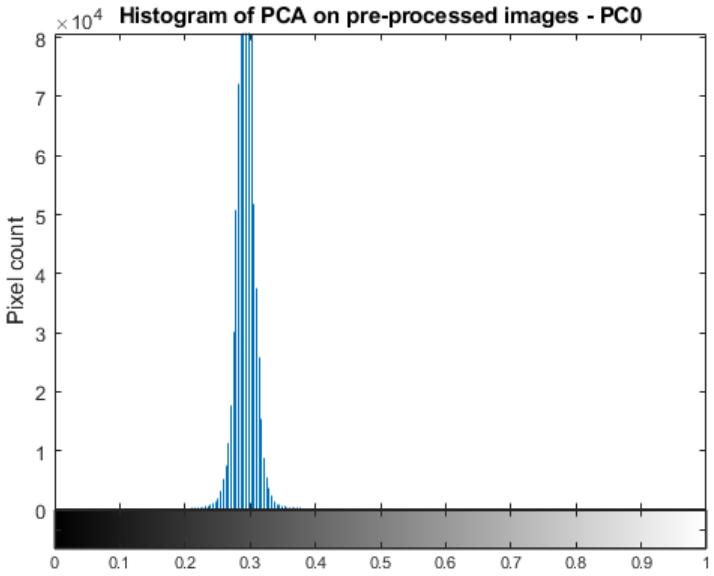


Figure 7.13: Histogram of normalised PC_0 from PCA on pre-processed images

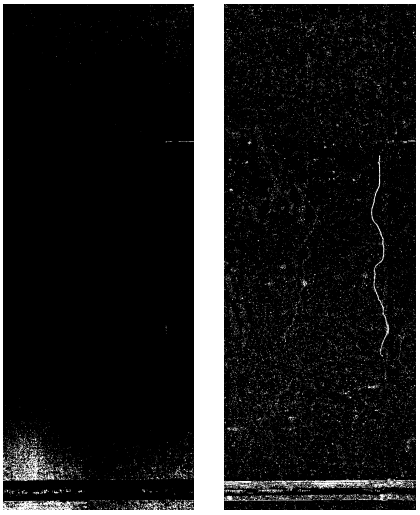
Hence, a double threshold is heuristically determined using:

$$\begin{aligned} T_{low} &= \mu - c\sigma \\ T_{high} &= \mu + c\sigma \end{aligned} \tag{7.9}$$

where μ and σ are the mean and the standard deviation of the PC_i respectively, c is a control constant set empirically.

Upon applying SPC on the PC_1 and PC_0 of the original and pre-processed images respectively, the CMs in Fig. 7.14 were generated following Eq. 7.10. As observed below, the actual real 'crack change' is only identified as a change when the pre-processed images are used.

$$CM_{PCA}(x,y) = \begin{cases} 1 & \text{if } PC_i(x,y) > T_{high} \\ 1 & \text{if } PC_i(x,y) < T_{low} \\ 0 & \text{otherwise} \end{cases} \tag{7.10}$$



(a) PC_1 (b) PC_0

Figure 7.14: Resulting CMs from PCA applied to different images (a) original images (PC_1) (b) pre-processed images (PC_0)

7.3.3 Structural Similarity Index (SSIM)

This is a well-known metric for image quality assessment. It was initially proposed in [225] and then further detailed in [226, 227] to improve upon a similar previous metric Universal Quality Index (UQI), replacing the average weight function by a Gaussian one. To compare two images, SSIM performs three different similarity measurements of luminance (l), contrast (c) and structure (s), and thereafter combines them to obtain a single value as shown in Fig. 7.15.

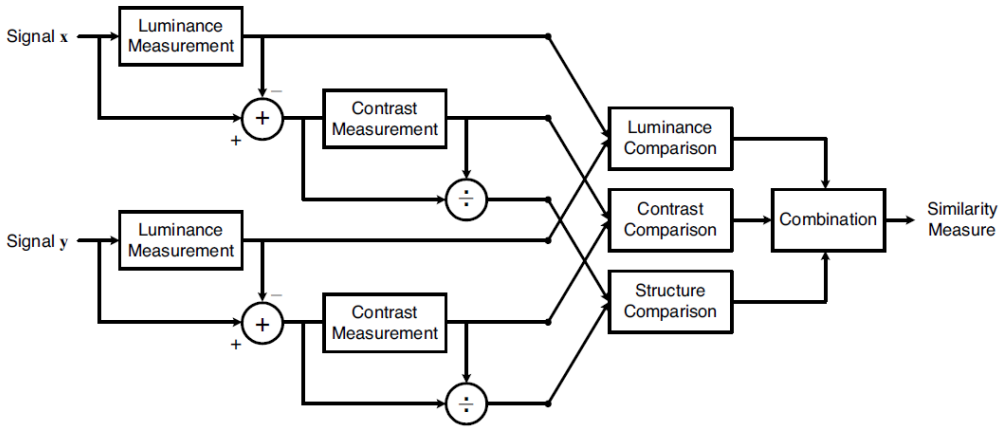


Figure 7.15: Diagram of the SSIM measurement system [226]

Considering two image blocks x and y, the individual comparison functions are defined by:

$$\begin{aligned} l(x,y) &= \frac{2\mu_x\mu_y + c_1}{{\mu_x}^2 + {\mu_y}^2 + c_1} \\ c(x,y) &= \frac{2\sigma_x\sigma_y + c_2}{\sigma_x + \sigma_y + c_2} \\ s(x,y) &= \frac{\sigma_{xy} + c_3}{\sigma_{xy} + c_3} \end{aligned} \tag{7.11}$$

where μ_x, μ_y are the averages and ${\sigma_x}^2$, ${\sigma_y}^2$ are the variances of x and y while σ_{xy} is the covariance between x and y. The constants c_1, c_2, c_3 are calculated using:

$$\begin{aligned} c_1 &= (K_1 L)^2 \\ c_2 &= (K_2 L)^2 \\ c_3 &= \frac{c_2}{2} \end{aligned} \tag{7.12}$$

where $K_1, K_2 \ll 1$, generally $K_1 = 0.01, K_2 = 0.03$ and L is the dynamic range of the pixel values ($L = 255$ for 8-bit greyscale images). SSIM is then a weighted combination of the above as defined by:

$$SSIM(x,y) = [l(x,y)^\alpha \cdot c(x,y)^\beta \cdot s(x,y)^\gamma] \tag{7.13}$$

Setting the weights α, β, γ to 1, the equation is reduced to:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{({\mu_x}^2 + {\mu_y}^2 + c_1)({\sigma_x}^2 + {\sigma_y}^2 + c_2)} \tag{7.14}$$

Apart from image assessment, the SSIM index has been widely used in various applications including image compression [228], image fusion [229], image watermarking [230], video hashing [231], target recognition [232], visual surveillance [233] and remote sensing [234]. Considering that change detection identifies changed ar-

eas by analysing the similarity of multi-temporal images, then, SSIM has a significant prospect in this process. Here, SSIM is used as a PBCD method to generate a CM between a pair of reference and survey images.

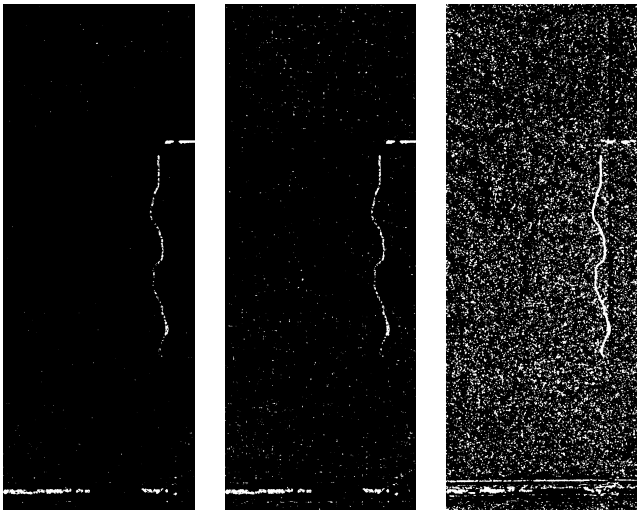
The resultant SSIM index is a value in the range [-1, 1] where 1 is only reached when the images compared are exactly the same, indicating ideal structural similarity. Here, the SSIM is calculated using Eq. 7.15 and its values are shifted to generate a CM with a dynamic range of $[0, 255]$. It is later thresholded using Eq. 7.16.

$$D(x, y) = 1 - \frac{SSIM(x, y) + 1}{2} \tag{7.15}$$

$$CM_{SSIM}(x, y) = \begin{cases} 1 & \text{if D(x,y)} \geq \text{T} \\ 0 & \text{otherwise} \end{cases} \tag{7.16}$$

where $D(x, y)$ represents the difference image and T is a pre-defined threshold. A fixed threshold value cannot however satisfy all scenarios, thus the same automatic thresholding method described in Section 7.3.1 is used with SSIM.

The SSIM formula is commonly applied only on the luma component. Here, an investigation of the performance in change detection is done using greyscale images, the V channel in HSV images and the pre-processed images corrected for uneven illumination. In general, the best results with minimum difference noise were obtained using greyscale images as shown in Fig. 7.16.



(a) greyscale (b) HSV (c) pre-processed

Figure 7.16: Resulting CMs from SSIM applied to the (a) greyscale images (b) V channel in HSV images and (c) pre-processed images

7.4 Decision-level fusion

Analysing the complementary advantages of the three implemented PBCD methods, the generated CMs from image differencing (CM_D), PCA (CM_{PCA}) and SSIM (CM_{SSIM}) are fused into a single CM using decision-level fusion. Different fusion methods including logical operations, PCA and majority voting were implemented.

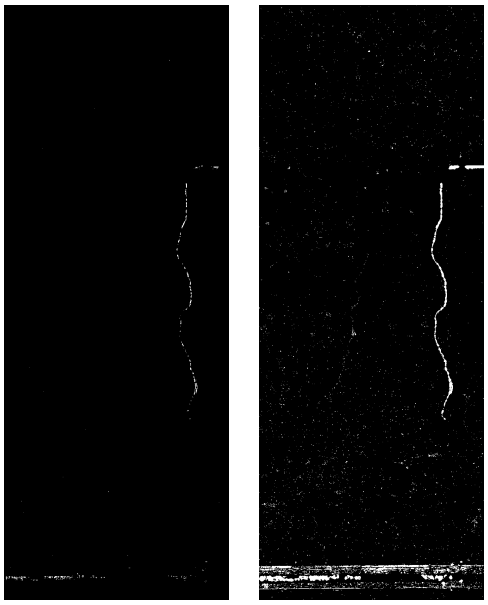
7.4.1 Fusion using logical operations

Logical operators AND and OR are used to combine the CMs using:

$$\begin{aligned} CM_{AND}(x,y) &= CM_D \wedge CM_{PCA} \wedge CM_{SSIM} \\ CM_{OR}(x,y) &= CM_D \vee CM_{PCA} \vee CM_{SSIM} \end{aligned} \tag{7.17}$$

The resulting CM_{AND} combines the presence of change only if confirmed by all the CMs thus, while it reduces noise pixels, some change pixels are ignored if any one of the methods eliminates them. On the other hand, the resulting CM_{OR} combines the presence of change if the pixel is '1' in any of the CMs. Hence, it closes any

gaps which are present in the individual CMs at the expense of having more noise pixels. This can be observed in the example shown in Fig. 7.17.



(a) CM_{AND} (b) CM_{OR}

Figure 7.17: Fusion of CMs using logical operators (a) AND (b) OR

7.4.2 Fusion using PCA-weighted summation

The information flow diagram of the PCA-based fusion algorithm is illustrated in Fig. 7.18. PCA is applied to the three CMs; CM_D, CM_{PCA} and CM_{ssim}. The resulting principal components PC_i are then used as weights multiplied to each of the CMs. A summation of these weighted terms generates the fused CM using:

$$CM_{PCA}(x,y) = CM_D(x,y)\cdot PC_0 + CM_{PCA}(x,y)\cdot PC_1 + CM_{SSIM}(x,y)\cdot PC_2 \quad (7.18)$$

As shown in Fig. 7.19, this method performs better than the logical operators as it generates less noise pixels while at the same time the actual changes, in this case those belonging to the crack, are retained.

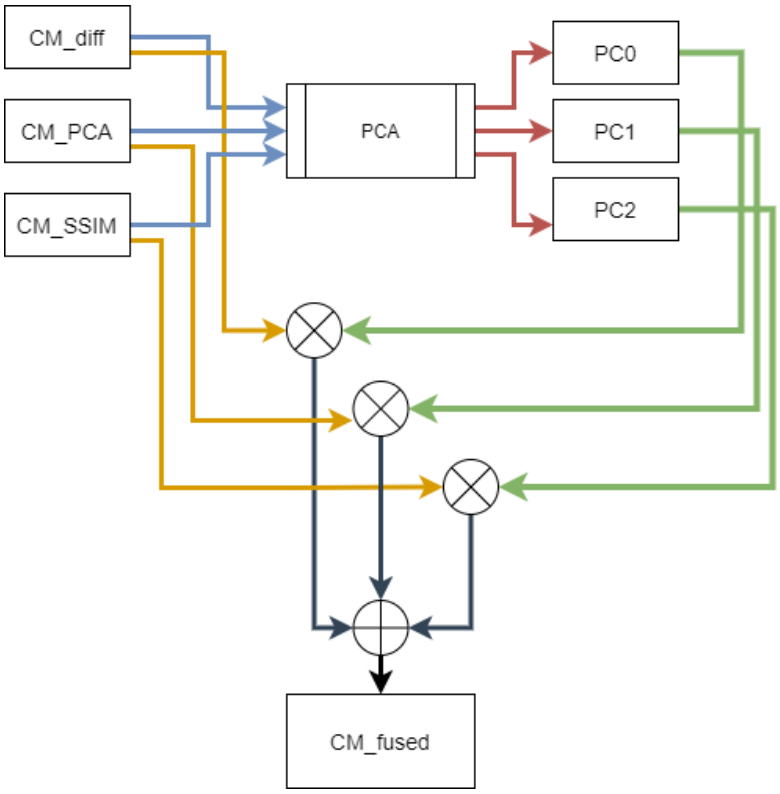


Figure 7.18: Diagram of CM fusion by PCA-weighted summation

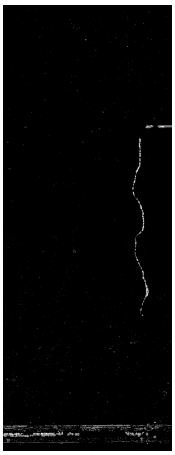


Figure 7.19: CM decision-level fusion by PCA-weighted summation

7.4.3 Fusion using majority voting

This is a basic and simple decision integration method, designed to combine several inputs by multiple processors using specific voting rules. Here, the three different CMs; CM_D, CM_{PCA} and CM_{SSIM} cast a unit vote and if at least two of the CMs register a change, then the corresponding pixel in the fused CM is assigned '1' (change), otherwise '0' (no change). Similar to the previous method, this fusion approach generates only a few noise pixels while retaining the actual changes belonging to the crack.

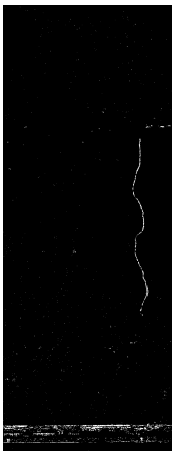


Figure 7.20: CM decision-level fusion by majority voting

7.5 Change map analysis

At this point, the fused CM may still contain 'nuisance change' areas that should not be considered as 'changes'. Hence, the CM analysis process illustrated in Fig. 7.21 was developed. Details of each step are given in the subsequent sections.

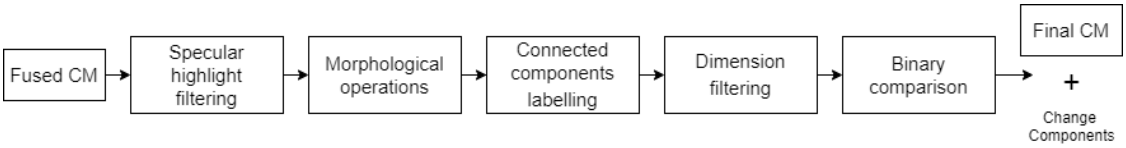


Figure 7.21: CM analysis process

7.5.1 Specular highlight filtering

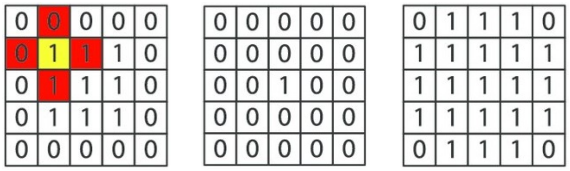
As indicated in Section 7.1.2, those false changes occurring from specular reflections are masked out. Fusion between the binary image $SpecH(x, y)$ and the final CM is done through an AND operation defined by:

$$CM_{filtered}(x, y) = CM_{fused}(x, y) \wedge SpecH(x, y) \tag{7.19}$$

7.5.2 Morphological operations

Furthermore, the $CM_{filtered}(x, y)$ may contain some small 'change areas' coming from image noise and minor registration errors. Such areas are eliminated using binary image enhancement through morphological operations, connected components labeling and filtering by dimensions.

Morphological operations, process every pixel in the image based on the neighbourhood pixel values. Here, to remove 'change noise' and close gaps in the CM, erosion and dilation are respectively applied.



(a) original (b) eroded (c) dilated

Figure 7.22: Concept of morphological operations

Erosion When any of the elements making up the structuring element (red boxes in Fig. 7.22) overlaps the background such that at least one '0' in the image is overlapped, then the pixel on which the origin of the structuring element falls, is set to a value of '0' otherwise set to '1'. Hence, erosion leads to thinning of the original binary image as shown in Fig. 7.22(b).

Dilation This operation is the opposite of erosion. The origin of the structuring element traverses over different locations in the image. When the origin is translated to points such that the structuring element overlaps at least one '1', then the pixel corresponding to the origin of the structuring element is assigned a value of '1'. Otherwise, where the structuring elements do not overlap any '1' then the origin's pixel is assigned a '0'. Consequently, dilation leads to the closing of holes within an image as shown in Fig. 7.22(c).

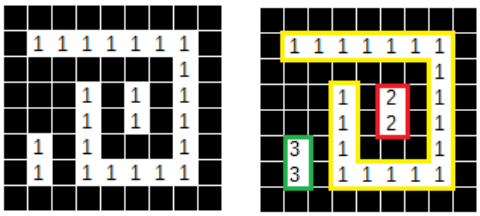
Opening and closing The opening operation successively erodes an image and dilates the resulting eroded image, using the same structuring element for both operations. This is essential to discard objects that are small in dimension while maintaining the size and shape of larger image objects. In contrast, the closing operation dilates an image and then erodes the result, using the same structuring element for both operations. This is used to fill small gaps while maintaining the size and shape of the image objects. Here, morphological closing is applied to the fused CM in order to join any change segments by filling gaps, such as in 'crack changes' while at the same time ignoring the 'noise changes'.

7.5.3 Connected components labelling

This is used to assemble neighbouring pixels in the CM into 'change components'. This process scans the binary image and groups pixels into components depending on pixel connectivity as illustrated in Fig. 7.23. Here, 8-connectivity is used to identify and group neighbouring pixels in the CM.

7.5.4 Dimension filtering

The components are now filtered by size. A 'change component' is only retained if its width or height satisfies a pre-defined threshold. The latter is a configurable constant S_{min} calculated on the actual dimensions captured by the image. Using the GDAL library [235], the scale in the orthophoto raster is obtained. Following



(a) original image (b) labelled image

Figure 7.23: Concept of connected components labelling

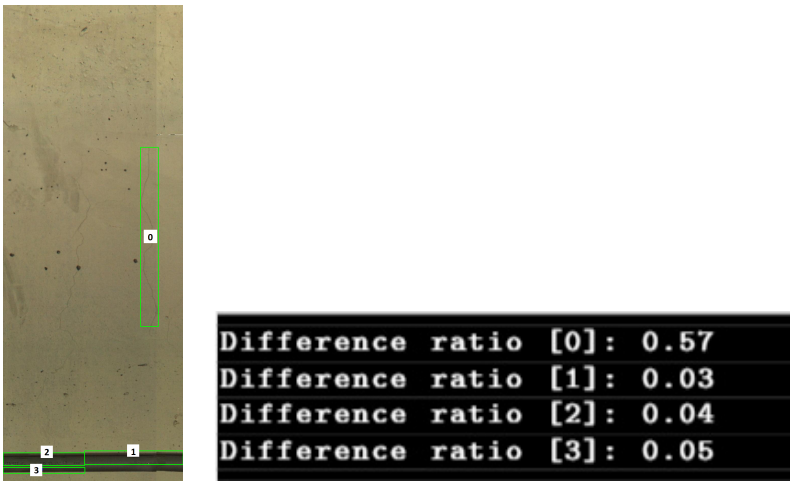
that, using the simple proportion principle, the physical dimensions of the FoV of a photo segment is calculated. Using the latter, the ratio of the photo dimensions and the configurable S_{min}, the size thresholds Th_{min} are calculated using:

$$\begin{aligned} ThW_{min} &= \frac{dim_{min} \times width_{img}}{width_{FOV}} \\ ThH_{min} &= \frac{dim_{min} \times height_{img}}{height_{FOV}} \end{aligned} \tag{7.20}$$

If a candidate 'change component' has a width larger than ThW_{min} and/or a height larger than ThH_{min} then the component is confirmed as a 'change component' otherwise eliminated.

7.5.5 Binary comparison

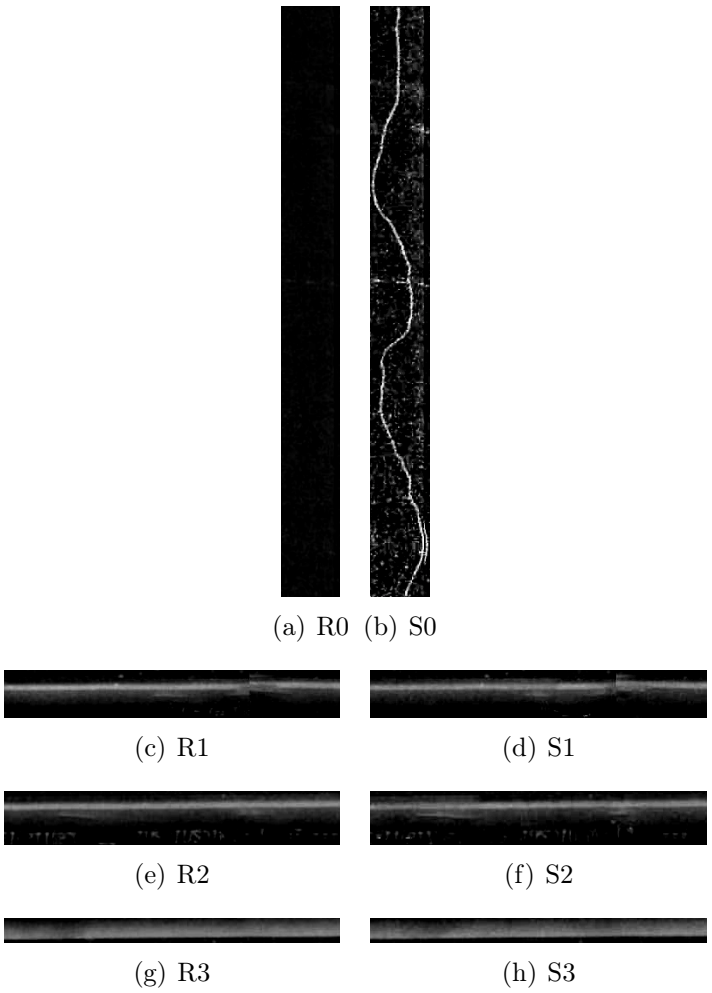
A further analysis is done to reduce false changes due to reflections, shadows and parallax errors. Wall images consist of a white background and darker areas where cracks, marks etc. appear. The images are inverted to generate images with a black background and a foreground of white pixels. Then, the bounding rectangle of each 'change candidate' is masked out for both the reference and survey images using the corresponding area in the CM as a mask. The difference in number of white pixels is divided by the total number of mask pixels.



(a) candidates (b) ratios

Figure 7.24: Change candidates and their difference ratios

Considering the same example, the 'change candidates' in Fig. 7.24(a), generated the difference ratios listed in Fig. 7.24(b) corresponding to the image patches displayed in Fig. 7.25. This shows that the difference ratio for component '0' which is the 'actual change', is much larger than for the others. Thus a threshold is empirically set to filter out the 'false changes'. If the ratio is higher than a threshold, this is considered as a 'change', otherwise ignored such that in this case for example, only 'change candidate 0' is considered as a change.



(a) R0 (b) S0

(c) R1 (d) S1

(e) R2 (f) S2

(g) R3 (h) S3

Figure 7.25: Change candidates (a)-(b) reference and survey patch '0 (c)-(d) reference and survey patch '1' (e)-(f) reference and survey patch '2' (g)-(h) reference and survey patch '3'

7.6 Performance evaluation

To demonstrate the effectiveness of the proposed change detection module, a set of experiments were conducted by simulating different changes such as cracks and other markings on the walls. In addition, some markings were made on the images during post-processing using a graphical editing software. For each test scenario, the areas manually identified as changes were marked with a red dot. The change detection output marked with green boxes and indices, was manually compared

to the corresponding reference-survey image pair. An actual 'change component' was marked as a True Positive (TP). Each actual 'change component' that was not identified by the algorithm was added to the False Negative (FN) list. A region which was falsely marked as a change as it does not relate to any of the actual changes, was recorded as a False Positive (FP). To quantitatively evaluate the performance of the change detection algorithm, the following metrics were used.

7.6.1 Evaluation metrics

In predictive analytics, a confusion matrix as illustrated in Fig. 7.26 is generally used. Usually this matrix also includes an entry for the True Negative (TN)s, which is not applicable in this scenario.

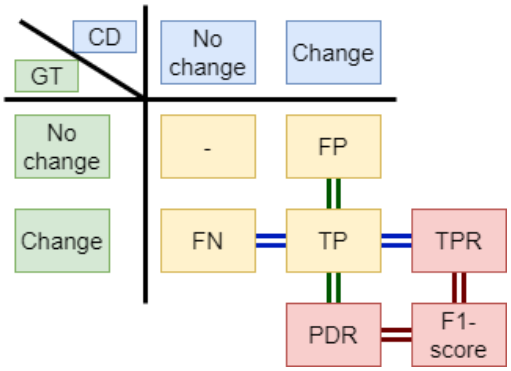


Figure 7.26: Confusion matrix of the ground-truth (GT) and the results from the change detection (CD) algorithm

This matrix allows a more detailed analysis than a mere proportion of correct guesses. Using its values, different metrics are calculated. The recall of the proposed system is calculated using Eq. 7.21. This is the fraction of changes that are actually detected, implying the sensitivity or ability of the system to find the changes. The precision is calculated using Eq. 7.22. This is the ratio of the identified true changes to the total number of changes detected, hence it implies the ability of the system to identify only the actual changes. Furthermore, using both the precision and recall values, the F1-score is calculated using Eq. 7.23. This combines the precision and recall and is useful to find an optimal blend of both.

$$TPR(Recall) = \frac{TP}{TP + FN} \times 100\% \tag{7.21}$$

$$PDR(Precision) = \frac{TP}{TP + FP} \times 100\% \tag{7.22}$$

$$F1{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100\% \tag{7.23}$$

7.6.2 Quantitative results

During the development of the change detection module, different decision-level fusion methods were implemented however, the PCA-weighted summation and majority voting methods resulted in the best CMs. Therefore, subsequent testing was undertaken using these two methods while varying the threshold of the final stage binary comparison.

The quantitative results recorded in Table 7.1 show that the decision-level fusion by PCA-weighted summation generated a higher precision rate. As the threshold of the final binary comparison was increased from 0.1 to 0.2, the precision value increased from 83.03% to 94.5%. When the majority voting approach was used, a precision of 78.84% and 92.99% was achieved at the same thresholds of 0.1 and 0.2 in the final comparison stage. This implies that, the PCA-weighted sum approach distinguished better between actual and nuisance changes.

However, it is also important to evaluate the effectiveness of the algorithm with respect to its ability to find all the data points of interest, in this case the identified changes. This is given by the recall rate, which had higher values of 83.71% and 81.11% for the majority voting approach with binary comparison stage threshold values of 0.1 and 0.2 respectively. This implies that the majority voting approach could identify more actual changes with fewer misses.

It is beneficial if the algorithm can correctly classify the changes, to avoid false alarms, however, it is important that changes due to defects on the tunnel lining

are not missed. Hence, a trade-off between precision and recall is essential. This is found by analysing the $F1-score$ which combines both metrics. As observed in the Table 7.1, the fusion using the majority voting approach achieved a better general performance with respect to the $F1-score$.

Table 7.1: Quantitative results from the change detection algorithm using different decision-level fusion methods; majority voting (MV) and PCA-weighted summation (PCA) with different threshold values for the binary comparison in the change component analysis stage

Method	**Binary TH**	**TP**	**FP**	**FN**	**Recall %**	**Precision %**	**F1-score**
MV	0.1	149	40	29	**83.71**	78.84	81.20
MV	0.2	146	11	34	81.11	92.99	**86.65**
PCA	0.1	137	28	39	77.84	83.03	80.35
PCA	0.2	103	6	73	58.52	**94.50**	72.28

7.6.3 Qualitative results

Further to the quantitative results, a qualitative analysis was made on different scenarios with 'crack changes', other defects and also 'nuisance changes' caused by varying light conditions and shadows.

In the example presented in Fig. 7.27, both of the fusion approaches identified the actual changes correctly. However, the majority voting approach generated a more confined bounding box around the 'crack change' identified by '1'.

Using the reference and survey images in Fig. 7.28, the change detection algorithm using majority voting correctly identified both of the 'crack changes'. On the other hand, the connectivity and binary comparison stages following the PCA-weighted summation method incorrectly identified this as a 'nuisance change' and thus discarded it.

In Fig. 7.29, another 'defect' was simulated on the wall. In this case, both methods correctly identified the change. The final example in Fig. 7.30 only

exhibits 'nuisance changes' with respect to the light. Both CMs show white pixels in different areas in the image, implying possible changes due to specular highlights, shadows and light changes. However, the CM analysis stage ignored most of these regions except for the small shadow area at the bottom of the image when using fusion by the PCA-weighted sum method, generating a 'false change'.

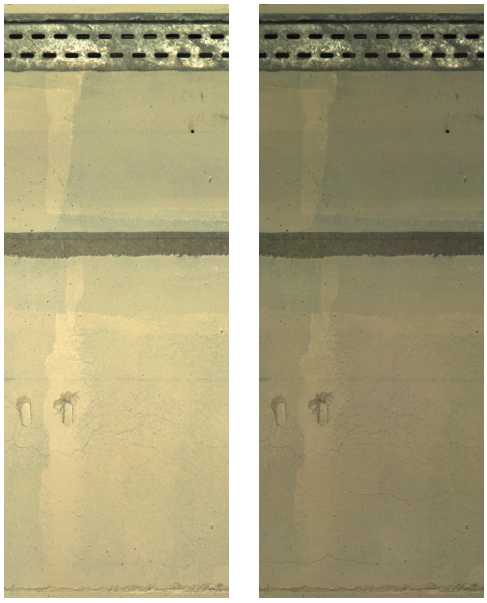
Considering both the quantitative and qualitative results, the final implementation of the proposed solution uses a majority voting approach for the decision-level fusion and a threshold of 0.2 for the final binary comparison stage.

For further examples showing results from the change detection module, the reader is referred to Appendix D.

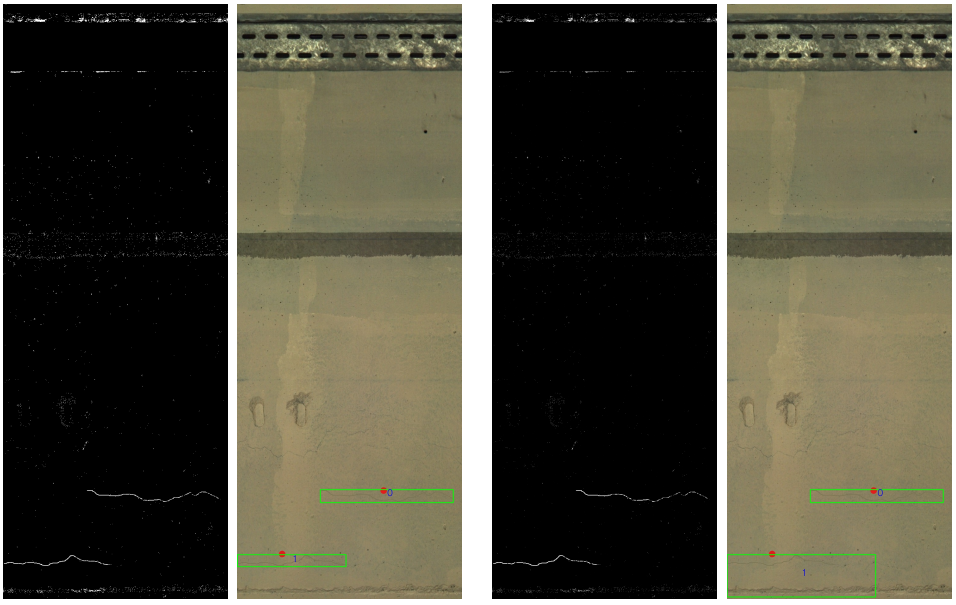
7.7 Contributions summary

The main contributions of this chapter include the:

- study, implementation and analysis of different bi-temporal image fusion techniques for image comparison and change map generation
- implementation and evaluation of two decision-level fusion techniques for robust change detection
- use of image processing for uneven illumination correction and application of specular highlight localisation to provide an illumination-invariant solution

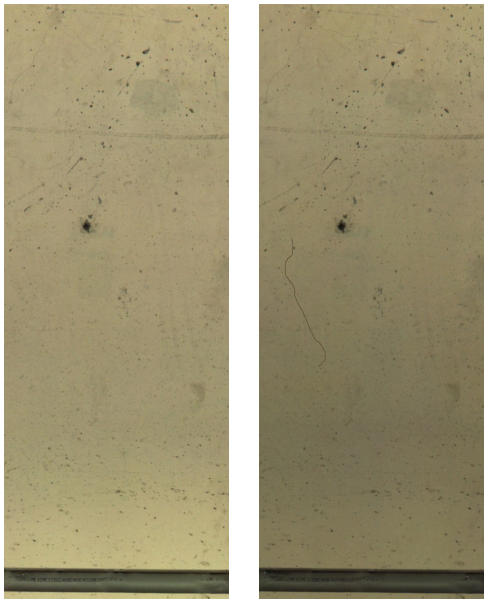


(a) reference (b) survey

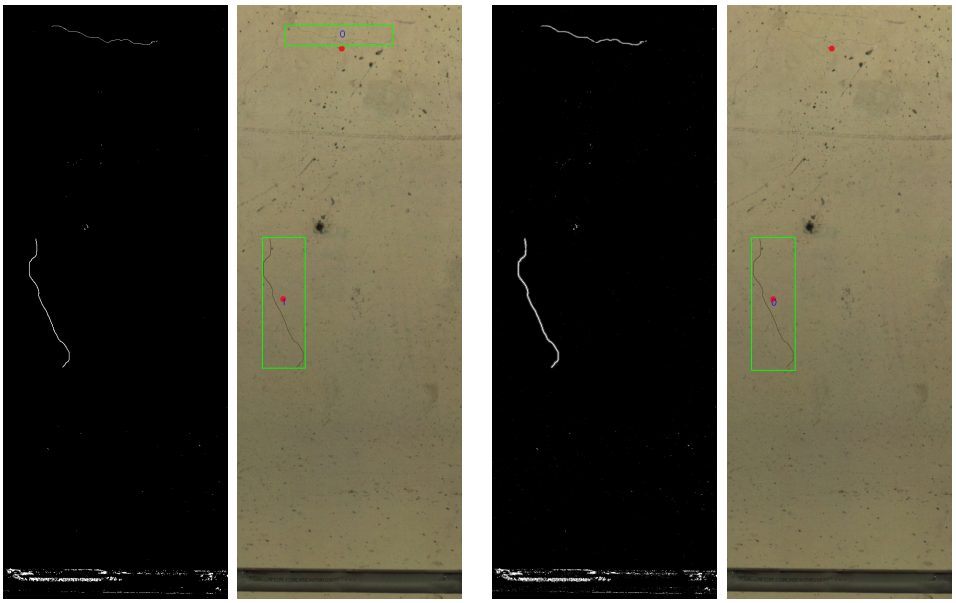


(c) majority voting (d) PCA-weighted summation

Figure 7.27: An example showing similar results for both majority voting and PCA

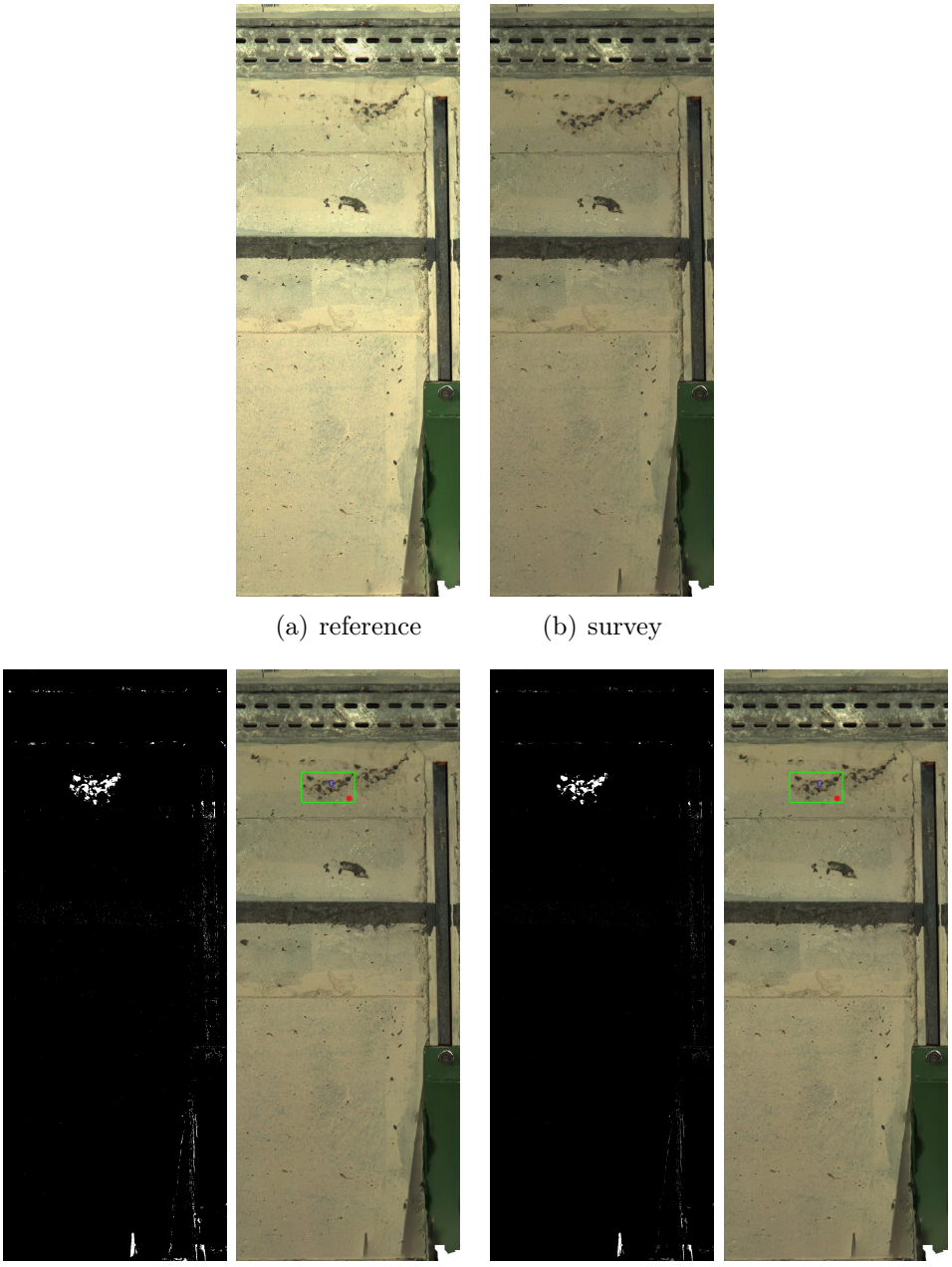


(a) reference (b) survey



(c) majority voting (d) PCA-weighted summation

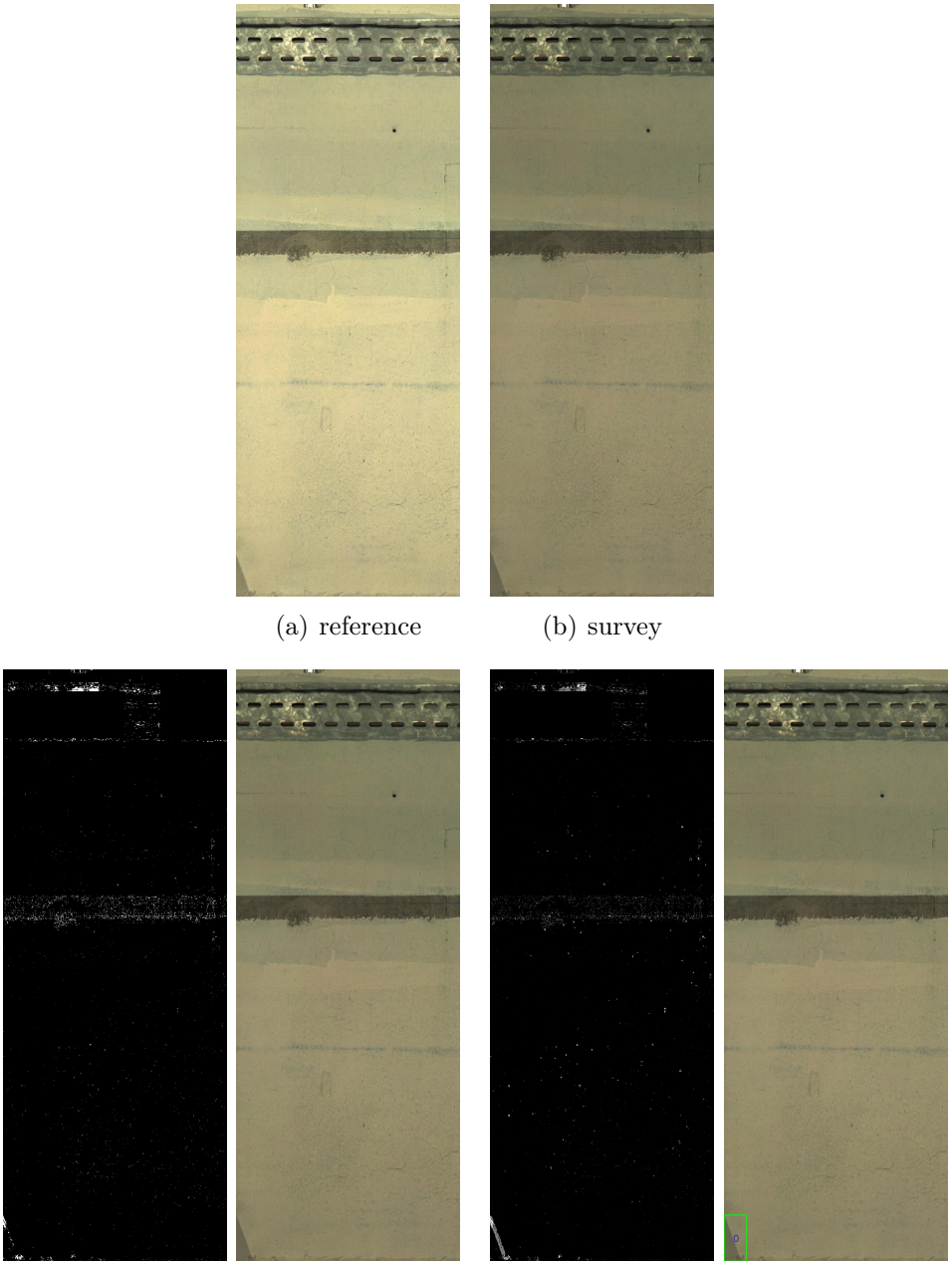
Figure 7.28: An example showing different detection results from majority voting and PCA-weighted summation



(a) reference (b) survey

(c) majority voting (d) PCA-weighted summation

Figure 7.29: An example showing a different simulated defect on the wall whereby the changed area was identified correctly by both methods



(a) reference (b) survey

(c) majority voting (d) PCA-weighted summation

Figure 7.30: An example exhibiting lighting changes, that are correctly identified as a nuisance and not detected as a change

8 | Conclusion and Future work

8.1 Conclusion

Periodic tunnel structural inspections are a necessity. Construction defects, aging, unexpected overloading and natural phenomena may lead to problems in structural integrity over time. These cause failures and possible fatal accidents if not pre-empted in time. Inspections are predominantly performed through visual observations which involve looking for structural defects and making sketches for civil engineers to assess them and in turn suggest the required maintenance and/or repairs. Associated with this method, there are several drawbacks including personnel exposure to hazardous conditions and outcome subjectivity that is highly dependent on human intervention which may lead to inaccuracies or misinterpretations. Furthermore, manual inspections are costly and require downtime to conduct the observations. All this has led to an increase in the need for automatic inspections.

Hence, using robotics, computer vision and data fusion, a tunnel inspection solution to monitor for changes on tunnel linings was proposed in this thesis. The solution comprises data acquisition from a rig of cameras hosted on a robotic platform and the use of computer vision and data fusion techniques to implement automatic tunnel lining monitoring. A study of the different methods for crack detection was carried out and deep learning techniques were employed to devise a crack detection module in order to identify cracks on concrete walls. To alleviate the effects of different light conditions on change detection, pre-processing stages were implemented. These include a shading correction to adjust uneven

illumination and highlights localisation to reduce false changes due to reflections from electronic flash units. Subsequently, a change detection algorithm was developed through a combination of different bi-temporal pixel-based fusion methods and decision-level fusion of CMs. Qualitative evaluation of the resulting CMs was followed by a quantitative analysis indicating high recall and precision values of 81% and 93% respectively.

The proposed solution aids the process of structural health monitoring and provides a better means of tunnel surface documentation. Data acquisition is carried out on-site during shutdowns or short, infrequent maintenance periods while objective inspection through crack and change detection is executed off-site, on a high-performance computer. Although the prime purpose of the system is for deployment in the CERN LHC tunnel, with a few modifications and a different configuration, it could be adapted to other infrastructure monitoring scenarios.

8.2 Summary of contributions

The main contributions of this work in the related fields include the:

- study of a multiple image sensor set up to obtain data for tunnel inspection and the implementation of automatic image data acquisition from off-the shelf commercial cameras;
- study and implementation of crack detection using deep learning techniques;
- development of a change detection algorithm using computer vision techniques to implement different bi-temporal and decision-level image fusion.

8.3 Recommendations for future work

A tunnel goes through three phases; planning, construction, and maintenance. An essential part of its maintenance is the documentation and evaluation of its

structure in general. Thus, the quality requirements in structural health monitoring render the efficient administration and documentation of all data, indispensable. Immediate availability, clear visualisation and presentation of the data are therefore essential for a reliable tunnel inspection system.

Hence, a future improvement on the proposed solution may include the use of overlaid images, 3D models and VR. As illustrated in Fig. 8.1, the generated inspection information can be augmented to a VR model rendering mixed reality as suggested in [236] and [237]. This would further reduce the presence of personnel in the tunnels by providing a means of remote observation allowing familiarisation of the environment before going for a survey, reducing the time spent on-site. Moreover, off-site inspection can also be carried out remotely via the 3D and VR models. Hence, such an addition would be beneficial for wall surface documentation, remote inspection and post-survey analysis.

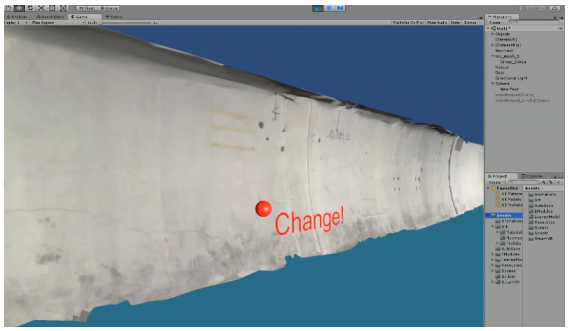


Figure 8.1: Augmentation of inspection findings on a VR model

During this research, an inquiry into the combination of data from multi-modal sensors was carried out. Considering the possible use of a TIR camera for further in-depth inspection, a Linux-based interface that is able to capture images remotely from the FLIR A300 TIR camera was developed. In addition, the actual temperature values are also stored in a Comma-Separated Values (CSV) file for possible future use. Furthermore, ThermoVis, a Graphical User Interface (GUI) based on the C# samples available online [238], was developed. This GUI can be used to connect to the TIR camera by searching the current devices or using

a known IP address and capture images or video sequences. Images can also be saved as temperature values in a CSV file. Moreover, the application can also be used to open saved thermal image and video files. A screenshot of ThermoVis is displayed in Fig. 8.2.

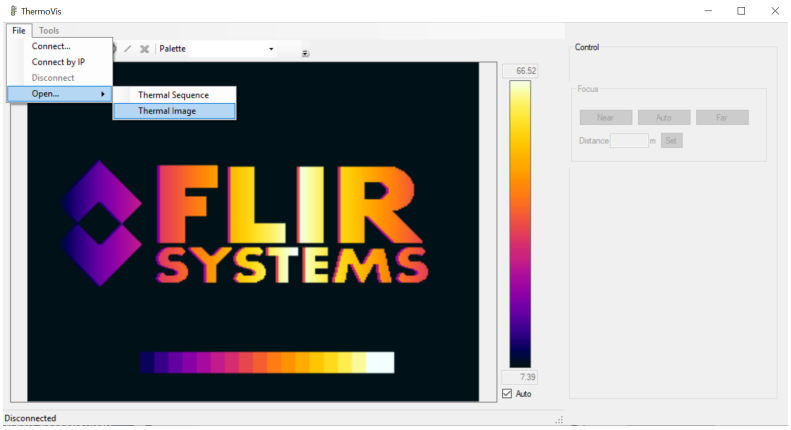


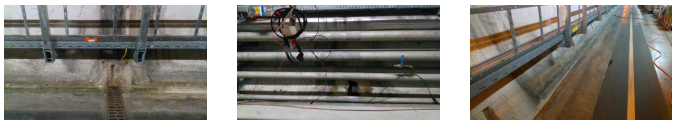
Figure 8.2: A screenshot of ThermoViss, a GUI implemented in C#, to be used with the TIR camera



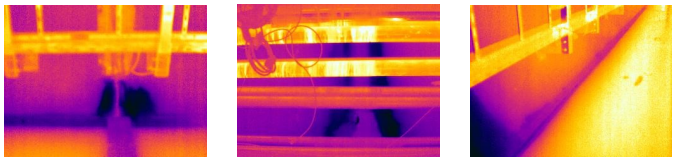
Figure 8.3: Thermal and RGB camera placed on a tripod

A small set of images was also acquired using a TIR and a colour camera fixed on a tripod as shown in Fig 8.3. A few samples are displayed in Fig. 8.4. In addition, a study of multi-modal camera calibration and data fusion methods was conducted. Future investigation into this can benefit from the complimentary properties of the

infrared and visible spectrum towards performing automatic in-depth inspection of cracks, spalling and water deposition areas.



(a) RGB, Scene 1 (b) RGB, Scene 2 (c) RGB, Scene 3



(d) TIR, Scene 1 (e) TIR, Scene 2 (f) TIR, Scene 3

Figure 8.4: A sample of the captured RGB and TIR images

Bibliography

[1] L. Attard, C. J. Debono, G. Valentino, and M. Di Castro, "Vision-based change detection for inspection of tunnel liners," *Automation in Construction*, vol. 91, pp. 142–154, 2018. [Online]. Available: https://doi.org/10.1016/j.autcon.2018.03.020

[2] L. Attard, "A vision-based abnormality detection system in the lhc tunnel," Master's thesis, University of Malta, Feb. 2017.

[3] CERN, "Cern accelerator complex," 2014. [Online]. Available: https://en.wikipedia.org/wiki/CERN

[4] R. Montero, J. Victores, S. Martanez, A. Jardon, and C. Balaguer, "Past, present and future of robotic tunnel inspection," *Automation in Construction*, vol. 59, pp. 99–112, 2015. [Online]. Available: https://doi.org/10.1016/j.autcon.2015.02.003

[5] K. G. Boving, *NDE Handbook*. Butterworth-Heinemann, 1989.

[6] C. Balaguer, R. Montero, J. G. Victores, S. Martínez, and A. Jardón, "Towards fully automated tunnel inspection : A survey and future trends," in *Proceedings of the 31st ISARC, Sydney, Australia*, 2014, pp. 19–33. [Online]. Available: https://doi.org/10.22260/ISARC2014/0005

[7] L. Attard, C. J. Debono, G. Valentino, and M. Di Castro, "Tunnel inspection using photogrammetric techniques and image processing: A review," *ISPRS*

Journal of Photogrammetry and Remote Sensing, vol. 144, pp. 180 – 188, 2018. [Online]. Available: https://doi.org/10.1016/j.isprsjprs.2018.07.010

[8] R. van Gosliga, R. Lindenbergh, and N. Pfeifer, "Deformation analysis of a bored tunnel by means of terrestrial laser scanning," in *Proceedings of the ISPRS Commission V Symposium Image Engineering and Vision Metrology*, 2006.

[9] L. Jian, W. Youchuan, and G. Xianjun, "A new approach for subway tunnel deformation monitoring: high-resolution terrestrial laser scanning," in *Proceedings of the XXII ISPRS Congress*, vol. XXXIX, no. B5, Sep 2012, pp. 223–228.

[10] Z. Kang, L. Tuo, and S. Zlatanovab, "Continuously deformation monitoring of subway tunnel based on terrestrial point clouds," in *Proceedings of the XXII ISPRS Congress*, vol. XXXIX, no. B5, Sep 2012, pp. 199–203.

[11] M. Scaioni, L. Barazzetti, A. Giussani, M. Previtali, F. Roncoroni, and M. Alba, "Photogrammetric techniques for monitoring tunnel deformation," *Earth Science Informatics*, vol. 7, no. 2, pp. 83–95, Mar 2014. [Online]. Available: https://doi.org/10.1007/s12145-014-0152-8

[12] T. T. Wang, J. J. Jaw, C. H. Hsu, and F. S. Jeng, "Profile-image method for measuring tunnel profile - improvements and procedures," *Tunnelling and Underground Space Technology*, vol. 25, no. 1, pp. 78–90, 2010. [Online]. Available: https://doi.org/10.1016/j.tust.2009.09.005

[13] B. Shen, W. Zhang, D. Qi, and X. Wu, "Wireless multimedia sensor network based subway tunnel crack detection method," *International Journal of Distributed Sensor Networks*, vol. 11, no. 6, pp. 1–10, 2015. [Online]. Available: https://doi.org/10.1155/2015/184639

[14] Q. Ai, Y. Yuan, and X. Bi, "Acquiring sectional profile of metro tunnels using charge-coupled device cameras," *Structure and Infrastructure*

Engineering, vol. 12, no. 9, pp. 1065–1075, 2016. [Online]. Available: https://doi.org/10.1080/15732479.2015.1076855

[15] MERMEC, "T-sight 5000," 2014. [Online]. Available: http://www.mermecgroup.com/northamerica/pageview2.php?i=1028&sl=1

[16] A. Mohan and S. Poobal, "Crack detection using image processing: A critical review and analysis," *Alexandria Engineering Journal*, pp. 1–12, 2017. [Online]. Available: https://doi.org/10.1016/j.aej.2017.01.020

[17] P. Wang and H. Huang, "Comparison analysis on present image-based crack detection methods in concrete structures," in *Proceedings of the 3rd International Congress on Image and Signal Processing, CISP*, vol. 5, Oct 2010, pp. 2530–2533.

[18] M. Ukai and N. Nagamine, "A high-performance inspection system of tunnel wall deformation using continuous scan image," *Railway Technical Research Institute*, 2011. [Online]. Available: http://www.railway-research.org/IMG/pdf/poster_ukai_masato.pdf

[19] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, Jan 1979.

[20] D. Qi, Y. Liu, Q. Gu, and F. Zheng, "An algorithm to detect the crack in the tunnel based on the image processing," *Journal of Computers*, vol. 26, no. 3, pp. 860–863, Oct 2014. [Online]. Available: http://www.csroc.org.tw/journal/JOC26_3/JOC26_3_2.pdf

[21] H. Huang, Y. Sun, Y. Xue, and F. Wang, "Inspection equipment study for subway tunnel defects by grey-scale image processing," *Advanced Engineering Informatics Journal*, vol. 32, pp. 188–201, 2017. [Online]. Available: https://doi.org/10.1016/j.aei.2017.03.003

[22] T. Yu, A. Zhu, and Y. Chen, "Efficient crack detection method for tunnel lining surface cracks based on infrared images," *Journal of Computing in Civil Engineering*, vol. 31, p. 04016067, 11 2016. [Online]. Available: https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29CP.1943-5487.0000645

[23] A. Ito, Y. Aoki, and S. Hashimoto, "Accurate extraction and measurement of fine cracks from concrete block surface image," in *Proceedings of the IEEE 28th Annual Conference of the Industrial Electronics Society. IECON 02*, vol. 3, Nov 2002, pp. 2202–2207.

[24] D. Hu, T. Tian, H. Yang, S. Xu, and X. Wang, "Wall crack detection based on image processing," in *Proceedings of the Third International Conference on Intelligent Control and Information Processing*, Jul 2012, pp. 597–600.

[25] Y. Fujita and Y. Hamamoto, "A robust automatic crack detection method from noisy concrete surfaces," *Machine Vision and Applications*, vol. 22, no. 2, pp. 245–254, 2011. [Online]. Available: https://doi.org/10.1007/s00138-009-0244-5

[26] T. Su, "Application of computer vision to crack detection of concrete structure," *International journal of engineering and technology*, vol. 5, no. 4, pp. 457–461, 2013. [Online]. Available: https://doi.org/10.7763/IJET.2014.V5.596

[27] B. Lee, Y. Y. Kim, S. Yi, and J. Kim, "Automated image processing technique for detecting and analysing concrete surface cracks," *Structure and Infrastructure Engineering*, vol. 9, no. 6, pp. 567–577, 2013. [Online]. Available: 10.1080/15732479.2011.593891

[28] S. Dorafshan and M. Maguire, "Automatic surface crack detection in concrete structures using otsu thresholding and morphological operations," *Utah State University CEE Faculty Publications*, 2016. [Online]. Available: https://doi.org/10.13140/RG.2.2.34024.47363

[29] M. Ayaho, K. Masa-Aki, and B. Eugen, "Automatic crack recognition system for concrete structures using image processing approach," *Asian Journal of Information Technology*, vol. 5, pp. 553–561, 2007. [Online]. Available: http://docsdrive.com/pdfs/medwelljournals/ajit/2007/553-561.pdf

[30] K. Y. Song, M. Petrou, and J. Kittler, "Texture crack detection," *Machine Vision and Applications*, vol. 8, no. 1, pp. 63–75, Jan 1995. [Online]. Available: https://doi.org/10.1007/BF01213639

[31] R. Medina, J. Llamas, J. Gómez-García-Bermejo, E. Zalama, and M. Segarra, "Crack detection in concrete tunnels using a gabor filter invariant to rotation," *Sensors (Basel, Switzerland)*, vol. 17, no. 7, pp. 1–16, 2017. [Online]. Available: https://doi.org/10.3390/s17071670

[32] M. Salman, S. Mathavan, K. Kamal, and M. Rahman, "Pavement crack detection using the gabor filter," in *Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, Oct 2013, pp. 2039–2044.

[33] W. Xu, Z. Tang, J. Zhou, and J. Ding, "Pavement crack detection based on saliency and statistical features," in *Proceedings of the 2013 IEEE International Conference on Image Processing*, Sep 2013, pp. 4093–4097. [Online]. Available: https://doi.org/10.1109/ICIP.2013.6738843

[34] P. Subirats, J. Dumoulin, V. Legeay, and D. Barba, "Automation of pavement surface crack detection using the continuous wavelet transform," in *Proceedings of the 2006 International Conference on Image Processing*, Oct 2006, pp. 3037–3040. [Online]. Available: https://doi.org/10.1109/ICIP.2006.313007

[35] W. Zhang, Z. Zhang, D. Qi, and Y. Liu, "Automatic crack detection and classification method for subway tunnel safety monitoring," *Sensors*,

vol. 14, no. 10, pp. 19 307–19 328, 2014. [Online]. Available: https://doi.org/10.3390/s141019307

[36] L. Weiguo, L. Yaru, and W. Fang, "Crack detection based on support vector data description," in *Proceedings of the 29th Chinese Control and Decision Conference (CCDC)*, May 2017, pp. 1033–1038.

[37] H. Oliveira and P. L. Correia, "Automatic road crack detection and characterization," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 155–168, March 2013. [Online]. Available: https://doi.org/10.1109/TITS.2012.2208630

[38] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, and P. Fieguth, "A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure," *Advanced Engineering Informatics*, vol. 29, no. 2, pp. 196 – 210, 2015, infrastructure Computer Vision. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474034615000208

[39] A. Cord and S. Chambon, "Automatic road defect detection by textural pattern recognition based on adaboost," *Computer-Aided Civil and Infrastructure Engineering*, vol. 27, no. 4, pp. 244–259, 2012. [Online]. Available: https://doi.org/10.1111/j.1467-8667.2011.00736.x

[40] Y. Shi, L. Cui, Z. Qi, F. Meng, and Z. Chen, "Automatic road crack detection using random structured forests," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 12, pp. 3434–3445, Dec 2016. [Online]. Available: https://doi.org/10.1109/TITS.2016.2552248

[41] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, and P. Fieguth, "A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure," *Advanced Engineering*

Informatics, vol. 29, no. 2, pp. 196–210, 2015. [Online]. Available: https://doi.org/10.1016/j.aei.2015.01.008

[42] Y.-J. Cha, W. Choi, and O. Büyüköztürk, "Deep learning-based crack damage detection using convolutional neural networks," *Computer-Aided Civil and Infrastructure Engineering*, vol. 32, no. 5, pp. 361–378, 2017.

[43] W. R. L. d. Silva and D. S. d. Lucena, "Concrete cracks detection based on deep learning image classification," *Sensors(Basel, Switzerland)*, vol. 2, no. 8, 2018. [Online]. Available: https://doi.org/10.3390/ICEM18-05387

[44] L. Zhang, F. Yang, Y. Daniel Zhang, and Y. J. Zhu, "Road crack detection using deep convolutional neural network," in *Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP)*, Sep. 2016, pp. 3708–3712. [Online]. Available: https://doi.org/10.1109/ICIP.2016.7533052

[45] A. Zhang, K. C. P. Wang, Y. Fei, Y. Liu, C. Chen, G. Yang, J. Q. Li, E. Yang, and S. Qiu, "Automated pixel-level pavement crack detection on 3d asphalt surfaces with a recurrent neural network," *Computer-Aided Civil and Infrastructure Engineering*, vol. 34, no. 3, pp. 213–229.

[46] K. Makantasis, E. Protopapadakis, A. Doulamis, N. Doulamis, and C. Loupos, "Deep convolutional neural networks for efficient vision based tunnel inspection," in *Proceedings of the IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, Sept 2015, pp. 335–342.

[47] S. Li and X. Zhao, "Image-based concrete crack detection using convolutional neural network and exhaustive search technique," *Advances in Civil Engineering*, vol. 2019, pp. 1–12, 04 2019. [Online]. Available: https://doi.org/10.1155/2019/6520620

[48] H. Xu, X. Su, Y. Wang, H. Cai, K. Cui, and X. Chen, "Automatic bridge crack detection using a convolutional neural network," *Applied Sciences*, vol. 9, p. 2867, 07 2019. [Online]. Available: https://doi.org/10.3390/app9142867

[49] H. Huang, Q. Li, and D. Zhang, "Deep learning based image recognition for crack and leakage defects of metro shield tunnel," *Tunnelling and Underground Space Technology*, vol. 77, pp. 166 – 176, 2018. [Online]. Available: https://doi.org/10.1016/j.tust.2018.04.002

[50] C. V. Dung and L. D. Anh, "Autonomous concrete crack detection using deep fully convolutional neural network," *Automation in Construction*, vol. 99, pp. 52 – 58, 2019. [Online]. Available: https://doi.org/10.1016/j.autcon.2018.11.028

[51] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.

[52] Z. Liu, Y. Cao, Y. Wang, and W. Wang, "Computer vision-based concrete crack detection using u-net fully convolutional networks," *Automation in Construction*, vol. 104, pp. 129–139, 2019. [Online]. Available: https://doi.org/10.1016/j.autcon.2019.04.005

[53] J. Cheng, W. Xiong, W. Chen, Y. Gu, and Y. Li, "Pixel-level crack detection using u-net," in *TENCON 2018 - 2018 IEEE Region 10 Conference*, Oct 2018, pp. 0462–0466.

[54] J. Ji, L. Wu, Z. Chen, J. Yu, P. Lin, and S. Cheng, "Automated pixel-level surface crack detection using u-net," in *Multi-disciplinary Trends in Artificial Intelligence*, M. Kaenampornpan, R. Malaka, D. D. Nguyen, and N. Schwind, Eds. Cham: Springer International Publishing, 2018, pp. 69–78.

[55] Q. Zou, Z. Zhang, Q. Li, X. Qi, Q. Wang, and S. Wang, "Deepcrack: Learning hierarchical convolutional features for crack detection," *IEEE Transactions on Image Processing*, vol. 28, no. 3, pp. 1498–1512, March 2019.

[56] C. Song, L. Wu, Z. Chen, H. Zhou, P. Lin, S. Cheng, and Z. Wu, "Pixel-level crack detection in images using segnet," in *Multi-disciplinary Trends in Artificial Intelligence*, R. Chamchong and K. W. Wong, Eds. Cham: Springer International Publishing, 2019, pp. 247–254.

[57] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, Dec 2017.

[58] C. Pohl and J. L. V. Genderen, "Review article multisensor image fusion in remote sensing: Concepts, methods and applications," *International Journal of Remote Sensing*, vol. 19, no. 5, pp. 823–854, 1998. [Online]. Available: 10.1080/014311698215748

[59] G. Simone, A. Farina, F. Morabito, S. Serpico, and L. Bruzzone, "Image fusion techniques for remote sensing applications," *Information Fusion*, vol. 3, no. 1, pp. 3 – 15, 2002. [Online]. Available: https://doi.org/10.1016/S1566-2535(01)00056-2

[60] S. Yenkanchi, "Multi sensor data fusion for autonomous vehicles," *University of Windsor Electronic Theses and Dissertations*, 2016.

[61] J. Kocic, N. Jovičić, and V. Drndarevic, "Sensors and sensor fusion in autonomous vehicles," in *Proceedings of the 26th Telecommunications Forum (TELFOR)*, 11 2018, pp. 420–425. [Online]. Available: 10.1109/TELFOR.2018.8612054

[62] J. C. Zapata, C. M. Duque, Y. Rojas-Idarraga, M. E. Gonzalez, J. A. Guzmán, and M. A. Becerra Botero, "Data fusion applied to biometric identification – a review," in *Advances in Computing*, A. Solano and H. Ordoñez, Eds. Cham: Springer International Publishing, 2017, pp. 721–733.

[63] J. Soh, F. Deravi, A. Triglia, and A. Bazin, *Multibiometrics and Data Fusion, Standardization.* Boston, MA: Springer US, 2009, pp. 973–980. [Online]. Available: https://doi.org/10.1007/978-0-387-73003-5_229

[64] M. Faundez-Zanuy, "Data fusion in biometrics," *IEEE Aerospace and Electronic Systems Magazine*, vol. 20, no. 1, pp. 34–38, Jan 2005.

[65] Y. Chen, J. Shu, S. Zhang, L. Liu, and L. Sun, "Data fusion in wireless sensor networks," in *2009 Second International Symposium on Electronic Commerce and Security*, vol. 2, May 2009, pp. 504–509.

[66] D. Izadi, J. Abawajy, S. Ghanavati, and T. Herawan, "A data fusion method in wireless sensor networks," *Sensors*, vol. 15, pp. 2964–2979, 02 2015. [Online]. Available: 10.3390/s150202964

[67] M. Koupaee and M. Reza, "Data fusion techniques in wireless sensor networks: Structured vs. structure-free approaches," *Journal of Networking Technology*, vol. 9, p. 41, 06 2018. [Online]. Available: 10.6025/jnt/2018/9/2/41-47

[68] D. Novak and R. Riener, "A survey of sensor fusion methods in wearable robotics," *Robotics and Autonomous Systems*, vol. 73, pp. 155 – 170, 2015, wearable Robotics. [Online]. Available: https://doi.org/10.1016/j.robot.2014.08.012

[69] K. Nagla, M. Uddin, and D. Singh, "Multisensor data fusion and integration for mobile robots: A review," *IAES International Journal of Robotics and Automation (IJRA)*, vol. 3, 09 2014. [Online]. Available: 10.11591/ijra.v3i2.4075

[70] S. B. Ayed, H. Trichili, and A. M. Alimi, "Data fusion architectures: A survey and comparison," in *Proceedings of the 15th International Conference on Intelligent Systems Design and Applications (ISDA)*, Dec 2015, pp. 277–282.

[71] E. Azimirad, J. Haddadnia, and A. Izadipour, "A comprehensive review of the multi-sensor data fusion architectures," *Journal of Theoretical and Applied Information Technology*, vol. 71, no. 1, pp. 33–42, 2015.

[72] C. L. Bowman and C. L. Morefield, "Multisensor fusion of target attributes and kinematics," in *Proceedings of the 19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, Dec 1980, pp. 837–839.

[73] R. C. Luo and M. G. Kay, "Multisensor integration and fusion in intelligent systems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, no. 5, pp. 901–931, Sep 1989.

[74] P. L. F., "Behavioral knowledge in sensor/data fusion systems," *Journal of Robotic Systems*, vol. 7, no. 3, pp. 295–308, 1992. [Online]. Available: https://doi.org/10.1002/rob.4620070303

[75] W. Elmenreich, "A review on system architectures for sensor fusion applications," in *Software Technologies for Embedded and Ubiquitous Systems*, R. Obermaisser, Y. Nah, P. Puschner, and F. J. Rammig, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 547–559.

[76] W. Elmenreich, "Sensor fusion in time-triggered systems," 2002.

[77] S. Masood, M. Sharif, Y. Mussarat, M. Shahid, and A. Rehman, "Image fusion methods: A survey," *Journal of Engineering Science and Technology Review*, vol. 10, pp. 186–195, 12 2017. [Online]. Available: 10.25103/jestr.106.24

[78] A. A. Goshtasby and S. G. Nikolov, "Image fusion: Advances in the state of the art," *Information Fusion*, vol. 8, pp. 114–118, 2007.

[79] D. Wu, A. Yang, L. Zhu, and C. Zhang, "Survey of multi-sensor image fusion," in *Life System Modeling and Simulation*, S. Ma, L. Jia, X. Li, L. Wang, H. Zhou, and X. Sun, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 358–367.

[80] L. Yan, L. Fei, C. Chen, Z. Ye, and R. Zhu, "A multi-view dense image matching method for high-resolution aerial imagery based on a graph network," *Remote Sensing*, vol. 8, no. 10, 2016. [Online]. Available: https://doi.org/10.3390/rs8100799

[81] A. Kuhn, H. Hirschmüller, and H. Mayer, "Multi-resolution range data fusion for multi-view stereo reconstruction," in *Pattern Recognition*, J. Weickert, M. Hein, and B. Schiele, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 41–50.

[82] M. Ylimki, J. Kannala, and J. Heikkil, "Accurate 3-d reconstruction with rgb-d cameras using depth map fusion and pose refinement," *CoRR*, vol. abs/1804.08912, 2018.

[83] T. Adali, C. Jutten, and L. K. Hansen, "Multimodal data fusion [scanning the issue]," *Proceedings of the IEEE*, vol. 103, no. 9, pp. 1445–1448, Sept 2015.

[84] D. Lahat, T. Adali, and C. Jutten, "Multimodal data fusion: An overview of methods, challenges, and prospects," *Proceedings of the IEEE*, vol. 103, no. 9, pp. 1449–1477, Sept 2015.

[85] L. Gómez-Chova, D. Tuia, G. Moser, and G. Camps-Valls, "Multimodal classification of remote sensing images: A review and future directions," *Proceedings of the IEEE*, vol. 103, no. 9, pp. 1560–1584, Sept 2015.

[86] A. K. Katsaggelos, S. Bahaadini, and R. Molina, "Audiovisual fusion: Challenges and new approaches," *Proceedings of the IEEE*, vol. 103, no. 9, pp. 1635–1653, Sep 2015.

[87] J. Ma, Y. Ma, and C. Li, "Infrared and visible image fusion methods and applications: A survey," *Information Fusion*, vol. 45, pp. 153 – 178, 2019. [Online]. Available: https://doi.org/10.1016/j.inffus.2018.02.004

[88] F. Bovolo and L. Bruzzone, "The time variable in data fusion: A change detection perspective," *IEEE Geoscience and Remote Sensing Magazine*, vol. 3, no. 3, pp. 8–26, Sep. 2015.

[89] J. Jan, *Medical Image Processing, Reconstruction and Restoration: Concepts and Methods*. Taylor & Francis Group, LLC, 2006, pp. 481–482.

[90] F. Abdullah, A. Wassai, N. V. Kalyankar, and A. A. A. Zuky, "The IHS transformations based image fusion," *International Journal of Advanced Research in Computer Science*, vol. 2, no. 5, 2011. [Online]. Available: http://arxiv.org/abs/1107.4396

[91] H. Jing and T. Vladimirova, "Novel pca based pixel-level multi-focus image fusion algorithm," in *Proceedings of NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, 2014, pp. 135–142. [Online]. Available: https://doi.org/10.1109/AHS.2014.6880169

[92] D. K. Sahu and M.P.Parsai, "Different image fusion techniques – a critical review," *International Journal of Modern Engineering Research (IJMER)*, vol. 2, no. 5, pp. 4298–4301, 2012.

[93] M. M. Mistry and B. Vala, "Survey on image fusion techniques," *International Journal of Engineering Research and General Science*, vol. 3, no. 3, pp. 933–936, 2015.

[94] R. Suthakar, J. Esther, D.Annapoorani, and R. S. Samuel, "Study of image fusion - techniques, method and applications," *International Journal of Computer Science and Mobile Computing*, vol. 3, no. 11, pp. 469–476, 2014.

[95] K. C. Rajini and S. Roopa, "A review on recent improved image fusion techniques," in *Proccedings of the International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, March 2017, pp. 149–153.

[96] M. Narsaiah, S. Vathsal, and D. V. Reddy, "A survey on image fusion requirements, techniques, evaluation metrics, and its applications," *International Journal of Engineering & Technology*, vol. 7, no. 2.20, pp. 260–266, 2018.

[97] L. Song, Y. Lin, W. Feng, and M. Zhao, "A novel automatic weighted image fusion algorithm," in *2009 International Workshop on Intelligent Systems and Applications*, May 2009, pp. 1–4.

[98] J. S. Deng, K. Wang, Y. H. Deng, and G. J. Qi, "Pca-based land-use change detection and analysis using multitemporal and multisensor satellite data," *International Journal of Remote Sensing*, vol. 29, no. 16, pp. 4823–4838, 2008. [Online]. Available: https://doi.org/10.1080/01431160801950162

[99] H. R. Shahdoosti and H. Ghassemian, "Spatial pca as a new method for image fusion," in *Proceedings of the 16th CSI International Symposium on Artificial Intelligence and Signal Processing (AISP 2012)*, May 2012, pp. 90–94.

[100] V. Naidu and J. Raol, "Pixel-level image fusion using wavelets and principal component analysis," *Defence Science Journal*, vol. 58, no. 3, pp. 338–352, 2008. [Online]. Available: http://dx.doi.org/10.14429/dsj.58.1653

[101] U. Patil and U. Mudengudi, "Image fusion using hierarchical pca," in *2011 International Conference on Image Information Processing*, Nov 2011, pp. 1–6.

[102] J. Wu, H. Huang, J. Liu, and J. Tian, "Remote sensing image data fusion based on IHS and local deviation of wavelet transformation," in *2004 IEEE International Conference on Robotics and Biomimetics*, Aug 2004, pp. 564–568.

[103] M. Choi, "A new intensity-hue-saturation fusion approach to image fusion with a tradeoff parameter," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 44, no. 6, pp. 1672–1682, June 2006.

[104] J. Harris, R. Murray, and T. Hirose, "IHS transform for the integration of radar imagery with other remotely sensed data," *Photogrammetric Engineering and Remote Sensing*, vol. 56, no. 12, pp. 1631–1641, 1990.

[105] R. Gharbia, A. H. El Baz, A. E. Hassanien, and M. F. Tolba, "Remote sensing image fusion approach based on brovey and wavelets transforms," in *Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014*, P. Kömer, A. Abraham, and V. Snášel, Eds. Cham: Springer International Publishing, 2014, pp. 311–321.

[106] H. Olkkonen and P. Pesola, "Gaussian pyramid wavelet transform for multiresolution analysis of images," *Graphical Models and Image Processing*, vol. 58, no. 4, pp. 394 – 398, 1996. [Online]. Available: https://doi.org/10.1006/gmip.1996.0032

[107] P. Burt and E. Adelson, "The laplacian pyramid as a compact image code," *IEEE Transactions on Communications*, vol. 31, no. 4, pp. 532–540, Apr 1983.

[108] A. Toet, "Image fusion by a ratio of low-pass pyramid," *Pattern Recognition Letters*, vol. 9, no. 4, pp. 245 – 253, 1989. [Online]. Available: https://doi.org/10.1016/0167-8655(89)90003-2

[109] U. Kumar, N. Gopaliya, U. Sharma, and S. Gupta, "Discrete transform based image fusion: A review," *Int. J. Multimed. Data Eng. Manag.*, vol. 8, no. 2, pp. 43–49, Apr. 2017. [Online]. Available: 10.4018/IJMDEM.2017040105

[110] R. Singh and A. Khare, "Multiscale medical image fusion in wavelet domain," *The Scientific World Journal*, vol. 2013, no. 521034, 2013. [Online]. Available: http://dx.doi.org/10.1155/2013/521034

[111] H. B. Kekre, A. Athawale, and D. Sadavarti, "Algorithm to generate wavelet transform from an orthogonal transform," *nternational Journal of Image Processing (IJIP)*, vol. 4, no. 4, 2010.

[112] H. B. Kekre, T. Sarode, and R. Dhannawat, "Implementation and comparison of different transform techniques using kekres wavelet transform for image fusion," *International Journal of Computer Applications*, vol. 44, no. 10, pp. 41–48, April 2012.

[113] V. Naidu, "Discrete cosine transform-based image fusion," *Defence Science Journal*, vol. 60, pp. 48–54, 01 2010.

[114] M. K. N. Tania Sultana, Md. Dulal Hossain, "Analysis on swt based image fusion techniques using intuitionistic fuzzy set operations," *International Journal of Technology Enhancements and Emerging Engineering Research*, vol. 4, no. 8, pp. 16–19, 2016.

[115] X. Jin, Q. Jiang, S. Yao, D. Zhou, R. Nie, S.-J. Lee, and K. He, "Infrared and visual image fusion method based on discrete cosine transform and local spatial frequency in discrete stationary wavelet transform domain," *Infrared Physics & Technology*, vol. 88, pp. 1–12, 2018. [Online]. Available: https://doi.org/10.1016/j.infrared.2017.10.004

[116] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image change detection algorithms: a systematic survey," *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 294–307, March 2005.

[117] D. Lu, P. Mausel, E. Brondizio, and E. Moran, "Change detection techniques," *International Journal of Remote Sensing*, vol. 25, no. 12, pp. 2365–2401, 2004. [Online]. Available: https://doi.org/10.1080/0143116031000139863

[118] R. Devi and D. Jiji, "Change detection techniques - a survey," *ternational Journal on Computational Science & Applications*, vol. 5, no. 2, pp. 45–57, 2015. [Online]. Available: 10.5121/ijcsa.2015.5205

[119] S. Singh, A. S. Mandal, C. Shekhar, and A. Vohra, "Real-time implementation of change detection for automated video surveillance system," *International Scholarly Research Notices - Electronics*, vol. 2013, no. 691930, 2013. [Online]. Available: http://dx.doi.org/10.1155/2013/691930

[120] B. A. Akram, A. Zafar, A. H. Akbar, B. Wajid, and S. A. Chaudhry, "Change detection algorithms for surveillance in visual iot: A comparative study visual internet of things," *Mehran University Research Journal of Engineering and Technology*, vol. 37, no. 1, pp. 77–94, 2018. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01676639

[121] M. Michael, C. Feist, F. Schuller, and M. Tschentscher, "Fast change detection for camera-based surveillance systems," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Nov 2016, pp. 2481–2486.

[122] A. Huertas and R. Nevatia, "Detecting changes in aerial views of man-made structures," *Image and Vision Computing*, vol. 18, no. 8, pp. 583 – 596, 2000. [Online]. Available: https://doi.org/10.1016/S0262-8856(99)00063-3

[123] L. Bruzzone and D. F. Prieto, "An adaptive semiparametric and context-based approach to unsupervised change detection in multitemporal remote-sensing images," *IEEE Transactions on Image Processing*, vol. 11, no. 4, pp. 452–466, Apr 2002.

[124] P. Lv, Y. Zhong, J. Zhao, A. Ma, and L. Zhang, "Change detection based on structural conditional random field framework for high spatial resolution remote sensing imagery," in *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2017, pp. 1059–1062.

[125] M. Bosc, F. Heitz, J.-P. Armspach, I. Namer, D. Gounot, and L. Rumbach, "Automatic change detection in multimodal serial mri: application to multiple sclerosis lesion evolution," *NeuroImage*, vol. 20, no. 2, pp. 643 – 656, 2003. [Online]. Available: https://doi.org/10.1016/S1053-8119(03)00406-3

[126] Q. Liu, M. Sun, and R. J. Sclabassi, "Illumination-invariant change detection model for patient monitoring video," in *Proceedings of the 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, vol. 1, Sep 2004, pp. 1782–1785.

[127] A. Naitsat, E. Saucan, and Y. Zeevi, "A differential geometry approach for change detection in medical images," in *Proceeding of the IEEE 30th International Symposium on Computer-Based Medical Systems (CBMS)*, Jun 2017, pp. 85–88.

[128] K. Lebart, E. Trucco, and D. M. Lane, "Real-time automatic sea-floor change detection from video," in *Proceedings of OCEANS 2000 MTS/IEEE Conference and Exhibition. Conference Proceedings (Cat. No.00CH37158)*, vol. 2, Sep 2000, pp. 1337–1343 vol.2.

[129] K. Seemakurthy and A. N. Rajagopalan, "Change detection in underwater imagery," *J. Opt. Soc. Am. A*, vol. 33, no. 3, pp. 301–313, Mar 2016. [Online]. Available: https://doi.org/10.1364/JOSAA.33.000301

[130] M. Radolko, F. Farhadifard, and U. F. von Lukas, "Dataset on underwater change detection," in *OCEANS 2016 MTS/IEEE Monterey*, Sept 2016, pp. 1–8.

[131] G. Nagy, T. Zhang, W. Franklin, E. Landis, E. Nagy, and D. T. Keane, "Volume and surface area distributions of cracks in concrete," in *Visual Form 2001*, C. Arcelli, L. P. Cordella, and G. S. di Baja, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 759–768. [Online]. Available: https://doi.org/10.1007/3-540-45129-3_70

[132] R. V. Patil and D. J. Pete, "Image change detection using stereo imagery and digital surface mode," in *2015 International Conference on Information Processing (ICIP)*, Dec 2015, pp. 192–197.

[133] H.-B. Yun, G. Sundaresan, Y. Jung, J.-W. Kim, and K.-T. Parkl, "Novel pattern detection algorithm for monitoring phase change of moisture on concrete pavement using surface temperature data," *Journal of Computing in Civil Engineering*, vol. 29, no. 2, p. 04014041, 2015. [Online]. Available: https://doi.org/10.1061/(ASCE)CP.1943-5487.0000330

[134] M. Baessler, H. Runge, S. Suchandt, and Y. Zhang, "Change detection for traffic measurement in multi-temporal terrasar-x spotlight images," in *EUSAR 2012; 9th European Conference on Synthetic Aperture Radar*, April 2012, pp. 328–331.

[135] C.-Y. Fang, S.-W. Chen, and C.-S. Fuh, "Automatic change detection of driving environments in a vision-based driver assistance system," *IEEE Transactions on Neural Networks*, vol. 14, no. 3, pp. 646–657, May 2003.

[136] J. Balcerek, A. Konieczka, T. Marciniak, A. Dabrowski, K. Mackowiak, and K. Piniarski, "Automatic detection of traffic lights changes from red to green and car turn signals in order to improve urban traffic," in *2014 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA)*, Sept 2014, pp. 110–115.

[137] B. Zitova, J. Flusser, and F. Sroubek, "Image registration: A survey and recent advances," in *Proceedings of 12th IEEE International Conference*

on Image Processing, Genova, Italy, 2005, online; accessed May 2020. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.477.1465&rep=rep1&type=pdf

[138] L. G. Brown, "A survey of image registration techniques," *ACM Computing Surveys*, vol. 24, pp. 325–376, 1992. [Online]. Available: http://dx.doi.org/10.1145/146370.146374

[139] Z. Xiong and Y. Zhang, "A critical review of image registration methods," *International Journal of Image and Data Fusion*, vol. 1, pp. 137–158, 2010. [Online]. Available: http://dx.doi.org/10.1080/19479831003802790

[140] J. R. G. Townshend, C. O. Justice, C. Gurney, and J. McManus, "The impact of misregistration on change detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 30, no. 5, pp. 1054–1060, Sept 1992.

[141] D. A. Stow, "Reducing the effects of misregistration on pixel-level change detection," *International Journal of Remote Sensing*, vol. 20, no. 12, pp. 2477–2483, 1999. [Online]. Available: https://doi.org/10.1080/014311699212137

[142] A. Sundaresan, P. K. Varshney, and M. K. Arora, "Robustness of change detection algorithms in the presence of registration errors," *Photogrammetric Engineering & Remote Sensing*, vol. 73, no. 4, pp. 375–383, 2007. [Online]. Available: https://doi.org/10.14358/PERS.73.4.375

[143] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 3431–3440.

[144] G. Lin, A. Milan, C. Shen, and I. Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation," 2016.

[145] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," 2015.

[146] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, April 2018.

[147] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017.

[148] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 6230–6239.

[149] M. Chen and X. L. Xu, "An improved method for motion detection by image difference," in *Proceedings of the IET International Conference on Information Science and Control Engineering 2012 (ICISCE 2012)*, Dec 2012, pp. 1–3.

[150] L. Xu, S. Zhang, Z. He, and Y. Guo, "The comparative study of three methods of remote sensing image change detection," in *2009 17th International Conference on Geoinformatics*, Aug 2009, pp. 1–4.

[151] S. Singh and R. Talwar, "Review on different change vector analysis algorithms based change detection techniques," in *2013 IEEE Second International Conference on Image Information Processing (ICIIP-2013)*, Dec 2013, pp. 136–141.

[152] I. Jolliffe, *Principal Component Analysis*. Springer Verlag, 1986.

[153] R. J. Kauth and G. S. Thomas, "The tasselled-cap—a graphic description of the spectral-temporal development of agricultural crops as seen by landsat," in *Proceedings of the Symposium on Machine Processing of Remotely Sensed Data*, July 1976, pp. 41–51. [Online]. Available: https://docs.lib.purdue.edu/lars_symp/159/

[154] L. Pursell and S. Y. Trimble, "Gram-schmidt orthogonalization by gauss elimination," *The American Mathematical Monthly*, vol. 98, no. 6, pp. 544–549, 1991. [Online]. Available: http://www.jstor.org/stable/2324877

[155] A. Alboody, F. Sedes, and J. Inglada, "Post-classification and spatial reasoning: new approach to change detection for updating gis database," in *2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications*, April 2008, pp. 1–7.

[156] C. Wu, B. Du, X. Cui, and L. Zhang, "A post-classification change detection method based on iterative slow feature analysis and bayesian soft fusion," *Remote Sensing of Environment*, vol. 199, pp. 241 – 255, 2017. [Online]. Available: https://doi.org/10.1016/j.rse.2017.07.009

[157] M. J. Black, D. J. Fleet, and Y. Yacoob, "Robustly estimating changes in image appearance," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 8 – 31, 2000. [Online]. Available: https://doi.org/10.1006/cviu.1999.0825

[158] C. Couvreur, *The EM Algorithm: A Guided Tour*. Boston, MA: Birkhäuser Boston, 1997, pp. 209–222. [Online]. Available: https://doi.org/10.1007/978-1-4612-1996-5_12

[159] F. Yang and J. Lishman, "Land cover change detection using gabor filter texture," 2003.

[160] S. Gopal and C. Woodcock, "Remote sensing of forest change using artificial neural networks," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 34, no. 2, pp. 398–404, 1996.

[161] X. Dai and S. Khorram, "Remotely sensed change detection based on artificial neural networks," *Photogrammetric Engineering & Remote Sensing*, vol. 65, p. 1187–1194, 1999.

[162] C. Huang, K. Song, S. Kim, J. R. Townshend, P. Davis, J. G. Masek, and S. N. Goward, "Use of a dark object concept and support vector machines to automate forest cover change analysis," *Remote Sensing of Environment*, vol. 112, no. 3, pp. 970 – 985, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0034425707003951

[163] F. Bovolo, L. Bruzzone, and M. Marconcini, "A novel approach to unsupervised change detection based on a semisupervised svm and a similarity measure," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 46, no. 7, pp. 2070–2082, 2008.

[164] J. Im and J. Jensen, "A change detection model based on neighborhood correlation image analysis and decision tree classification," *Remote Sensing of Environment*, vol. 99, pp. 326–340, 11 2005.

[165] A. Makkeasorn, N.-B. Chang, and J. Li, "Seasonal change detection of riparian zones with remote sensing images and genetic programming in a semi-arid watershed," *Journal of Environmental Management*, vol. 90, no. 2, pp. 1069 – 1080, 2009. [Online]. Available: https://doi.org/10.1016/j.jenvman.2008.04.004

[166] A. Smith, "Image segmentation scale parameter optimization and land cover classification using the random forest algorithm," *Journal of Spatial Science*, vol. 55, no. 1, pp. 69–79, 2010. [Online]. Available: https://doi.org/10.1080/14498596.2010.487851

[167] O. Miller, A. Pikaz, and A. Averbuch, "Objects based change detection in a pair of gray-level images," *Pattern Recognition*, vol. 38, no. 11, pp. 1976 – 1992, 2005. [Online]. Available: https://doi.org/10.1016/j.patcog.2004.07.010

[168] L. Durieux, E. Lagabrielle, and A. Nelson, "A method for monitoring building construction in urban sprawl areas using object-based analysis of spot 5 images and existing gis data," *ISPRS Journal of Photogrammetry*

and Remote Sensing, vol. 63, no. 4, pp. 399 – 408, 2008. [Online]. Available: https://doi.org/10.1016/j.isprsjprs.2008.01.005

[169] K. G. Mehrotra, C. K. Mohan, and H. Huang, *Clustering-Based Anomaly Detection Approaches*. Cham: Springer International Publishing, 2017, pp. 41–55.

[170] F. T. Liu, K. M. Ting, and Z. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*, Dec 2008, pp. 413–422.

[171] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: Identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '00, 2000, pp. 93–104. [Online]. Available: https://doi.org/10.1145/342009.335388

[172] M. D. Castro, A. Masi, G. Lunghi, and R. Losito, "An incremental slam algorithm for indoor autonomous navigation," in *Proceedings of the 20th Imeko TC4 International Symposium and 18th International Workshop on ADC Modelling and Testing*, 2014.

[173] M. D. Jenkins, T. Buggy, and G. Morison, "An imaging system for visual inspection and structural condition monitoring of railway tunnels," in *Proceedings of the IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS)*, Jul 2017, pp. 1–6.

[174] L. Attard, C. J. Debono, G. Valentino, and M. D. Castro, "Image mosaicing of tunnel wall images using high level features," in *Proceedings of the 10th International Symposium on Image and Signal Processing and Analysis*, Sep 2017, pp. 141–146.

[175] S. Stent, R. Gherardi, B. Stenger, K. Soga, and R. Cipolla, "An image-based system for change detection on tunnel linings," in *Proceedings of the 13th IAPR International Conference on Machine Vision Applications, Kyoto, Japan*, 2013, pp. 2–5.

[176] S. I. Granshaw, "Structure from motion: origins and originality," *The Photogrammetric Record*, vol. 33, no. 161, pp. 6–10, 2018. [Online]. Available: 10.1111/phor.12237

[177] S. Stent, R. Gherardi, B. Stenger, and R. Cipolla, "Detecting change for multi-view, long-term surface inspection," in *Proceedings of the British Machine Vision Conference (BMVC)*, September 2015, pp. 127–139.

[178] F. Crosilla, *Procrustes Analysis and Geodetic Sciences*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 287–292.

[179] M. Ukai, "Advanced inspection system of tunnel wall deformation using image processing," *Quarterly Report of RTRI*, vol. 48, no. 2, pp. 94–98, 2007.

[180] S. Stent, C. Girerd, P. Long, and R. Cipolla, "A low-cost robotic system for the efficient visual inspection of tunnels," in *Proceedings of the 32nd International Symposium on Automation and Robotics in Construction, ISARC*, 2015, pp. 1–8. [Online]. Available: https://doi.org/10.22260/ISARC2015/0070

[181] C. Wu, "Towards linear-time incremental structure from motion," in *Proceedings of the International Conference on 3D Vision - 3DV*, Jun 2013, pp. 127–134.

[182] K. Chaiyasarn, K. Tae-Kyun, F. Viola, R. Cipolla, and K. Soga, "Distortion-free image mosaicing for tunnel inspection based on robust cylindrical surface estimation," *Journal of Computing in Civil Engineering*, vol. 30, no. 3, pp. 1–9, 2013. [Online]. Available: http://dx.doi.org/10.1061/(ASCE)CP.1943-5487.0000516

[183] Pavemetrics TM, "Laser tunnel scanning system (ltss)," online; accessed May 2020. [Online]. Available: http://www.pavemetrics.com/wp-content/uploads/2016/03/LTSS_Flyer.pdf

[184] C. Frohlich and M. Mettenleiter, "Terrestrial laser scanning-new perspectives in 3d surveying," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 36, 01 2004. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.215.8213&rep=rep1&type=pdf

[185] G. Paar, A. Bauer, and H. Kontrus, "Texture-based fusion between laser scanner and camera for tunnel surface documentation," in *Proceedings of the 7th International Conference on Optical 3-D Measurement Techniques*, 2005. [Online]. Available: http://dibweb.joanneum.at/group{_}3DVision/3DVision/publications-presentations/literature/pdfs/PUB05DIB005.pdf

[186] A. Bauer, K. Gutjahr, G. Paar, H. Kontrus, and R. Glatzl, "Tunnel surface 3d reconstruction from unoriented image sequences," in *Proceedings of the 39th Annual Workshop of the Austrian Association for Pattern Recognition (OAGM)*, 2015. [Online]. Available: https://arxiv.org/abs/1505.06237

[187] S. Stent, R. Gherardi, B. Stenger, K. Soga, and R. Cipolla, "Visual change detection on tunnel linings," *Machine Vision and Applications Journal*, vol. 27, no. 3, pp. 319–330, Apr. 2014. [Online]. Available: https://doi.org/10.1007/s00138-014-0648-8

[188] C. Wu, "Towards linear-time incremental structure from motion," in *2013 International Conference on 3D Vision - 3DV 2013*, Jun 2013, pp. 127–134.

[189] E. Protopapadakis, C. Stentoumis, N. Doulamis, A. Doulamis, K. Loupos, K. Makantasis, G. Kopsiaftis, and A. Amditis, "Autonomous robotic inspection in tunnels," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. III-5, pp. 167–174, 06 2016. [Online]. Available: https://www.researchgate.net/publication/307530827_AUTONOMOUS_ROBOTIC_INSPECTION_IN_TUNNELS

[190] M. R. Jahanshahi and S. F. Masri, "Adaptive vision-based crack detection using 3d scene reconstruction for condition assessment of structures," *Automation in Construction*, vol. 22, no. Supplement C, pp. 567 – 576, 2012. [Online]. Available: https://doi.org/10.1016/j.autcon.2011.11.018

[191] M. Torok, M. Golparvar-Fard, and K. Kochersberger, "Image-based automated 3d crack detection for post-disaster building assessment," *Journal of Computing in Civil Engineering*, vol. 28, Jan 2013. [Online]. Available: https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29CP.1943-5487.0000334

[192] N. Singh and S. Singh, "Virtual reality: A brief survey," in *Proceedings of the 2017 International Conference on Information Communication and Embedded Systems (ICICES)*, Feb 2017, pp. 1–6.

[193] C. Fu, "A new research approach on the application of virtual reality technology in civil engineering," in *Proceedings of the International Conference on Materials Engineering and Information Technology Applications (MEITA 2015)*, Feb 2015, pp. 1014–1017.

[194] A. Z. Sampaio, A. M. Gomes, A. R. Gomes, and D. P. Rosário, "Virtual reality technology used to support the buildings inspection activity," in *Proceedings of The Third International Conferences on Advances in Multimedia, MMEDIA 2011*, 2011, pp. 80–86. [Online]. Available: https://www.thinkmind.org/download.php?articleid=mmedia_2011_4_10_40011

[195] "Troll tunnel inspection ROV piloted in virtual reality mode," https://www.offshore-mag.com/subsea/article/16759458/troll-tunnel-inspection-rov-piloted-in-virtual-reality-mode, pp. 141–142, 1996, [Online; accessed May 2019].

[196] M. Di Castro, M. L. Baiguera Tambutti, S. Gilardoni, R. Losito, G. Lunghi, and A. Masi, "LHC train control system for autonomous inspections and

measurements," in *Proceedings, 16th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS 2017)*, Oct 2017.

[197] M. Di Castro, L. R. Buonocore, M. Ferre, S. Gilardoni, R. Losito, G. Lunghi, and A. Masi, "A dual arms robotic platform control for navigation, inspection and telemanipulation," in *Proceedings of the 16th International Conference on Accelerator and Large Experimental Physics Control Systems(ICALEPCS 2017)*, Oct 2017.

[198] Nikon, "Nikon software development kit," online; accessed May 2020. [Online]. Available: https://sdk.nikonimaging.com/apply/

[199] SITES, "Scantubes." [Online]. Available: https://www.sites.fr/cas-pratique/inspection-scantubes/

[200] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018, doi: https://doi.org/10.1109/TPAMI.2018.2844175.

[201] D. Gupta, "Image Segmentation Keras : Implementation of Segnet, FCN, UNet, PSPNet and other models in Keras," https://github.com/divamgupta/image-segmentation-keras.

[202] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[203] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[204] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[205] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 580–587, doi: https://doi.org/10.1109/CVPR.2014.81. [Online]. Available: https://doi.org/10.1109/CVPR.2014.81

[206] R. Girshick, "Fast r-cnn," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 1440–1448, doi: https://doi.org/10.1109/ICCV.2015.169.

[207] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, pp. 91–99. [Online]. Available: http://dl.acm.org/citation.cfm?id=2969239.2969250

[208] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778, doi: https://doi.org/10.1109/CVPR.2016.90.

[209] L. Attard, C. J. Debono, G. Valentino, M. Di Castro, A. Masi, and L. Scibile, "Automatic crack detection using mask r-cnn," in *2019 11th International Symposium on Image and Signal Processing and Analysis (ISPA)*, Sep. 2019, pp. 152–157.

[210] W. Abdulla, "Mask r-cnn for object detection and instance segmentation on keras and tensorflow," 2017. [Online]. Available: https://github.com/matterport/Mask_RCNN

[211] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer*

Vision - ECCV 2014, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 740–755, doi: https://doi.org/10.1007/978-3-319-10602-1_48.

[212] A. B. Jung, "imgaug," https://github.com/aleju/imgaug, 2018, [Online; accessed May 2020].

[213] S. Dorafshan, R. J. Thomas, and M. Maguire, "SDNET2018: An annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks," *Data in Brief*, vol. 21, pp. 1664–1668, 2018. [Online]. Available: https://doi.org/10.1016/j.dib.2018.11.015

[214] A. Bréhéret, "Pixel Annotation Tool," 2017. [Online]. Available: https://github.com/abreheret/PixelAnnotationTool

[215] H. A. Khan, J.-B. Thomas, and J. Y. Hardeberg, "Analytical survey of highlight detection in color and spectral images," in *Computational Color Imaging*, S. Bianco, R. Schettini, A. Trémeau, and S. Tominaga, Eds. Cham: Springer International Publishing, 2017, pp. 197–208.

[216] S. Lee, T. Yoon, K. Kim, K. Kim, and W. Park, "Removal of specular reflections in tooth color image by perceptron neural nets," in *2010 2nd International Conference on Signal Processing Systems*, vol. 1, July 2010, pp. V1–285–V1–289.

[217] A. Rodríguez-Sánchez, D. Chea, G. Azzopardi, and S. Stabinger, "A deep learning approach for detecting and correcting highlights in endoscopic images," in *2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)*, Nov 2017, pp. 1–6.

[218] L. Attard, C. J. Debono, G. Valentino, and M. di Castro, "Specular highlights detection using a u-net based deep learning architecture," in *27th IEEE International Conference on Image Processing (ICIP 2020)*, 2020, [under review].

[219] M. D. Zeiler, "ADADELTA: an adaptive learning rate method," *CoRR*, vol. abs/1212.5701, 2012. [Online]. Available: http://arxiv.org/abs/1212.5701

[220] J. B. Park and A. C. Kak, "A truncated least squares approach to the detection of specular highlights in color images," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2003)*, 2003.

[221] H.-F. Ng, "Automatic thresholding for defect detection," *Pattern Recognition Letters*, vol. 27, no. 14, pp. 1644 – 1649, 2006. [Online]. Available: https://doi.org/10.1016/j.patrec.2006.03.009

[222] J.-L. Fan and B. Lei, "A modified valley-emphasis method for automatic thresholding," *Pattern Recognition Letters*, vol. 33, no. 6, pp. 703 – 708, 2012. [Online]. Available: https://doi.org/10.1016/j.patrec.2011.12.009

[223] H. Ng, D. Jargalsaikhan, H. Tsai, and C. Lin, "An improved method for image thresholding based on the valley-emphasis method," in *Proceedings of the 2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, Oct 2013, pp. 1–4.

[224] D.-M. Tsai and T.-Y. Huang, "Automated surface inspection for statistical textures," *Image and Vision Computing*, vol. 21, no. 4, pp. 307 – 323, 2003. [Online]. Available: https://doi.org/10.1016/S0262-8856(03)00007-6

[225] Zhou Wang and A. C. Bovik, "A universal image quality index," *IEEE Signal Processing Letters*, vol. 9, no. 3, pp. 81–84, March 2002.

[226] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, April 2004.

[227] Z. Wang and A. C. Bovik, *Modern Image Quality Assessment*. Morgan & Claypool, 2006. [Online]. Available: https://ieeexplore.ieee.org/document/6812930

[228] Z. Wang and X. Shang, "Spatial pooling strategies for perceptual image quality assessment," in *2006 International Conference on Image Processing*, Oct 2006, pp. 2945–2948.

[229] G. Piella and H. Heijmans, "A new quality metric for image fusion," in *Proceedings 2003 International Conference on Image Processing (Cat. No.03CH37429)*, vol. 3, Sep. 2003, pp. III–173.

[230] A. M. Alattar, E. T. Lin, and M. U. Celik, "Digital watermarking of low bit-rate advanced simple profile mpeg-4 compressed video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 8, pp. 787–800, Aug 2003.

[231] C.-Y. Hsu and C.-S. Lu, "Geometric distortion-resilient image hashing system and its application scalability," in *Proceedings of the 2004 Workshop on Multimedia and Security*, New York, NY, USA, 2004, p. 81–92. [Online]. Available: https://doi.org/10.1145/1022431.1022448

[232] Honghua Chang and Jianqi Zhang, "New metrics for clutter affecting human target acquisition," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 42, no. 1, pp. 361–368, Jan 2006.

[233] L. Snidaro and G. L. Foresti, "A multi-camera approach to sensor evaluation in video surveillance," in *IEEE International Conference on Image Processing 2005*, vol. 1, Sep. 2005, pp. I–1101.

[234] E. Christophe, D. Leger, and C. Mailhes, "Quality criteria benchmark for hyperspectral imagery," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 43, pp. 2103 – 2114, 10 2005. [Online]. Available: 10.1109/TGRS.2005.853931

[235] GDAL/OGR contributors, *GDAL/OGR Geospatial Data Abstraction software Library*, Open Source Geospatial Foundation, 2020. [Online]. Available: https://gdal.org

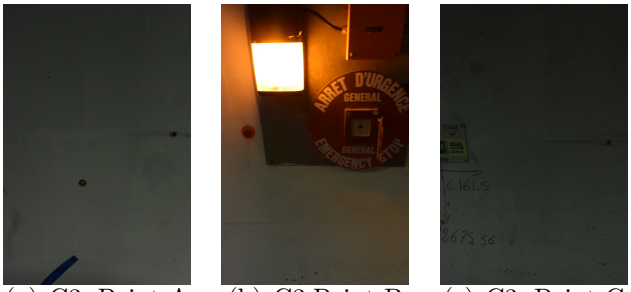
[236] L. Attard, C. J. Debono, G. Valentino, M. di Castro, J. A. Osborne, L. Scibile, and M. Ferre, "A comprehensive virtual reality system for tunnel surface documentation and structural health monitoring," in *Proceedings of the 2018 IEEE International Conference on Imaging Systems and Techniques (IST)*, 2018, pp. 1–6.

[237] L. Attard, C. J. Debono, G. Valentino, M. di Castro, and A. Masi, "Vr-shm - a structural health monitoring tool to assist crack detection using deep learning and virtual reality," in *Proceedings of the Malta Sustainable Built Environment conference (SBE)*, 2019.

[238] FLIR Systems, "Developers' resources," https://flir.custhelp.com/app/account/fl_downloads, [Online; accessed May 2020].

Appendices

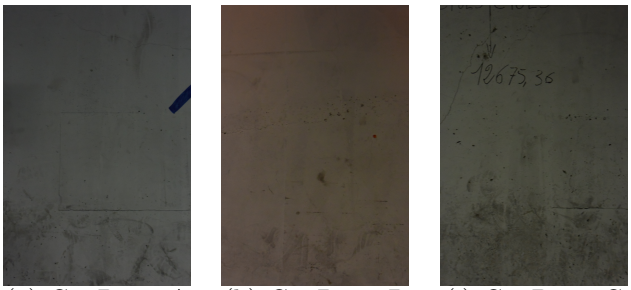
Appendix A | Examples of acquired LHC tunnel images



(a) C3, Point A (b) C3,Point B (c) C3, Point C

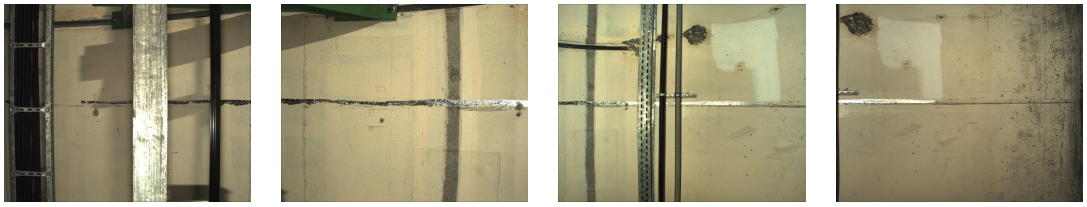


(d) C2, Point A (e) C2, Point B (f) C2, Point C

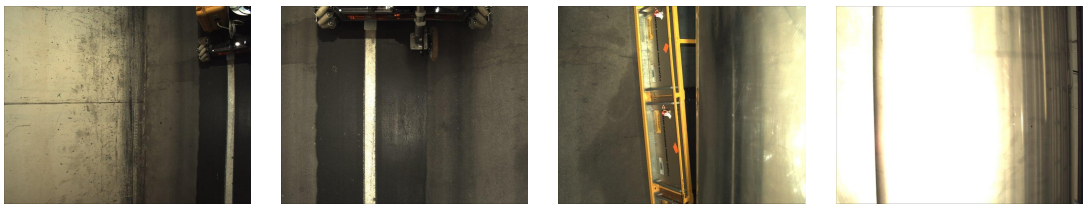


(g) C1, Point A (h) C1, Point B (i) C1, Point C

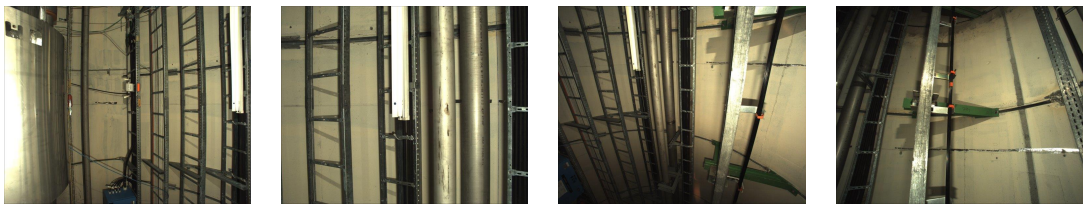
Figure A.1: Images captured by the three cameras (C1, C2, C3) on the vertical structure on the CERNBot as different points (Points A, B and C)



(a) Camera 1 (b) Camera 2 (c) Camera 3 (d) Camera 4

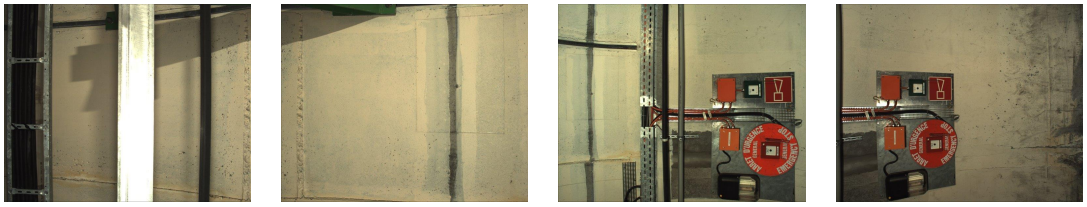


(e) Camera 5 (f) Camera 6 (g) Camera 7 (h) Camera 8



(i) Camera 9 (j) Camera 10 (k) Camera 11 (l) Camera 12

Figure A.2: Example 1 of a sample set of images captured using the provisional commercial camera system during the demo test in the LHC at a particular location



(a) Camera 1 (b) Camera 2 (c) Camera 3 (d) Camera 4



(e) Camera 5 (f) Camera 6 (g) Camera 7 (h) Camera 8



(i) Camera 9 (j) Camera 10 (k) Camera 11 (l) Camera 12

Figure A.3: Example 2 of a sample set of images captured using the provisional commercial camera system during the demo test in the LHC at a particular location

Appendix B | Crack detection

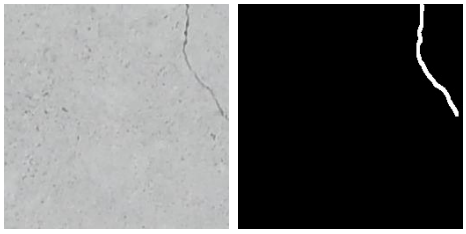


(a) Crack image and corresponding GT



(b) U-Net (c) SegNet (d) Mask R-CNN

Figure B.1: Example 1 of crack detection results from the SDNET subset using Mask R-CNN with ResNet-101 backbone, U-Net and SegNet with VGG16 encoder

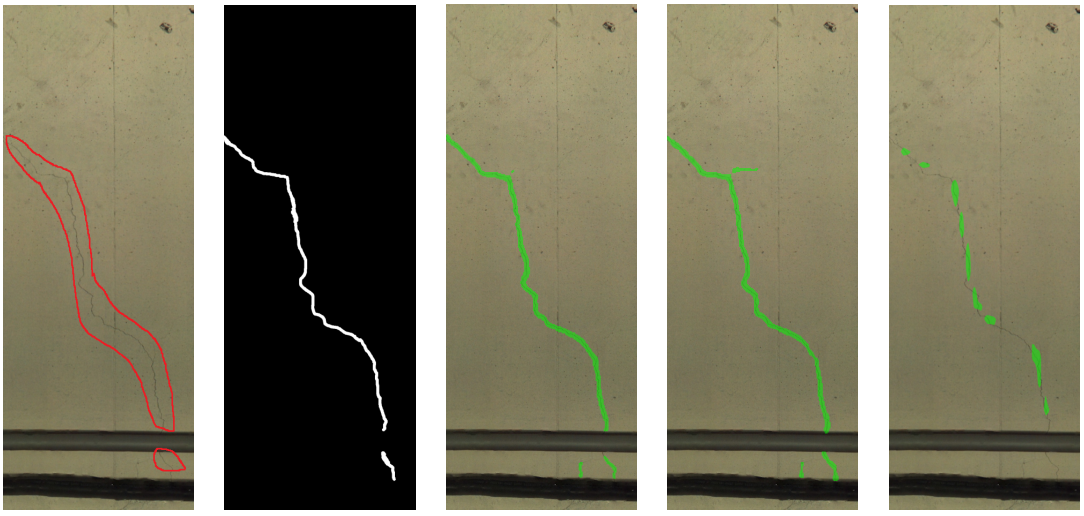


(a) Crack image and corresponding GT



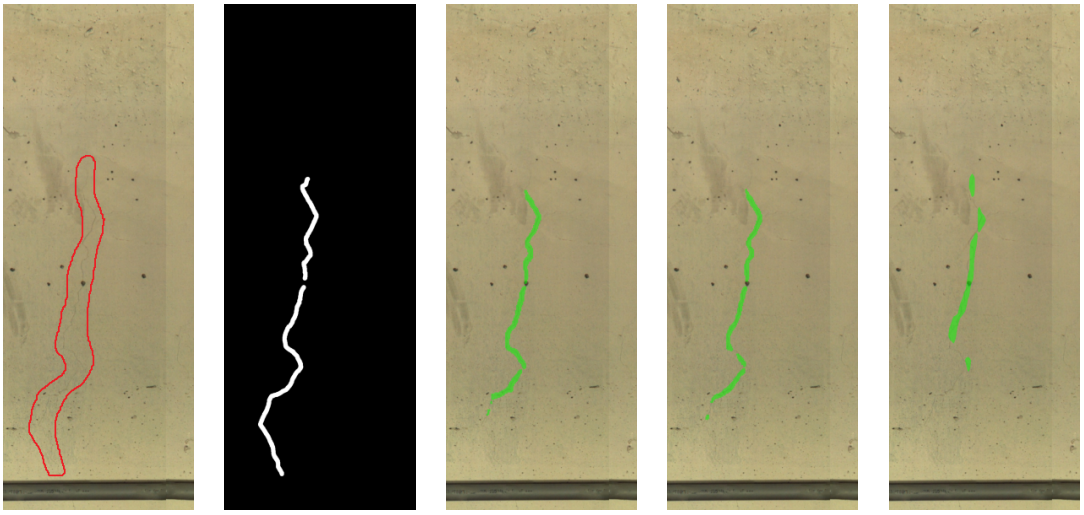
(b) U-Net (c) SegNet (d) Mask R-CNN

Figure B.2: Example 2 of crack detection results from the SDNET subset using Mask R-CNN with ResNet-101 backbone, U-Net and SegNet with VGG16 encoder



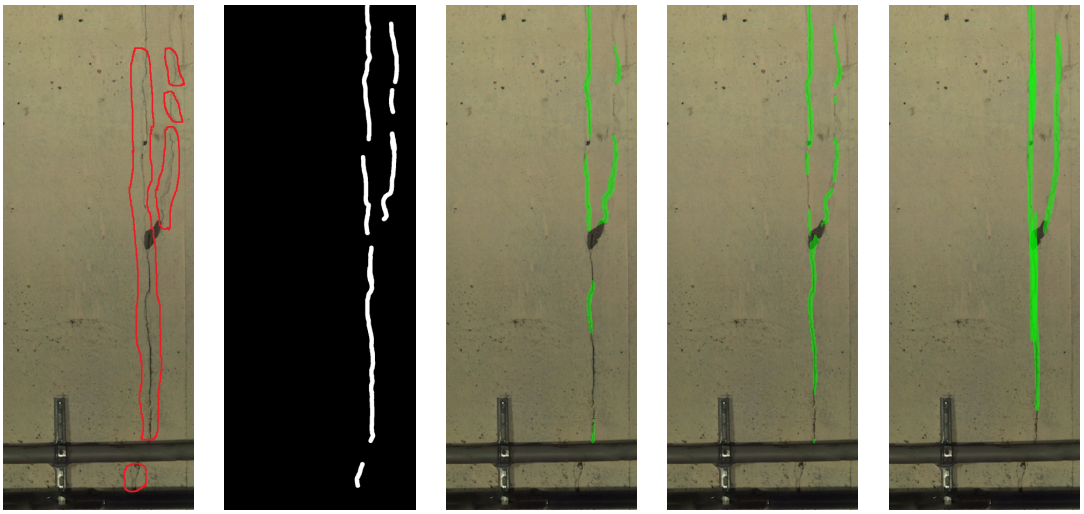
(a) Crack image (b) GT (c) U-Net (d) SegNet (e) Mask R-CNN

Figure B.3: Example 1 of crack detection results from the LHC dataset using Mask R-CNN with ResNet-101 backbone and both U-Net and SegNet with a ResNet-50 encoder



(a) Crack image (b) GT (c) U-Net (d) SegNet (e) Mask R-CNN

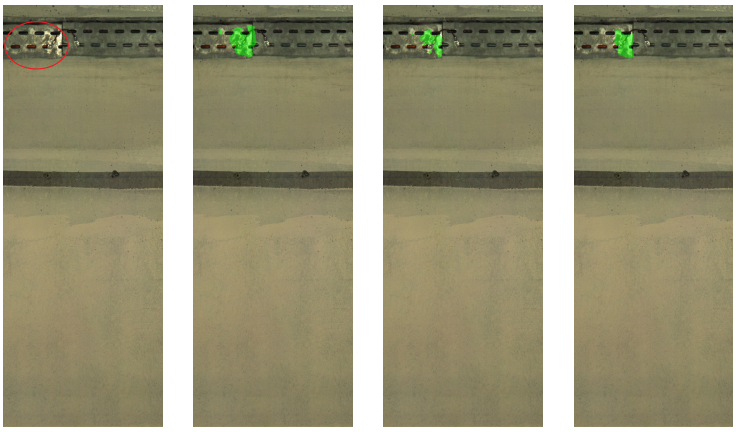
Figure B.4: Example 2 of crack detection results from the LHC dataset using Mask R-CNN with ResNet-101 backbone and both U-Net and SegNet with ResNet-50 encoder



(a) Crack image (b) GT (c) U-Net (d) SegNet (e) Mask R-CNN

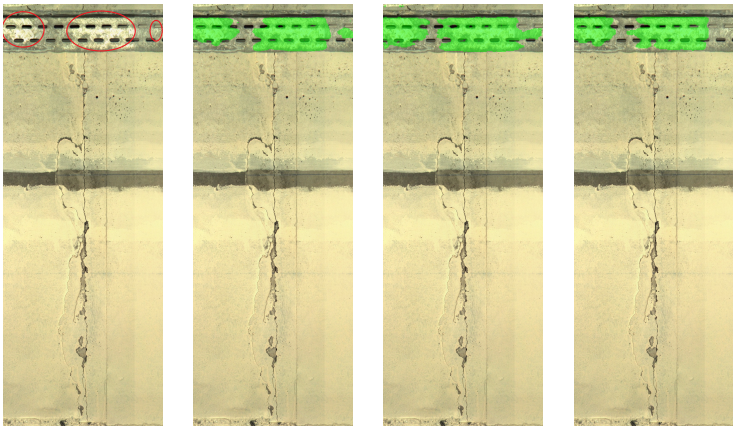
Figure B.5: Example 3 of crack detection results from the LHC dataset using Mask R-CNN with ResNet-101 backbone and both U-Net and SegNet with a ResNet-50 encoder

Appendix C | Highlight detection



(a) Highlight (b) Proposed (c) VGG16 (d) ResNet-50

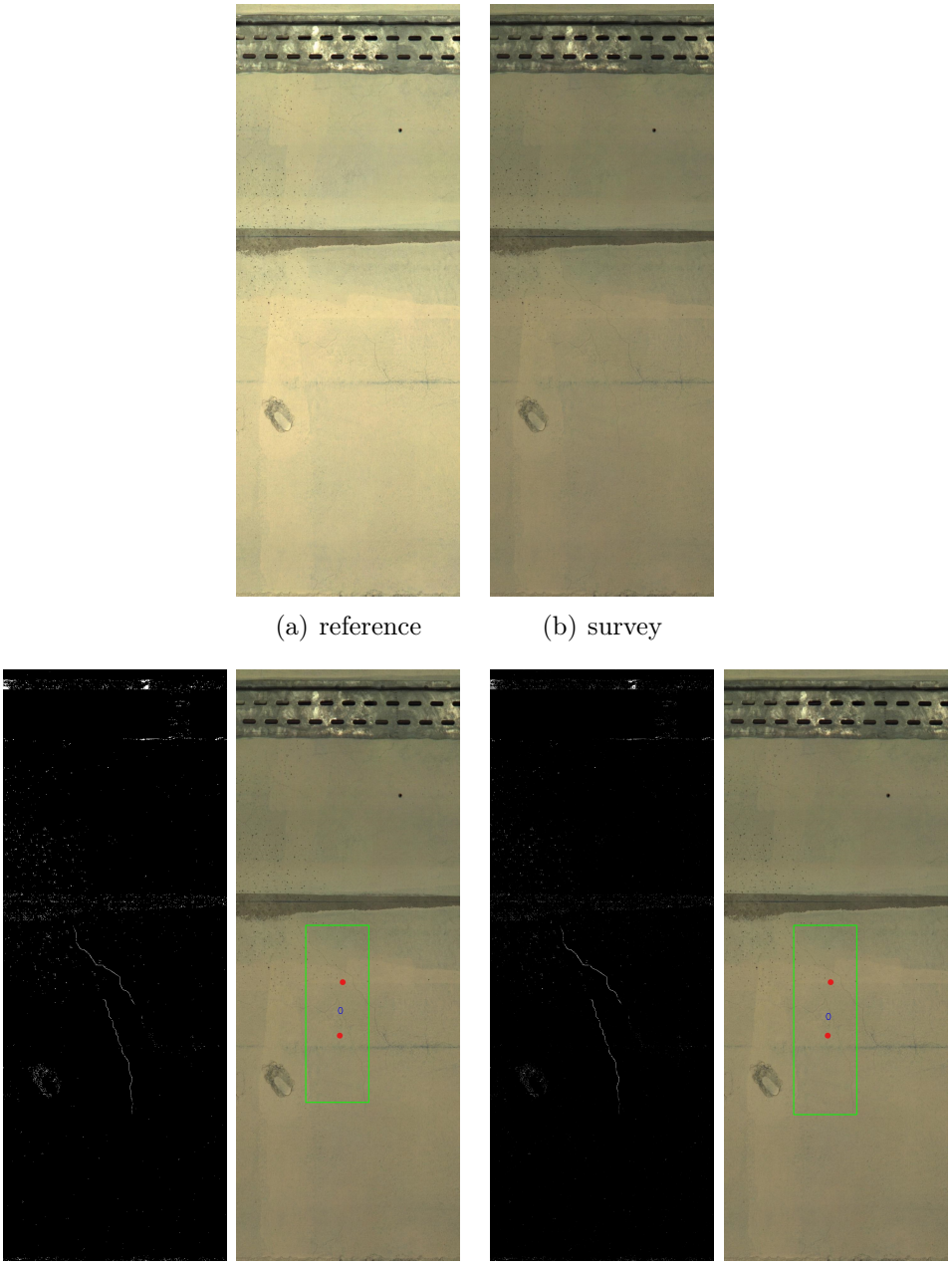
Figure C.1: A comparison of the specular highlights marked on image example 1 and the resulting highlight detection results using U-Net with (b) the proposed modified architecture and (c)-(d) other architectures



(a) Highlight (b) Proposed (c) VGG16 (d) ResNet-50

Figure C.2: A comparison of the specular highlights marked on image example 2 and the resulting highlight detection results using U-Net with (b) the proposed modified architecture and (c)-(d) other architectures

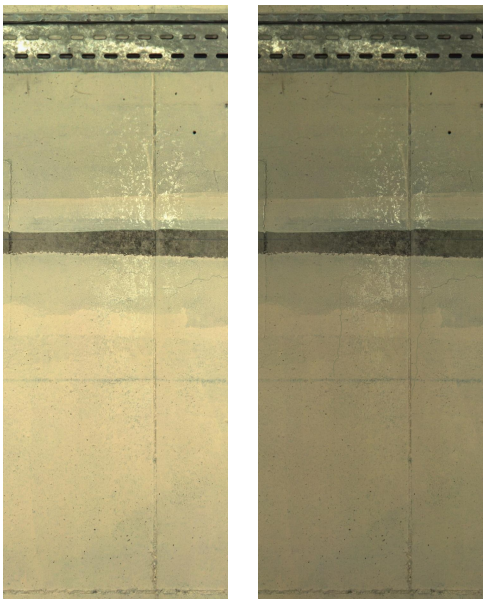
Appendix D | Change detection



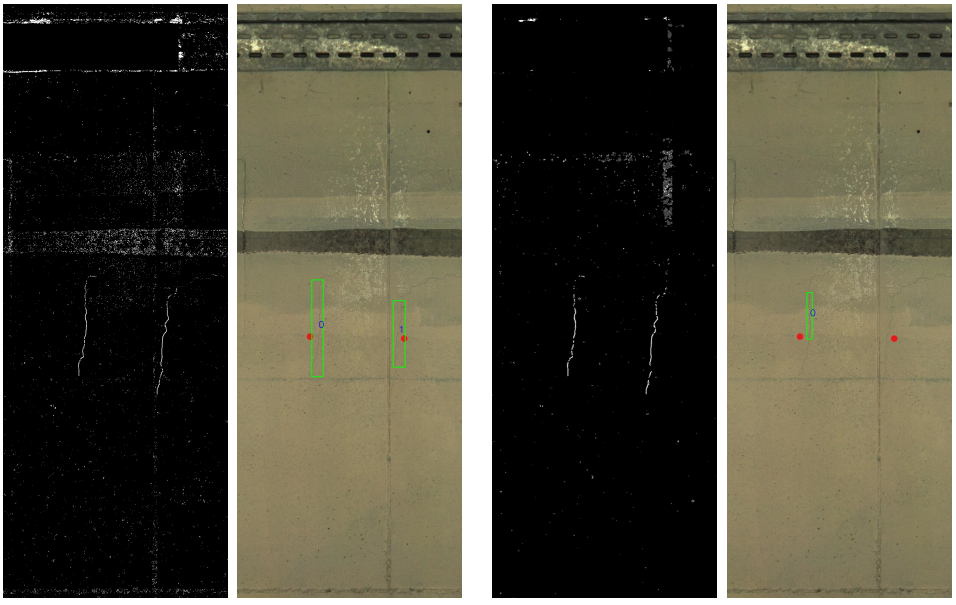
(a) reference (b) survey

(c) MV (d) PCA-weighted summation

Figure D.1: Example showing similar detection results from the majority voting and PCA-weighted summation methods

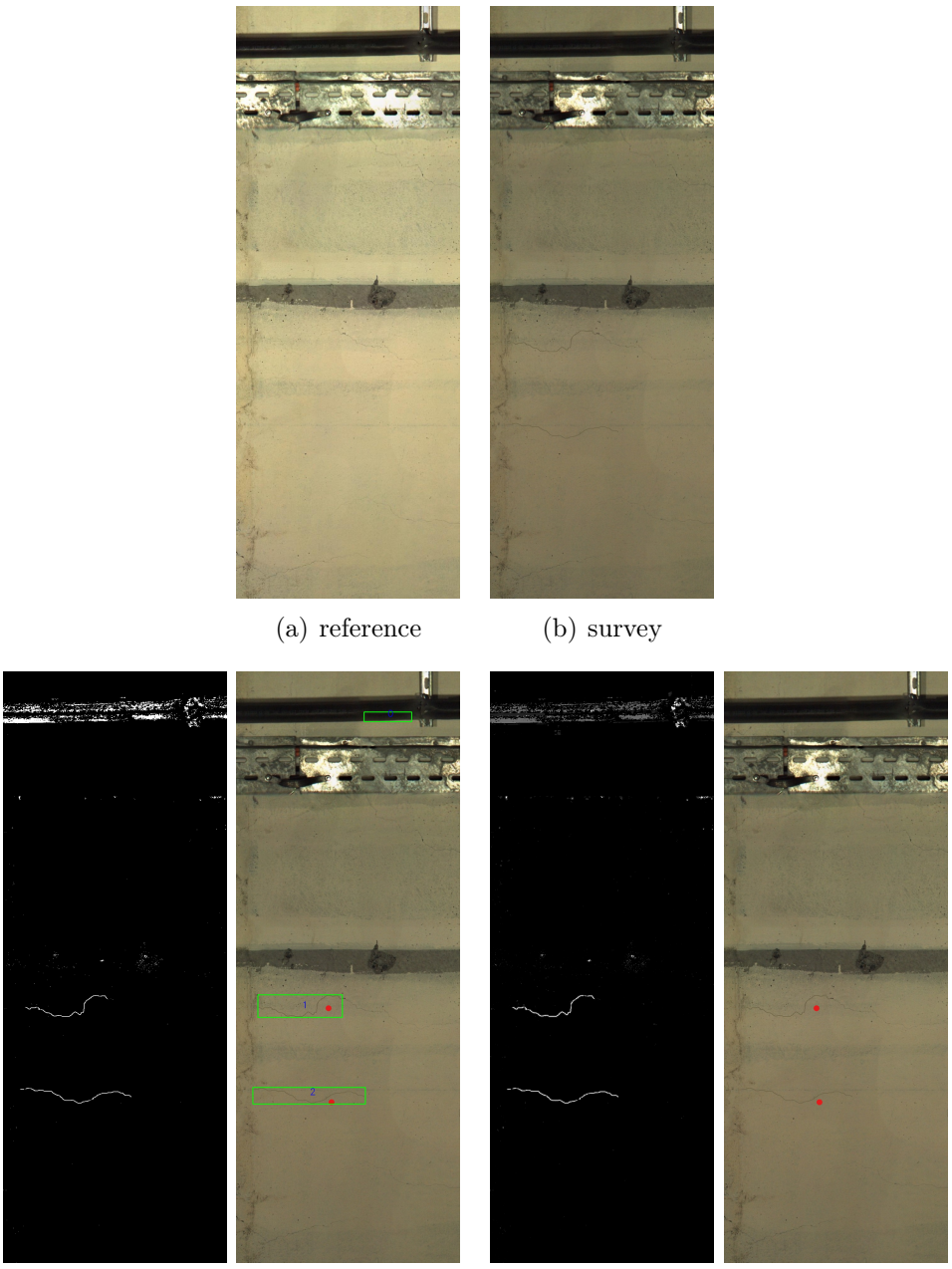


(a) reference (b) survey



(c) MV (d) PCA-weighted summation

Figure D.2: Example showing different change detection results from the majority voting and PCA-weighted summation methods, as a result of the change map analysis stage on the respective change maps



(a) reference (b) survey

(c) MV (d) PCA-weighted summation

Figure D.3: Example showing different change detection results from the majority voting and PCA-weighted summation methods, with a FP for the former and FN for the latter

Appendix E | Publications

ISPRS Journal of Photogrammetry and Remote Sensing 144 (2018) 180–188

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

Review Article

Tunnel inspection using photogrammetric techniques and image processing: A review

Leanne Attard[a], Carl James Debono[a,*], Gianluca Valentino[a], Mario Di Castro[b]

[a] Department of Communications and Computer Engineering, University of Malta, Msida, Malta
[b] Engineering Department, CERN, Meyrin, Switzerland

ARTICLE INFO

Keywords:
Tunnel inspection
Image processing
Photogrammetry

ABSTRACT

During the last few decades many tunnelling projects were conducted in order to use limited land surface area more efficiently. Such underground constructions are used for transportation such as for railways, subways and roads, to host equipment used for experiments like particle accelerators, as well as for pipelines and mines. Independent of their purpose, tunnels should be regularly inspected in order to avoid accidents resulting from structure failure and to simultaneously extend their lifetime by identifying deterioration at an early stage and perform the required maintenance. Traditional methods of tunnel inspection rely on manual vision monitoring and sensing equipment that requires installation and contact with the tunnel surface. Apart from being time consuming, tedious and expensive, manual inspection is also highly dependent on human subjectivity and exposes inspection personnel to possible dangerous environments. Taking these issues into consideration, various systems were proposed to automate different procedures of tunnel inspection using photographic equipment to capture photos of the tunnel environment, apply photogrammetric and computer vision (CV) techniques and conduct image processing (IP) on them to achieve different surveying goals. This manuscript provides a collective review of the current state of the art in tunnel inspection based on photogrammetric techniques and IP.

1. Introduction

A considerable amount of tunnelling was performed in the last few decades, and concerns have been raised over the need to improve the current methods employed in civil construction management, monitoring and inspection in general. The use of photogrammetry and CV is already being utilized to provide better automated approaches for these tasks. Using IP techniques, 3D maps are generated to help with Building Information Modelling (BIM) Eastman et al. (2008) as in Ptrucean et al. (2015) and Martin et al. (2016). Continuous area monitoring to analyse the progress on a construction site is also being improved by the introduction of CV systems Lukins et al. (2007). Over time, much of the infrastructure shows signs of deterioration due to ageing and stresses which may eventually cause problems in structural integrity. Consequently, to ensure safety in concrete tunnels, periodic inspections have to be conducted.

Currently, structural tunnel inspection is predominantly performed through periodic visual observations by trained inspectors. They try to detect structural defects such as cracking, spalling and water leakage as well as to identify possible changes in the infrastructure with respect to a previous survey. It is important that such inspections are made without creating a negative effect on the structure itself. Thus non-destructive (ND) inspection methods are commonly used other than destructive ones. ND methods (Montero et al., 2015; Boving, 1989) can be divided in visual observation, strength-based, sonic and ultrasonic, electrical, thermography, radar and endoscopy methods, each requiring specific equipment. In order to conduct such methods, presently, personnel often are required to be physically present in the tunnel and move around with the equipment. This approach has several drawbacks including the cost involved for hiring and training personnel to do the inspections and the considerable amount of time necessary to perform them. In addition, it is highly dependent on human subjectivity leading to possible inaccuracies, false and missing detections. Furthermore, tunnel inspections may demand personnel to access hazardous environments characterized by lack of light, inadequate temperatures, dust and possibly lack of adequate ventilation or presence of poisonous gases. For these reasons, research on automated health monitoring of tunnel structures has received significant attention in recent years in order to facilitate the process of visual inspection as in Balaguer et al. (2014) and Montero et al. (2015).

The use of cost-effective photographic equipment and photogrammetric techniques and CV (Linder, 2013; Förstner and Wrobel, 2016;

* Corresponding author.
E-mail address: carl.debono@um.edu.mt (C.J. Debono).

https://doi.org/10.1016/j.isprsjprs.2018.07.010
Received 20 February 2018; Received in revised form 6 July 2018; Accepted 18 July 2018
0924-2716/ © 2018 International Society for Photogrammetry and Remote Sensing, Inc. (ISPRS). Published by Elsevier B.V. All rights reserved.

Ikeuchi, 2014; Szeliski, 2011) techniques implemented through IP has led to various solutions that deal with different aspects of automated tunnel inspection. Such systems aim to achieve time-saving automated surveying solutions with fast data acquisition, identification and documentation of cracks as well as detection of structural changes. This publication reviews the use of CV to facilitate and automate tunnel inspections. Although reviews on general crack identification, image mosaicking and change detection are available in literature, these generally focus on natural scene images. In contrast, this paper provides an extensive survey of previous works presented within the whole image-based tunnel inspection spectrum, including tunnel profile monitoring, crack and leakage detection as well as tunnel surface documentation and visualization.

The remainder of this article is structured as follows. Section 2 reviews the state of the art with respect to techniques used for tunnel profile measurement and deformation monitoring. Section 3 gives an overview of methods for tunnel interior visualization. The latter includes both image mosaicking of the tunnel wall as well as 3D reconstruction. Section 4 discusses works related to crack and defect detection using different methods. Image-based water leakage detection systems are reviewed in Section 5. Section 6 investigates change identification systems. Future trends are then discussed in Section 7. A summary of the state of the art in tunnel inspection using CV concludes this publication.

2. Tunnel profile deformation

The deformation of a tunnel's cross-section indicates the structural condition of the tunnel in general. Measurement and monitoring of the tunnel profile are thus critical proactive maintenance activities to ensure tunnel safety. Several methods can be used to measure tunnel profiles such as mechanical gauge, tape extensometer, Terrestrial Laser Scanning (TLS) (van Gosliga et al., 2006; Jian et al., 2012; Kang et al., 2012) and geodetic instruments.

A tunnel profile measurement system that makes use of physical indicators was proposed in Scaioni et al. (2014). Relative deformations of traversal cross-section of tunnels are achieved by installing targets on the tunnel vault and measuring their coordinates in images captured along the wall. First, targets are independently measured on the images with the Least Square Template Matching (LSTM) Gruen (1985) algorithm. The 3D coordinates of the targets are then estimated using free-net bundle adjustment Luhmann et al. (2013). Finally, the scale ambiguity is removed using an invar wire and gauge as shown in Fig. 1 and the relative distances between the targets are computed. Photogrammetric levelling using a calibrated camera and photogrammetric rods and three circular targets as shown in Fig. 2 is then used for the measurement of vertical deformations. In these experiments, the camera was set on a topographic tripod to avoid blurring effects, making it inadequate for moving platforms.

A solution that installs physical objects in the tunnel is not an optimal one as it is installation and the maintenance is time consuming, especially in long and wide tunnels. The following works, instead focused on using laser light projections to create 'virtual targets' instead of physical ones.



Fig. 1. The invar wire and the gauge used to remove the scale ambiguity in Scaioni et al. (2014). Reprinted by permission from: Springer Nature Earth Science Informatics (Photogrammetric techniques for monitoring tunnel deformation, M. Scaioni, L. Barazzetti, A. Giussani, M. Previtali, F. Roncoroni, M. Alba), © (2014).

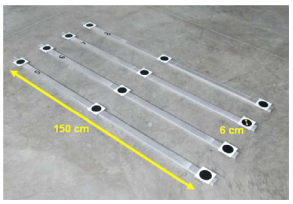


Fig. 2. An example of the basic operational scheme of 'photogrammetric levelling' as used in Scaioni et al. (2014). Reprinted by permission from: Springer Nature Earth Science Informatics (Photogrammetric techniques for monitoring tunnel deformation, M. Scaioni, L. Barazzetti, A. Giussani, M. Previtali, F. Roncoroni, M. Alba), © (2014).

The tunnel cross-section measurement method proposed in Wang et al. (2010) makes use of the profile-image method proposed earlier in Wang et al. (2009). This method uses laser pointers to beam the surface and capture the resulting tunnel profile using a camera. It is important that the planes of the laser-lit profile and the camera image are parallel, thus, Wang et al. (2010) parallelizes the image, by locating all the calibration points on the periphery of the profile as shown in Fig. 3 instead of adjusting the camera orientation as in Wang et al. (2009). The transformation relationship of the global 3D coordinates and the local 2D coordinates is found using perspective projection.

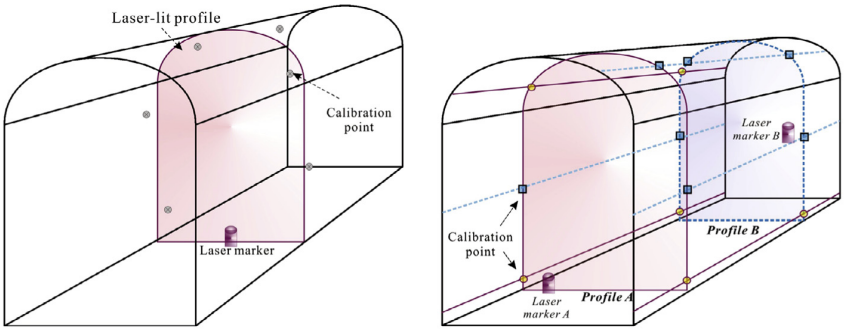
Multiple structured light projectors and cameras mounted on a dedicated vehicle, were used for 3D tunnel clearance inspection in Shen et al. (2015). The optical triangulation principle is used to reconstruct the 3D metric information of the tunnel. This is achieved by a global calibration, whereby the intrinsic and extrinsic parameters of each neighboring camera are found using the pinhole camera model. Based on this, the mapping of a 3D world point $P = (x\ y\ z\ 1)^T$ to a 2D image point $p = (u\ v\ 1)^T$ can be described by:

$$sp = A(R\ T)P, \quad A = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

where s is a non-zero arbitrary scale factor, A contains the intrinsic parameters, including α and β which are the scale factors in the image axes u and v. The principal point is represented by (u_0, v_0) and γ is the skew of the two image axes. The extrinsic parameters are represented by R for the rotation and T for the translation. The equation of the structured light plane is then used together with the latter found parameters to obtain the global model.

Similarly, in Ai et al. (2016), a set of CCD cameras and a laser emitter were placed on a cart fitted with a distance encoder. The laser emitter generates a plane perpendicular to the longitudinal axis of a metro tunnel and when the cart reaches a predefined location, the cameras are synchronously triggered to acquire images of the profile. Using photogrammetry-based algorithms and the related geometry equations, features of the sectional profile are obtained using a transmissive projection system. First, the calibration of the coordinate system transformation is conducted using an exterior target. Two neighboring cameras capture a picture of the target. Next, IP is applied on the acquired images to convert the color image to grayscale. Binary images are obtained using thresholding (TH) segmentation and used to extract the geometric information. The coordinates of points from the obtained profile are fitted to an ellipse using the Least-Squares method and the severity of the deformation is directly analyzed. The accuracy of the this system is about ± 25 mm at a speed of at least 5 km/h.

The use of laser pointers and cameras is becoming increasingly common to measure tunnel profiles as a cheaper, simpler and faster method to conduct than using targets, gauges and geodetic instruments.



(a) Original profile-image method proposed in [19] (b) Improved profile-image method in [18]

Fig. 3. Schematic illustration of the profile-image method proposed in (a) Wang et al. (2009) and (b) the improved method in Wang et al. (2010). Reprinted from Tunnelling and Underground Space Technology, vol. 25, No. 1, T.T. Wang, J.J. Jaw, C.H. Hsu, F.S. Jeng, Profile-image method for measuring tunnel profile - improvements and procedures, pp. 78–90, © (2010) with permission from Elsevier.

However, the precision of these approaches is highly influenced by the type of laser used, number of calibration points and contrast of the laser profile to the background.

As previously discussed, during the last few decades there have been multiple efforts to replace traditional methods for tunnel profile monitoring with vision-based approaches as recorded in literature and used in commercial products, such as the Tunnel & Clearance products from MERMEC MERMEC (2014). However, still, literature on tunnel clearance measurement based on IP is lacking and TLS is still the most commonly adopted method. Considering the capacity of TLS to automatically scan large tunnels and generate 3D models together with the ability to monitor work progress, it is still preferred over image-based methods.

3. Tunnel interior visualization

Visualization, in this context, is a means of organizing large image datasets to create a layout plan of tunnel lining, to aid inspection. Technical condition evaluation can be conducted offline and analyzed further using digital processes on the constructed models, reducing the presence of personnel in the tunnels while providing a more objective observation. Various means of visualizing a tunnel layout exist, including: image stitching/mosaicking and 3D reconstruction.

3.1. Image stitching/mosaicking

During tunnel inspection, a large amount of photographic data is generated, which needs to be effectively organized. A typical solution is to apply image mosaicking. This technique stitches individual images together to form a larger image, hence reducing the number of images requiring successive inspection. Moreover, having a larger field of view of the tunnel surface can help identifying minor faults such as fine cracks, which would have otherwise been missed in the context of the original small image.

Mosaicking applications spatially align the images such that they are on the same coordinate system. They are then blended together on a common canvas to form the final mosaic image. A lot of previous work (Pravenaa, 2016; Arya, 2015; Shaskank et al., 2014; Ghosh and Kaabouch, 2016) exists in the field of image stitching and mosaicking, however, these mainly dealt with natural images rather than tunnel environments where images lack brightness, contrast and features. The following is a review of image mosaicking in tunnel environments.

A system composed of line sensor cameras was proposed in Ukai (2007) to create panoramic images of a tunnel surface. The acquired images are spliced together by detecting characteristic points based on differences in color and texture followed by matching. Unfortunately further details on the algorithms used were not presented. Although the example image given in Ukai (2007) shows the matched points, the algorithm used to obtain these results is not described. Furthermore, the transformation matrix, if any, used to align the images in order to join them, is not mentioned.

During automated tunnel inspections, images are normally captured from a moving platform, therefore it is highly unlikely that images during different inspections are taken from the same points, requiring a position offset correction. In Attard et al. (2017) image mosaicking is proposed as a means for this correction. A shading correction is applied as a pre-processing step to adjust the uneven illumination. The method then uses binary edges as features for image registration via template matching. After alignment, the images are attached to each other to form a single image.

A modified subway train carriage fitted with nine line scan cameras placed at different angles and five laser light sources was designed in Zhang et al. (2014). The acquired discontinuous images are stitched together into a mosaic for subsequent IP. Assuming a horse-shoe shaped subway tunnel geometry model and the fact that the covering area of each image is also fixed, the images are directly put together. Although the mosaic is presented as having low complexity and low execution time, it lacks the presentation of details how warping to correct for the roundness of the tunnel structure is achieved if any.

Similarly, an image mosaicking algorithm using the equation of a horse-shoe-shaped cross section was proposed in Lee et al. (2013). Consumer level Digital Single-Lens Reflex (DSLR) cameras are used to capture wall images. Laser markers, are used to provide control points used in mosaicking. The images are rectified for geometric distortion and spliced together to form a mosaic which can then be used for manual inspection and identification of cracks as shown in Fig. 4.

Rather than using geometry to create a parametric model of the tunnel surface and then transform the 3D coordinates to this model, information from Structure-from-Motion (SfM) Granshaw (2018) was used in the following works to register the images into a common coordinate frame.

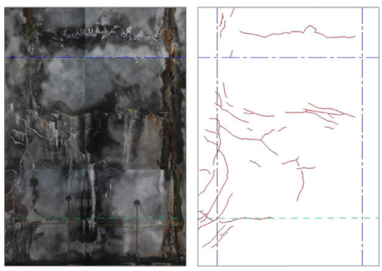


Fig. 4. Mosaic image and manually identified cracks in Lee et al. (2013). Reprinted from Tunnelling and Underground Space Technology, vol. 34, C.H. Lee, Y.C. Chiu, T.T. Wang, T.H. Huang, Application and validation of simple image-mosaic technology for interpreting cracks on tunnel lining, pp. 61–72, © (2013) with permission from Elsevier.

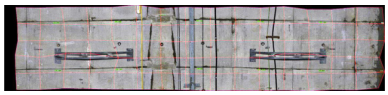


Fig. 5. Sample mosaic highlighting individual image segments in red Stent et al. (2015). Reprinted from Proceedings of the 32nd International Symposium on Automation and Robotics in Construction, S. Stent, C. Girerd, P. Long, R. Cipolla, A low-cost robotic system for the efficient visual inspection of tunnels, pp. 1–8, © (2015) with permission from the International Association for Automation and Robotics in Construction (IAARC).

The system in Stent et al. (2015), consisting of two DSLR cameras and LED arrays on a rotating unit, was set up on a monorail track to capture an area of a tunnel wall. Inferred camera poses for each image and sparse 3D point clouds are produced by a SfM pipeline based on Wu (2013). Using this information, a cylindrical projection is used to map the 2D coordinates to a 2D image mosaic plane defined by the translation of the robot along the tunnel and the angle of rotation around the sensor axis. Central quadrilaterals are selected from individual images and pieced together with planar perspective warping to produce a jigsaw-like image as displayed in Fig. 5. In addition to this, a simple image filtering pipeline is employed to retrieve location barcode stickers on the wall to create a catalogue of tunnel segments.

A system that mosaics the tunnel images via robust quadric surface estimation was presented in Chaiyasarn et al. (2013). Wall images are input to the sparse multi-view reconstruction algorithm based on the work in Wu (2011), to find an estimate of the 3D model. A support vector machine (SVM) classifier is then applied to discriminate the tunnel surface points from non-surface points. The feature vectors used as input to the SVM are extracted using the scale-invariant feature transform (SIFT) algorithm. Given the constraints on the image collection process where cameras are put inside the cylinder and each ray intersects the surface in only a single visible point, defining a bijection between an image sample and a point on the surface, these allow a warping definition producing the flattened versions of the input images. The warped images are then stitched using Microsoft Image Compositor Editor (ICE) Microsoft Research (2017).

The SVM classifier used in the latter approach, requires training for different tunnel environments. Moreover, this method uses a cylinder as the surface proxy for warping which leads to notable distortion for non-cylindrical tunnel profiles. Taking this into consideration, the algorithm proposed in Zhu et al. (2016) models the projection surface using tunnel design geometry instead. Images are first fed to the incremental SfM software Wu (2011) to reconstruct the 3D scene, generating a sparse 3D point cloud. In addition, the 3D tunnel lining shape is also modelled using streamline sweeping along a path line. This projection surface and the point cloud are input to the shape estimation component, finding a rigid transformation which ensures the best overlap between the point cloud and the target shape. The images are hence rectified and later stitched using Microsoft Research (2017). This approach is able to create a layout panorama of tunnels including non-circular shaped ones.

3.2. 3D reconstruction

Having a 3D model of the tunnel provides an actual comprehensive visual and geometric image of its environment. It is useful to examiners in terms of contextualising the location of damages found during inspection. The provision of 3D information further facilitates the evaluation of defects relative to the neighboring areas.

During the last decade, TLS has been the most commonly adopted method to survey tunnel surfaces, providing sufficient data to be able to reconstruct the true geometry of a tunnel. A commercial system using this approach is Pavemetrics TM (2017) and an overview on the use of laser scanning can be found in Frohlich and Mettenleiter (2004). Concurrently, such 3D models lack image data that would be more useful for a thorough inspection of the tunnel. Photogrammetry methods on the other hand require significantly cheaper and probably smaller equipment while providing image data.

The fusion between active and passive imaging sensors was proposed in Paar et al. (2005) to combine the benefits of both technologies to generate dense and high resolution surface reconstruction. In order to fuse the data, the orientation of the camera must be known in the scanners coordinate system, thus both data are projected on a common regular grid on the tunnel surface. Once projected, Hierarchical Feature Vector Matching (HFVM) Paar and Polzleitner (1992) is applied to match the interest points. Using this correspondence, the laser information and the RGB camera data are combined to fill the surface grid, creating a 3D textured model of the tunnel walls.

In Bauer et al. (2015), a 3D surface model is produced using a single camera with a 24 mm lens. Six images are acquired to cover the whole tunnel profile with adequate overlaps. A laser pointer is used to signalize targets for geo-referencing and alignment. Tie points are identified in the images using speeded up robust features (SURF), and the correspondences are found using Fast Library for Approximate Nearest Neighbors (FLANN) Muja and Lowe (2009). Local block bundle adjustment is then applied to create a locally consistent set of orientations for all the images involved. The targets are detected in the images using Hough transform and morphological operations. Global orientation of all the images is then conducted using a seven parameter transformation between the known 3D signalized points from a TS and their local 3D coordinates from local bundle adjustment. Dense stereo matching Gerhard Paar (1994) is then utilized and disparities are used to project the resulting 3D textural information on a surface grid.

In Stent et al. (2016), SfM techniques are used to recover the 3D geometry and model of the tunnel by locally fitting quadratic surfaces to the resulting point cloud. A 3D wire frame surface model is generated and then texture mapped with captured images. The 3D models from the previously-mentioned two systems are later used to detect changes occurring on the tunnel walls. Similarly, in Jenkins et al. (2017), 3D reconstruction is achieved using SfM techniques. An array of cameras and lights are used to acquire generic tunnel surface images. In order to deal with different tunnel geometries, rather than assuming a cylindrical shape as in Stent et al. (2016) or Chaiyasarn et al. (2013), the authors utilize the SfM algorithm in Wu (2013) which does not rely on geometric priors. A dense point cloud is generated and processed to create a 3D mesh frame which is later textured using the same images fed to the SfM.

To further analyze the cracks detected on the tunnel wall, in Protopapadakis et al. (2016) a stereo camera pair captures images which are later used to create a full 3D reconstruction of high fidelity models of the areas of cracks. On the other hand, Jahanshahi and Masri (2012) and Torok et al. (2013)use 3D scene reconstruction to do the actual crack detection in general structures.

4. Crack and defect detection

During concrete tunnel inspection, the most sought after defects are cracks as they are the earliest indications of structure degradation. If cracks are identified at an early stage, preventive measures can be taken to avoid larger infrastructural damages as well as to avoid accidents that might otherwise take place. Several factors can cause cracks in tunnels, amongst which are: ageing, fluctuations between contraction and expansion of concrete due to temperature changes, heavy seasonal rainfall, topographic change, cyclic weight loading and poor repair.

The conventional approach for lining defects detection and monitoring involves physical visual inspection, manual sketches and physical measurements on site. Such a method depends on the inspectors' knowledge and experience, lacking objectivity. Therefore, the use of IP for crack detection and monitoring has been studied considerably to provide systems and techniques to be able to objectively assess the status of cracks in concrete structures Wang and Huang (2010). A

design study, supporting analysis, visualization and rendering of cracks on tunnel linings can be found in Ortner et al. (2016). Approaches using binary TH and morphological operations are the most commonly used. Although literature dealing specifically with crack detection in tunnels is limited, various works using IP for crack detection were realised in other infrastructure fields such as pavements and bridges. Even though the scenario conditions might be different with respect to the image acquisition, illumination present and other limitations, certain basic principles still hold and can be applied to crack detection in tunnels. A further review of image-based methods used for crack detection in general concrete surfaces can be found in Wang and Huang (2010), Koch et al. (2015), and Mohan and Poobal (2017).

4.1. Methods based on TH techniques and morphological operators

Generally, crack areas are darker than those of their surroundings, resulting in lower intensity values compared to the background. Such a property allows TH techniques to be used as a first step to segment the image and extract potential crack features.

In Ukai and Nagamine (2007), a tunnel scanner capable of taking panoramic annular images of the tunnel lining was proposed. It consists of multiple line sensor cameras and lighting mounted on a rail-car that can be driven at speeds of 10–30 km/h. Once the acquisition is completed, images are pre-processed to correct for blurring and misalignment. Utilising the existence of luminance gradient variations along the line edges, cracks with larger luminance variation are selected. A hysteresis threshold method is then applied to select only edges joined to others detected by high threshold values. The resulting image contains segments of line pixels implying cracks, for which the width, height and direction are found. When two or more cracks connect to generate a closed crack region, the risk of exfoliation and concrete failure is higher, thus a blob region analysis is further employed to detect such regions.

An automated crack detection method based on a wireless multimedia sensor network was developed for subway tunnels in Shen et al. (2015). The system is composed of vehicular wireless multimedia sensor nodes, sink node stations and a data centre. Each sensor node, having a laser source and CCD cameras, captures an image, stores it, performs binarization and compresses it to keep up with the limited bandwidth. When the train arrives at a sink node, it transfers the image to this station which in turn sends it to a central data centre. This central server processes the data using a crack detection algorithm. Images are preprocessed using median filtering and high cap transformation. The Otsu method Otsu (1979) for threshold segmentation is then applied to crack regions. Crack width, length and areas are calculated from the resulting image such as that in Fig. 6. The latter properties are compared against respective thresholds to distinguish true crack areas.

Generally, crack information occupies only a small portion of the images, making it difficult to distinguish from the background. Furthermore, the inter-class variance between the background and the crack is affected by other items on the wall such as pipes and cables. To counteract these problems, a block binarization was proposed in Qi et al. (2014). Histogram equalisation, median filtering and top-hat transformation Serra (1983) are applied to enhance the contrast and remove the noise from the images. Segmentation is then performed through local binarization using the average intensity value of a square region of pixels as the threshold. Areas with a number of pixels lower than a pre-defined value are eliminated. The algorithm is simple and easy to implement however no detailed results were presented in Qi et al. (2014). The authors suggest that machine learning can be used for automatic threshold setting.

In Zhang et al. (2014), tunnel wall images are first merged into a mosaic and then smoothed by an averaging filter to reduce the noise. A black top hat transformation Serra (1983) is then applied to detect local dim regions. Crack segmentation is then achieved through a TH operation. The resulting binary images give an indication of the cracks present. However, there are some local dark regions misidentified as cracks. Thus, an opening operation is applied to filter these irrelevant regions. The crack detection and recognition system proposed in Zhiwei et al. (2002), makes use of two thresholds to produce a binarized image. The resulting object edge image shows crack areas distinguished from the background. Such an image is then processed further to classify whether the areas are actually cracks or not.

A rig of CCD line scan cameras as shown in Fig. 7 to capture grayscale images and identify crack defects was used in Huang et al. (2017). The images are first pre-processed to compensate for the image shift caused by vibration of the moving platform. The original image is divided into local image elements. The grayscale values of the pixels are used to calculate the brightness and contrast of the area. The overall gray value difference of the region is calculated and the product of the latter two values results in a value ranging between 1 and 0. The lower the value, the greater is the probability that the area includes a crack. This difference value is compared to a pre-defined value and if it is below this value, the centre pixel is recorded as a crack seed. A crack is recognized by the connecting line between the crack seeds.

Rather than intensity data, Yu et al. (2016) used infrared images to detect tunnel lining surface cracks through a three-step method. First, the images are pre-processed in the frequency domain. Each image is then divided into sub-blocks and the directional dependence of the texture in each of them is calculated. The optimum threshold is obtained by an iterative method. Crack areas are identified through a comparison against this threshold value and the cracks in each sub-block are connected.

Threshold-based methods for crack detection in general concrete structures include: Ito et al. (2002), Hu et al. (2012,), Fujita and Hamamoto (2011), Su (2013), Lee et al. (2013), Dorafshan and Maguire (2016), and Ayaho et al. (2007). The TH technique is relatively simple and computationally inexpensive rendering it as the most commonly used, at least in preliminary stages of crack detection. On the other hand, its accuracy highly depends on the predefined value at which the

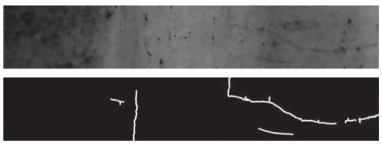


Fig. 6. Original image and crack segmentation result from Shen et al. (2015). Reprinted from International Journal of Distributed Sensor Networks, vol. 11, No. 6, B. Shen, W. Zhang, D. Qi, X. Wu, Wireless Multimedia Sensor Network Based Subway Tunnel Crack Detection Method, © (2015) This is an open access article distributed under the Creative Commons Attribution License.

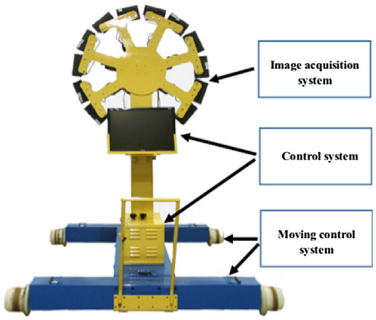
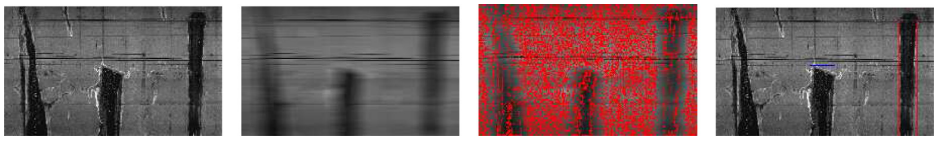


Fig. 7. Image acquisition system used in Huang et al. (2017), to capture images and then identify tunnel defects. Reprinted from Advanced Engineering Informatics Journal, vol. 32, H. Huang, Y. Sun, Y. Xue, F. Wang, Inspection equipment study for subway tunnel defects by grayscale image processing, pp. 188–201, © (2017) with permission from Elsevier.



(a) original image (b) smoothed image (c) thresholded image (d) output image

Fig. 8. Detection of water leakage using the method in Ukai and Nagamine (2007). (a) Original image showing generation of a water leakage (b) Stronger smoothing in the horizontal direction (c) Dynamic threshold value processing to extract darker regions (d) Minimum bounding rectangles around water leakage areas. Reprinted from Railway Technical Research Institute, M. Ukai, A High-performance Inspection System of Tunnel Wall Deformation Using Continuous Scan Image, 2011© (2016) with permission from International Union of Railways (UIC).

threshold is set, implying some difficulty in scenarios where crack sizes vary considerably.

4.2. Texture analysis methods

Visual texture is a vital characteristic that can be used to distinguish one surface from another. In addition, changes in texture along a surface can be used to identify defects or flaws in it. Using texture-analysis, a crack detection method using a rotation invariant Gabor Filter was suggested in Medina et al. (2017). This method allows cracks to be analyzed at the pixel level and to be detected regardless of their direction. The filter's parameters are set using a modified genetic algorithm based on the Differential Evolution optimization method. A railroad mobile platform equipped with a linear camera was developed to test this method. Gabor filters were also used to detect cracks in pavements in Salman et al. (2013).

4.3. Pattern recognition methods

Various image-based crack detection algorithms recorded in literature make use of pattern recognition techniques. In Zhang et al. (2014), the crack areas identified by the TH stage are analyzed through features such as standard deviation of shape distance histogram, pixel number and average gray level. Such features are used as inputs to a radial basis function neural network (RBF), Extreme Learning Machine (ELM), SVM and K-nearest neighbor (KNN) algorithm to classify the candidate objects as cracks or not. The different classifiers achieved similar accuracies, with ELM being the most accurate. Following binarization, the image is then segmented into local regions and SVM is used to classify the sub-images into three types of classes: crack, non-crack and intermediate.

A defect detector using Convolutional Neural Networks (CNN) was proposed in Makantasis et al. (2015). The information extracted from the RGB images includes edges, frequency, texture, entropy and Histogram of Oriented Gradients (HOG). The CNN takes these features and constructs high-level features as inputs to a Multi-Layer Perceptron (MLP) which is trained to identify defects on the tunnel lining.

An image recognition algorithm for semantic segmentation of cracks and leakage defects of metro shield tunnel using hierarchies of features extracted by fully convolutional network (FCN) was presented in Huang et al. (2018). FCN models of crack and leakage are separately trained through several iterations of forward inference and backward learning. Following this, semantic segmentation of defect images is implemented via the corresponding FCN models using a two-stream algorithm. One stream is used to recognize the crack while the other is adopted for the leakage. Similarly, Cha et al. (2017) uses a deep architecture of CNNs for detecting concrete cracks without calculating the defect features. The algorithm was tested on tunnel RGB images captured by a DSLR camera.

In order to detect cracks in general concrete surfaces, a Support Vector Data Description (SVDD) approach was undertaken in Weiguo et al. (2017). The proposed method converts the color image to grayscale and then segments it using a threshold. A morphological closing operation is then applied on the binary image. Once, pre-processed, properties including eccentricity, circularity and packing density are compiled into a vector and input into a trained SVDD to identify cracks.

5. Water leakage detection

To ensure the safety of the concrete structure, tunnels should also be monitored for water leakages. Detection for water ingress is often performed during human visual inspection, however during the last decade, efforts to automate this process using IP were done to reduce the subjectivity and improve efficiency. Leakage detection and recognition can be treated as an object detection problem similar to crack detection, using similar IP principles. In general, leakage areas are darker than the rest due to low reflectance, where edges of leakage areas have larger gradient gray values and form closed irregular shapes.

The inspection system in Huang et al. (2017) also includes a leakage recognition component. It uses edge detection followed by the Otsu algorithm to calculate the threshold in order to binarize the image, segmenting the leakage areas from the surrounding regions. The work presented only outlines the approach taken and neither gives further details nor quantitative results of the proposed method.

Water leakage on walls is commonly identified through darker areas occurring near the ingress location, traced vertically down implying the water flow. Using, this observation, Ukai and Nagamine (2007) smooths tunnel images in the horizontal direction to eliminate pattern content in this direction. Dynamic threshold value processing is then utilized to extract regions which are darker than the background as the leakage areas, as shown in Fig. 8.

In ChuanPeng et al. (2010), color images are first converted to grayscale and then Canny Edge detection is applied to extract the edges. Non-maxima suppression is then used to remove the noise while a hysteresis threshold is utilized to obtain more accurate edges. Since objects present on the tunnel lining, such as pipes and segment joints, generally have similar intensity features, their edges will also be extracted. Taking this into consideration, a classifier using an Artificial Neural Network (ANN) was proposed to distinguish them. The gradient magnitude and orientation, RGB value, line width and line length are first extracted. Their mean and variance values are then used as inputs to the ANN. The detection result is claimed to be satisfactory when the background is relatively simple with few occlusions, however the performance reduces significantly in the presence of more complex backgrounds and light reflection.

Further to visible images, other methods currently used to find a leak include: acoustics using noise logger and sensitive microphones and tracer gas. Thermal Infrared (TIR) imagery has also been used to detect moist areas and perform leakage detection in underground pipes. In Fahmy and Moselhi (2009) a study was conducted for the detection of water leaks, and identification of their respective locations in underground pipelines using a TIR camera. In Parida et al. (2013), an automated water leakage detection system using a wireless sensor network created along the distribution pipes based on TIR was proposed. IP techniques such as edge detection are used to determine if there is a water leakage in the region of interest in the imaged zone.

Although several systems that use IP to detect water leaks in different environment scenarios were previously presented, works recorded in the specific field of tunnel inspection are very limited.

Moreover, none of the above mentioned water leakage detection systems present any quantitative results and they display only a few resulting detection images. General statistics on the detection accuracy could have been used in order to provide a better evaluation of the results from such works. These include the true positive rate (TPR) and false positive rate (FPR) given by:

$$TPR = \left(\frac{TP}{TP + FN}\right) \tag{2}$$

$$FPR = \left(\frac{FP}{FP + TN}\right) \tag{3}$$

where *TP* is the number of true leakage detection areas, *FP* is the number of falsely detected areas, *TN* is the number of leakage-free areas and *FN* is the number of missed leakage detections. Moreover, such rates could be further utilized in order to perform receiver operating characteristic (ROC) analysis, comparing the behavior of each of the above rates as they vary with each other.

6. General change detection

There are several types of faults that structural inspectors look for when inspecting a tunnel, including cracks, spalls and water ingress. As discussed in the previous sections, most of the inspection research dealt with detection of such faults rather than deformation monitoring. Sometimes it is more beneficial to study the evolution of such deformations as this gives a better indication of the structural health status of the tunnel and its deterioration if any. Deformation on the tunnel lining results in changes which are visible on the wall. Early detection of such visual changes is a critical requirement for structural failure prevention. Observing the tunnel for such changes is often the work of human inspectors. They, have to traverse the tunnel and check for any changes occurring since a previous inspection by comparing to previous records. Similar to all the manual detections mentioned previously, this is a costly and time-consuming process and given that some tunnels present adverse working environment conditions, it is very beneficial to automate this process.

Change detection in 2D images is a well researched problem, particularly in the fields of video surveillance, remote sensing and medical imaging. Reviews of such change identification methods are found in Lu et al. (2004) and Radke et al. (2005). However, literature on the detections of changes on tunnel linings is still lacking, possibly due to the challenges encountered in this area. Accurate image registration is an important prerequisite for precise image comparison for change detection. In remote sensing, GPS is commonly used for registration, however, this is unavailable in tunnels, thus, Simultaneous Localization and Mapping (SLAM) can be used to achieve this as well as to help with navigation as in Castro et al. (2014) or to assist in generating 3D models. Furthermore, images in tunnel environments are characterized by lack of features, low contrast and low brightness. Despite these problems, the following are some change detection systems that were proposed for tunnel environments.

A change detection system for railway tunnels is described in Jenkins et al. (2017). It uses an array of overlapping cameras placed on a railway trolley. Change detection is conducted by comparing an image from a previous scan as a template and the best matched image from the current scan as the query. Images are first aligned, then normalized cross-correlation based filtering is applied to detect the differences between the two images. The method is claimed to be more robust than comparing single pixels as it takes into account the surrounding pixels as well as it corrects lighting variation through mean-based intensity normalization. Despite such claims, insufficient theory details are given in each step involved.

TInspect, a tunnel wall change monitoring system making use of low-cost camera equipment placed on a train inspection monorail, was proposed in Attard et al. (2018). The system corrects for position offset variances between the query and a previous survey image using the mosaicking solution given in Attard et al. (2017). A hybrid change detection algorithm that uses image differencing, binary image pixel comparison and optical flow is then applied. Using a combined weighting model, the 'actual change' areas are identified and false detections due to parallax, misalignment and shadows are discarded. This system achieves high sensitivity and precision rates while it is able to detect changes within a resolution of around 10 cm in width and/or height. Despite the good result obtained, only a limited area of the tunnel could be monitored by one camera, and thus multiple sensors are needed to cover the whole tunnel.

An automated system using five synchronized cameras with flash units was presented in Stent et al. (2013). Inspection images are registered to a 3D tunnel surface model recovered by SfM techniques. A change map is estimated by defining a distance function between the query and its matching image from a previously obtained dataset. The information from SfM is used to form a geometric prior, mapping image locations to corresponding 3D points. Two-dimensional SIFT features are categorized in two groups, inlier (on-surface) and outlier (off-surface), based on the distance of their corresponding 3D points to their closest point on the locally fitted surface. Mean shift segmentation is then applied to the query image to delineate the image into groups of similar color and textures. Inliers and outliers contained within a pixel group vote towards its overall classification. The geometric prior implies a lower weight to detected changes where the geometry is either known to be off surface or known to be unreliable, reducing false detections of changes caused by cables and other objects on the walls.

In Stent et al. (2015), overlapping 360° rings of images are gathered by an autonomous calibrated camera running along a monorail, combined with polarized lighting and orthogonally polarized lens filters to remove or attenuate image variations due to scene secularities. SfM is used to build panoramas of the surface. Neighboring reconstructed subsets are registered across time using a similarity transform estimated via Procrustes alignment Crosilla (2003) on a subset of confident feature correspondences. A CNN architecture is used to classify the input pair between changed and unchanged states. The quantitative evaluation reported in the paper shows that the CNN outperformed the change detection implemented using absolute differencing (RGB) and NCC approach used in Stent et al. (2013) as well as manual detection when a slightly higher false positive rate was allowed.

7. Future trends

The low cost of ubiquitous photographic equipment as well as the availability of very fast processors that can be used to execute IP algorithms makes this approach a better alternative to manual inspection. Nonetheless, the use of photogrammetry has not been fully exploited yet in the field of tunnel inspection due to the various challenges in the area. In general, tunnel environments are usually characterized by low lighting, thus leading to dark and low contrast images, making them unsuitable for subsequent processing. Also, when considering tunnel walls, in particular painted ones, images contain low texture leading to a lack of features. The latter are fundamental in common IP tasks such as registration, recognition and 3D reconstruction. In order to exploit better the use of CV in tunnel inspection, investigation into new methods that improve feature extraction and matching should be done. Furthermore, the application of already established deep learning techniques applied in other fields can help recognition tasks improve the identification of faults in the tunnel structure, including cracks and water detection. In addition, the use of multiple sensors, such as thermal cameras and RGB-D cameras, generating different data and fusing them together can combine the advantages of each modality and provide better data for analysis in inspections. A recent technology being used in civil infrastructure is that of unmanned aerial vehicles (UAVs) which can also be used to inspect tunnels, making the whole process that much safer and more effective as discussed in Commercial

UAV News (2018). Several commercial UAVs that can be used for inspection are PRODRONE (2018), FLYABILITY (2018) and Orbital Technical Solutions (2018).

8. Conclusion

Due to ageing, continuous loading and other environmental factors, tunnel structures deteriorate over time, reducing the safety of such infrastructures. Thus, regular tunnel inspection is necessary to identify any faults at an early stage and perform the required maintenance. Traditional methods involved manual inspection through visual observation and measurement using geodetic devices. In order to improve inspection in terms of efficiency, safety of personnel as well as survey objectivity, there has been an increasing interest in automating such inspections. This publication provided a collective review of automated tunnel inspection systems based on IP and photogrammetric techniques. Surveys on general crack identification, change detection and image mosaicking based on IP already exist in literature, however, a comprehensive review focusing on image-based tunnel inspection was still missing. This manuscript thus contributes a study and review of the state of the art in vision-based automation used in different tunnel inspection procedures. These include: tunnel profile monitoring, crack and leakage detection as well as tunnel surface documentation and visualization. As discussed in this paper, each of these inspection tasks has been improved in different respects over time. Unfortunately some of the literature does not give extensive details of the methods implemented while others lack the presentation of results and statistics and therefore a fair comparison cannot be made. Although considerable advancements have been made in the field of tunnel inspection, the use of photogrammetry has not been fully exploited yet due to the various challenges in the area. Improving on this, recent technology innovations such as UAV's, advanced and compact photographic equipment together with the use of data fusion from multi-modal sensors are continuously being introduced to the field.

Acknowledgment

We thank various authors for their contribution in providing us with authorization to reprint some of the images published in their previous works. Each image indicates the respective permission and reference to the previous work.

References

Ai, Q., Yuan, Y., Bi, X., 2016. Acquiring sectional profile of metro tunnels using charge-coupled device cameras. Struct. Infrastruct. Eng. 12 (9), 1065–1075. https://doi.org/10.1080/15732479.2015.1076855.

Arya, S., 2015. A review on image stitching and its different methods. Int. J. Adv. Res. Comput. Sci. Softw. Eng. 05 (05), 299–303. <http://ijarcsse.com/Before_August_2017/docs/papers/Volume_5/5_May2015/V5I5-0168.pdf>.

Attard, L., Debono, C.J., Valentino, G., Castro, M.D., 2017. Image mosaicing of tunnel wall images using high level features. In: Proceedings of the 10th International Symposium on Image and Signal Processing and Analysis, pp. 141–146. https://doi.org/10.1109/ISPA.2017.8073585.

Attard, L., Debono, C.J., Valentino, G., Castro, M.D., 2018. Vision-based change detection for inspection of tunnel liners. Automat. Constr. 91, 142–154. https://doi.org/10.1016/j.autcon.2018.03.020.

Ayaho, M., Masa-Aki, K., Eugen, B., 2007. Automatic crack recognition system for concrete structures using image processing approach. Asian J. Inform. Technol. 5, 553–561. <http://docsdrive.com/pdfs/medwelljournals/ajit/2007/553-561.pdf>.

Balaguer, C., Montero, R., Victores, J.G., Martínez, S., Jardón, A., 2014. Towards fully automated tunnel inspection: a survey and future trends. In: Proceedings of the 31st ISARC, Sydney, Australia, pp. 19–33. https://doi.org/10.22260/ISARC2014/0005.

Bauer, A., Gutjahr, K., Paar, G., Kontrus, H., Glatzl, R., 2015. Tunnel surface 3d reconstruction from unoriented image sequences. In: Proceedings of the 39th Annual Workshop of the Austrian Association for Pattern Recognition (OAGM). < https://arxiv.org/abs/1505.06237 > .

Boving, K.G., 1989. Nde handbook. Butterworth-Heinemann, p. iii. https://doi.org/10.1016/B978-0-408-04392-2.50001-1. < http://www.sciencedirect.com/science/article/pii/B9780408043922500011 > .

Castro, M.D., Masi, A., Lunghi, G., Losito, R., 2014. An incremental slam algorithm for indoor autonomous navigation.

Cha, Y.J., Choi, W., Bykztrk, O., 2017. Deep learning-based crack damage detection using convolutional neural networks. Comput.-Aided Civ. Infrastruct. Eng. J. 32 (5), 361–378. https://doi.org/10.1111/mice.12263.

Chaiyasarn, K., Tae-Kyun, K., Viola, F., Cipolla, R., Soga, K., 2013. Distortion-free image mosaicing for tunnel inspection based on robust cylindrical surface estimation. J. Comput. Civ. Eng. 30 (3), 1–9. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000516.

ChuanPeng, H., HeHua, Z., XiaoJun, L., 2010. Detection of tunnel water leakage based on image processing. Inform. Technol. Geo-Eng. 254–262.

Commercial UAV News. UAVs in Civil Infrastructure. < https://www.expouav.com/news/report/uavs-civil-infrastructure/ > (online; accessed June 2018).

Crosilla, F., 2003. Procrustes Analysis and Geodetic Sciences. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 287–292. https://doi.org/10.1007/978-3-662-05296-9_29.

Dorafshan, S., Maguire, M., Automatic Surface Crack Detection in Concrete Structures Using OTSU Thresholding and Morphological Operations. Utah State University CEE Faculty Publications. https://doi.org/10.13140/RG.2.2.34024.47363.

Eastman, C., Teicholz, P., Sacks, R., Liston, K., 2008. BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors. Wiley Publishing.

Fahmy, M., Moselhi, O. Automated detection and location of leaks in water mains using infrared photography.(author abstract)(report), J. Perform. Construct. Facil. 24(3).

FLYABILITY . ELIOS, Inspect & Explore Indoor and Confined Spaces. < https://www.flyability.com/elios/ > (online; accessed June 2018).

Förstner, W., Wrobel, B.P., 2016. Photogrammetric Computer Vision – Statistics, Geometry, Orientation and Reconstruction. Springer.

Frohlich, C., Mettenleiter, M. Terrestrial laser scanning-new perspectives in 3d surveying 36. < http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.215.8213&rep=rep1&type=pdf > .

Fujita, Y., Hamamoto, Y., 2011. A robust automatic crack detection method from noisy concrete surfaces. Mach. Vis. Appl. 22 (2), 245–254. https://doi.org/10.1007/s00138-009-0244-5.

Gerhard Paar, A.A., 1994. Fast hierarchical stereo reconstruction. In: Proceedings of The International Society for Optical Engineering SPIE, vol. 2252. https://doi.org/10.1117/12.169849.

Ghosh, D., Kaabouch, N., 2016. A survey on image mosaicing techniques. J. Vis. Commun. Image Rep. 34 (C), 1–11. https://doi.org/10.1016/j.jvcir.2015.10.014.

Granshaw, S.I., 2018. Structure from motion: origins and originality. Photogrammetr. Rec. 33 (161), 6–10. https://doi.org/10.1111/phor.12237.

Gruen, A., 1985. Adaptive least squares correlation: a powerful image matching technique. South Afr. J. Photogram., Remote Sens. Cartogr. 14, 175–187. https://doi.org/10.1117/12.952246.

Huang, H., Sun, Y., Xue, Y., Wang, F., 2017. Inspection equipment study for subway tunnel defects by grey-scale image processing. Adv. Eng. Inform. J. 32, 188–201. https://doi.org/10.1016/j.aei.2017.03.003.

Huang, H., Li, Q., Zhang, D., 2018. Deep learning based image recognition for crack and leakage defects of metro shield tunnel. Tunn. Undergr. Space Technol. 77, 166–176. https://doi.org/10.1016/j.tust.2018.04.002.

Hu, D., Tian, T., Yang, H., Xu, S., Wang, X., 2012. Wall crack detection based on image processing. In: Proceedings of the Third International Conference on Intelligent Control and Information Processing, pp. 597–600. https://doi.org/10.1109/ICICIP.2012.6391474.

Ikeuchi, K. (Ed.), 2014. Computer Vision: A Reference Guide, Springer Reference. Springer. https://doi.org/10.1007/978-0-387-31439-6.

Ito, A., Aoki, Y., Hashimoto, S., 2002. Accurate extraction and measurement of fine cracks from concrete block surface image. In: Proceedings of the IEEE 28th Annual Conference of the Industrial Electronics Society. IECON 02, vol. 3, pp. 2202–2207. https://doi.org/10.1109/IECON.2002.1185314.

Jahanshahi, M.R., Masri, S.F., 2012. Adaptive vision-based crack detection using 3d scene reconstruction for condition assessment of structures. Automat. Constr. 22 (Suppl. C), 567–576. https://doi.org/10.1016/j.autcon.2011.11.018.

Jenkins, M.D., Buggy, T., Morison, G., 2017. An imaging system for visual inspection and structural condition monitoring of railway tunnels. In: Proceedings of the IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS), pp. 1–6. https://doi.org/10.1109/EESMS.2017.8052679.

Jian, L., Youchuan, W., Xianjun, G., 2012. A new approach for subway tunnel deformation monitoring high-resolution terrestrial laser scanning. In: Proceedings of the XXII ISPRS Congress, vol. XXXIX, pp. 223–228.

Kang, Z., Tuo, L., Zlatanovab, S., 2012. Continuously deformation monitoring of subway tunnel based on terrestrial point clouds. In: Proceedings of the XXII ISPRS Congress, vol. XXXIX, pp. 199–203.

Koch, C., Georgieva, K., Kasireddy, V., Akinci, B., Fieguth, P., 2015. A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure. Adv. Eng. Inform. 29 (2), 196–210. https://doi.org/10.1016/j.aei.2015.01.008. (infrastructure Computer Vision).

Lee, C.H., Chiu, Y.C., Wang, T.T., Huang, T.H., 2013. Application and validation of simple image-mosaic technology for interpreting cracks on tunnel lining. Tunn. Undergr. Space Technol. 34, 61–72. https://doi.org/10.1016/j.tust.2012.11.002.

Lee, B., Kim, Y.Y., Yi, S., Kim, J., 2013. Automated image processing technique for detecting and analysing concrete surface cracks. Struct. Infrastruct. Eng. 9 (6), 567–577. https://doi.org/10.1080/15732479.2011.593891. <http://www.tandfonline.com/doi/abs/10.1080/15732479.2011.593891>.

Linder, W., 2013. Digital Photogrammetry: Theory and Applications. Springer Berlin Heidelberg. <https://books.google.ch/books?id=icYhBQAAQBAJ>.

Lu, D., Mausel, P., Brondzio, E., Moran, E., 2004. Change detection techniques. Int. J. Remote Sens. 25 (12), 2365–2401. https://doi.org/10.1080/0143116031000139863.

Luhmann, T., Robson, S., Kyle, S., Boehm, J., 2016. Close Range Photogrammetry and 3D Imaging.
Lukins, T., Ibrahim, Y., Kaka, A., Trucco, E., 2007. Now you see it: the case for measuring progress with computer vision. In: Proceedings of the 4th International SCRI Research Symposium, pp. 409–422.
Makantasis, K., Protopapadakis, E., Doulamis, A., Doulamis, N., Loupos, C., 2015. Deep convolutional neural networks for efficient vision based tunnel inspection. In: Proceedings of the IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), pp. 335–342. https://doi.org/10.1109/ICCP.2015.7312681.
Martin, H., Chevallier, S., Monacelli, E., 2016. Adaptive visualisation system for construction building information models using saliency. CoRR abs/1603.02028. Available from arXiv:1603.02028. < http://arxiv.org/abs/1603.02028 > .
Medina, R., Llamas, J., Gómez-García-Bermejo, J., Zalama, E., Segarra, M., 2017. Crack detection in concrete tunnels using a gabor filter invariant to rotation. Sensors (Basel, Switzerland) 17 (7), 1–16. https://doi.org/10.3390/s17071670.
MERMEC, 2014. T-sight 5000. < http://www.mermecgroup.com/northamerica/pageview2.php?i=1028&sl=1 > .
Microsoft Research. Image Compositor Editor (ICE). < https://www.microsoft.com/en-us/research/product/computational-photography-applications/image-composite-editor/ > (online; accessed December 2017).
Mohan, A., Poobal, S. Crack detection using image processing: a critical review and analysis. Alex. Eng. J. https://doi.org/10.1016/j.aej.2017.01.020.
Montero, R., Victores, J., Martnez, S., Jardn, A., Balaguer, C., 2015. Past, present and future of robotic tunnel inspection. Automat. Constr. 59, 99–112. https://doi.org/10.1016/j.autcon.2015.02.003.
Muja, M., Lowe, D.G., 2009. Fast approximate nearest neighbors with automatic algorithm configuration. In: Proceedings of the VISAPP International Conference on Computer Vision Theory and Applications, pp. 331–340. < http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.160.1721 > .
Orbital Technical Solutions. Unmanned Aircraft Systems. < http://www.orbital-ots.com/drone-services/ > (online; accessed June 2018).
Ortner, T., Sorger, J., Piringer, H., Hesina, G., Gröller, E., 2016. Visual analytics and rendering for tunnel crack analysis: a methodological approach for integrating geometric and attribute data. Visual Comput. 32 (6–8), 859–869. https://doi.org/10.1007/s00371-016-1257-5.
Otsu, N., 1979. A threshold selection method from gray-level histograms. IEEE Trans. Syst., Man, Cybernet. 9 (1), 62–66. https://doi.org/10.1109/TSMC.1979.4310076.
Paar, G., Polzleitner, W., 1992. Robust disparity estimation in terrain modeling for spacecraft navigation. In: Proceedings of the 11th IAPR International Conference on Pattern Recognition, pp. 738–741. https://doi.org/10.1109/ICPR.1992.201666.
Paar, G., Bauer, A., Kontrus, H., 2005. Texture-based fusion between laser scanner and camera for tunnel surface documentation. In: Proceedings of the 7th International Conference on Optical 3-D Measurement Techniques. < http://dibweb.joanneum.at/group_3DVision/3DVision/publications-presentations/literature/pdfs/PUB05DIB005.pdf > .
Parida, R.K., Thyagarajan, V., Menon, S., 2013. A thermal imaging based wireless sensor network for automatic water leakage detection in distribution pipes. In: 2013 IEEE International Conference on Electronics, Computing and Communication Technologies, pp. 1–6.
Pavemetrics TM. Laser Tunnel Scanning System (LTSS). < http://www.pavemetrics.com/wp-content/uploads/2016/03/LTSS_Flyer.pdf > (online; accessed December 2017).
Pravenaa, S., 2016. A methodical review on image stitching and video stitching techniques. Int. J. Appl. Eng. Res. 11 (5), 3442–3448. <https://www.ripublication.com/ijaer16/ijaerv11n5_80.pdf>.
PRODRONE . Tunnel Inspection Drone. < https://www.prodrone.com/release-en/2845/ > (online; accessed June 2018).
Protopapadakis, E., Stentoumis, C., Doulamis, N., Doulamis, A., Loupos, K., Makantasis, K., Kopsiaftis, G., Amditis, A., 2016. Autonomous robotic inspection in tunnels III-5, pp. 167–174. < https://www.researchgate.net/publication/307530827_AUTONOMOUS_ROBOTIC_INSPECTION_IN_TUNNELS > .
Ptrucean, V., Armeni, I., Nahangi, M., Yeung, J., Brilakis, I., Haas, C., 2015. State of research in automatic as-built modelling. Adv. Eng. Inform. 29 (2), 162–171. https://doi.org/10.1016/j.aei.2015.01.001. (infrastructure Computer Vision).
Qi, D., Liu, Y., Gu, Q., Zheng, F. An algorithm to detect the crack in the tunnel based on the image processing. J. Comput. 26(3). < http://www.csroc.org.tw/journal/JOC26_3/JOC26_3_2.pdf > .
Radke, R.J., Andra, S., Al-Kofahi, O., Roysam, B., 2005. Image change detection algorithms: a systematic survey. IEEE Trans. Image Process. 14 (3), 294–307. https://doi.org/10.1109/TIP.2004.838698.
Salman, M., Mathavan, S., Kamal, K., Rahman, M., 2013. Pavement crack detection using the gabor filter. In: Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), pp. 2039–2044. https://doi.org/10.1109/ITSC.2013.6728529.
Scaioni, M., Barazzetti, L., Giussani, A., Previtali, M., Roncoroni, F., Alba, M., 2014. Photogrammetric techniques for monitoring tunnel deformation. Earth Sci. Inf. 7 (2), 83–95. https://doi.org/10.1007/s12145-014-0152-8.
Serra, J., 1983. Image Analysis and Mathematical Morphology. Academic Press, Inc., Orlando, FL, USA.
Serra, J., 1983. Image Analysis and Mathematical Morphology. Academic Press, Inc., Orlando, FL, USA.
Shaskank, K., Chatianya, N., Manikanta, G., Balaji, C., Murthy, V.V.S.A., 2014. A survey and review over image alignment and stitching methods. Int. J. Electron. Commun. Technol. (IJECT) 05 (03), 20–52. <http://www.iject.org/vol5/spl3/ec1152.pdf>.
Shen, B., Zhang, W., Qi, D., Wu, X. Wireless multimedia sensor network based subway tunnel crack detection method. Int. J. Distrib. Sensor Networks 11(6). https://doi.org/10.1155/2015/184639.
Stent, S., Gherardi, R., Stenger, B., Soga, K., Cipolla, R., 2013. An Image-Based System for Change Detection on Tunnel Linings, pp. 2–5.
Stent, S., Girerd, C., Long, P., Cipolla, R., 2015. A low-cost robotic system for the efficient visual inspection of tunnels. In: Proceedings of the 32nd International Symposium on Automation and Robotics in Construction, ISARC, pp. 1–8. https://doi.org/10.22260/ISARC2015/0070.
Stent, S., Gherardi, R., Stenger, B., Cipolla, R., 2015. Detecting change for multi-view, long-term surface inspection. In: Proceedings of the British Machine Vision Conference (BMVC), pp. 127–139. https://doi.org/10.5244/C.29.127.
Stent, S., Gherardi, R., Stenger, B., Soga, K., Cipolla, R., 2016. Visual change detection on tunnel linings. Mach. Vision Appl. J. 27 (3), 319–330. https://doi.org/10.1007/s00138-014-0648-8.
Su, T., 2013. Application of computer vision to crack detection of concrete structure. Int. J. Eng. Technol. 5 (4), 457–461. https://doi.org/10.7763/IJET.2014.V5.596.
Szeliski, R., 2011. Computer vision algorithms and applications. https://doi.org/10.1007/978-1-84882-935-0.
Torok, M., Golparvar-Fard, M., Kochersberger, K. Image-based automated 3d crack detection for post-disaster building assessment 28. < https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29CP.1943-5487.0000334 > .
Ukai, M., 2007. Advanced inspection system of tunnel wall deformation using image processing. Quart. Rep. RTRI 48 (2), 94–98. https://doi.org/10.2219/rtriqr.48.94.
Ukai, M., Nagamine, N. A High-performance Inspection System of Tunnel Wall Deformation Using Continuous Scan Image. Railway Technical Research Institute. < http://www.railway-research.org/IMG/pdf/poster_ukai_masato.pdf > .
van Gosliga, R., Lindenbergh, R., Pfeifer, N., 2006. Deformation analysis of a bored tunnel by means of terrestrial laser scanning. In: Proceedings of the ISPRS Commission V Symposium Image Engineering and Vision Metrology.
Wang, P., Huang, H., 2010. Comparison analysis on present image-based crack detection methods in concrete structures. In: Proceedings of the 3rd International Congress on Image and Signal Processing, CISP, vol. 5, pp. 2530–2533. https://doi.org/10.1109/CISP.2010.5647496.
Wang, T.T., Jaw, J.J., Chang, Y.H., Jeng, F.S., 2009. Application and validation of profile-image method for measuring deformation of tunnel wall. Tunn. Undergr. Space Technol. 24 (2), 136–147. https://doi.org/10.1016/j.tust.2008.05.008.
Wang, T.T., Jaw, J.J., Hsu, C.H., Jeng, F.S., 2010. Profile-image method for measuring tunnel profile - improvements and procedures. Tunn. Undergr. Space Technol. 25 (1), 78–90. https://doi.org/10.1016/j.tust.2009.09.005.
Weiguo, L., Yaru, L., Fang, W., 2017. Crack detection based on support vector data description. In: Proceedings of the 29th Chinese Control And Decision Conference (CCDC), pp. 1033–1038. https://doi.org/10.1109/CCDC.2017.7978671.
Wu, Changchang, 2011. VisualSFM: A Visual Structure from Motion System. < http://ccwu.me/vsfm/ > (online; accessed December 2017).
Wu, C., 2013. Towards linear-time incremental structure from motion. In: Proceedings of the International Conference on 3D Vision - 3DV, pp. 127–134. https://doi.org/10.1109/3DV.2013.25.
Wu, C., 2013. Towards linear-time incremental structure from motion. In: 2013 International Conference on 3D Vision - 3DV 2013, pp. 127–134. https://doi.org/10.1109/3DV.2013.25.
Yu, T., Zhu, A., Chen, Y., 2016. Efficient crack detection method for tunnel lining surface cracks based on infrared images. J. Comput. Civ. Eng. 31, 04016067. <https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29CP.1943-5487.0000645>.
Zhang, W., Zhang, Z., Qi, D., Liu, Y., 2014. Automatic crack detection and classification method for subway tunnel safety monitoring. Sensors 14 (10), 19307–19328. https://doi.org/10.3390/s141019307.
Zhiwei, L., Suandi, S.A., Ohashi, T., Ejima, T., 2002. Tunnel crack detection and classification system based on image processing. In: Proceedings of the SPIE Machine Vision Applications in Industrial Inspection, vol. 4664. https://doi.org/10.1117/12.460191.
Zhu, Z.H., Fu, J.Y., Yang, J.S., Zhang, X.M., 2016. Panoramic image stitching for arbitrarily shaped tunnel lining inspection. Comput.-Aided Civ. Infrastruct. Eng. 31 (12), 936–953. https://doi.org/10.1111/mice.12230.

This full text paper was peer-reviewed at the direction of IEEE Instrumentation and Measurement Society prior to the acceptance and publication.

A comprehensive virtual reality system for tunnel surface documentation and structural health monitoring

Leanne Attard *, Carl James Debono †, Gianluca Valentino †
Department of Communications and Computer Engineering
University of Malta
Msida, Malta
*leanne.attard@um.edu.mt †c.debono@ieee.org ‡gianluca.valentino@um.edu.mt

Mario di Castro §, John Andrew Osborne ¶, Luigi Scibile ‖
§ *Engineering Department* ¶ ‖*Site Management and Buildings Department*
CERN
Meyrin, Switzerland
§mario.di.castro@cern.ch ¶ john.andrew.osborne@cern.ch ‖ luigi.scibile@cern.ch

Manuel Ferre
Centre for Automation and Robotics
Universidad Politcnica de Madrid
Madrid, Spain
m.ferre@upm.es

***Abstract*—Infrastructures may develop defects over time and thus periodic monitoring is required to evaluate their health. Structural inspection of such constructions can sometimes be restricted due to short time windows in which humans can access the area as well as due to various hazardous conditions that may be present. This work advances the state of the art in structural inspection by contributing to the field of robotics, vision and inspection by proposing a comprehensive system to provide a better means of surface documentation and to aid structural health monitoring. A mobile robotic platform is equipped with one or more cameras to capture images of walls. Such images are then reconstructed into a 3D model that can be visualised through Virtual Reality (VR). The model can then be further analysed via subsequent image processing stages. Although the prime purpose of the system is for deployment in tunnels, it can be adapted to various other scenarios.**

***Index Terms*—robotic tunnel inspection, visualisation, image processing, virtual reality**

I. Introduction

The underground infrastructure is becoming increasingly important to use the available land area more efficiently. A broad variety of tunnel projects ranging from railways, subways and roads to pipelines, mines and further on to tunnels hosting equipment used for research experiments, have been developed. Due to ageing, ground motion, environmental elements, structural stress such as increased loading as well as neglect, including poor maintenance and deferred repairs, such underground structures may develop defects over time. The latter consist of opening of joints, concrete reinforcement corrosion, cracks, spalls and even deformation of the tunnel profile. Consequently to ensure safety, periodic monitoring of such tunnels is needed. Traditionally, structural inspection is carried out manually through visual surveys by inspectors. Due to surveillance conditions and large-scale requirements, this approach is challenging and demanding. It involves a high amount of time, high cost for hiring the personnel and is human subjective. Considering this, recently, the automation of structural health monitoring received significant consideration.

By using robotics, photographic gear and visualization techniques, several systems were built to carry out varying automated inspection tasks. We propose a vision-based robotic system to gather image data from a tunnel for its surface documentation and use such data for structural monitoring purposes. It automatically captures images of the tunnel walls, feeds them into a reconstruction module to create the 3D model of the tunnel surface and visualizes the latter through VR. In this paper, the LHC tunnel [1] of the European Organization for Nuclear Research (CERN), was used as a tunnel environment scenario.

The rest of the paper is as follows. The state of the art relevant to the approaches used in the proposed system is presented in Section II. An overview of the system is given in Section III. Section IV describes the image capturing component. The 3D image reconstruction is discussed in section V while the visualization element is explained in section VI. A summary and an indication of future work conclude the paper.

978-1-5386-6628-9/18/$31.00 ©2018 IEEE

II. Background Information

During the last few decades there were several efforts in automating inspection of tunnel infrastructure through the use of laser scanners, photography equipment and robotics. General surveys on existing automated tunnel inspection systems can be found in [2] and [3].

In this paper we propose a robotic system composed of three modules. The first component automatically captures images which are then reconstructed into a 3D model by the second module. The final module visualizes the model through VR. The following sub-sections provide some background information on each of these system parts.

A. Image Acquisition

The choice of the image acquisition system is generally dependent on the constraints present in the particular scenario. Theses include space limitations, available time and environmental conditions. Related works in vision-based tunnel inspection have used different equipment for data acquisition. For the detection of cracks, the system in [4] uses a line-scanning camera. A line-scanning camera array combined with powerful lighting was used in [5]. Other systems using CCD line-scan cameras were proposed as in [6]. A fish-eye camera and structured light were utilised in [7] for inspection as well as to find the robot's location using visual odometry. A Digital Single Lens Reflex (DSLR) camera was utilised in [8] to capture photos for image mosaicking. A similar system was proposed in [9] however the image capturing process is manual. A mirror-less camera installed on a monorail system to capture images along the tunnel and then perform change detection on these images was used in [10].

B. 3D Reconstruction

The availability of a 3D model for surface documentation of a tunnel wall provides comprehensive visual and geometric images of its environments, aiding inspectors to contextualise better the location of damages found during observations. In addition, 3D information enables visual validation of defects with respect to the areas around them. Laser scanning is predominantly used for such reconstructions and a review on its use can be consulted in [11]. Such laser-based 3D models are devoid of image data that can be more useful in inspection. The combination of passive and active imaging sensors was proposed in [12]. A 3D surface model was generated through a stereo-based photogrammetry workflow in [13]. Structure from motion (SfM) techniques were used in [9] and [14] to recover the 3D geometry of a tunnel by fitting local quadric surfaces to the generated point cloud. In [15], 3D reconstruction of the crack areas on a tunnel wall is used to further analyse previously detected cracks.

C. Virtual Reality

VR is a blend of digital image processing, computer graphics and multimedia technology used to create an interactive computer simulation, which senses the user's state and movements and subsequently replaces sensory feedback information in such a way that the user experiences a sense of being immersed in the simulation (virtual environment). Work on VR has been recorded for a long time as listed in [16] however, it became more popular in the current decade. The various applications of VR include health-care, entertainment, design, education and engineering. VR technology has also been used in the civil engineering field [17] such as for planning and design, construction progress demonstration as well as for monitoring and inspection. Two prototype solutions based on VR technology for use in maintenance planning of buildings were proposed in [18]. These support the execution of periodic inspections and the monitoring of interior and exterior wall maintenance. Furthermore, an inspection and reporting system that uses VR, multimedia and 3D modeling techniques was developed for the Troll Gas landfall tunnels [19].

III. System Overview

The proposed system consists of three components. First, it captures wall photos automatically from camera/s on a moving platform. Then, the second module uses this image data in order to make the 3D reconstruction of the tunnel wall. This model is fed to the final module that uses VR technology to render the tunnel wall structures. The end user visualizes this on a VR headgear for the purpose of monitoring and inspection.

IV. Image Data Acquisition

In order to keep up with space and time constraints, inspection systems must be simple to set up and small in dimensions. In this work, we use the LHC tunnel environment in which its restricted access areas, low lighting and dust impose various other limitations on the choice of image acquisition set-up. The proposed system uses a mobile platform to move a camera around the LHC tunnel and captures images of the walls.

A. Mobile Platforms

There are currently two mobile platforms that can be used for the proposed system: Train Inspection Monorail (TIM) [20] and CERNbot [21]. TIM is a modular inspection train remotely operated to move on an overhead track installed on the LHC tunnel ceiling. For image capturing, a camera is fixed on a robotic arm extending downwards from one of the TIM wagons as shown in Fig. 1. CERNbot is an in-house developed remotely operated vehicle (ROV), on which different devices such as sensors and robotic arms can be placed to conduct different interventions. For image capturing, it is equipped with a metal structure that houses multiple cameras as shown in Fig. 2. The CERNbot is remotely operated via a graphical user interface [22] and for our solution it is driven roughly in a straight line in parallel to the tunnel wall.

B. Camera Setup

Either a DSLR or a mirror-less camera can be used with both mobile platforms however the latter type is preferred due to its compactness and light weight. In the case of the CERNbot, multiple cameras can be used, such that a number



Fig. 1. Camera on the arm extending from one of the wagons of the TIM

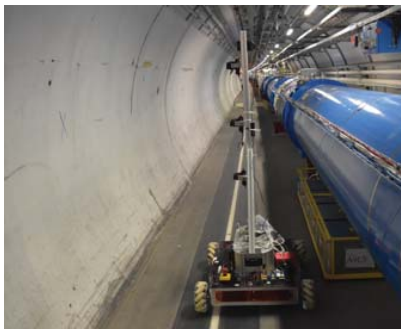


Fig. 2. Camera on the CERNbot robotic platform

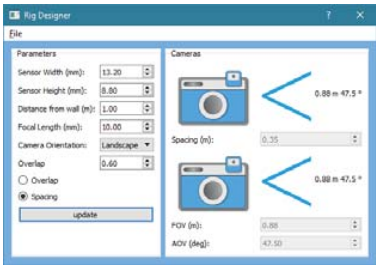


Fig. 3. Screenshot of the application used to find the camera set-up parameters

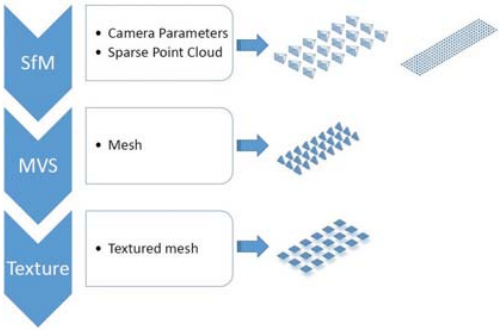


Fig. 4. Zephyr 3D reconstruction process

of overlapping images can be captured. This method allows a larger surface area of the wall to be reconstructed from a single run along the tunnel. A simple application was developed to calculate the camera set-up configuration parameters. For the given camera's sensor dimensions, lens focal length and distance from the wall, the application calculates the image overlap when setting the spacing between the cameras or vice versa. A screenshot of the application is displayed in Fig. 3.

C. Automatic image capturing

Images are captured by the cameras automatically while the robotic platform is moving. This is possible through the interface developed using the camera software development kit (SDK) [23]. The camera interface can capture both images and videos and saves them to the SD card and/or the host computer according to the previously defined configuration parameters.

V. 3D Image Reconstruction

In general, 3D reconstruction algorithms use salient points extracted from images such as corners, blobs, etc. In our scenario images from the tunnel walls, mainly contain a white surface with some dirt, cracks and some equipment, which may lie on the wall, implying lack of features in the image. While there has been a surge of interest in generating 3D models of objects and scenes from photos, reconstruction from images lacking texture and consequently reliable features is still very limited. After evaluating various existing 3D reconstruction software applications, 3DFlow Zephyr Aerial [24] commercial software was chosen due to its high quality output even when the image content lacks features. This software can operate either on a set of single images or on a frame sequence from a video file. It uses SfM techniques to calculate the camera parameters and generates a sparse point cloud. A multi-view stereo algorithm is then used to create a dense point cloud followed by a mesh generation. The mesh is then textured using the content in the original images. The following subsections give further details on each part of the 3D reconstruction process displayed in Fig. 4 as applied to the captured images and video frames of the LHC tunnel wall. Such reconstruction is done offline once the image dataset is captured.

A. Structure from Motion

SfM deals with the recovery of the 3D geometry of a scene, i.e. structure, when observed through a moving camera, i.e. motion. Zephyr first extracts keypoints from all n input images. It uses a scale-space feature extractor based on Difference of Gaussian, with a radial and symmetric descriptor. A spanning tree defining the overlap relationship between the images is then built to establish the order in which the images must be processed. Feature matching then follows a nearest neighbour approach [25]. M-estimator Sample Consensus (MSAC) [26], is subsequently used to compute homographies and fundamental matrices between pairs of matching images. MSAC gives outliers a fixed penalty but scores inliers on how well they fit the data. Next, the images are organized into

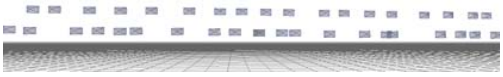


Fig. 5. Camera positions as calculated by SfM



Fig. 6. Sparse reconstruction corresponding to the scene in Fig. 5

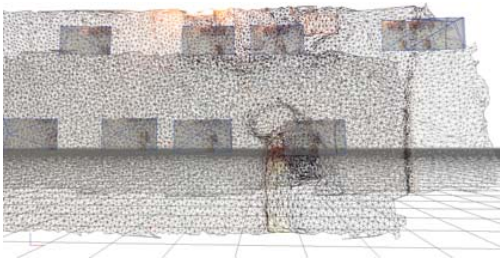


Fig. 7. Zoomed in wire-frame mesh corresponding to the scene in Fig. 5



Fig. 8. Final 3D model corresponding to the tunnel section in Fig. 5

a tree with agglomerative clustering in a bottom-up manner, using the overlap measure as the distance. The dendrogram generated by the clustering stage imposes a hierarchical organization of the views. At each node of the dendrogram, three operations are possible: two views reconstruction, one view addition or fusion. This stage establishes the camera location for each frame as shown in Fig. 5 and the corresponding sparse reconstruction of the scene as in Fig. 6. For details on the SfM stage of the reconstruction the reader is referred to [27].

B. Multi-View Stereo

In order to reconstruct a dense model using several images captured from multiple known camera viewpoints, which in our case were found using SfM, MVS is used. For each pixel m, candidate depths are extracted by considering the reference image I_i and $N(I_i)$ neighbouring views. The latter are chosen using an overlap measure based on the Jaccard index. Candidate depths for each m are searched along the epipolar line of each neighboring image using block matching and Normalised Cross Correlation (NCC). The final depth map is built from the depth hypothesis using a discrete Markov random field (MRF) optimization method over the (regular) image grid. The MRF assigns a label $l \in \{l_1 ... l_k, l_{k+1}\}$ to each m, where the candidate depths are represented by the first k labels and l_{k+1} is the undetermined state. A sequential tree re-weighted message passing optimization [28] was used to solve the MRF. The Poisson algorithm [29] is then used to create a surface. For every 3D point, a normal is computed by fitting a plane to its nearest neighbours. At this stage, a mesh of the wall surface is available as shown in Fig. 7.

C. Texture mapping

In the final stage of the reconstruction, the mesh from the previous stage is textured using the content of the original images fed into the software such that the 3D model has close resemblance to the real tunnel wall surface. Zephyr's texturing stage is based on the color balance method and although the approach is quite different it is based on a modified multiband algorithm [30]. Once the mesh is textured using the image content, the complete 3D model of the tunnel wall is available as displayed in Fig. 8.

D. Experiments

The reconstruction module was tested on different image datasets captured using both the TIM and CERNBot. With the current robotic arm on the TIM only a single camera can be used while multiple ones can be placed on the CERNbot. Using multiple cameras, overlapping images in both directions (horizontal and vertical) can be captured and stitched, allowing a larger area of the tunnel wall to be observed. Consequently, the CERNbot was selected as the mobile platform for the proposed system.

In order to have an overlap of approximately 60% between each images considering that the cameras are at around $1m$ away from the wall with $35cm$ between their centres, the focal length (F) was set to 10mm. For a wider field of view, a shorter focal length can be chosen, however this may then introduce lens distortion. To avoid blurring, ideally, the shutter speed should be set at $\frac{1}{F}s$ or faster so, in this case, the speed should be at least $\frac{1}{10}s$. When the camera was set at a speed of $\frac{1}{15}s$, with an aperture of $\frac{f}{5.6}$ and $ISO400$, the exposure was relatively dark. Improving the brightness by increasing the exposure time would introduce blurring. Furthermore, the camera is on a moving platform, thus there is already the possibility of blurring from this egomotion. To improve the brightness, the exposure triangle principle was used. This states that if the exposure is to be kept the same, if one element of aperture, ISO or shutter speed is changed with x stops then the combination of the other two should change by x stops too. Consequently, as the speed was increased by four stops to $1/60s$, the ISO was pushed up to 1600 as shown in Fig. 9. The CERNBot was driven at different speeds, according to the overlap required. For an overlap of 60%, taking a photo every second, the spacing in between each capture should be around $0.33m$, thus the optimal speed was found to be $0.3m/s$.

Two different dataset types were used to reconstruct the tunnel wall and a summary of the properties for each is given in table I. The use of video enabled the dataset gathering to be done faster as it does not require the robotic platform to be moved slowly to avoid photo blurring. Furthermore, the reconstruction was more complete due to a large overlap between the frames. While both single images and video frames achieved high quality reconstruction, some blurring

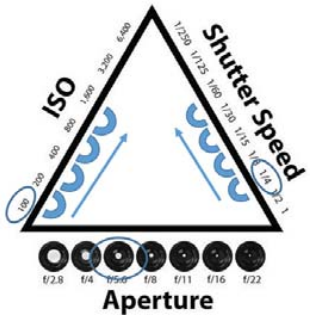


Fig. 9. Camera settings involving the ISO, Aperture and Shutter speed using the Exposure Triangle Principle

TABLE I
DATASET SUMMARY, INCLUDING DATA TYPE, CAMERA AND RESOLUTION

Dataset type	Camera used	Resolution	FPS
Images	Nikon 1 V3	2607 x 1744	N/A
Video	Nikon 1 V3	1920 x 1080	59

was observed when using videos. In some cases, holes appeared in the mesh, however these were filled using the water-tightness feature of the software. Although mesh-filling produced good results in general, some abnormalities occurred in places where the images had nearly no features.

For each of the models created, a processing report was exported from Zephyr and analyzed manually to check the performance. Considering the sample model used in the presented figures and its corresponding report, we now discuss the quantitative values. Although this is a sample model it is representative of the other generated models. As observed in Fig. 5, there are two sets of cameras, one on top of each other, as datasets from two cameras set as shown in Fig. 2 were used. When using video as a data source, video frames were extracted at an empirically set frame rate of 1 FPS. When extracting frames, a similarity score between the current frame and the previous one is calculated; if the deviation is lower than a defined threshold, then the frames are considered too similar and the current frame is discarded. Once the images/video frames are available, using SfM, the 3D reconstruction module finds the camera parameters as shown in table II. To decrease the computation time, the images/frames are resized by a factor of 0.5 for subsequent steps of the reconstruction. The quality of the 3D reconstruction of the tunnel wall depends on the overlap of the neighboring images which is directly related to the speed of the moving platform and the texture on the wall. The slower the speed, the larger the overlap and thus the better the results. As for the lack of features in the images, on-going work is focusing on improving the feature extraction capabilities of the system. On average, the images/frames had an overlap between 60-70% for a successful reconstruction.

TABLE II
INTERNAL CAMERA PARAMETERS OBTAINED BY THE 3D RECONSTRUCTION MODULE FOR THE SCENE IN FIG. 5

Skew	Focals	Optical Center	Radial Distortion	Tangential Distortion
0	X: 1035.1 Y: 1035.1	X: 982.38 Y: 538.87	K1: 0.0080082 K2: -0.062066 K3: 0.022664	P1: 0 P2: 0
0	X: 1086.6 Y: 1086.6	X: 965.05 Y: 534.76	K1: 0.011798 K2: -0.078762 K3: 0.032365	P1: 0 P2: 0



Fig. 10. Virtual model of the tunnel wall being refined in Unity

VI. TUNNEL WALL MODEL VISUALISATION

Building and maintaining infrastructures typically involves four phases: construction, monitoring, preventive maintenance and repair; throughout which engineers increasingly demand a visual and geometrical digital representation of the structure surfaces. Using the image data capture on-site and generating 3D models of the environment helps with better documentation and also offers a means of remote inspection. Moreover, viewing such 3D models using VR technology offers further benefits, thus the final module of our proposed system uses the generated 3D models and renders them such that they can be viewed via VR.

VR is a computer-generated scenario that simulates experience. Through such a model, tele-presence comes into play, where a user is able to view the walls as if s/he is in the tunnel itself and thus can perform inspection remotely in a better contextualised scenario than through 2D or 3D models. In addition, such a model can also be used by personnel to familiarise themselves to the environment before going on a mission. Our system uses Unity3D, a cross-platform game engine [31] to generate the virtual model and refine it by changing the scale, adding lights and other modifications through a user interface as demonstrated in Fig. 10. In turn, an HTC VIVE headset together with an HTC controller are then used to view the VR model and navigate through the scene as illustrated in Fig. 11.

VII. CONCLUSION AND FUTURE WORK

Infrastructures such as tunnels and bridges may develop some defects due to ageing and stresses, thus they need to be regularly monitored. Traditionally this process is done by visually observing the structure itself, however this is not very efficient. Some tunnels cannot be closed for traffic and others may not be accessible to humans due to hazardous conditions



Fig. 11. Virtual model of the tunnel wall viewed using an HTC headset

that may be present. Furthermore, manual inspection is highly dependant on human subjectivity. Taking this into consideration we developed a comprehensive system to automatically capture images using one or more cameras placed on a moving platform. The images are then reconstructed into 3D models and viewed in VR. Such a system is beneficial for wall surface documentation, remote inspection and analysis. Further improvements to the system can be the use of more sensors to provide a more accurate 3D model as well as the automation of 3D reconstruction via an SDK rather than the user interface. In addition, some information on the environment can be augmented to the VR model rendering mixed reality.

ACKNOWLEDGMENT

We thank our colleague Mengping Zheng from the EN Department at CERN for her valuable contribution in the VR aspect. She provided help with the set up and the use of Unity3D to create the VR models.

REFERENCES

[1] CERN, "Cern LHC design report," online; accessed May 2018. [Online]. Available: https://edms.cern.ch/ui/file/445918/5/Vol_2_Chapter_3.pdf

[2] R. Montero, J. Victores, S. Martnez, A. Jardn, and C. Balaguer, "Past, present and future of robotic tunnel inspection," *Automation in Construction*, vol. 59, pp. 99–112, 2015.

[3] L. Attard, C. J. Debono, G. Valentino, and M. D. Castro, "Tunnel inspection using photogrammetric techniques and image processing: A review," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 144, pp. 180 – 188, 2018.

[4] S. N. Yu, J. H. Jang, and C. S. Han, "Auto inspection system using a mobile robot for detecting concrete cracks in a tunnel," *Automation in Construction*, vol. 16, pp. 255–261, 2007.

[5] M. Ukai, "Advanced inspection system of tunnel wall deformation using image processing," *Quarterly Report of RTRI*, vol. 48, no. 2, pp. 94–98, 2007.

[6] S. Y. Lee, S. H. Lee, D. I. Shin, Y. K. Son, and S. Han, C, "Development of an inspection system for cracks in a concrete tunnel lining," *Canadian Journal of Civil Engineering*, vol. 34, pp. 966–975, 2007.

[7] P. Hansen, H. Alismail, P. Rander, and B. Browning, "Visual mapping for natural gas pipe inspection," *International Journal of Robotics Research*, vol. 34, pp. 532–558, 2015.

[8] C. H. Lee, Y. C. Chiu, T. T. Wang, and T. H. Huang, "Application and validation of simple image-mosaic technology for interpreting cracks on tunnel lining," *Tunnelling and Underground Space Technology* , vol. 34, pp. 61–72, 2013.

[9] S. Stent, R. Gherardi, B. Stenger, K. Soga, and R. Cipolla, "Visual change detection on tunnel linings," *Machine Vision and Applications Journal*, vol. 27, no. 3, pp. 319–330, Apr. 2016.

[10] L. Attard, C. J. Debono, G. Valentino, and M. Di Castro, "Vision-based change detection for inspection of tunnel liners," *Automation in Construction*, vol. 91, pp. 142 – 154, 2018.

[11] C. Frohlich and M. Mettenleiter, "Terrestrial laser scanning-new perspectives in 3d surveying," vol. 36, 01 2004.

[12] G. Paar, A. Bauer, and H. Kontrus, "Texture-based fusion between laser scanner and camera for tunnel surface documentation," in *Proceedings of the 7th International Conference of Optical 3-D Measurement Techniques*.

[13] A. Bauer, K. Gutjahr, G. Paar, H. Kontrus, and R. Glatzl, "Tunnel surface 3d reconstruction from unoriented image sequences," in *Proceedings of the 39th Annual Workshop of the Austrian Association for Pattern Recognition (OAGM)*, 2015.

[14] Z. H. Zhu, J. Y. Fu, J. S. Yang, and X. M. Zhang, "Panoramic image stitching for arbitrarily shaped tunnel lining inspection," *Computer-Aided Civil and Infrastructure Engineering*, vol. 31, no. 12, pp. 936–953, 2016.

[15] E. Protopapadakis, C. Stentoumis, N. Doulamis, A. Doulamis, K. Loupos, K. Makantasis, G. Kopsiaftis, and A. Amditis, "Autonomous robotic inspection in tunnels," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. III-5, pp. 167–174, 2016.

[16] N. Singh and S. Singh, "Virtual reality: A brief survey," in *Proceedings of the 2017 International Conference on Information Communication and Embedded Systems (ICICES)*, Feb 2017, pp. 1–6.

[17] C. Fu, "A new research approach on the application of virtual reality technology in civil engineering," in *Proceedings of the International Conference on Materials Engineering and Information Technology Applications (MEITA 2015)*, Feb 2015, pp. 1014–1017.

[18] A. Z. Sampaio, D. P. Rosrio, and A. R. Gomes, "Monitoring interior and exterior wall inspections within a virtual environment," *Advances in Civil Engineering*, vol. 2012, no. 780379, pp. 1568–1583, Oct 2012.

[19] "Troll tunnel inspection ROV piloted in virtual reality mode," *Offshore Magazine*, vol. 56, no. 8, pp. 141–142, 1996.

[20] M. Di Castro, M. L. Baiguera Tambutti, S. Gilardoni, R. Losito, G. Lunghi, and A. Masi, "LHC train control system for autonomous inspections and measurements," in *Proceedings, 16th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS 2017)*, 2017.

[21] M. Di Castro, L. R. Buonocore, M. Ferre, S. Gilardoni, R. Losito, G. Lunghi, and A. Masi, "A dual arms robotic platform control for navigation, inspection and telemanipulation," in *Proceedings, 16th International Conference on Accelerator and Large Experimental Physics Control Systems(ICALEPCS 2017)*, Oct 2017.

[22] G. Lunghi, R. M. Prades, and M. D. Castro, "An advanced, adaptive and multimodal graphical user interface for human-robot teleoperation in radioactive scenarios," in *Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2016)*, 2016, pp. 224–231.

[23] Nikon, "Nikon software development kit," online; accessed May 2018. [Online]. Available: https://sdk.nikonimaging.com/apply/

[24] 3Dflow, online; accessed May 2018. [Online]. Available: https://www.3dflow.net/3df-zephyr-pro-3d-models-from-photos/

[25] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov 2004.

[26] P. Torr and A. Zisserman, "MLESAC: A new robust estimator with application to estimating image geometry," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138 – 156, 2000.

[27] R. Toldo, R. Gherardi, M. Farenzena, and A. Fusiello, "Hierarchical structure-and-motion recovery from uncalibrated images," *Computer Vision and Image Understanding*, vol. 140, pp. 127 – 143, 2015.

[28] V. Kolmogorov, "Convergent tree-reweighted message passing for energy minimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1568–1583, Oct 2006.

[29] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, ser. (SGP'06), 2006.

[30] C. Allene, J. P. Pons, and R. Keriven, "Seamless image-based texture atlases using multi-band blending," in *Proceedings of the 19th International Conference on Pattern Recognition*, Dec 2008, pp. 1–4.

[31] Unity3d, online; accessed May 2018. [Online]. Available: https://unity3d.com/

Automatic Crack Detection using Mask R-CNN

Leanne Attard *, Carl James Debono †, Gianluca Valentino ‡
Department of Communications and Computer Engineering
University of Malta
Msida, Malta
*leanne.attard@um.edu.mt † c.debono@ieee.org ‡ gianluca.valentino@um.edu.mt

Mario di Castro §, Alessandro Masi¶, Luigi Scibile ‖
§¶*Engineering Department, Survey, Mechatronics and Measurements group* ‖*Site Management and Buildings Department*
CERN
Meyrin, Switzerland
§mario.di.castro@cern.ch ¶alessandro.masi@cern.ch ‖ luigi.scibile@cern.ch

***Abstract*—In order to avoid possible failures and prevent damage in civil infrastructures, such as tunnels and bridges, inspection should be done on a regular basis. Cracks are one of the earliest indications of degradation, hence, their detection allows preventive measures to be taken to avoid further damage. In this paper, we demonstrate that Mask R-CNN can be used to localize cracks on concrete surfaces and obtain their corresponding masks to aid extract other properties that are useful for inspection. Such a tool can help mitigate the drawbacks of manual inspection by automating crack detection, lowering time consumption in executing this task, reducing costs and increasing the safety of the personnel. To train Mask R-CNN for crack detection we built a groundtruth database of masks on images from a subset of a standard crack dataset. Tests on the trained model achieved a precision value of 93.94% and a recall of 77.5%.**

***Index Terms*—object detection, crack detection, mask r-cnn, vision-based inspection**

I. Introduction

As civil infrastructure (e.g. bridges, tunnels and dams) ages due to weathering, corrosion, carbonation and thermal cycles, it becomes susceptible to structural deterioration which may lead to deviations from their original design functions. It is therefore of utmost importance that such structures are inspected on a regular basis to proactively respond to prevent damage and possible failures which may also lead to fatal accidents. Cracks on concrete surfaces are one of the earliest indication of degradation of a structure. The number of cracks together with their type, width and length show the degradation level and carrying capacity of the concrete structure of a surface. Their early detection allows preventive measures to be taken in order to avoid further damage.

The acclaimed traditional method used to inspect cracks is through manual, visual surveys. Inspectors conduct site visits, either in person or through the use of drones or other robotic or remotely operated equipment and traverse the structure looking at surfaces and noting conditions of the irregularities through manual sketches of cracks. Such on-site inspections require closing bridge and tunnel systems, disrupting traffic flow, building structures around high buildings as well as shutting down facilities within the structures being surveyed. All this, leads to high expenses, time consumption and inefficiency. Furthermore, this manual approach depends on the surveyor's knowledge and experience, thus it lacks objectivity in the quantitative analysis.

To mitigate the above, various research groups have proposed automatic crack detection methods as a partial replacement of manual inspections. Over the last few decades, numerous works on automatic crack detection on different structural surfaces such as roads, bridge decks, pavements and tunnel walls were published. Surveys reviewing such works can be found in [1], [2] and [3]. A number of image processing techniques were implemented. Early works relied on a combination of techniques such as thresholding, mathematical morphology and edge detection. More recent approaches study crack detection under more challenging conditions using other methods including saliency detection, texture analysis, wavelet transform, minimal path finding and machine learning. Whilst being reliable in some applications, these methods use shallow abstractions and use rule-based approaches which cannot overcome inherent challenges associated with crack images. The latter include inhomogeneity of cracks, diversity of surface texture, background complexity, inference of noises with similar texture to cracks such as joints and difficult topology of cracks. Such challenges make it impossible to use a rule-based method which is capable to extract generalized features effectively under varying conditions. To overcome these challenges, deep learning using convolutional neural networks (CNNs) has been recently proposed, featuring high level of abstractions and generalization without any need of extracting hand-crafted features.

In this study, we use Mask R-CNN [4], a region-based CNN classifier that not only detects targets in the image but also gives the predicted mask for each target which is useful for further processing. This model has been used and proved to perform very well on natural images. Here we use it to detect cracks and other artifacts on concrete surfaces, and obtain their corresponding masks to aid with extracting further properties of the crack such as length and width.

978-1-7281-3140-5/19/$31.00 ©2019 IEEE

II. Background Information and Related Works

A. Crack Detection

Generally, crack areas are darker than those of their surroundings, resulting in lower intensity values compared to the background. Such a property has been used to fix one or more thresholds for segmentation, creating binary images that distinguish crack and non-crack pixels. The classical intensity thresholding technique is relatively simple and computationally inexpensive however, its accuracy depends merely on the predefined threshold value, implying difficulty in scenarios where crack sizes, backgrounds and lighting conditions vary considerably.

When surfaces are highly textured, the patterns in the texture along the surface can be used to identify defects in it. An algorithm that uses a Wigner model to identify cracks in complex textural backgrounds was proposed in [5]. Texture-analysis based methods using a rotation invariant Gabor Filter were suggested to detect cracks in concrete tunnels and pavements in [6] and [7] respectively.

Salient regions are visually more conspicuous due to their contrast with the surroundings. Although existing methods demonstrate their effectiveness in detecting salient regions in natural content images, they perform poorly on the completeness and continuity of the detected crack. Works using saliency for crack detection such as [8] are very limited in number.

A 2D continuous wavelet transform is used to build complex coefficient maps, where wavelet coefficients maximal values are obtained for crack detection in [9]. Due to the anisotropic characteristic of wavelets, these approaches cannot handle scenarios with cracks of high curvature or low continuity.

The above image processing based techniques have limited learning capabilities and sometimes rely on parameters fine-tuned manually as they do not encompass the complexity of conditions that a concrete surface might exhibit. A better solution that has more real-world situation adaptability is to use machine learning algorithms. An integrated system, CrackIT, for automatic detection and characterization of cracks in flexible pavement surfaces using a combination of unsupervised learning (clustering) followed by supervised learning (classification) was proposed in [10]. A pavement crack detection algorithm based on fuzzy logic was introduced in [11]. In [12], AdaBoost was used to select textural descriptors that can describe crack images. In CrackForest [13], a descriptor based on random structured forests to characterize cracks is suggested. A comprehensive review of the computer vision based defect detection on pavements presented in [14] identified Support Vector Machine (SVM) as the most popular machine learning technique for image-based road crack detection.

1) Crack Detection using Deep Learning: The performance of these machine learning methods is high but very dependent on the extracted features. However, due to complicated surface conditions, it is hard to find features effective for all structural scenarios. Considering this, deep learning algorithms have been recently applied to overcome such adaptability limitations. In [15] and [16], a vision-based method using a deep architecture of CNNs for detecting concrete cracks without calculating the defect features was proposed. However, both works can only find patch level cracks without considering the pixel level. In [17], a CNN is used to predict whether an individual pixel belongs to a crack based on the local patch information, however this method still ignores the spatial relations between pixels and overestimates crack width. In [18], a CNN is used to predict class for each pixel of the image. However, it still needs manually designed feature extractors at a pre-processing stage, such that the CNN is only used as a classifier.

B. Mask R-CNN

The last years were characterized by dramatic advances in the state of the art solutions for fundamental tasks in computer vision. This was mainly based on the use of CNN for object detection, semantic segmentation and object localization.

The Region-based CNN (R-CNN) approach [19], is noted to be the pioneering work of using deep learning for object detection. A manageable number of candidate object regions are generated at the first stage. Then, for each candidate region, features are extracted. R-CNN was later extended to Fast R-CNN [20] attending to regions of interest (RoIs) on feature maps using RoIPool, leading to higher speed and better accuracy. Following that, Faster R-CNN [21] was introduced, replacing the slow selective search algorithm with a fully convolutional neural network on top of the already generated features, specifically using a Region Proposal Network (RPN). The latter works by sliding a window over the CNN feature map and at each window, outputting k potential bounding boxes and scores.

Mask R-CNN [4] was proposed to extend Faster R-CNN for pixel level segmentation. It adds a branch for predicting an object mask in parallel with the existing branch for bounding box recognition while replacing the ROI-Pooling operation with ROI-Align that allows very accurate instance segmentation masks to be constructed. Mask R-CNN is simple to train and adds only a slight overhead to Faster R-CNN. Recently, Mask R-CNN has been used for object detection of various classes including pedestrians, cars and traffic signs for surveillance and self-driving cars, building extraction using aerial imaging as well as nucleus segmentation in medical imaging. In this paper we demonstrate that Mask R-CNN can also be used to detect cracks in concrete surfaces to aid the automation of infrastructure inspection when monitoring structural health.

III. Methodology

Our framework is based on Mask R-CNN [4], with the pipeline shown in Fig. 1. First, the Region Proposal Network (RPN) outputs a set of bounding boxes (ROIs) with scores indicative of how probable they contain an object within them. Then, a combination of a Faster R-CNN classifier and a binary mask prediction branch are used to find the class of the object within the ROIs and the corresponding mask respectively. Our detection framework is based on the implementation released

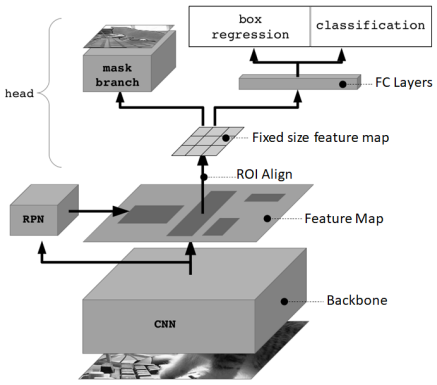
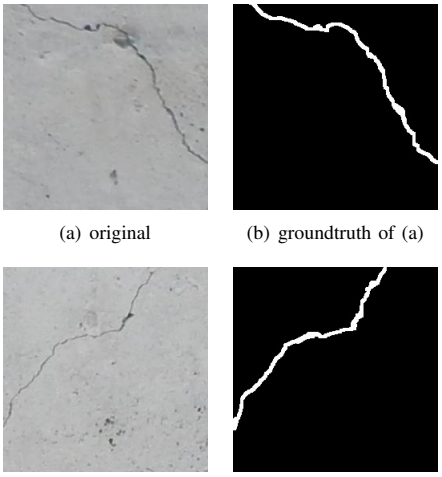


Fig. 1. Mask R-CNN pipeline diagram adapted from [22]



(a) original (b) groundtruth of (a)

(c) original (d) groundtruth of (c)

Fig. 2. Sample images from the annotated crack dataset we built

by Matterport under the MIT license [23]. This is itself based on the open-source libraries of Keras and Tensorflow.

A. Architecture

a) Backbone: The backbone architecture of Mask R-CNN consists of a standard neural network that serves as a feature extractor. The early layers detect low level features and the later layers successively detect higher level features. While this is a good backbone, Mask R-CNN improves upon it using a feature pyramid network (FPN). This adds a second pyramid that takes the high level features from the first pyramid and passes them down to lower layers allowing features at every level to have access to both lower and higher level features. This implementation of Mask R-CNN uses a ResNet [24] architecture with a FPN backbone.

b) RPN: The RPN is a lightweight neural network that finds areas that contain images using a sliding window fashion. The regions that the RPN scans over are boxes distributed over the image area and are referred to as $anchors$. The RPN anchor scales, ratios, strid and non-maximum suppression (NMS) threshold are related hyperparameters that were heuristically modified during training until satisfactory results were obtained.

B. Transfer Learning

As only a relatively small dataset could be built, to enable a robust training of a complete deep learning model, a transfer learning methodology is recommended. Hence, rather than training the network end-to-end from scratch, we initialize the model with pre-trained weights from training on the COCO [25] and Imagenet [26] datasets. By tweaking several hyperparameters we could fine-tune the network to adapt it to our own data. The Matterport implementation [23] provides the possibility to change various parameters, for instance; learning rate, learning momentum and train ROIs per image.

C. Data Augmentation

Since the data available for training is not very large, we introduced an augmentation pipeline to provide different variations of the available images. Augmentation mimics a larger dataset which improves the training performance of the network as will be discussed later. We experimented with several transformations for augmentation, such as vertical and horizontal flips, different rotations, changes in the brightness and addition of blurring using a Gaussian kernel. To investigate the benefits of using data augmentation, we built and tested different pipelines using the various functions from the imgaug library [27]. A brief description of each pipeline is given in Table III.

IV. Data

To demonstrate the effectiveness of using Mask R-CNN for crack detection, we train the network using images of cracks and other defects to counteract for any possible false detections. To train the model to detect cracks, we built a dataset based on the SDNET dataset [28], an annotated image set for training, validation, and benchmarking of artificial intelligence based crack detection algorithms for concrete. However this dataset provides the groundtruth for images, classified as crack vs non crack only rather than groundtruth masks for instance object segmentation as required by Mask R-CNN. Consequently we built a mask dataset from a subset of 200 images of the complete SDNET set. Once the subset was chosen, mask annotation was then conducted. We used the PixelAnnotationTool [29] which enabled us to use a brush with a small radius to mark the 'crack objects' which are quite fine, very narrow and long in nature. Two samples from the developed mask dataset are shown in Fig. 2. The RGB color images have a resolution of 256×256. We used the 80/20 rule to divide the data in 128 images for training and 32 for validation. The remaining 40 images were used for testing.

V. Experiments & Results

In order to evaluate the varying models trained using different hyperparameters configurations we used the Precision and Recall metrics. Precision shows how much of the detected

cracks were actually cracks. It was calculated using the ratio of the 'true cracks' detected to the total number of cracks identified. On the other hand, Recall is a measure of how much of the actual cracks were detected. It was calculated using the ratio of the 'true cracks' detected to the total number of actual cracks. When a groundtruth crack mask overlapped a detected crack mask by 30% or more, a match is defined and taken as a true positive. Otherwise, if the overlap is greater than 0% but less than 30%, the detected crack mask is labelled as a false negative. If a detected crack mask did not match with any of the groundtruth masks, this is taken as a false positive.

a) Backbone Architecture: In order to train our network, we started from weights pre-trained on the Imagenet and COCO datasets for the ResNet-50 and RestNet-101 backbones respectively. As noted in Table I the ResNet-101 pre-trained on the COCO dataset performed slightly better in general.

b) Training schedule: Since we used a small dataset and started from pre-trained weights, we did not need to train for a long time. We experimented with different training schedules as shown in Table I, where H and AL define the numbers of epochs used to train the Heads of the network and all the layers respectively. When fine-tuning our network by re-training only the heads of the network as in Tests 1 and 2, a high precision was obtained however the recall value was low, this implies that while the detections were correctly made, a lot of misses also occurred. When the first few epochs re-trained the heads for a fraction of these epochs and then all the layers for the remaining epochs (Tests 3 and 4), the precision value was still high however less misses were incurred when compared to Tests 1 and 2. When the number of training epochs was increased, comparing Tests 3 and 4, the validation loss did not improve much further as shown in Fig. 3 and the accuracy of the detections remained very similar, implying that training for a longer time will not do any mayor improvements to the network. In another training schedule, instead of increasing the number of epochs, we increased the number of steps per epoch however and, whilst the precision value remained similar, the recall value decreased considerably for Test 5. Hence, the best training schedule was that of training the heads of the network for 50 epochs and all the layers for the next 150 epochs as recorded in Test 3.

The learning rate hyperparameter controls how much the weights of a network are adjusted with respect to the loss gradient. While using a low learning rate might be a good idea not to miss any local minima, it could also cause the network to take a long time to converge, especially if it gets stuck in a plateau region for some time. Consequently, we experimented with setting both a fixed learning as well as changing it along the epochs as shown in Table II, where H and AL define the numbers of epochs used to train the Heads of the network and all the layers respectively. The original paper reporting Mask R-CNN [4] used a learning rate of 0.02 however on the Tensorflow implementation [23] weights increased too much thus, a lower value of 0.001 was used. Taking into consideration a training schedule of 50 epochs on heads only, keeping the learning rate fixed resulted in a high precision value however the recall achieved was very low. Reducing the learning rate by half at 25 epochs did improve the recall value however, the optimum values were achieved when the learning rate was reduced as the validation loss came to a plateau as shown for Test 9. When training for the heads and all the layers, changing learning rate throughout the epochs did not achieve better results.

Considering the results from the experiments above, it can be concluded that the best achieved results were obtained with the trained model in Test 3. This consists of a ResNet-101 backbone, trained starting from pre-trained weights from the COCO dataset. The learning rate was set at 0.001 and kept fixed throughout the 50 epochs fine tuning the heads of the network while training all the layers for the remaining 150 epochs.

c) Augmentation: First, we trained our Mask R-CNN model without any augmentation and then proceeded with training using the three augmentation pipelines described in Section III-C. As observed in Table III, the lowest Precision and Recall values occurred when no augmentation was involved. When flipping was introduced the values were slightly higher and then improved even further when rotation, brightness variation and blurring were introduced. When the augmentation pipeline included also contrast normalization as well as cropping, the performance deteriorated only slightly. Such results imply that using data augmentation when training our Mask R-CNN, in general improved the model performance.

Further to the tests conducted on the modified SDNET annotated subset [28], to test the generalization of the model, we used the model with the configuration applied in Test 3 to detect cracks on images randomly found on the Internet as well as on images we captured in a tunnel. As seen in the images in Fig. 5, varying cracks were successfully detected under different exposure settings and wall surfaces.

VI. Conclusion and Future Work

During the last few decades automatic vision-based detection has been proposed as a replacement to mitigate the drawbacks of manual inspection. Various approaches were recorded in literature including; saliency detection, texture analysis, wavelet transform, minimal path finding and machine learning. In this paper we adapt the state of the art detection model Mask R-CNN to automate crack detection on concrete surfaces. We train this model using our own groundtruth dataset which was built on a subset of images from a recent benchmark annotated image set aimed for training and validation of artificial intelligence based crack detection algorithms for concrete. In order to adapt our framework to scenarios with more varying surfaces and lighting conditions, in the future we aim to retrain the network on a larger and more varied dataset. Furthermore, multi-class detection for other infrastructure components and defects can help to reduce the false detections as well as provide a better means for civil structures inspection.

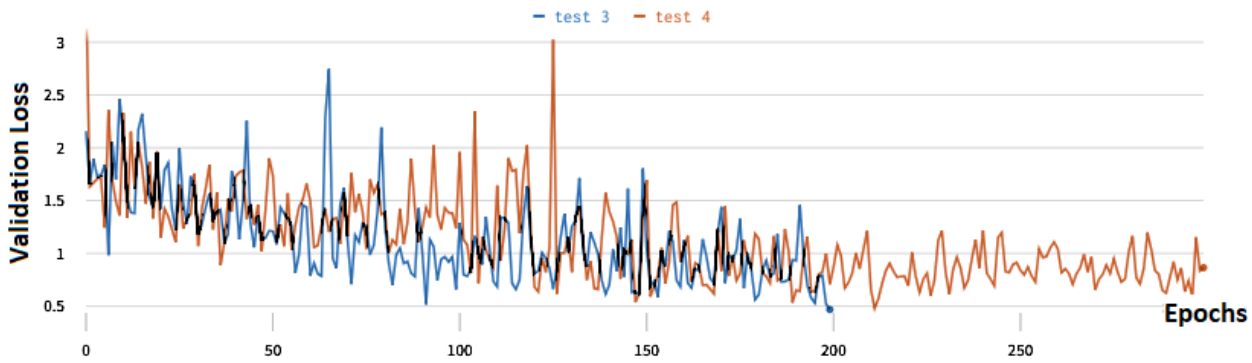
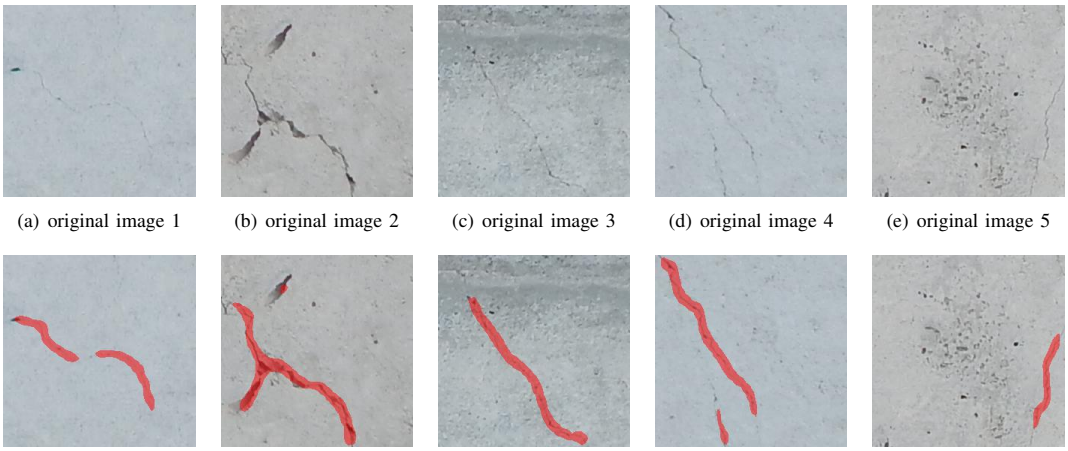


Fig. 3. Plot of the validation loss against the number of epochs for Test 3 and Test 4

TABLE I
PRECISION AND RECALL VALUES WHEN VARYING THE PRE-TRAINED WEIGHTS AND THE TRAINING SCHEDULE.

Test No.	Pre-trained Weights	Backbone Architecture	Training Schedule			Results %	
			H	AL	steps	Precision	Recall
1	COCO	ResNet-101	50	N/A	200	85.7	15
2	COCO	ResNet-101	200	N/A	200	95	47.5
3	COCO	ResNet-101	50	150	200	**93.9**	**77.5**
4	COCO	ResNet-101	100	200	200	93.6	72.5
5	COCO	ResNet-101	50	150	400	94.7	45
6	Imagenet	ResNet-50	50	150	200	92	57.5
7	Imagenet	ResNet-50	100	200	200	90.9	50



(a) original image 1 (b) original image 2 (c) original image 3 (d) original image 4 (e) original image 5

(f) detection in (a) (g) detection in (b) (h) detection in (c) (i) detection in (d) (j) detection in (e)

Fig. 4. Sample images from the crack detection results

REFERENCES

[1] P. Wang and H. Huang, "Comparison analysis on present image-based crack detection methods in concrete structures," in *Proceedings of the 3rd International Congress on Image and Signal Processing*, vol. 5, Oct 2010, pp. 2530–2533.

[2] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, and P. Fieguth, "A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure," *Advanced Engineering Informatics*, vol. 29, no. 2, pp. 196–210, 2015, doi: https://doi.org/10.1016/j.aei.2015.01.008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474034615000208

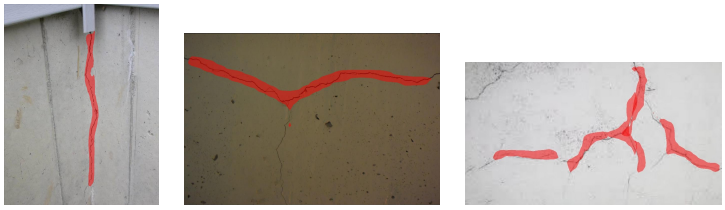
[3] L. Attard, C. J. Debono, G. Valentino, and M. D. Castro, "Tunnel inspection using photogrammetric techniques and image processing: A review," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 144, pp. 180 – 188, 2018, doi: https://doi.org/10.1016/j.isprsjprs.2018.07.010.

[4] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018, doi: https://doi.org/10.1109/TPAMI.2018.2844175.

[5] K. Y. Song, M. Petrou, and J. Kittler, "Texture crack detection," *Machine Vision and Applications*, vol. 8, no. 1, pp. 63–75, Jan 1995, doi: https://doi.org/10.1007/BF01213639.

[6] R. Medina, J. Llamas, J. Gmez-Garca-Bermejo, E. Zalama, and M. Segarra, "Crack detection in concrete tunnels using a gabor filter invariant to rotation," *Sensors (Switzerland)*, vol. 17, 07 2017, doi: https://doi.org/10.3390/s17071670.

[7] M. Salman, S. Mathavan, K. Kamal, and M. Rahman, "Pavement crack detection using the gabor filter," in *Proceedings of the 16th International*



(a) (b) (c)

Fig. 5. Samples from the crack detection results on images retrieved from the Internet and captured by us in a tunnel

TABLE II
PRECISION AND RECALL VALUES WHEN CHANGING THE LEARNING RATE.

Test No.	LR variation (init_LR = 0.001)	Training Schedule			Results %	
		H	AL	steps	Precision	Recall
1	fixed	50	N/A	200	85.7	15
8	$LR/2$, $epochs/2$	50	N/A	200	78.6	27.5
9	Reduce on plateau	50	N/A	200	77.4	60
3	fixed	50	150	200	**93.9**	**77.5**
13	Reduce on plateau	50	150	200	90.0	67.5

TABLE III
PRECISION AND RECALL VALUES WHEN VARYING THE AUGMENTATION.

Test No.	Functions	Precision %	Recall %
10	None	92.3	60
11	horizontal, vertical flips	92.9	65
3	horizontal, vertical flips, rotation, brightness, blur	**93.9**	**77.5**
12	horizontal, vertical flips, rotation, brightness, blur, contrast normalization, crop	91.7	55

IEEE Conference on Intelligent Transportation Systems, Oct 2013, pp. 2039–2044, doi: https://doi.org/10.1109/ITSC.2013.6728529.

[8] W. Xu, Z. Tang, J. Zhou, and J. Ding, "Pavement crack detection based on saliency and statistical features," in *Proceedings of the 2013 IEEE International Conference on Image Processing*, Sep. 2013, pp. 4093–4097, doi: https://doi.org/10.1109/ICIP.2013.6738843.

[9] P. Subirats, J. Dumoulin, V. Legeay, and D. Barba, "Automation of pavement surface crack detection using the continuous wavelet transform," in *Proceedings of the 2006 International Conference on Image Processing*, Oct 2006, pp. 3037–3040, doi: https://doi.org/10.1109/ICIP.2006.313007.

[10] H. Oliveira and P. L. Correia, "Automatic road crack detection and characterization," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 155–168, March 2013, doi: https://doi.org/10.1109/TITS.2012.2208630.

[11] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, and P. Fieguth, "A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure," vol. 29, no. 2, 2015, pp. 196 – 210, infrastructure Computer Vision. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474034615000208

[12] A. Cord and S. Chambon, "Automatic road defect detection by textural pattern recognition based on adaboost," *Computer-Aided Civil and Infrastructure Engineering*, vol. 27, no. 4, pp. 244–259, 2012, doi: https://doi.org/10.1111/j.1467-8667.2011.00736.x.

[13] Y. Shi, L. Cui, Z. Qi, F. Meng, and Z. Chen, "Automatic road crack detection using random structured forests," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 12, pp. 3434–3445, Dec 2016, doi: https://doi.org/10.1109/TITS.2016.2552248.

[14] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, and P. Fieguth, "A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure," *Advanced Engineering Informatics*, vol. 29, no. 2, pp. 196–210, 2015, infrastructure Computer Vision.

[15] Y.-J. Cha, W. Choi, and O. Bykztrk, "Deep learning-based crack damage detection using convolutional neural networks," *Computer-Aided Civil and Infrastructure Engineering*, vol. 32, no. 5, pp. 361–378, 2017, doi: https://doi.org/10.1111/mice.12263.

[16] W. R. L. d. Silva and D. S. d. Lucena, "Concrete cracks detection based on deep learning image classification," *Sensors(Basel, Switzerland)*, vol. 2, no. 8, 2018, doi: https://doi.org/10.3390/ICEM18-05387.

[17] L. Zhang, F. Yang, Y. Daniel Zhang, and Y. J. Zhu, "Road crack detection using deep convolutional neural network," in *Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP)*, Sep. 2016, pp. 3708–3712, doi: https://doi.org/10.1109/ICIP.2016.7533052.

[18] A. Zhang, K. C. P. Wang, Y. Fei, Y. Liu, C. Chen, G. Yang, J. Q. Li, E. Yang, and S. Qiu, "Automated pixel-level pavement crack detection on 3d asphalt surfaces with a recurrent neural network," *Computer-Aided Civil and Infrastructure Engineering*, vol. 34, no. 3, pp. 213–229, 2019, doi: https://doi.org/10.1111/mice.12409.

[19] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 580–587, doi: https://doi.org/10.1109/CVPR.2014.81. [Online]. Available: https://doi.org/10.1109/CVPR.2014.81

[20] R. Girshick, "Fast r-cnn," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 1440–1448, doi: https://doi.org/10.1109/ICCV.2015.169.

[21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, pp. 91–99. [Online]. Available: http://dl.acm.org/citation.cfm?id=2969239.2969250

[22] C. Lim, "Mask R-CNN architecture pipeline," https://www.slideshare.net/IldooKim/deep-object-detectors-1-20166, [Online; accessed May 2019].

[23] W. Abdulla, "Mask r-cnn for object detection and instance segmentation on keras and tensorflow," 2017. [Online]. Available: https://github.com/matterport/Mask_RCNN

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778, doi: https://doi.org/10.1109/CVPR.2016.90.

[25] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision - ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 740–755, doi: https://doi.org/10.1007/978-3-319-10602-1_48.

[26] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[27] A. B. Jung, "imgaug," https://github.com/aleju/imgaug, 2018, [Online; accessed May 2019].

[28] "SDNET2018: An annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks," *Data in Brief*, vol. 21, pp. 1664–1668, 2018, doi: https://doi.org/10.1016/j.dib.2018.11.015.

[29] A. Bréhéret, "Pixel Annotation Tool," 2017. [Online]. Available: https://github.com/abreheret/PixelAnnotationTool

VR-SHM - A structural health monitoring tool to assist crack detection using deep learning and virtual reality

L Attard[1,3], C J Debono[1,4], G Valentino[1,5], M di Castro[2,6], A Masi[2,7]

1 Department of Communications and Computer Engineering, University of Malta
2 Engineering Department, Survey, Mechatronics and Measurements group, CERN

E-mail: [3] leanne.attard@um.edu.mt
[4] c.debono@ieee.org [5] gianluca.valentino@um.edu.mt
[6] mario.di.castro@cern.ch [7] alessandro.masi@cern.ch

Abstract. Monitoring concrete surfaces of infrastructures to identify defects at an early stage is paramount to cost-effectively manage structural assets such that preventive measures can be taken to avoid larger infrastructural damages as well as to prevent accidents that might otherwise take place. During concrete inspection, the most sought after defects are cracks as they are the earliest indications of structure degradation. Traditional methods of crack detection rely on on-site visual monitoring which is time consuming, tedious and expensive, human-subjective and exposes inspectors to possible hazardous environments. To mitigate this, a number of computer vision-based crack detection techniques were developed to automate the crack detection process. Nevertheless, these techniques have not yet replaced visual inspection. This work aims to facilitate regular inspection of concrete structures and speed up the assessment of crack identification by providing a structural health monitoring tool to assist crack detection. It identifies cracks automatically through a deep learning architecture and then displays the identified cracks on a 3D model which can then be viewed using Virtual Reality (VR) for better contextualisation, enabling the identified defects to be further analysed remotely. The proposed system contributes 1) a crack detection technique using deep learning 2) 3D reconstruction of the infrastructure and 3) visualisation of detected cracks using VR.

1. Introduction

To prevent the impending degradation of the critical infrastructure that serves society, periodic inspection and condition assessment for long-term monitoring has recently become more critical. Large concrete structures including motorways, bridges, dams, tunnels and other buildings may develop defects as they age, becoming susceptible to losing their primary functions. Hence, the idea of Structural Health Monitoring (SHM) technology emerged with the goal of safeguarding the operational safety of structures by monitoring the on-going performance in terms of safety and serviceability. In addition, SHM supports planning and decision making on maintenance programs and predicts future conditions, hence identifying long-term needs that allow preventive measures to be taken in due time.

Such monitoring is traditionally done through deploying various sensor types and inspectors monitoring diversified physical quantities during routine observations. However, such inspections can be laborious, time-consuming, expensive and sometimes even dangerous. To address some

of these problems, improved inspection and monitoring approaches requiring less intervention from humans have been proposed through the use of computer vision techniques.

Since images capture visual information similar to that acquired by human inspectors, automatic structural surveys that are similar to visual inspection can be anticipated through their use. Images can encode information from a complete field of view in a non-contact manner, implicitly addressing the challenges of monitoring using contact sensors. Moreover, such vision-based approaches, used in conjunction with cameras integrated on robots and unmanned aerial vehicles (UAVs), offer the potential for rapid and automated inspection and monitoring. Thus, a significant amount of research in the civil engineering community has concentrated on developing and adapting computer vision techniques for various inspection tasks. Such vision-based tasks include recognition of structural elements, 3D modelling, detection and localisation of defects such as cracks and spalling and structural displacement measurement to name a few.

Building and maintaining infrastructures typically involves four phases: construction, monitoring, preventive maintenance and repair; throughout which civil engineers increasingly demand a visual and geometrical digital representation of the structure's surfaces. Achieving effective flow of information both to and from infrastructure sites and conducting actionable analytics for condition assessment require intuitive visualisation of information provided throughout the process.

Computer vision techniques can also help with respect to surface structure documentation. Using the image data captured on site and generating 3D models of the infrastructure helps with better documentation and reporting. Moreover such models offer a means of remote inspection and better post-survey analysis. Furthermore, viewing 3D models using virtual reality (VR) technology provides additional benefits including a better context when viewing inspection results and can also provide a means of tele-presence without visiting the actual site.

Through the use of robotics, photographic equipment, machine learning and visualisation techniques, the work in this paper builds on [1] whilst integrating object detection, 3D reconstruction and VR technology to propose VR-SHM, a SHM tool to assist crack detection using deep learning and its analysis on 3D and VR models. Although we consider the CERN LHC Tunnel as our scenario, such a tool can be applied to other tunnels as well as other structures in general.

The rest of the paper is organised as follows. Background information relevant to the approaches used is presented in Section 2. An overview of the proposed tool is given in Section 3. Section 4 explains the mobile platform, camera setup and image capturing. Crack detection is then described in Section 5. 3D image reconstruction and VR visualisation are presented in Section 6 and Section 7 respectively. A summary and ideas for future work conclude the paper.

2. Background information

2.1. Structural Health Monitoring (SHM)

SHM involves non-destructive sensing to analyse structural characteristics to identify the occurrence of damage, define its location and estimate its severity. In addition, behavioural data of the in-service condition of a structure is gathered and continuous assessment is carried out. Early detection of damages helps to save the structure following timely repair. It improves the safety and reliability of the structure, reduces maintenance costs and extends service life. Rapid developments in sensors, wireless communications, micro electro-mechanical systems (MEMS), integrated circuits and information technology led to an improvement in SHM through the use of smart sensors with embedded microprocessors and wireless communication links as reviewed in [2]. Among such sensors, there are accelerometers, acoustic emission (AE) and fibre optic sensors (FOS).

Lately, a significant number of innovative sensing and monitoring systems based on computer vision have been exploited for SHM. This technology has various distinctive benefits including

non-contact, non-destructive, long distance, high precision, immunity to electromagnetic interference and multiple target monitoring. As reviewed in [3], [4] and [5] some of the applications of machine vision for SHM recorded in literature include:

- 3D surface reconstruction to transform models into a Building Information Model (BIM)
- automatic vision-based recognition of structural components such as columns, joints etc.
- 2D and 3D structural displacement monitoring
- dynamic monitoring applications such as vibration, structural stress and strain monitoring
- defect inspection including characterisation of concrete spalling, fatigue cracks in steel, asphalt defects, concrete cracks and steel corrosion
- characterisation of defects such as measuring width, length etc.

2.2. Crack Detection

Cracks on concrete surfaces are one of the earliest indication of structural degradation. The number of cracks, their type and other properties such as width and length show the degradation level and carrying capacity of the concrete structure of a surface. Their early detection allows preventive measures to be taken in order to avoid further damage. On-site inspections for cracks may require closing bridge and tunnel systems, disrupting traffic flow, building structures around high buildings as well as the shutting down of facilities within the structures being surveyed. This, leads to high expenses, time consumption and inefficiency. Furthermore, this approach lacks objectivity in the quantitative analysis as it depends on the surveyor's knowledge and experience.

To mitigate the above, various research works proposing vision-based automatic crack detection methods as a partial replacement of manual inspections on roads, bridge decks, pavements and tunnel walls were published. Surveys reviewing these works can be found in [6], [7] and [8]. Early works used a combination of image processing techniques such as thresholding, mathematical morphology and edge detection. More recent approaches cater for more challenging conditions using other methods including saliency detection, texture analysis, wavelet transform, minimal path finding and machine learning. In this work, we implement crack detection using a state of the art deep learning model for object detection.

2.3. 3D reconstruction

The availability of a 3D model of an infrastructure provides a comprehensive visual and geometric image of its environment. This is useful to structural health examiners in terms of contextualising the location of damages found during observations. Furthermore, 3D information facilitates the evaluation of defects relative to the neighbouring areas. Laser scanning is the acclaimed method used to generate 3D data. A review on its use can be retrieved in [9]. The combination of passive and active imaging sensors was proposed in [10]. In recent years, image-based 3D reconstruction of civil infrastructure has gained significant interest as surveyed in [11]. Different algorithms using this approach, could be roughly divided into two groups. Structure from Motion (SfM) detects and matches a set of visual features to find the most likely 3D structure and camera trajectory that fit the data. The second group produce produce depth maps recording the distance of object points from a viewpoint by enforcing consistency constraints.

2.4. Virtual Reality (VR)

VR involves the use of components to create and visualise a physical environment or an imaginary world through an immersive technology. It replaces the user's physical world with a virtual space with which one can interact and navigate through. There are multiple types of VR systems, each providing a different virtual experience. In Desktop-VR, the virtual environment is displayed

on a screen and one can navigate through it using traditional desktop devices such as a mouse. In Immersive-VR, a user wears a head-mounted device and experiences a high level of presence as the environment surrounds him/her rather than being on a screen. 3D-Game-Based-VR uses 3D game technology which aims to enhance user interactions through the integration of visual, interactive, network and multi-user operating technologies.

In the last few years, VR has also been recognized and implemented for various applications in the field of civil engineering. As discussed in [6], [12] and [13], VR was applied in project planning, architecture and design visualisation, construction health and safety training, equipment and operational task training. VR technology can also be utilised for structural analysis, providing an inspector with remote presence and the ability to analyse the structure without physically being on site, reducing time consumption, improving efficiency and reducing personnel risks that might otherwise be faced if persons are present on site.

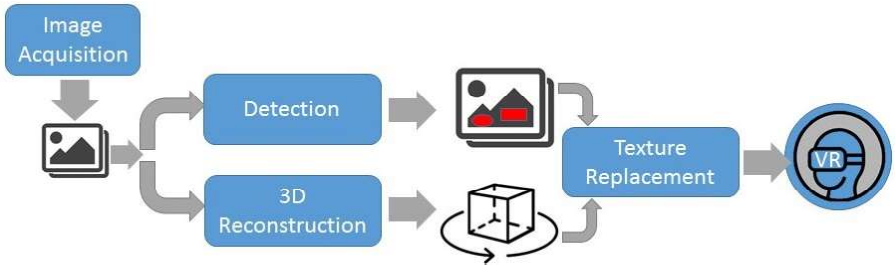


Figure 1. System pipeline of the proposed tool

3. Methodology

The VR-SHM pipleline consists of five modules as shown in Fig.1. First, the Image Acquisition component captures wall photos automatically from camera/s on a moving platform. Then, in parallel, this image data is used by the 3D reconstruction module in order to generate the 3D model of the wall while the detection module identifies and locates cracks on the wall. The next module overlays the results from the detection module on the 3D model. The final module uses VR technology to render the wall structures. The end user visualizes this on a VR headgear for the purpose of monitoring and inspection.

4. Image acquisition

In order to keep up with time and space constraints, inspection systems should be simple to set up and small in dimensions. The LHC tunnel, which we use for our environment scenario, has restricted access areas and time-windows, low lighting and dust which impose various other limitations on the choice of image acquisition set-up.

The proposed system uses a mobile platform to move a camera around the tunnel and captures images of the walls. There are currently two mobile platforms that can be used for the proposed system: Train Inspection Monorail (TIM) [14] and CERNbot [15]. The TIM uses a camera that is fixed on a robotic arm extending downwards from one of the TIM wagons as shown in Fig. 2(a). The CERNbot is equipped with a metal structure that houses multiple cameras as shown in Fig. 2(b) and is remotely operated via a graphical user interface [16] and for our solution it is driven roughly in a straight line in parallel to the tunnel wall. Images are captured by the cameras automatically and saved locally on the on-board computer while the robotic platform moves continuously. Later, the images are transferred remotely to the main processing computer.

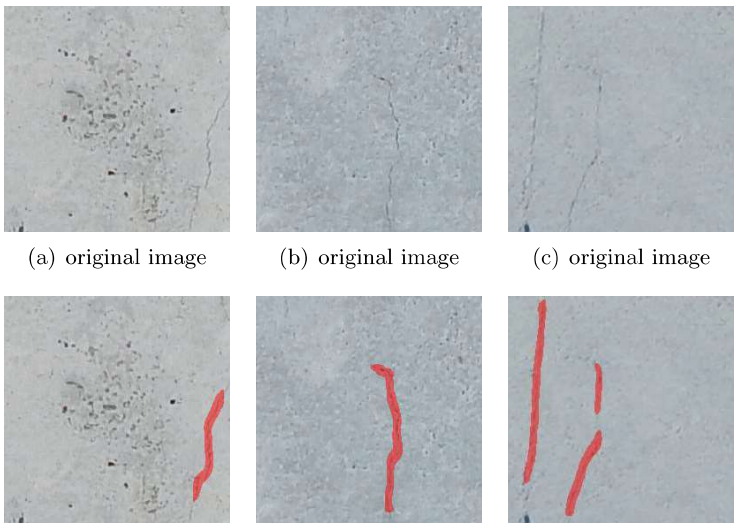


(a) TIM (b) CERNbot

Figure 2. A camera on the arm extending from one of the wagons of the TIM and multiple cameras on the CERNbot robotic platform

5. Crack Detection

The second module in Fig.1 uses a state of the art object detector [17] that not only detects targets in the image but also gives the predicted mask for each target which is useful for further processing. This detector consists of two stages. First, it generates proposals about the regions where there might be an object. Second, it predicts the class of the object, refines the bounding box and generates a mask at pixel level of the object based on the first stage proposal. This detection model has been used and proved to perform very well on natural images. Here, we use it to detect cracks and joints on concrete surfaces.

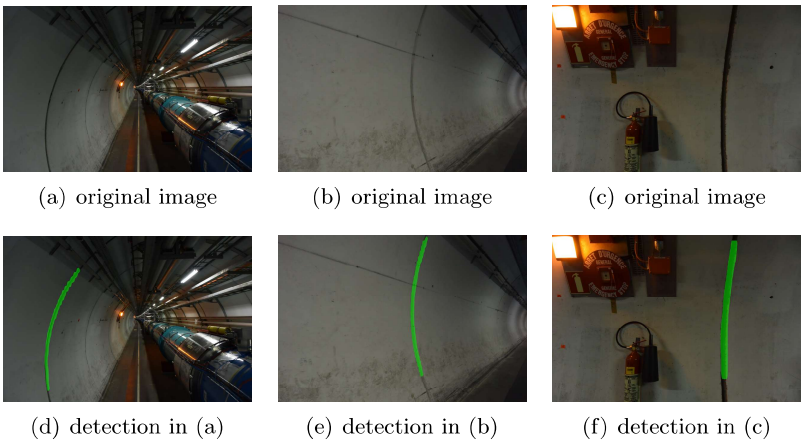


(a) original image (b) original image (c) original image

(d) detection in (a) (e) detection in (b) (f) detection in (c)

Figure 3. Sample images from the crack detection results

The detection model was therefore trained using images with cracks as well as concrete joints. Detection of the latter structure component is important to prevent any joints being falsely detected as cracks. To train the proposed model, instance object segmentation masks were required. For the crack training images we built an annotated dataset based on the SDNET dataset [18]. For joint training images, we built a dataset from images captured in the LHC tunnel and generated the ground-truth masks using an online annotation tool [19]. Various network configurations were analysed and the one providing the best detection results, with a good trade-off between the true positives and false negatives was chosen. A sample of the results



(a) original image (b) original image (c) original image

(d) detection in (a) (e) detection in (b) (f) detection in (c)

Figure 4. Sample images from the joint detection results

for the crack and joint detection can be observed in Fig.3 and Fig.4 respectively. Rather than merely classifying images whether they have a crack/joint or not, using the proposed model, the component's localisation could be also identified. Furthermore, the corresponding masks are extracted to aid with measuring further properties of the structural defects and components such as the crack width or joint length for instance.

6. 3D Reconstruction

After evaluating various existing 3D reconstruction software applications, 3DFlow Zephyr Aerial [20] commercial software was chosen due to its high quality output even when the image content lacks features. It uses SfM techniques to calculate the camera parameters and generate a sparse point cloud. Then, it uses a multi-view stereo algorithm to create a dense point cloud. A mesh is then generated and later textured using the content in the original images. Such a model can be kept as a reference model when comparing with future generated models to identify any changes that could have occurred during this time. A sample 3D reference model is shown in Fig. 5. Using the resulting images from the detection module, texture replacement is carried out on the reference model such that the cracks and joints identified earlier can now be displayed on the 3D model as shown in Fig. 6. 3D models provide both geometric and textural information that can be easily viewed any time in case the structure is inaccessible or whenever remote access is required.

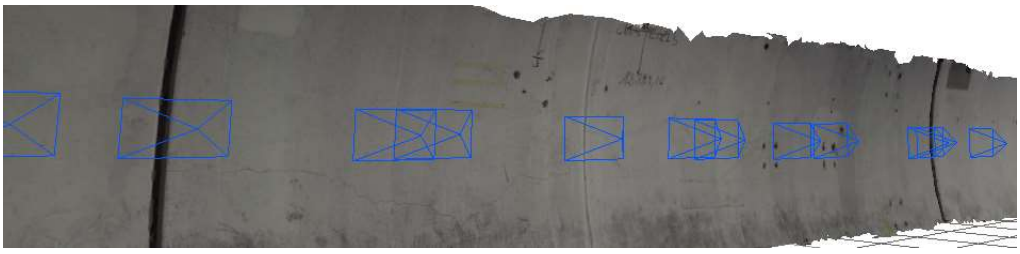


Figure 5. 3D model of the tunnel wall reconstructed using SfM techniques

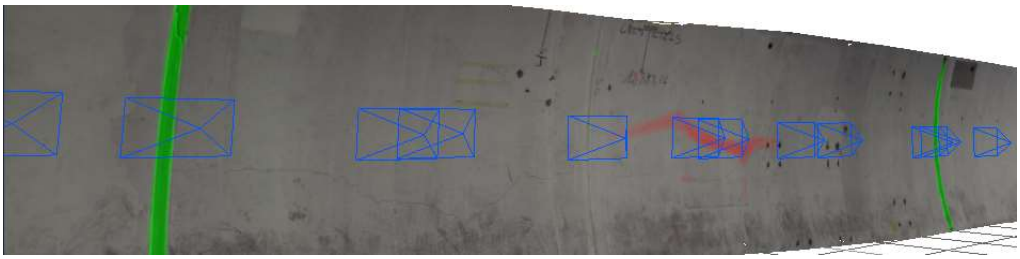


Figure 6. Detected cracks (highlighted in red) and joints (highlighted in green) overlayed on the 3D reference model of the tunnel wall

7. Virtual Reality

In SHM, VR improves the efficiency of inspection and enhances the effectiveness of monitoring. The ability to view the structure using the reference model or review the previously detected cracks and other structural components in an immersive environment allows in-detail analysis without the need of being present on site as shown in Fig. 7.

Thus, in the final module of the proposed tool, VR technology is introduced to complement 3D modeling. The 3D generated reference model as well as the overlayed model can be viewed in VR as shown in Fig. 8. Such a visualisation enables easier comparison to data from previous models in case of qualitative changes such as cracks, spalling and delamination. The VR model of a structure can also support identification and scheduling of the relevant maintenance work and future data gathering surveys.



Figure 7. Viewing the detected cracks and joints overlayed on the 3D model in VR-SHM



(a) (b)

Figure 8. Screenshots of the virtual tunnel wall viewing using VR

8. Conclusion and Future work

Concrete infrastructures should be monitored regularly to identify defects at an early stage, preventing the impending degradation of structures. Manual inspection is time-consuming, tedious, expensive, subjective and may expose humans to hazardous environments. To mitigate such drawbacks, a SHM tool is proposed to facilitate regular inspection through remotely assisted image capturing and automatic crack detection, reducing administrative time in transposing information from paper to digital records otherwise required by manual inspection. Furthermore, VR-SHM speeds up structural surveys through the possibility of remote inspection and provides a contextualised means of post-inspection analysis using 3D models and VR technology. The proposed tool's contributions include crack detection using deep learning, 3D reconstruction of an infrastructure and visualisation of detected cracks using VR. Future work intended for the tool involves detection of other defects and better VR interaction such as remote measurements.

References

[1] Attard L, Debono C J, Valentino G, di Castro M, Osborne J A, Scibile L and Ferre M 2018 A comprehensive virtual reality system for tunnel surface documentation and structural health monitoring *Proc. IEEE Int. Conf. on Imaging Systems and Techniques (IST)* pp 1–6 ISSN 1558-2809

[2] Teresa R, Jasper D and Lakshmanan I 2018 Development of structural health monitoring system the state-of-the art-review *Proc. Int. Conf. on Emerging and Sustainable trends in civil engineering* pp 33–8

[3] Spencer B F, Hoskere V and Narazaki Y 2019 *J. Engineering* **5**(2) 199–222 ISSN 2095-8099 URL `http://www.sciencedirect.com/science/article/pii/S2095809918308130`

[4] Koch C, Paal S G, Rashidi A, Zhu Z, Knig M and Brilakis I 2014 *J. Advances in Structural Engineering* **17**(3) 303–18

[5] X W Y, Chuan-Zhi D and Ts L 2016 *J. of Sensors* **2016** 1–10

[6] Wang P and Huang H 2010 Comparison analysis on present image-based crack detection methods in concrete structures *Proc. 3rd Int. Congress on Image and Signal Processing* vol 5 pp 2530–3

[7] Koch C, Georgieva K, Kasireddy V, Akinci B and Fieguth P 2015 *Advanced Engineering Informatics* **29**(2) 196–210 ISSN 1474-0346 infrastructure Computer Vision URL `http://www.sciencedirect.com/science/article/pii/S1474034615000208`

[8] Attard L, Debono C J, Valentino G and Castro M D 2018 *ISPRS J. of Photogrammetry and Remote Sensing* **144** 180–8 ISSN 0924-2716

[9] Frohlich C and Mettenleiter M 2004 *Int. Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* **36**

[10] Paar G, Bauer A and Kontrus H 2005 Texture-based fusion between laser scanner and camera for tunnel surface documentation *Proc. 7th Int. Conf. of Optical 3-D Measurement Techniques*

[11] Fathi H, Dai F and Lourakis M 2015 *J. Advanced Engineering Informatics* **29**(2) 149–161 ISSN 1474-0346 infrastructure Computer Vision URL `http://www.sciencedirect.com/science/article/pii/S1474034615000245`

[12] Li X, Yi W, Chi H L, Wang X and Chan A P 2018 *J. Automation in Construction* **86** 150–162 ISSN 0926-5805 URL `http://www.sciencedirect.com/science/article/pii/S0926580517309962`

[13] Wang P, Wu P, Wang J, Chi H L and Wang X 2018 *Int. J. of Environmental Research and Public Health* **15** 1204

[14] Di Castro M, Baiguera Tambutti M L, Gilardoni S, Losito R, Lunghi G and Masi A 2017 LHC train control system for autonomous inspections and measurements *Proc. 16th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS 2017)*

[15] Di Castro M, Buonocore L R, Ferre M, Gilardoni S, Losito R, Lunghi G and Masi A 2017 A dual arms robotic platform control for navigation, inspection and telemanipulation *Proc. 16th Int. Conf. on Accelerator and Large Experimental Physics Control Systems(ICALEPCS 2017)*

[16] Lunghi G, Prades R M and Castro M D 2016 An advanced, adaptive and multimodal graphical user interface for human-robot teleoperation in radioactive scenarios *Proc. 13th Int. Conf. on Informatics in Control, Automation and Robotics (ICINCO 2016)* pp 224–31 ISBN 978-989-758-198-4

[17] He K, Gkioxari G, Dollar P and Girshick R 2018 *IEEE Transactions on Pattern Analysis and Machine Intelligence*

[18] Dorafshan S, Thomas R J and Maguire M 2018 *J. Data in Brief* **21** 1664–1668 ISSN 2352-3409

[19] Dutta A and Zisserman A 2019 *arXiv preprint arXiv:1904.10699*

[20] 3Dflow online; accessed May 2018 URL `https://www.3dflow.net/3df-zephyr-pro-3d-models-from-photos/`

Specular Highlights Detection Using a U-Net Based Deep Learning Architecture

***Abstract*—Different lighting conditions surrounding an object can cause specular reflections, resulting in specular highlights in the captured image. These can interfere with image processing algorithms, leading to false interpretation of results. Hence, applications that demand consistent object appearance require that such highlights are detected and localized for subsequent processing. Using a slightly modified U-Net architecture, we propose a semantic segmentation model to localize specular highlights. The model achieved a frequency-weighted and mean IoU of 0.83 and 0.75 respectively when tested on a benchmark dataset. Furthermore, the proposed network was also trained and tested on a custom dataset, focusing on flash light reflections in an underground tunnel environment. For this custom dataset, the model achieved a frequency-weighted and mean IoU of 0.98 and 0.80 respectively.**

***Index Terms*—specular highlights detection, semantic segmentation, U-Net**

I. Introduction

While highlights from continuous or flash lights are ubiquitous in the physical world, they can disrupt results in computer vision applications involving segmentation, detection or matching. When light is incident on a boundary between two different media, it immediately reflects back to the medium it came from. The visual appearance of specular reflections is known as a specular highlight. Its identification provides useful information for applications that need consistent object appearance such as stereo reconstruction, change detection, visual recognition and tracking. There are different types of segmentation approaches that can be applied for specular highlights detection such as those using thresholding, edge detection and clustering. While these are relatively easy to implement and incur low computational cost, they have various limitations.

In this work, we use semantic segmentation to detect specular highlights on objects in images. With the introduction of convolutional neural networks (CNN) and deep learning, semantic segmentation advanced rapidly in the last few years. Here, we use the U-Net architecture [1] with a few modifications, mainly reducing the size of the baseline model and introducing batch normalization (BN). Compared to other image processing methods such as thresholding, this method generalizes better and does not depend on any predefined values. The U-Net architecture lends itself to applications where the amount of training samples is small, such as this case, and hence was adopted as the base architecture. U-Net combines the location information from the downsampling path with the contextual information in the upsampling path to finally obtain a general information combining localization and context, which is necessary to predict a good segmentation map as required by specular highlights detection.

The rest of the paper is organized as follows. Background information on specular highlights detection is presented in Section II. Then, in Section III we briefly explain the baseline architecture upon which we base our network. Section IV describes our method in detail. Section V discusses the training and optimization techniques used. Experiments for different configurations and their results are analyzed in Section VI. A summary and ideas for future work conclude the paper.

II. Background information

An analytical survey of different approaches to detect specular highlights is presented in [2]. A common approach is that of intensity thresholding using either a fixed or adaptive threshold. The main problem of this method is the over/under estimation of highlight areas. By thresholding the Y channel at the last peak in the Y histogram of a YUV colorspace image, specularities are isolated in [3]. This method is used in endoscopic images where the context is darker and the image dynamic range is generally well distributed. However, with specular-free images this method might produce misdetections, such as white objects, while images with specularities do not necessarily have a peak at the end of the histogram. In [4], images are first converted to the HSV colorspace and then absolute bright regions are isolated by two threshold values on the V and S channels. Another approach of specularities detection using thresholding in the RGB colorspace and grayscale image is proposed in [5]. An automatic thresholding technique applied in the HSV colorspace is used in [6]. The threshold on the V channel is estimated dynamically using information from the histogram of the same channel and brightness values calculated using the RGB channel intensities.

Dimensionality reduction and optimization algorithms can also be used to isolate specular highlights. A truncated least squares approach was proposed in [7] to map color distribution between images of an object under different illumination conditions to detect specular highlights. In [8] a bi-dimensional histogram allows the exploitation of the relations between the signals of intensity and saturation of a color image. Thresholding is then applied on this histogram to isolate specularities. A histogram equalization is used to keep the threshold value constant. This method produces fast results however using histogram equalization can lead to false detections as it can emphasize other outliers as well as white surfaces and noise.

Machine learning has also been applied to the task of specular highlights detection. A perceptron neural network

is used in [9] to classify specular regions. A deep learning approach based on the SegNet [10] segmentation architecture was used to detect highlights in endoscopic images in [11]. The SegNet architecture is trained on pairs of images and dense per-pixel labels. For reflection segmentation, the labels specify whether a pixel is part of a reflection or not.

III. U-Net baseline model

The U-Net architecture consists of convolutional (CONV) layers arranged in a top-down and bottom-up manner in two paths forming a U-shaped network. The first path is referred to as the contraction or encoder path. It is made up of CONV and max-pooling layers. This path is used to extract features and capture context in an image. The second path, referred to as the expansion or decoder part is used to enable precise localization using transposed convolutions.

The architecture proposed in [1] involves the repeated application of two 3×3 unpadded convolutions, each followed by a rectified linear unit (ReLU) and a 2×2 max-pooling operation with a stride of 2 for downsampling. In each step of the decoder path, the feature map is upsampled and then a 2×2 convolution that halves the number of channels is applied. Following this, a concatenation with the corresponding feature map from the contracting path is done. Two successive 3×3 convolutions, each followed by a ReLU, are then applied. At the final layer a 1×1 convolution is used to map each feature vector to the desired number of classes. In total, the model has four levels in each path and a bridge connection in between.

IV. Methodology

The proposed model is based on the U-Net architecture described in Section III. Using a smaller network while adding other layers such as dropout and BN, the architecture in Fig. 1 is proposed to segment images to identify specular highlights.

A. Pre-processing

First, the input image is resized to the input size of the network. Following this, mean subtraction is applied for faster convergence. It involves subtracting the mean across every individual feature in the data, and has the geometric interpretation of centering the cloud of data around the origin along every dimension. Normalization is implicit as the image pixel values are all within the 0-255 range. In our work, we use the sample mean computed on a large training set of the ImageNet dataset [12] and subtract 123.68, 116.779 and 103.939 from the R, G and B channels respectively.

B. Modified U-Net architecture

The proposed model in Fig. 1 consists of three CONV blocks for each of the downsampling and upsampling paths. Each block contains two 3×3 CONV layers each followed by a ReLU. A 2×2 max-pooling layer follows each CONV block. In this path, the number of channels increases from the input three-channel image to $N = 32$ for the first block up to $N = 256$. In the upsampling phase, CONV blocks are correspondingly symmetric to those in the downsampling path, decreasing the number of channels from $N = 256$ to $N = 32$. As opposed to the the model in [1], we inserted a BN layer after each CONV layer. Furthermore, we experimented with dropout at different locations within the architecture.

C. Batch Normalization

As shown in Fig. 1, we add a BN layer after each 3×3 CONV in the CONV block. This method normalizes activations in a network across the mini-batch during training. For each feature in the mini-batch, BN computes the mean and variance of that feature. It then subtracts the mean and divides the result by the standard deviation of the mini-batch. In this way, it restricts the activations to have a zero mean and unit variance. BN rescales the normalized activations and adds a constant, ensuring the expressiveness of the network does not change. In general, BN reduces the internal covariate shift in the network during training. We thus added BN to our model to speed up the convergence during training and to apply an indirect regularization term to avoid overfitting.

D. Dropout

Neurons develop co-dependency amongst each other during training which restrains the individual power of each neuron leading to overfitting of training data. Dropout is generally used to mitigate this by providing implicit data augmentation. When using dropout, individual nodes are dropped with a probability p at each training stage, indirectly reducing the network size. The dropout step has no trainable parameters, and does not change the volume size of the output. We tested different configurations with no dropout, dropout $p = 0.2$ after each level or dropout $p = 0.2$ at the end.

V. Training and Optimization

During training, the Adadelta optimizer [13] was used with default parameters. In order to initialize the weights of the network we use the Xavier uniform initializer. The latter draws samples from a uniform distribution within limits calculated using the number of input and output units in the weight tensor. The categorical cross entropy [14] was used as the loss function.

A. Data Augmentation

The successful implementation of deep learning models demands a large amount of varied training data. When this is not feasible, data augmentation can be used such that the network learns the desired invariance and robustness properties. We generate smooth deformations of the existing image samples through vertical and horizontal flipping, vertical and horizontal displacement of -20% to 20% and rotation of -45°to 45°.

VI. Experiments and Results

The Keras deep learning framework was used to implement the proposed model and to train it using different configurations by making modifications to the code in [15]. All experiments are preformed using an Intel®Xeon®CPU E5-1630 v3, 3.70GHz $\times$ 8 and an Nvidia GeForce GTX 1080.

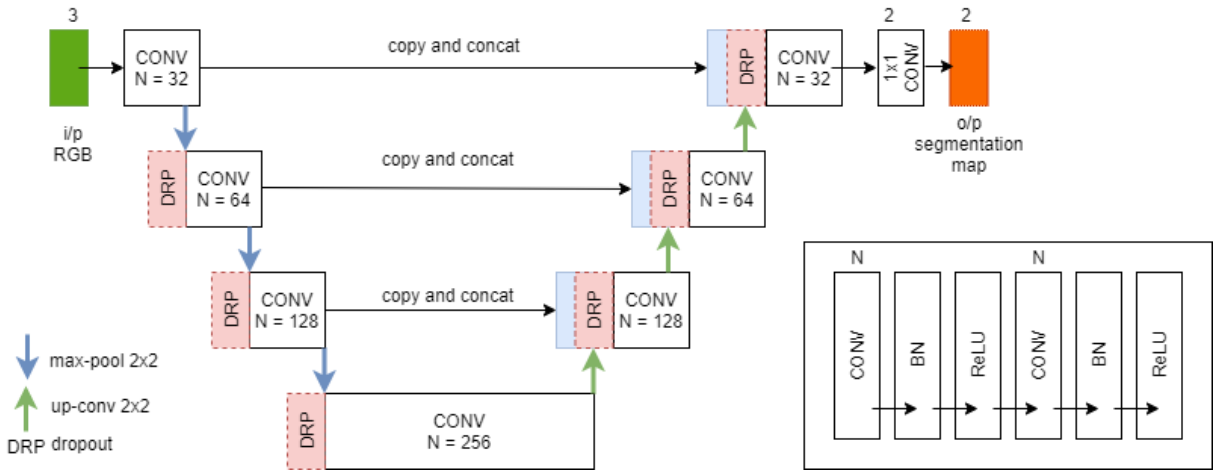


Fig. 1. U-Net model with proposed modifications.

A. Datasets

To train our network, we use the publicly available dataset PURDUE RVL SPEC-DB [7]. This dataset contains 300 images with specular highlights under three different conditions, namely ambient, directed and diffused. The images in this dataset consist of objects having different sizes, colors and materials. A ground-truth segmentation corresponding to 200 of these images is included. We used the 80/20 rule to divide the data in 128 images for training and 32 for validation. The remaining 40 images were used for testing.

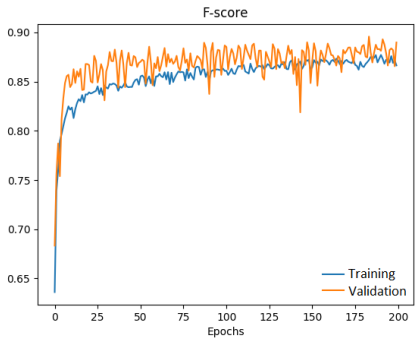
Furthermore, the proposed network was also trained and tested on a custom dataset, focusing on flash light reflections in a tunnel environment. This set contains images with a resolution of 1885×711. To generate the masks, the specular highlights in each of these images were manually marked using an annotation application [16]. Similar to the previous dataset, '0' was assigned to the background and '1' to highlight areas. The 80/20 rule was used to divide the data in 76 images for training and 18 for validation. The remaining 24 images were used for testing.

B. Evaluation Metrics

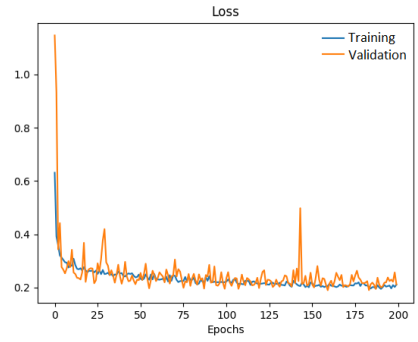
Experiments were conducted to determine the optimal configuration of the proposed model by considering different evaluation metrics. In class imbalance scenarios pixel accuracy can easily give a false good performance impression.

Thus, we use more reliable metrics consisting of the training and validation loss, intersection over union (IoU), and F-score referred to also as Dice similarity coefficient. By monitoring the loss, we could analyse the different configurations to empirically find the optimal one avoiding any underfitting or overfitting issues. The F-score/Dice and IoU are very similar, however the former divides the intersection area by the total number of pixels in both images instead of the union.

In general, the IoU tends to penalize single instances of bad classification more than the F-score quantitatively as it tends to have a "squaring" effect on the errors relative to the F-score. In summary, the F-score tends to measure the average performance, while the IoU score gives an indication of the worst case performance.



(a) F-score for batch size = 20, using BN



(b) Loss for batch size = 20, using BN

Fig. 2. Training and validation curves using batch normalization with a batch size of 20.

TABLE I
SUMMARY OF RESULTS ON THE PURDUE DATASET DURING THE VALIDATION OF THE U-NET MODEL WITH DIFFERENT ENCODER ARCHITECTURES

Backbone	Frequency-weighted IoU	Mean IoU
ResNet	0.80	0.70
VGG	0.77	0.68
MobileNet	0.81	0.70
Proposed model	0.83	0.75

C. Quantitative results

a) PURDUE RVL SPEC-DB dataset: Using the proposed model with a batch size of 1 without BN, the F-score kept increasing during training and validation. As for the loss, the curve behaved differently for training and validation. During validation the loss constantly oscillated at higher values than during training. We then trained the network using a batch size of 20 and inserting a BN layer after each CONV layer. As shown in Fig. 2, the performance of the network improved, with the training and validation curves for both the Cross-Entropy loss and Dice being very close to each other implying no overfitting. In addition to this, as depicted in Fig. 2, the validation loss did not improve after 120 epochs. Considering this, in the experiments that followed, we trained the network for 120 epochs.

Following the above, using a batch size of 20 with BN, we also experimented with the use of dropout within the network with the optimal configuration empirically found to be a dropout after each stage as shown in Fig. 1. Considering the small amount of images that we had at our disposal we also introduced a data augmentation pipeline as described in Section V-A.

In addition, we also trained the U-Net using different encoder architectures including VGG-16 and ResNet50. Metric results in Table I show that, for the relatively small dataset we had, our model with a smaller and simpler architecture, could achieve better overall results, achieving a highest frequency weighted and mean IoU of 0.83 and 0.75 respectively.

TABLE II
VALIDATION RESULTS ON THE CUSTOM TUNNEL WALL DATASET FOR DIFFERENT ENCODER ARCHITECTURES

Encoder architecture	Frequency-weighted IoU	Mean IoU
ResNet	0.97	0.73
VGG	0.98	0.76
Proposed	0.98	0.80

b) Tunnel wall dataset: Similarly, several experiments were conducted to train and test the model on the tunnel image set. These include varying configurations of the proposed encoder architecture, with and without BN, different batch sizes and dropout at different stages within the network. In addition, the U-Net model was also trained with different encoder architectures, to compare the performance with the proposed architecture.

The model was trained with a batch size of 1 and later with a batch size of 20 with BN. For both experiments, different configurations with no dropout or dropout $p = 0.2$ after each level or at the end, were tested. The optimal configuration was empirically found to be a dropout after each stage as shown in Fig. 1. However, for this dataset, a better general performance is observed when using a batch size of 1, where during both training and validation, the values of F-score were higher, while the loss was lower than when using a batch size of 20.

The U-Net model was trained using the VGG16 and ResNet-50 architectures for the encoder. In general, the training and validation curves for F-score and the loss implied that the U-Net with a VGG16 encoder architecture performed better. From these plots, we observed that the network did not exhibit any significant improvement after 75 epochs. Thus, the checkpoint at 75 epochs was used to test the models on the 'test' subset. From the results in Table II, the proposed modified architecture with a smaller and simpler architecture than VGG16 or ResNet-50, achieved better overall results when tested on a subset of new images, achieving the highest frequency weighted and mean IoU values of 0.98 and 0.80 respectively.

D. Qualitative results

As observed in the test images in Fig. 3, when comparing the segmentation results with the ground-truth masks, the proposed architecture identified the specular highlights very well. When U-Net was used with larger backbones such as VGG, ResNet and MobileNet, in general, our segmentation gives less false positive areas than VGG and with respect to MobileNet and ResNet, it gives less false negative areas.

Similarly, as observed in the custom tunnel wall test images in Fig. 4, the U-Net model with the proposed encoder architecture generated segmentation maps identifying the highlight locations as those in ground-truth ones. Compared to the larger architectures, VGG16 and ResNet-50, the proposed encoder, produced less false positive areas.

VII. CONCLUSION

When specular highlights are present, they change image features drastically, disrupting computer vision applications involving segmentation, detection or matching. Hence, their detection is essential as a preprocessing step in various applications. We presented the use of semantic segmentation to identify specular highlights with a reduced version of the U-Net model architecture. Using this model we achieved a frequency weighted and mean IoU of 0.83 and 0.75 respectively. Furthermore, the proposed network was also employed as a preprocessing stage in an inspection application focusing on change detection in tunnel environments, to prevent nuisance changes caused by flashlights. Future improvements may include training on a larger dataset comprising various surfaces, objects and backgrounds for a more generalized detection.



(a) samples images



(b) ground-truth



(c) proposed



(d) VGG



(e) MobileNet



(f) ResNet

Fig. 3. A few samples of the specular highlight detection using the proposed U-Net-based architecture compared to the corresponding ground-truth masks and other encoder architectures.

REFERENCES

[1] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.

[2] H. A. Khan, J. Thomas, and J. Hardeberg, "Analytical survey of highlight detection in color and spectral images," in *Computational Color Imaging*, S. Bianco, R. Schettini, A. Trémeau, and S. Tominaga, Eds. Cham: Springer International Publishing, 2017, pp. 197–208.

[3] T. Stehle, "Removal of specular reflections in endoscopic images," *Acta Polytechnica: Journal of Advanced Engineering*, vol. 46, pp. 32–36, 01 2006.

[4] J. Oh, S. Hwang, J. Lee, W. Tavanapong, J. Wong, and P. C. de Groen, "Informative frame classification for endoscopy video," *Medical Image Analysis*, vol. 11, no. 2, pp. 110 – 127, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S136184150600079X

[5] S. A. M. Arnold, A. Ghosh and G. Lacey, "Automatic segmentation and inpainting of specular highlights for endoscopic imaging," *EURASIP Journal on Image and Video Processing*, 2010.

[6] A. Morgand and M. Tamaazousti, "Generic and real-time detection of specular reflections in images," in *Proceedings of the 2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, vol. 1, Jan 2014, pp. 274–282.

[7] J. B. Park and A. C. Kak, "A truncated least squares approach to the detection of specular highlights in color images," in *Proceedings of the 2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, vol. 1, Sep. 2003, pp. 1397–1403 vol.1.

[8] F. Ortiz and F. Torres, "A new inpainting method for highlights elimination by colour morphology," in *Proceedings of the International Conference on Pattern Recognition and Image Analysis*, S. Singh, M. Singh, C. Apte, and P. Perner, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 368–376.

[9] S. Lee, T. Yoon, K. Kim, K. Kim, and W. Park, "Removal of specular reflections in tooth color image by perceptron neural nets," in *Proceedings of the 2010 2nd International Conference on Signal Processing Systems*, vol. 1, July 2010, pp. V1–285–V1–289.

[10] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, Dec 2017.

[11] A. Rodríguez-Sánchez, D. Chea, G. Azzopardi, and S. Stabinger, "A deep learning approach for detecting and correcting highlights in endoscopic images," in *Proceedings of the 2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)*, Nov 2017, pp. 1–6.

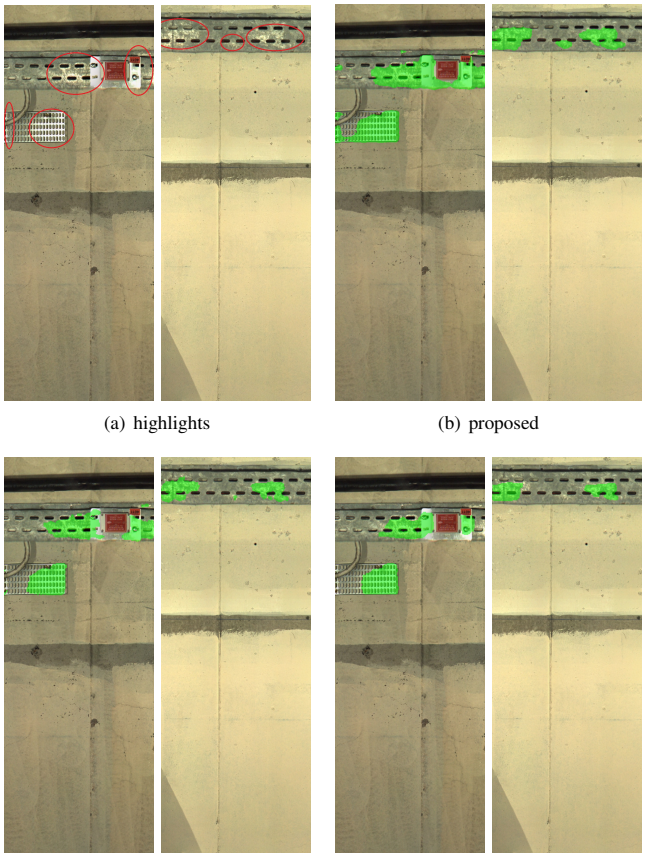
[12] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[13] M. D. Zeiler, "ADADELTA: an adaptive learning rate method," *CoRR*, vol. abs/1212.5701, 2012. [Online]. Available: http://arxiv.org/abs/1212.5701

[14] P. de Boer, D. Kroese, S. Mannor, and R. Rubinstein, "A tutorial on the cross-entropy method," *Annals of operations research*, vol. 134, no. 1, pp. 19–67, 1 2005.

[15] D. Gupta, "Image Segmentation Keras : Implementation of Segnet, FCN, UNet, PSPNet and other models in Keras," https://github.com/divamgupta/image-segmentation-keras.

[16] A. Bréhéret, "Pixel Annotation Tool," 2017. [Online]. Available: https://github.com/abreheret/PixelAnnotationTool



(a) highlights (b) proposed

(c) VGG16 (d) ResNet-50

Fig. 4. Examples of a comparison of the highlight detection results using U-Net with (b) the proposed modified architecture and (c)-(d) other architectures

Automatic crack detection in concrete infrastructure using deep learning models - a comparative anlaysis

Leanne Attard[a], Carl James Debono[a,*], Gianluca Valentino[a], Mario di Castro[b]

[a] *Department of Communications and Computer Engineering, University of Malta, Msida, MSD 2080, Malta*
[b] *Engineering Department, Survey, Mechatronics and Measurements group, CERN, Switzerland*

Abstract

Cracks are the earliest indications of infrastructure degradation. The conventional approach to localise them involves visual inspection, manual sketches and physical measurements on-site, possibly exposing persons to environments with hazardous conditions. Moreover, this approach is highly dependent on human subjectivity leading to possible inaccuracies. To mitigate these drawbacks, various works proposed automatic crack detection using image processing and object classification techniques. Whilst reliable in some scenarios, these methods use shallow abstractions and rule-based procedures which cannot overcome the inherent challenges associated with images containing cracks. In this work, we propose the identification and localisation of cracks from images through the use of deep learning. The U-Net, SegNet and Mask R-CNN models are used to compare their effectiveness at detecting concrete cracks in an image. These methods detect the areas occupied by cracks and generate the corresponding mask which is useful for further processing.

Keywords: crack detection, automatic inspection, deep learning, object detection, object segmentation, supervised learning

*Corresponding author, Tel. +356 2340 2076
Email addresses: leanne.attard@um.edu.mt (Leanne Attard), c.debono@ieee.org (Carl James Debono), gianluca.valentino@um.edu.mt (Gianluca Valentino), mario.di.castro@cern.ch (Mario di Castro)

1. Introduction

Over time, concrete infrastructures may develop cracks due to ageing, topographic changes, fluctuations between expansion and contraction of concrete due to temperature changes, heavy rainfall, cyclic weight loading and poor repair. Cracks are the earliest evidence of structural degradation hence, if detected at an early stage, preventive measures can be made to avoid larger infrastructural damages such as collapses and accidents.

The conventional approach to locate cracks in concrete structures demands visual inspection, manual sketches and physical measurements on site. Such a method depends on the inspectors' knowledge and experience, lacking objectivity. Hence, considerable effort has been made to objectively identify and assess the status of cracks in structures using image processing and pattern recognition techniques. However, challenges due to the environment and characteristics of concrete cracks, make it difficult to apply rule-based methods that are capable of effectively extracting generalised features, as these methods often rely on manually fine-tuned parameters which do not encompass the complexity of conditions that a concrete surface might exhibit. A more adaptive solution relies on the use of pattern recognition and machine learning algorithms. Although the performance of these methods is high, it is very dependent on the extracted features. Due to complicated surface conditions, it is difficult to find features effective for diverse structural scenarios.

Considering this, deep learning has been recently applied to overcome such adaptability limitations. Here, three different deep learning models; U-Net [1], SegNet [2] and Mask R-CNN [3] are trained and a comparison of their effectiveness at detecting cracks in images capturing concrete infrastructure is made.

The rest of this paper is structured as follows. The motivation behind this work is presented in Section 2. Previous works in the field of crack detection using image processing, pattern recognition and deep learning techniques are reviewed in Section 3. Automatic detection of cracks using semantic segmentation is explained in Section 4. Here an introduction on the U-Net and SegNet

models is made and the methodology applied is described. In Section 5, crack detection using instance segmentation through the Mask R-CNN model is discussed, giving related background information and details of the methodology used. The datasets used for training and testing of the models are presented in Section 6. A comparative analysis of the models used for crack detection is then made, where quantitative and qualitative results are discussed in Section 7 and 8 respectively.

2. Motivation

Our research deals with incorporating a vision-based automatic crack detection module on a robotic platform to automate structural health monitoring in tunnel environments. In this work, we first review the currently existing image-based techniques used to detect cracks. Following this, we train different deep learning models on crack datasets and make a comparative analysis of the results. This allows us to determine the optimal deep learning solution for our scenario.

3. Background information

To mitigate the drawbacks of manual crack detection, a substantial number of works to automate this process through images-based techniques have been recorded in literature. Mainly these works involve the use of image processing, pattern recognition and machine learning techniques as discussed in the following subsections.

3.1. Crack detection using image processing

Generally, crack areas are darker than those of their surroundings, resulting in lower intensity values compared to the background. Such a property allows thresholding and edge detection techniques to segment the image and extract potential crack features. In [4], crack detection in tunnel linings is proposed. Utilising the existence of luminance gradient variations along the line

edges, cracks with larger luminance variation are selected. A hysteresis threshold method is then applied to select only edges joined to others detected by high threshold values. In [5], a wireless multimedia sensor network was developed to detect cracks in subway tunnels. After pre-processing the images, threshold segmentation using the Otsu method [6] is applied to find crack regions.

Generally, cracks occupy only a small portion of the image and the inter-class variance between the background and the crack is affected by other items on the wall such as pipes and cables. To counteract these problems, a block binarisation is used in [7]. After pre-processing to enhance the contrast and remove the noise from the images, segmentation through local binarisation using the average intensity value of a square region of pixels as the threshold, is made. In [8], the grayscale values of the image pixels are used to calculate the brightness and contrast of the local image area. The overall gray value difference of the region is calculated and compared to a threshold, if it is below this predefined value, the centre pixel is recorded as a crack seed. A crack is recognised by the line connecting the crack seeds.

Other threshold-based methods for crack detection in general concrete structures include [9–15]. The thresholding technique is computationally inexpensive and relatively simple to implement, rendering it the most commonly used method, at least in preliminary stages of crack detection. On the other hand, its accuracy depends merely on the predefined value at which the threshold is set, implying some difficulty in scenarios where crack sizes vary considerably.

Visual texture is a vital characteristic to distinguish surfaces while changes in texture along a surface can identify defects or flaws in it. An algorithm using a Wigner model to identify cracks in complex textural backgrounds was proposed in [16]. By using a rotation invariant Gabor Filter, crack detection through texture-analysis was suggested in [17] and [18]. This method allows cracks to be analysed at the pixel level and to be detected regardless of their direction. Salient regions are visually more conspicuous due to their contrast with the surroundings. Although existing methods demonstrate their effectiveness in detecting salient regions in natural content images, they perform poorly

on the completeness and continuity of the detected crack. Works using saliency for crack detection such as [19] are very limited in number. Other works in crack detection involve wavelet transforms. In [20], a 2D continuous wavelet transform is used to generate complex coefficient maps, from where, wavelet coefficients maximal values are obtained for crack detection. Due to the anisotropic characteristic of wavelets, these approaches cannot handle scenarios with cracks of high curvature or low continuity.

3.2. Crack detection using pattern recognition

The previous methods have limited learning capabilities and sometimes rely on manually fine-tuned parameters as they do not encompass the complexity of conditions that a concrete surface might exhibit. A solution with better adaptability is to use pattern recognition techniques and machine learning algorithms. In [21], the crack areas identified by the thresholding stage are analysed through different features. The standard deviation of shape distance histogram, pixel number and average gray level are used as inputs to a neural network to classify the candidate objects as cracks or not. In order to detect cracks in general concrete surfaces, a support vector data description (SVDD) approach was undertaken in [22]. Properties including eccentricity, circularity and packing density are compiled into a vector and input into a trained SVDD network to identify cracks. CrackIT [23] is an integrated system, for automatic detection and classification of cracks in pavement surfaces. It uses a combination of unsupervised learning (clustering) followed by supervised learning (classification). A pavement crack detection algorithm based on fuzzy logic was introduced in [24]. To characterise cracks CrackForest [25], adopts a descriptor based on random structured forests.

3.3. Crack detection using deep learning

The performance of the previous methods is high but very dependent on the extracted features. Due to complicated surface conditions, it is hard to find features effective for all structural scenarios. Considering this, deep learning

algorithms were recently applied to overcome such adaptability limitations and automate the feature engineering and extraction process.

In [26–28], a vision-based method using a deep CNN architecture is used to detect concrete cracks. However, these works can only find patch level cracks and do not provide labels at the pixel level. In [29], a CNN is used to predict the class for each pixel of the image. However, it still needs manually designed feature extractors at a pre-processing stage, such that the CNN is only used as a classifier. In [30], edges, frequency, texture, entropy and histogram of oriented gradients (HOG) are used as inputs to a multilayer perceptron (MLP) which is trained to identify defects on the tunnel lining. In [31], crack detection using a CNN is designed through modifying the AlexNet model. Using the trained classifier and an exhaustive search with a sliding window, cracks can be separated from images accordingly. Taking the advantage of atrous convolution, atrous spatial pyramid pooling (ASPP) module and depthwise separable convolution, an end-to-end crack detection model is proposed in [32].

Semantic segmentation of cracks and leakage defects of metro shield tunnel, using hierarchies of features extracted by a fully convolutional network (FCN), is presented in [33]. Similarly, in [34], an encoder-decoder FCN with the VGG16-based encoder is trained on a subset of annotated crack-labeled images for semantic segmentation to detect cracks on concrete. Works in [35–37] use the U-Net model [1] with several modifications, to achieve pixel-level surface crack detection. To detect cracks on bridges, [38] proposes a model based on SegNet [2]. To segment crack images, DeepCrack [39] uses a skip-layer fusion to connect the encoder network and decoder network in the original SegNet model in order to utilise both sparse and continuous feature maps at each scale.

4. Semantic segmentation method

In order to understand a scene, visual information has to be associated to an entity while simultaneously considering spatial information. A better comprehension of the environment is useful in many fields. For an autonomous

car to move by itself, it needs to delimitate the roadsides with a high precision. In robotics, production machines need to delimitate the exact shape of the object to understand how to grab, turn and place components. Semantic segmentation can be used to achieve this. It consists of assigning a class to each pixel in an image, to understand the image at pixel level enabling the concept of scene understanding.

In this paper, semantic segmentation is used to segment images of walls to detect cracks using the U-Net and SegNet models. For each model, an explanation, including background information on their architectures, followed by the methodology applied, is presented.

4.1. U-Net model

The U-Net model [1] is based on an encoder-decoder architecture. This consists of multiple convolutional layers arranged in a top-down and bottom-up manner in two paths creating a U-shaped network. The first path is referred to as the contracting or encoder path. This is made up of multiple convolutional and max-pooling layers. While capturing the context in an image, this path extracts features. The second part, is referred to as the expansion or decoder path. This uses transposed convolutions to enable precise localisation.

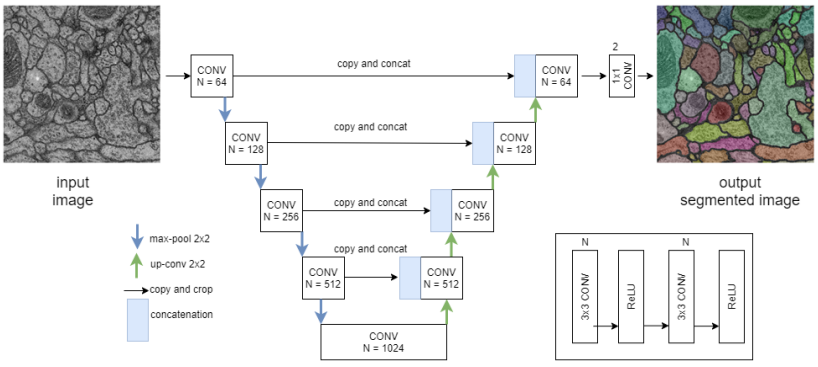


Figure 1: U-Net model architecture

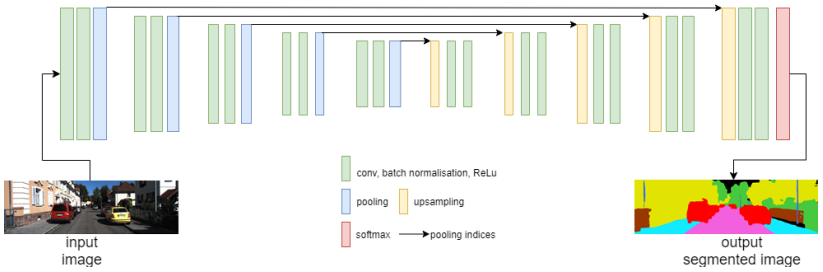


Figure 2: SegNet model architecture

As illustrated in Fig. 1, U-Net's pipeline involves the repeated application of two 3×3 unpadded convolutions. Every convolution is followed by a rectified linear unit (ReLU) and a 2×2 max-pooling operation using a stride of 2 for downsampling. In the decoder path, at each step, the feature map is upsampled and then a 2×2 convolution is applied, reducing the number of channels by a factor of two. After, the generated feature maps and the corresponding feature maps from the contracting path are concatenated. Next, two successive 3×3 convolutional layers each followed by a ReLU, are applied. At the final layer, a 1×1 convolution is used to map each feature vector to the desired number of classes. In total, the model has four levels in each of its two paths with a bridge connection in between.

4.2. SegNet model

The SegNet architecture [2] consists of an encoder and a corresponding decoder network followed by a pixel-wise classification layer. The encoder part consists of 13 convolutional layers corresponding to the VGG16 [40] network's first 13 convolutional layers. In favour of retaining higher resolution feature maps at the deepest encoder output and at the same time reducing the number of parameters, SegNet does not use the fully connected layers. For each layer in the encoder part, there is a corresponding layer in the decoder network, hence the latter has 13 layers also. At the end, to produce class probabilities for each pixel, SegNet has a multi-class soft-max classifier.

The SegNet architecture is illustrated in Fig. 2. A set of feature maps is produced using convolutions by a filter bank in each block in the encoder network. Batch normalisation (BN) and an element-wise ReLU are then applied consecutively. A max-pooling operation with a stride of 2 and a 2×2 non-overlapping window is performed. Using max-pooling, translation invariance over small spatial shifts in the input image is achieved. Sub-sampling allows a large input image context (spatial window) for each pixel in the feature map. When using multiple layers of max-pooling and sub-sampling, a loss in the spatial resolution of the feature maps occurs. To cater for this, before applying sub-sampling, SegNet captures and stores the encoder feature maps' boundary information using max-pooling indices. For each encoder feature map, the locations of the maximum feature value in each pooling window, are kept. Hence, in this respect, SegNet requires less memory than U-Net which transfers entire feature maps from the encoder to the decoder instead of using pooling indices.

Each block in the decoder network upsamples its input feature maps using the recorded max-pooling indices. The produced feature maps are convolved with a decoder filter bank generating dense feature maps on which BN is then applied. At the output of the final decoder block, the high dimensional feature representation is fed to a soft-max classifier to classify each pixel independently. Its output is a K-channel image of probabilities where K is the number of classes. For each pixel, the class with maximum probability is assigned.

4.3. Methodology used for the SegNet and U-Net architectures

To train these semantic segmentation models, the Keras deep learning framework was used. The code in [41] was adopted as a basis and then, various modifications were made to adapt the implementation to our scenario. Although the SegNet and U-Net model architectures are different, the same common training pipeline was used. First, the training and validation datasets are verified, checking that each image has its corresponding mask image. Every image-mask pair is pre-processed to the expected format at the input. The model instance is then created followed by training and validation on the respective subsets.

4.3.1. Pre-processing

The input image is first resized to the input dimensions of the network. Such dimensions are configurable and set empirically. For faster convergence during training, mean subtraction is applied. This involves subtracting the image mean from every pixel in the image. This has the geometric interpretation of centring the cloud of data around the origin along every dimension. Since image pixel values are all within the 0-255 range, normalisation is implicit. The sample mean R, G and B values, computed on a large training set of the ImageNet dataset [42] are subtracted from the R, G and B channels respectively.

4.3.2. Encoder architectures

Both SegNet and U-Net models can be used in their original architecture format or with other known architectures for the encoder part. In this work, Vanilla CNN, VGG16 [40] and ResNet-50 [43] based encoders are used with the two models and a comparative analysis of the trained models is made.

4.3.3. Data augmentation

A deep learning model requires training on a large amount of data in order to learn the desired invariant features and have robustness properties. In this work, since the available data was limited, data augmentation is used to increase the diversity of the available data. Smooth deformations of the existing image samples are generated through vertical and horizontal flips, vertical and horizontal displacements in the range [-20%, 20%] and rotations in the range [-45°, 45°].

5. Instance segmentation method

Semantic segmentation detects objects within an image, isolates them from the background and groups them based on their type. Further to this, instance segmentation detects each individual object within a cluster of similar objects, identifying their boundaries individually. The Mask R-CNN model is an example of a deep learning approach to instance segmentation.

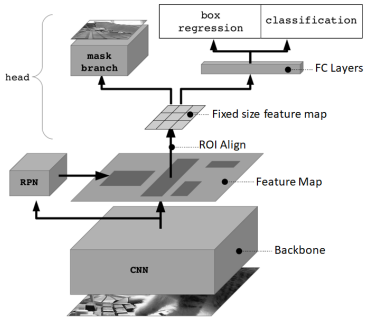


Figure 3: Mask R-CNN pipeline from [44]

5.1. Mask R-CNN model

The deep learning approach for object detection is based on the region-based CNN (R-CNN) [45]. At the first stage, a number of candidate object regions are generated. Then, for each candidate region, features are extracted.

Fast R-CNN [46] was later proposed to extend R-CNN by attending to multiple ROIs on feature maps using RoIPool. The latter led to a lower computation time and better accuracy. After that, Faster R-CNN [47] was introduced. Instead of the slow selective search algorithm, it uses a regional proposal network (RPN) to generate proposals using an end-to-end network in a single stage.

Mask R-CNN [3] was later proposed, extending Faster R-CNN to achieve pixel level segmentation. In parallel with the existing branch for bounding box recognition, Mask R-CNN adds a branch for predicting an object mask. Furthermore, in Mask R-CNN the ROI-Pooling operation is replaced by ROI-Align. ROI-Align does not adjust the input proposal from RPN to fit the feature map correctly as ROI-Pooling. Instead, it takes the object proposal and divides it into a certain number of bins. A number of points are sampled from each bin and the value at those points is determined using the bilinear interpolation. This allows more accurate instance segmentation masks to be generated.

Mask R-CNN adds only a slight overhead to Faster R-CNN. It has been used for object detection of varying classes including animals, pedestrians, cars, traffic

signs for surveillance and self-driving cars, nucleus segmentation in medical imaging and for building extraction using aerial imaging. Here, Mask R-CNN is used to detect cracks in concrete surfaces.

The model pipeline is illustrated in Fig. 3. First, the RPN outputs a set of bounding boxes (ROIs) with scores indicating the probability of having an object within them. Then, the combination of a Faster R-CNN classifier and the binary mask prediction branch is used to find the class of the object lying within the ROIs and the corresponding mask.

5.2. Methodology used for the Mask R-CNN architecture

The Mask R-CNN implementation in this work is based on that released by Matterport under the MIT license [48]. It uses the open-source libraries of Tensorflow and Keras.

5.2.1. Backbone architecture

A standard neural network that serves as a feature extractor is used for the backbone architecture. Low level features are identified by the early layers while the deeper layers successively detect features at a higher level. Mask R-CNN improves on this base architecture by using a FPN. High level features from the first pyramid are fed into a second pyramid which then passes them down to lower layers. This allows features at every level to have access to both lower and higher level features. This implementation of Mask R-CNN uses a ResNet [43] architecture with a FPN backbone.

In RPN, a lightweight neural network, finds areas containing objects. To do this, a sliding window is moved over the feature maps, using regions distributed over the image, referred to as anchors, to identify whether or not there is an object, per location per anchor box. In this work, the RPN anchor scales, ratios, stride and NMS threshold are related hyperparameters that were heuristically modified during training until satisfactory results were obtained.

Table 1: Different augmentation pipelines.

Pipeline	Functions
1	horizontal, vertical flips
2	horizontal, vertical flips, rotation, brightness, blur
3	horizontal, vertical flips, rotation, brightness, blur, contrast normalisation, crop

5.2.2. Transfer learning

Since only a limited number of training samples were available, rather than training the network end-to-end from scratch, a transfer learning methodology was used. The model is first initialised with pre-trained weights obtained by training the network on the COCO [49] and Imagenet [42] datasets. By tweaking several hyperparameters such as learning momentum, learning rate and train ROIs per image, the network is fine-tuned to adapt it to crack images.

5.2.3. Data augmentation

To further counteract the lack of training data, an augmentation pipeline was used to train the Mask R-CNN. Experimentation with several transformations for augmentation included vertical and horizontal flips, different rotations, changes in the brightness and addition of blurring using a Gaussian kernel. To investigate the benefits of using data augmentation, various pipelines were built using several functions from the *imgaug* library [50] and tested by training using the respective augmentation pipelines. A brief description of each pipeline is given in Table 1.

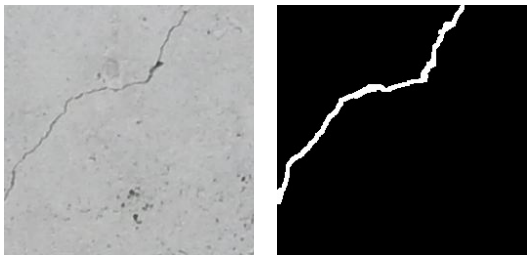
6. Datasets

To demonstrate the effectiveness of a deep learning approach for crack detection, the U-Net, SegNet and Mask R-CNN models were trained using crack

images from two different datasets. One set is based on a subset of the publicly available SDNET [51] dataset and the other is a dataset built from images captured in CERN's Large Hadron Collider (LHC) tunnel.

6.1. SDNET subset

The SDNET dataset is an annotated image set used for training and benchmarking of AI-based crack detection algorithms. It provides the crack vs non-crack ground-truth classification only, rather than masks as required by U-Net, SegNet and Mask R-CNN networks. Consequently, a mask dataset was built using a subset of 200 images from the complete SDNET set. The images have a resolution of 256×256. Using the PixelAnnotationTool [52], a brush with a small radius was used to mark the cracks, which are very narrow and long in nature. A sample from the developed mask dataset is shown in Fig. 4. The 80/20 rule was used to randomly divide the data in 128 images for training, 32 for validation and 40 for testing.



(a) crack image (b) ground-truth

Figure 4: A sample of crack markings from the SDNET dataset subset

6.2. LHC dataset

This dataset was built from images captured in the CERN LHC tunnel. This tunnel is 27 km long and lies at around 100 m below the ground, with most of it being located in France. Using a set of images captured in this tunnel, the cracks in this dataset, were manually marked using the same annotation tool [52] to generate the mask annotations. A sample from the generated mask dataset is

shown in Fig. 5. The images have a resolution of 1885×711. The 80/20 rule was used to divide the data into 110 images for training and 28 for validation. The remaining 34 images were used for testing.



(a) crack image (b) ground-truth

Figure 5: A sample of crack markings from the LHC dataset

7. Quantitative analysis

Crack images from both the SDNET subset and the LHC dataset were used to train the U-Net, SegNet and Mask R-CNN models using different configurations and hyperparameters. Experiments using different configurations of the three models were conducted to define the optimal one by analysing the resulting values of different evaluation metrics.

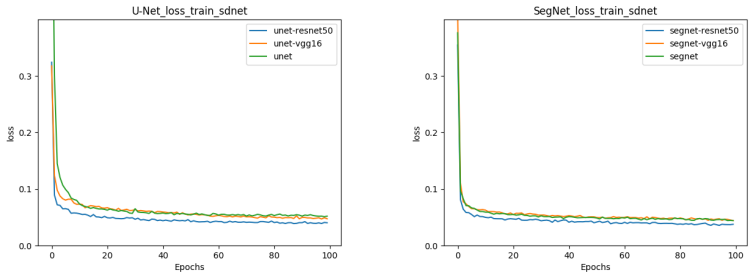
7.1. Evaluation metrics

In class-imbalanced scenarios, pixel accuracy can easily give a false impression of good performance. Hence, more reliable metrics, namely the training and validation loss and intersection over union (IoU), were used. By monitoring the model loss, different configurations can be analysed to empirically find the

optimal one, avoiding underfitting or overfitting situations. The IoU divides the intersection area of the predicted segmentation and ground-truth by the total number of pixels in both images. It measures how well a predicted segmentation matches the corresponding ground-truth annotation by dividing the intersection of two segments by their union. For the Mask R-CNN, the class and mask loss were monitored. The class loss is the RPN anchor classifier loss which reflects the confidence at which the model predicts the class labels. The mask loss is the output of a cross entropy loss function applied to the mask branch of the network and penalises wrong per-pixel binary classifications.

7.2. Analysis on the SDNET subset

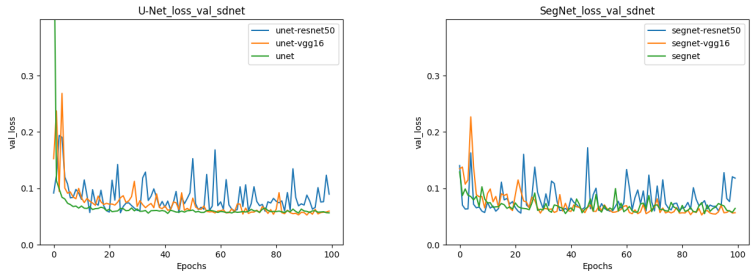
When considering the semantic segmenation method, the U-Net and SegNet models were individually trained for 200 epochs using the SDNET subset, however the models' loss reduced to a plateau even before 100 epochs were reached as can be observed in Fig. 6. When comparing these curves with the validation ones displayed in Fig. 7, one observes that the latter are not as consistent. In general U-Net performed better and had a more consistent decaying loss when using the U-Net with Vanilla CNN and VGG16 encoder architectures.



(a) U-Net (b) SegNet

Figure 6: A plot of the cross entropy loss during training of different models on the SDNET subset

During training, the IoU value was also monitored. This had a fairly consistently increasing behaviour for any of the trained models, however during



(a) U-Net (b) SegNet

Figure 7: A plot of the cross entropy loss during validation of U-Net and SegNet models with different encoder architectures, on the SDNET subset

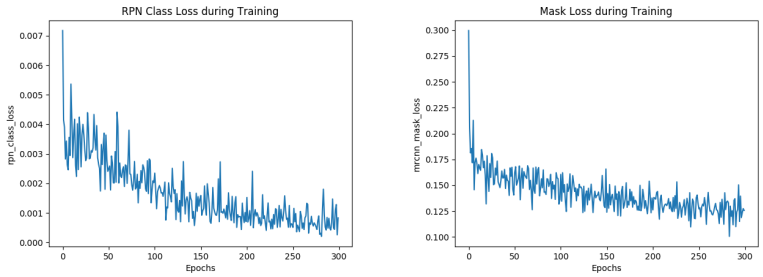
Table 2: Mean IoU from the Mask R-CNN model trained for different number of epochs on the SDNET subset

Number of Epochs	**Mean IoU**
200	**0.68**
250	0.66
300	0.67

validation, the U-Net model performed better implying improved generalisation. Furthermore, testing the models on the testing dataset confirmed that the U-Net model with a VGG-based encoder had the best segmentation performance with the highest mean IoU of 0.73 as recorded in Table 3.

The Mask R-CNN model was initialised with weights pre-trained on the Imagenet and COCO datasets for the ResNet-50 and RestNet-101 backbones respectively. Upon training the model with these two backbones and using different hyperparameters, it was noted that ResNet-101 performed slightly better in general. Hence, the Mask R-CNN model with a ResNet-101 backbone was trained with different hyperparameters and the class and mask losses were monitored to identify the number of epochs at which the model had a high probability of giving the best performance. This model was trained in different

training schedules; training only the heads of the network, training all the layers of the network and a combination of both, with the latter outperforming the others. The plots in Fig. 8, show the losses when training the heads of the network for 50 epochs followed by training all the layers for another 250 epochs using a fixed learning rate of 0.001. As observed here, training further than 200 epochs did not add any major improvements to the network. To confirm this, predictions on the testing subset were done with the trained model at different epochs, obtaining the highest IoU at 200 epochs as observed in Table 2.



(a) Class Loss (b) Mask Loss

Figure 8: The plots of the class and mask loss while training Mask R-CNN on the SDNET subset

The different augmentation pipelines listed in Table 1 were used to train the model and the optimal results were obtained when using Pipeline 3. Hence, using the Mask R-CNN model with a ResNet-101 backbone, trained for 200 epochs with a fixed learning rate and a data augmentation pipeline involving horizontal and vertical flipping, rotation, brightness, blur, contrast normalisation and cropping resulted in the optimal configuration generating the highest mean IoU of 0.68.

When comparing all the trained networks, Table 3 shows that, for the SDNET subset dataset, the U-Net with a VGG16-based encoder generated the highest mean IoU with a value of 0.73.

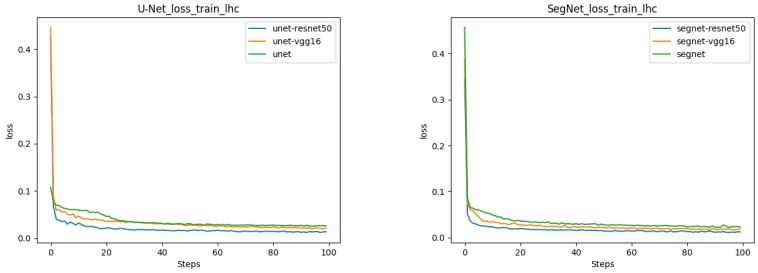
Table 3: IoU from the U-Net, SegNet and Mask R-CNN models trained on the SDNET subset

Model	**Mean IoU**
U-Net	0.70
U-Net with VGG16	**0.73**
U-Net with ResNet-50	0.56
SegNet	0.54
SegNet with VGG16	0.68
SegNet with ResNet-50	0.53
Mask R-CNN with ResNet-101	0.68

7.3. Analysis on the LHC dataset

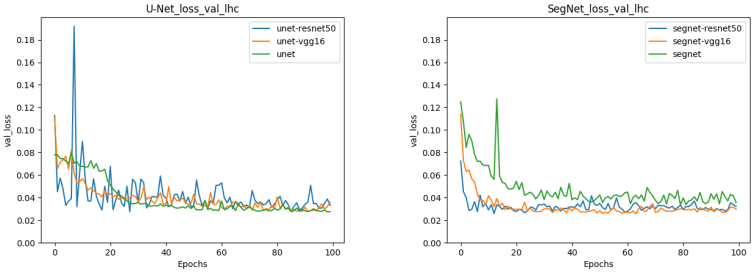
Both the U-Net and SegNet models were trained for 200 epochs however the models' loss reduced to a plateau even before 100 epochs as can be observed in Fig. 9. Furthermore, when comparing the training loss curves with the validation ones displayed in Fig. 10, the latter were not as consistent. When monitoring the IoU, we observed a consistently increasing behaviour. In contrast, during validation, the U-Net model performed better implying better generalisation. Furthermore, testing the models on the testing subset confirmed that for the LHC dataset, the U-Net model with a ResNet-based encoder had the best segmentation performance with the highest mean IoU of 0.72 as recorded in Table 5.

For this dataset, the Mask R-CNN model was also initialised with weights pre-trained on the Imagenet and COCO datasets, for the ResNet-50 and RestNet-101 backbones respectively. Upon training the model using different configurations, the one with a ResNet-101 backbone performed slightly better in general. Hence, the Mask R-CNN model with a ResNet-101 backbone architecture was trained with different hyperparameters, and the classification loss and the mask loss were monitored to identify the number of epochs at which the model had a high probability of giving the best performance.



(a) U-Net (b) SegNet

Figure 9: A plot of the cross entropy loss during training of the U-Net and SegNet models with different encoder architectures, on the LHC dataset

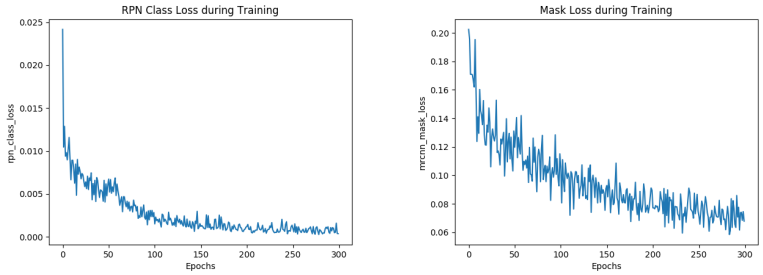


(a) U-Net (b) SegNet

Figure 10: A plot of the cross entropy loss during validation of the U-Net and SegNet models with different encoder architectures, on the LHC dataset

To train this dataset, the same training schedules were used. The plots displayed in Fig. 11, show the losses when training the heads of the network for 50 epochs followed by training all the layers for another 250 epochs using a fixed learning rate of 0.001. As noted here, training further than 200 epochs did not result in any major improvements to the network. To confirm this, predictions on the testing subset were done with the trained model at 200, 225 and 250 epochs, obtaining the highest IoU at 200 epochs as observed in Table 4.

The augmentation pipelines in Table 1 were also applied to this dataset. In this case, the optimal results were obtained when using only horizontal and



(a) Class Loss (b) Mask Loss

Figure 11: The plots of the class loss and mask loss while training the Mask R-CNN model on the LHC dataset

Table 4: Mean IoU from the Mask R-CNN model trained on the LHC dataset

Number of Epochs	**Mean IoU**
200	**0.57**
225	0.55
250	0.54

vertical flipping. The Mask R-CNN model with a ResNet-101 backbone, trained for 200 epochs with a fixed learning rate and a data augmentation pipeline involving flipping resulted in the optimal configuration generating the highest mean IoU with a value of 0.57.

When comparing all the trained models, Table 5 shows that, for the LHC dataset, both the U-Net and SegNet with a ResNet-50-based encoder generated the highest mean IoU with a value of 0.72. This implies that the semantic segmentation models performed better in our scenario.

8. Qualitative Analysis

A further qualitative interpretation of the results from training the different networks on both datasets was done. A sample of these is presented in the following subsections.

Table 5: IoU from the U-Net, SegNet and Mask R-CNN models trained on the dataset built from images captured in the LHC Tunnel

Model	**Mean IoU**
U-Net	0.70
U-Net with VGG16	0.61
U-Net with ResNet-50	**0.72**
SegNet	0.63
SegNet with VGG16	0.61
SegNet with ResNet-50	**0.72**
Mask R-CNN with ResNet-101	0.57

8.1. Results from the SDNET subset

When comparing the sample results in Fig. 12 - 14 with their corresponding ones in Fig. 15 - 17, the U-Net model's performance is in general better, with the U-Net model with a VGG16 based encoder generating segmentation results closest to the ground-truth mask. The Mask R-CNN model also generated segmentations very close to the ground-truth, as shown in Fig. 18, Fig. 19 and Fig. 20. However, drawbacks include a larger architecture and longer training time.

8.2. Results from the LHC dataset

The quantitative results from the LHC dataset presented in Section 7 imply that the semantic segmentation models with a ResNet-50 encoder network both resulted in the highest IoU. This is also observed in the sample image in Fig. 22 where the U-Net and SegNet model's performance outcome was better than that of Mask R-CNN, with the segmentation maps being very close to the corresponding ground-truth of each image.

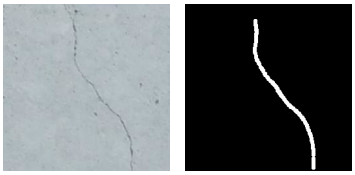


(a) original (b) ground-truth



(c) vanilla CNN (d) VGG16 (e) ResNet-50

Figure 12: Crack detection results of example 1 from the SDNET subset using the U-Net model with different encoder architectures

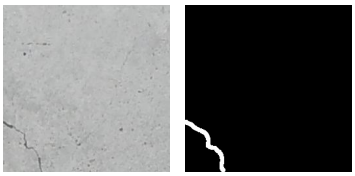


(a) original (b) ground-truth



(c) vanilla CNN (d) VGG16 (e) ResNet-50

Figure 13: Crack detection results of example 2 from the SDNET subset using the U-Net model with different encoder architectures



(a) original (b) ground-truth



(c) vanilla CNN (d) VGG16 (e) ResNet-50

Figure 14: Crack detection results of example 3 from the SDNET subset using the U-Net model with different encoder architectures

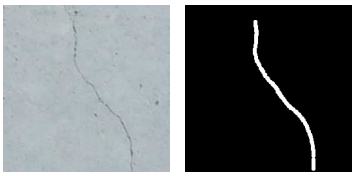


(a) original (b) ground-truth



(c) vanilla CNN (d) VGG16 (e) ResNet-50

Figure 15: Crack detection results of example 1 from the SDNET subset using the SegNet model with different encoder architectures

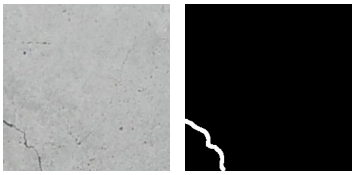


(a) original (b) ground-truth



(c) vanilla CNN (d) VGG16 (e) ResNet-50

Figure 16: Crack detection results of example 2 from the SDNET subset using the SegNet model with different encoder architectures



(a) original (b) ground-truth



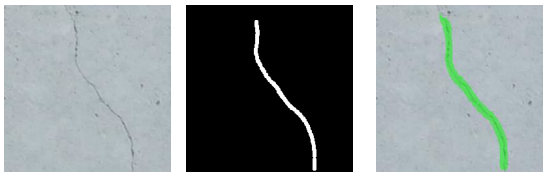
(c) vanilla CNN (d) VGG16 (e) ResNet-50

Figure 17: Crack detection results of example 3 from the SDNET subset using the SegNet model with different encoder architectures



(a) original (b) ground-truth (c) Mask R-CNN

Figure 18: Crack detection results of example 1 from the SDNET subset using the Mask R-CNN model



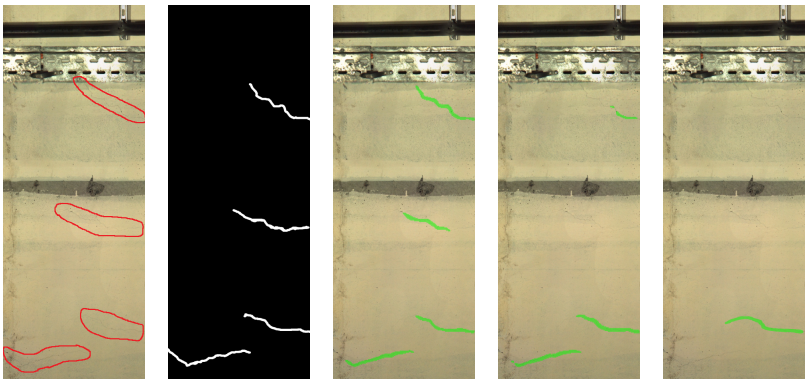
(a) original (b) ground-truth (c) Mask R-CNN

Figure 19: Crack detection results of example 2 from the SDNET subset using the Mask R-CNN model



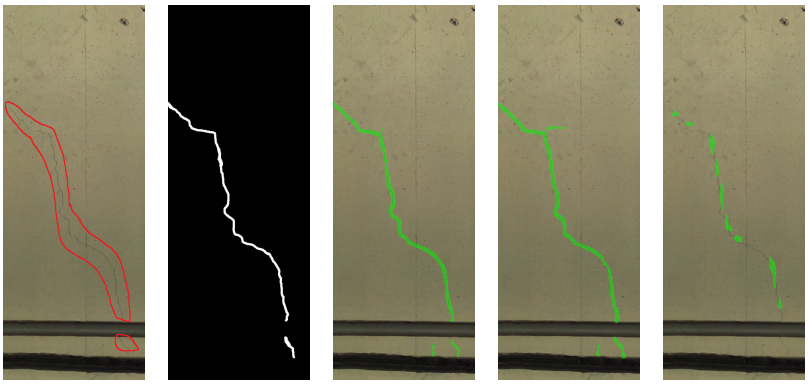
(a) original (b) ground-truth (c) Mask R-CNN

Figure 20: Crack detection results of example 3 from the SDNET subset using the Mask R-CNN model



(a) crack image (b) ground-truth (c) U-Net (d) SegNet (e) Mask R-CNN

Figure 21: First example of crack detection results from the LHC dataset using Mask R-CNN with ResNet-101 backbone and both U-Net and SegNet with a ResNet-50 encoder



(a) crack image (b) ground-truth (c) U-Net (d) SegNet (e) Mask R-CNN

Figure 22: Second example of crack detection results from the LHC dataset using Mask R-CNN with ResNet-101 backbone and both U-Net and SegNet with a ResNet-50 encoder

9. Conclusion

Our current research deals with automating structural health monitoring in tunnel environments by using computer vision solutions to detect cracks in concrete linings. In this work, we reviewed the currently existing techniques used for crack detection and focused on the application of deep learning techniques. Different deep learning models for semantic and instance segmentation were trained on crack image datasets. A comparative analysis of the results was made to find the optimal solution to be used for crack detection in tunnel linings. These results show that the semantic segmentation models with a ResNet-50 encoder are the best solutions for detecting cracks in the CERN LHC tunnel infrastructure.

References

[1] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional networks for biomedical image segmentation, in: N. Navab, J. Hornegger, W. M. Wells, A. F. Frangi (Eds.), Proc. International Conference on Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015, Springer International Publishing, 2015, pp. 234–241. doi:https://doi.org/10.1007/978-3-319-24574-4_28.

[2] V. Badrinarayanan, A. Kendall, R. Cipolla, SegNet: a deep convolutional encoder-cecoder architecture for image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (12) (2017) 2481–2495. doi:https://doi.org/10.1109/TPAMI.2016.2644615.

[3] K. He, G. Gkioxari, P. Dollar, R. Girshick, Mask R-CNN, IEEE Transactions on Pattern Analysis and Machine Intelligencedoi:https://doi.org/10.1109/TPAMI.2018.2844175.

[4] M. Ukai, N. Nagamine, A high-performance inspection system of tunnel wall deformation using continuous scan image, Proceedings of the 9th

World Congress on Railway Research.
URL http://www.railway-research.org/IMG/pdf/poster_ukai_masato.pdf

[5] B. Shen, W. Zhang, D. Qi, X. Wu, Wireless multimedia sensor network based subway tunnel crack detection method, International Journal of Distributed Sensor Networks 11 (6) (2015) 1–10. doi:https://doi.org/10.1155/2015/184639.

[6] N. Otsu, A threshold selection method from gray-level histograms, IEEE Transactions on Systems, Man, and Cybernetics 9 (1) (1979) 62–66. doi:http://dx.doi.org/10.1109/TSMC.1979.4310076.

[7] D. Qi, Y. Liu, Q.Gu, F. Zheng, J. of Comput.doi:https://doi.org/10.1109/IIH-MSP.2014.217.

[8] H. Huang, Y. Sun, Y. Xue, F. Wang, Inspection equipment study for subway tunnel defects by grey-scale image processing, Advanced Engineering Informatics 32 (2017) 188–201. doi:https://doi.org/10.1016/j.aei.2017.03.003.

[9] A. Ito, Y. Aoki, S. Hashimoto, Accurate extraction and measurement of fine cracks from concrete block surface image, in: Proc. of the IEEE 28th Annu. Conf. of the Ind. Electron. Soc. IECON 02, Vol. 3, 2002, pp. 2202–2207. doi:https://doi.org/10.1109/IECON.2002.1185314.

[10] D. Hu, T. Tian, H. Yang, S. Xu, X. Wang, Wall crack detection based on image processing, in: Proc. of the Third Int. Conf. Intell. Control and Inf. Process., 2012, pp. 597–600. doi:https://doi.org/10.1109/ICICIP.2012.6391474.

[11] Y. Fujita, Y. Hamamoto, A robust automatic crack detection method from noisy concrete surfaces, Machine Vision and Applications 22 (2) (2011) 245–254. doi:https://doi.org/10.1007/s00138-009-0244-5.

[12] T. Su, Application of computer vision to crack detection of concrete structure, International Journal of Engineering and Technology 5 (4) (2013) 457–461. doi:https://doi.org/10.7763/IJET.2014.V5.596.

[13] B. Lee, Y. Y. Kim, S. Yi, J. Kim, Automated image processing technique for detecting and analysing concrete surface cracks, Structure and Infrastructure Engineering 9 (6) (2013) 567–577. doi:https://doi.org/10.1080/15732479.2011.593891.

[14] S. Dorafshan, M. Maguire, Automatic surface crack detection in concrete structures using OTSU thresholding and morphological operations, Utah State University CEE Faculty Publications. doi:https://doi.org/10.13140/RG.2.2.34024.47363.

[15] M. Ayaho, K. Masa-Aki, B. Eugen, Automatic crack recognition system for concrete structures using image processing approach, Asian Journal of Information Technology 5 (2007) 553–561.
URL http://medwelljournals.com/abstract/?doi=ajit.2007.553.561

[16] K. Y. Song, M. Petrou, J. Kittler, Texture crack detection, Machine Vision and Applications 8 (1) (1995) 63–75. doi:https://doi.org/10.1007/BF01213639.

[17] R. Medina, J. Llamas, J. Gómez-García-Bermejo, E. Zalama, M. Segarra, Crack detection in concrete tunnels using a gabor filter invariant to rotation, IEEE Sensors 17 (7) (2017) 1–16. doi:https://doi.org/10.3390/s17071670.

[18] M. Salman, S. Mathavan, K. Kamal, M. Rahman, Pavement crack detection using the gabor filter, in: Proc. of the 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), 2013, pp. 2039–2044. doi:https://doi.org/10.1109/ITSC.2013.6728529.

[19] W. Xu, Z. Tang, J. Zhou, J. Ding, Pavement crack detection based on saliency and statistical features, in: Proc. of the 2013 IEEE Int. Conf. on Image Process., 2013, pp. 4093–4097. doi:https://doi.org/10.1109/ICIP.2013.6738843.

[20] P. Subirats, J. Dumoulin, V. Legeay, D. Barba, Automation of pavement surface crack detection using the continuous wavelet transform, in: Proc. of the 2006 Int. Conf. Image Proc., 2006, pp. 3037–3040. doi:https://doi.org/10.1109/ICIP.2006.313007.

[21] W. Zhang, Z. Zhang, D. Qi, Y. Liu, Automatic crack detection and classification method for subway tunnel safety monitoring, IEEE Sensors 14 (10) (2014) 19307–19328. doi:https://doi.org/10.3390/s141019307.

[22] L. Weiguo, L. Yaru, W. Fang, Crack detection based on support vector data description, in: Proc. of the 29th Chinese Control and Decis. Conf. (CCDC), 2017, pp. 1033–1038. doi:https://doi.org/10.1109/CCDC.2017.7978671.

[23] H. Oliveira, P. L. Correia, Automatic road crack detection and characterization, IEEE Transactions on Intelligent Transportation Systems 14 (1) (2013) 155–168. doi:https://doi.org/10.1109/TITS.2012.2208630.

[24] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, P. Fieguth, A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure, Advanced Engineering Informatics 29 (2) (2015) 196 – 210, infrastructure Computer Vision. doi:https://doi.org/10.1016/j.aei.2015.01.008.

[25] Y. Shi, L. Cui, Z. Qi, F. Meng, Z. Chen, Automatic road crack detection using random structured forests, IEEE Transactions on Intelligent Transportation Systems 17 (12) (2016) 3434–3445. doi:https://doi.org/10.1109/TITS.2016.2552248.

[26] Y.-J. Cha, W. Choi, O. Büyüköztürk, Deep learning-based crack damage detection using convolutional neural networks, Computer Aided Civil and Infrastructure Engineering 32 (5) (2017) 361–378. doi:https://doi.org/10.1111/mice.12263.

[27] W. R. L. d. Silva, D. S. d. Lucena, Concrete cracks detection based on deep learning image classification, IEEE Sensors 2 (8). doi:https://doi.org/10.3390/ICEM18-05387.

[28] L. Zhang, F. Yang, Y. Daniel Zhang, Y. J. Zhu, Road crack detection using deep convolutional neural network, in: Proc. of the 2016 IEEE Int. Conf. on Image Process. (ICIP), 2016, pp. 3708–3712. doi:https://doi.org/10.1109/ICIP.2016.7533052.

[29] A. Zhang, K. C. P. Wang, Y. Fei, Y. Liu, C. Chen, G. Yang, J. Q. Li, E. Yang, S. Qiu, Automated pixel-level pavement crack cetection on 3d asphalt surfaces with a recurrent neural network, Computer Aided Civil and Infrastructure Engineering 34 (3) (2019) 213–229. doi:https://doi.org/10.1111/mice.12409.

[30] K. Makantasis, E. Protopapadakis, A. Doulamis, N. Doulamis, C. Loupos, Deep convolutional neural networks for efficient vision based tunnel inspection, in: Proc. IEEE Int. Conf. Intell. Comput. Commun. Proc. (ICCP), 2015, pp. 335–342. doi:https://doi.org/10.1109/ICCP.2015.7312681.

[31] S. Li, X. Zhao, Image-based concrete crack detection using convolutional neural network and exhaustive search technique, Advances in Civil Engineering 2019 (2019) 1–12. doi:https://doi.org/10.1155/2019/6520620.

[32] H. Xu, X. Su, Y. Wang, H. Cai, K. Cui, X. Chen, Automatic bridge brack detection using a convolutional neural network, Applied Sciences 9 (2019) 2867. doi:https://doi.org/10.3390/app9142867.

[33] H. Huang, Q. Li, D. Zhang, Deep learning based image recognition for crack and leakage defects of metro shield tunnel, Tunnelling and Underground

Space Technology 77 (2018) 166 – 176. doi:https://doi.org/10.1016/j.tust.2018.04.002.

[34] C. V. Dung, L. D. Anh, Autonomous concrete crack detection using deep fully convolutional neural network, Automation in Construction 99 (2019) 52 – 58. doi:https://doi.org/10.1016/j.autcon.2018.11.028.

[35] Z. Liu, Y. Cao, Y. Wang, W. Wang, Computer vision-based concrete crack detection using U-Net fully convolutional networks, Automation in Construction 104 (2019) 129 – 139. doi:https://doi.org/10.1016/j.autcon.2019.04.005.

[36] J. Cheng, W. Xiong, W. Chen, Y. Gu, Y. Li, Pixel-level crack detection using U-Net, in: Proc. of TENCON 2018 - 2018 IEEE Region 10 Conference, 2018, pp. 0462–0466. doi:https://doi.org/10.1109/TENCON.2018.8650059.

[37] J. Ji, L. Wu, Z. Chen, J. Yu, P. Lin, S. Cheng, Automated pixel-level surface crack detection using U-Net, in: M. Kaenampornpan, R. Malaka, D. D. Nguyen, N. Schwind (Eds.), Proc. International Conference on Multi-disciplinary Trends in Artificial Intelligence, Springer International Publishing, Cham, 2018, pp. 69–78. doi:https://doi.org/10.1007/978-3-030-03014-8_6.

[38] C. Song, L. Wu, Z. Chen, H. Zhou, P. Lin, S. Cheng, Z. Wu, Pixel-level crack detection in images using SegNet, in: R. Chamchong, K. W. Wong (Eds.), Proc. International Conference on Multi-disciplinary Trends in Artificial Intelligence, Springer International Publishing, Cham, 2019, pp. 247–254. doi:https://doi.org/10.1007/978-3-030-33709-4_22.

[39] Q. Zou, Z. Zhang, Q. Li, X. Qi, Q. Wang, S. Wang, Deepcrack: learning hierarchical convolutional features for crack detection, IEEE Transactions on Image Processing 28 (3) (2019) 1498–1512. doi:https://doi.org/10.1109/TIP.2018.2878966.

[40] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: Y. Bengio, Y. LeCun (Eds.), Proc. 3rd International Conference Learning Representations, ICLR, 2015.
URL http://arxiv.org/abs/1409.1556

[41] D. Gupta, Image segmentation keras: implementation of Segnet, FCN, UNet, PSPNet and other models in Keras.
URL https://github.com/divamgupta/image-segmentation-keras

[42] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, ImageNet: a large-scale hierarchical image database, in: Proc. 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009. doi:https://doi.org/10.1109/CVPR.2009.5206848.

[43] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778. doi:https://doi.org/10.1109/CVPR.2016.90.

[44] L. Attard, C. J. Debono, G. Valentino, M. Di Castro, A. Masi, L. Scibile, Automatic crack detection using Mask R-CNN, in: Proc. of the 11th International Symposium on Image and Signal Processing and Analysis (ISPA), 2019, pp. 152–157. doi:https://doi.org/10.1109/ISPA.2019.8868619.

[45] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proc. 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR '14, IEEE Computer Society, Washington, DC, USA, 2014, pp. 580–587. doi:https://doi.org/10.1109/CVPR.2014.81.

[46] R. Girshick, Fast R-CNN, in: Proc. 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1440–1448. doi:https://doi.org/10.1109/ICCV.2015.169.

[47] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (6) (2017) 1137–1149. doi:https://doi.org/10.1109/TPAMI.2016.2577031.

[48] W. Abdulla, Mask R-CNN for object detection and instance segmentation on keras and tensorflow (2017).
URL https://github.com/matterport/Mask_RCNN

[49] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft COCO: Common objects in context, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), European Conference on Computer Vision ECCV 2014, Springer International Publishing, Cham, 2014, pp. 740–755. doi:https://doi.org/10.1007/978-3-319-10602-1_48.

[50] A. B. Jung, imgaug, [Online; accessed May 2019] (2018).
URL https://github.com/aleju/imgaug

[51] S. Dorafshan, R. J. Thomas, M. Maguire, SDNET2018: an annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks, Data in Brief 21 (2018) 1664–1668. doi:https://doi.org/10.1016/j.dib.2018.11.015.

[52] A. Bréhéret, Pixel Annotation Tool (2017).
URL https://github.com/abreheret/PixelAnnotationTool

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

A machine vision solution for change detection on tunnel linings using fusion

Leanne Attard · Carl James Debono · Gianluca Valentino · Mario di Castro

Received: date / Accepted: date

Abstract Tunnel inspections are predominantly done through periodic visual observations, requiring humans to be physically present on-site, possibly exposing them to hazardous environments. Furthermore, such surveys are subjective, time consuming and may require operation shutdown, thus raising the need for accurate automatic inspection systems. To improve structural health monitoring, this work proposes a remotely operated machine vision change detection application. It comprises data acquisition from a rig of cameras hosted on a robotic platform that is driven parallel to the tunnel walls. Once the data is acquired, image processing and deep learning techniques are used to pre-process the images to reduce nuisance changes cause by light variations. Data fusion techniques are then applied to identify the changes occurring in the tunnel structure. Different pixel-based change detection approaches are used to generate temporal change maps. In addition, for a more reliable detection of changes, decision-level fusion methods are used to combine the change maps obtained earlier. This is further discussed through a quantitative and qualitative analysis of the results achieved. The proposed change detection application achieved high recall and precision values of 81% and 93% respectively.

Carl James Debono
Faculty of ICT, University of Malta
Msida MSD2080, Malta
Tel.: +356-23402250
E-mail: c.debono@ieee.org

Leanne Attard
Faculty of ICT, University of Malta, Msida, Malta

Gianluca Valentino
Faculty of ICT, University of Malta, Msida, Malta

Mario di Castro
Engineering Department, SMM group, CERN, Meyrin, Switzerland

Keywords change detection · tunnel inspection · computer vision · data fusion

1 Introduction

Over time, infrastructure shows signs of deterioration due to construction defects, ageing, unexpected overloading and natural phenomena, possibly leading to problems in structural integrity. Consequently, to ensure safety in concrete tunnels, periodic inspections have to be conducted. These are predominantly performed through periodic visual observations, looking for structural defects such as cracking, spalling and water leakage to identify possible changes in the infrastructure with respect to a previous survey. To conduct such observations, personnel are required to be physically present in the tunnel. Associated with this, there are several drawbacks including the human presence in hazardous environments and the financial cost involved to train and hire people to do the inspections. In addition, these inspections require a considerable amount of time to perform, leading to longer operation down-times and thus higher monetary losses. In addition, the outcome is highly dependent on human subjectivity, leading to possible inaccuracies, false and missing detections.

All this has led to an increase in the need for robotic operations to reduce direct human intervention and machine vision applications can be used to obtain more objective results. Hence, substantial effort has been done to automate inspections using image processing to detect and classify cracks, deformities and the presence of water along the tunnel linings. Whilst defect identification is essential to automate inspection, regular monitoring of tunnel linings can provide a more informative survey to further automate inspection and analysis. Us-

ing robotics, computer vision and data fusion, here we propose a tunnel inspection application to monitor for changes on tunnel linings. The tunnel in this scenario is within CERN, the European Organization for Nuclear Research. The considered tunnel is a 27 km long tunnel lying at around 100m below the ground, hosting the world's largest particle accelerator, the Large Hadron Collider (LHC).

The remainder of this article is structured as follows. Sect.2 reviews the state of the art with respect to automated tunnel inspection and the techniques used here. The proposed solution is presented in Sect.3. Sect.4 explains the image acquisition part. In Sect.5 pre-processing tasks are described. Bi-temporal image fusion is described in Sect.6. Sect.7 discusses decision-level fusion in the context of change detection, followed by the change map (CM) analysis process presented in Sect.8. A performance evaluation is made in Sect.9. A summary and suggestions for future work conclude this article.

2 Background Information

2.1 General tunnel inspection

Research on automated health monitoring of tunnel structures has received significant attention in recent years as recorded in [17,8]. Various solutions that deal with different aspects of automated tunnel inspection were proposed through the use of cost-effective photographic equipment and computer vision. In [5] an extensive survey of works within the whole image-based tunnel inspection spectrum is presented. This includes tunnel profile monitoring, crack and leakage detection as well as tunnel surface documentation and visualisation.

2.2 Change Detection

Change detection is a well researched problem in the fields of video surveillance, remote sensing and medical imaging amongst others. Reviews of change identification methods are found in [16] and [19]. However, literature on the detection of changes on tunnel linings is still lacking, possibly due to the challenges encountered in this area. Some of these can be referred to in [14], [3], [24], [23], [10]. The goal of a change detection algorithm is to detect significant changes. Apparent intensity changes resulting from camera motion and different lighting, ideally should not be detected as changes. Hence, pre-processing steps involving geometric, radiometric adjustments and semantic segmentation are generally required as a primary stage to change detection.

2.3 Data Fusion

Data fusion combines data from different methods for increased reliability, higher redundancy and improved identification. Surveys of different fusion architectures are presented in [6] and [7]. Image fusion is a specific type of data fusion, classified into pixel, feature and decision levels. Image fusion applications can also be categorised by the time, view or modality at which the images are taken. Multi-view applications such as [20] and [26] fuse images from the same modality but from different viewpoints. Images taken at different times are combined using multi-temporal fusion to detect changes between them or to synthesise images not photographed at a desired time as in [9] and [13]. In multi-modal fusion, images coming from different sensors are combined such as in [1] and [15].

3 Solution Overview

The proposed solution is illustrated in Fig. 1. Image acquisition is made by a mobile robotic platform. Preprocessing steps involving radiometric adjustments and specular highlight localisation are applied. Bi-temporal fusion, involving image differencing, principal component analysis (PCA) and structural similarity index (SSIM) and decision-level image fusion are employed at respective stages to achieve change detection.

4 Image Data Acquisition

4.1 Acquisition system

A camera system [22] designed to inspect cylindrical environments, was identified on the market. The system is composed of the twelve unit camera rig in Fig. 2(b), two flash lights, an encoder wheel, two batteries and computer unit with software for camera synchronisation. During a demo test in the LHC tunnel, this system was integrated on CERNBot [11], one of the robotic platforms at CERN as shown in Fig. 2(a). The encoder wheel was attached to the CERNBot as shown in Fig. 2(c).

4.2 Dataset

CERNbot was driven parallel to the wall at a speed of around 0.2m/s along a section of the LHC tunnel while capturing synchronised images. This image set is referred to as $DataT_1$. Changes were then simulated by marks on the wall. The CERNbot was again driven along the same section capturing $DataT_2$.

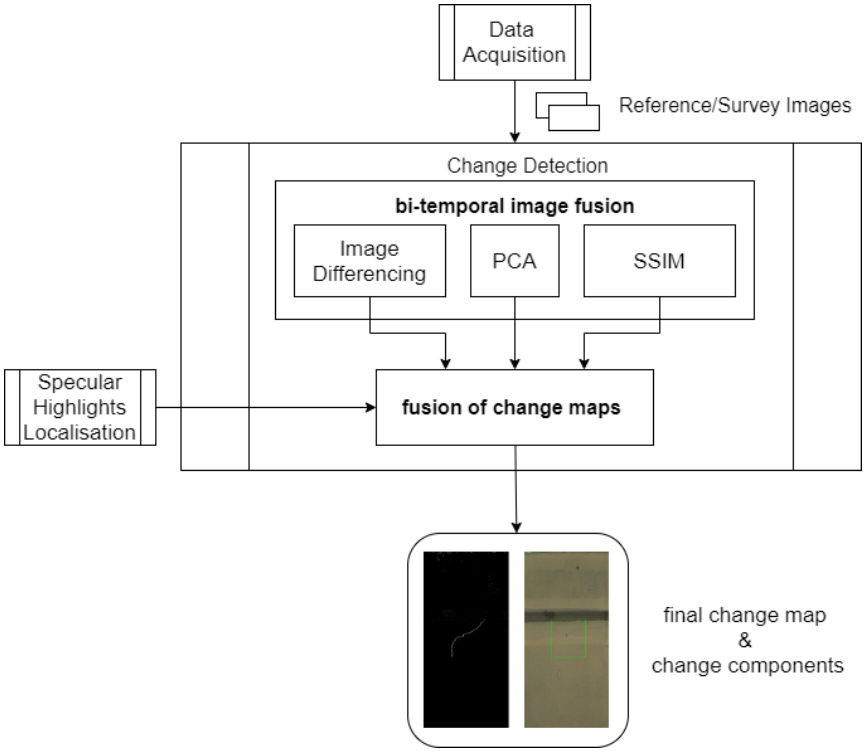


Fig. 1 Block diagram of the proposed inspection solution



(a) integration (b) camera rig (c) encoder wheel

Fig. 2 Commercial camera system [22] integrated on the CERNBot

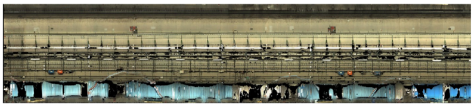
Using this data, 3D models were generated and unwrapped into orthophotos using scripts run by the company supplying the same camera system [22]. Using location information from the encoder wheel, orthophotos could be accurately registered as seen in Fig. 3 such that pixel-based change detection (PBCD) techniques could be applied. Each orthophoto is segmented in ten parts along its height and each of the image crops covers 0.5m of the tunnel length. Such images were used for training and testing of different algorithms of the solution.

5 Image Pre-processing

Changes can be due to new defects or from the evolution of already existing ones. Other changes caused by lighting sources should be identified as nuisance, preventing them from being propagated in a change detection pipeline.

5.1 Uneven illumination correction

An uneven amount of light falling on different areas causes non-uniformity in images leading to nuisance when comparing images. To adjust the uneven illumination, we use the shading algorithm in [2]. A low-pass fil-



(a) Orthophoto from $DataT_1$



(b) Orthophoto from $DataT_2$

Fig. 3 Orthophotos generated from $DataT_1$ and $DataT_2$ captured during the demo test

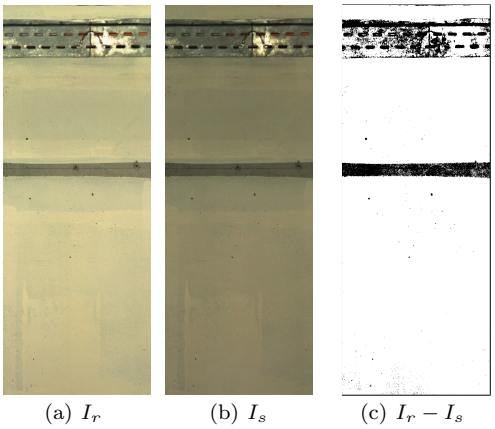
ter is applied on the original image using a median filter with a large kernel. The illumination corrected image is obtained through a pixel-wise division of the original image by the low-pass filtered image. As observed in Fig. 4(c), subtracting the original images generates a difference image full of 'white change areas', however this is due to uneven illumination. On the other hand, when the images are pre-processed to correct the uneven illumination, their difference image does not have any 'white areas' even if there is a change in lighting as shown in Fig. 4(f). Thus, this method is an effective pre-processing method to provide useful images for subsequent processing.

5.2 Specular highlight localisation

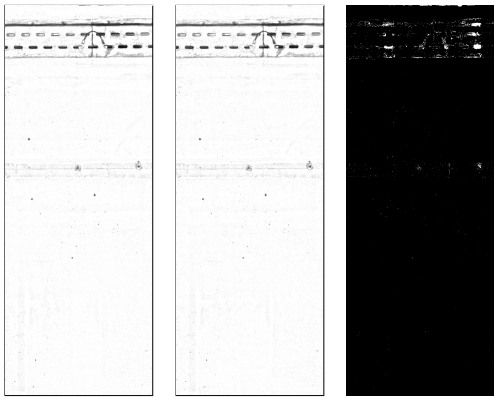
During image acquisition, flash lights cause reflections on metal racks/pipes on the wall, resulting in specular highlights in the images. Such highlights are not constant neither in time nor in place, leading to false detections when subtracting images to identify changes as shown in Fig. 4. Thus, highlight detection was implemented to localise these regions in the image pair as displayed Fig. 5. For this, semantic segmentation using the modified U-Net [21] architecture proposed in [4], is implemented. Morphological operations and connectivity analysis are then applied to the segmentation images to generate bounding boxes around highlight areas in the image pair as illustrated in Fig. 5(c). Such masks are later fused with the CM to mask out these false change candidates.

6 Bi-temporal image fusion

Multi-temporal fusion combines data from images of the same scene, acquired at different times. Hence, this approach can be used to identify changes in a scene by



(a) I_r (b) I_s (c) $I_r - I_s$



(d) C_r (e) C_s (f) $C_r - C_s$

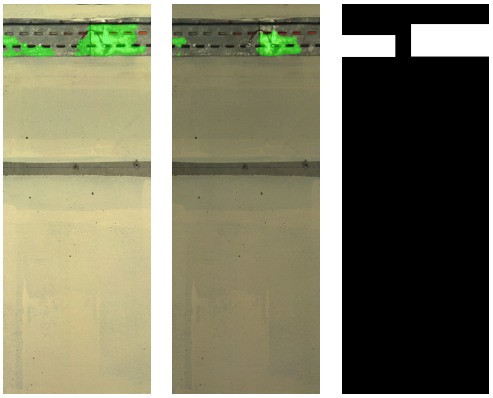
Fig. 4 The original reference (a) and survey (b) images, the difference image (c), the pre-processed reference C_r (d) and survey C_s (e) images, and the difference image of the pre-processed images (f)

comparing images. In this scenario, bi-temporal image fusion is applied between the two temporal images; reference and survey. The pair of a reference and survey image in Fig. 6 is used in the explanation of the subsequent methods.

6.1 Image Difference

In this method, two images of the same scene taken at separate times t_1 and t_2 are subtracted pixel-wise. After the subtraction, the magnitude of the difference value is compared against a threshold. Pixels with a difference magnitude higher than the pre-defined threshold are classified as 'change', otherwise noted as 'no change'. The CM is generated using:

$$Diff(x,y) = |I(x,y,t_1) - I(x,y,t_2)|$$
$$CM(x,y) = \begin{cases} 1 & \text{if } Diff(x,y) \geq T \\ 0 & \text{otherwise} \end{cases} \quad (1)$$



(a) I_r (b) I_s (c) mask

Fig. 5 Highlight localisation on the reference image (a), survey image (b) and their corresponding combined mask (c)

where $I(x, y, t_1)$ is the image at time t_1, $I(x, y, t_2)$ is the image at time t_2 and T is the threshold on the difference magnitude. This method is simple and requires low computation however, its accuracy depends on the threshold set. As T is increased, the number of change pixels decreases, implying the elimination of lower difference magnitudes, thus more noise suppression. However, the 'valid change' pixels are lost at $T \geq 30$ in this particular example.

A fixed threshold value cannot satisfy all scenarios, thus a better approach is to set the threshold automatically depending on the images being compared. The Gaussian valley emphasis (VE) method proposed in [18] is used, generating a CM with only a few 'noise changes' while retaining the 'crack change' as observed in Fig. 7.

6.2 Principal component analysis (PCA)

PCA reduces the dimensionality of a dataset while maintaining the variances. Independent data transformation analysis applies PCA on each of the temporal images separately. The derived principal components are then analysed by applying other change detection techniques such as image differencing or regression. On the other hand, merged data transformation analysis stacks N temporal images of p channels each, fuses them into a single $N \times p$-channel image and applies PCA on the latter. In this bi-temporal scenario, the merged data approach is used and the reference and survey images are stacked on each other. The method was investigated in terms of the original RGB images and the pre-processed images ie. illumination corrected images.

When RGB images ($p = 3, N = 2$) are used, the stacked images are merged into a 6-channel image. The first component (C_0), corresponding to the highest eigenvalues, contains most of the information from both images. C_1 represents the difference between temporal images while later components contain noise information. Experimental results show that PCA is scene-dependent, thus comparison between different data is often difficult to interpret using a fixed condition, implying the need of scenario-dependent thresholds. When considering C_1, the histogram shape is not clearly defined at its tails, making it difficult to find an adaptive threshold pair. When pre-processed images ($p = 1, N = 2$) are used, a stacked 2-channel image is generated. From PCA, the first component C_0 represents the difference between temporal images while C_1 contains most of the information from both images. In this case, when considering C_0, the 'crack change' has a high value (white), the 'pipe reflections change' has a low value (black) and the rest of the wall has a medium value (grey). This again, implies that the histogram contains changes at both of its tails. In this case, however, as observed in Fig. 8, the histogram shape follows a Gaussian distribution. To automatically find a threshold pair, the statistical process control (SPC) principle [25] is used.

A double threshold is heuristically determined using:

$$\begin{aligned} T_{low} &= \mu - c\sigma \\ T_{high} &= \mu + c\sigma \end{aligned} \tag{2}$$

where μ, σ are the mean and standard deviation of C_i, c is a constant.

$$CM(x,y) = \begin{cases} 1 & \text{if } C_i(x,y) > T_{high} \\ 1 & \text{if } C_i(x,y) < T_{low} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

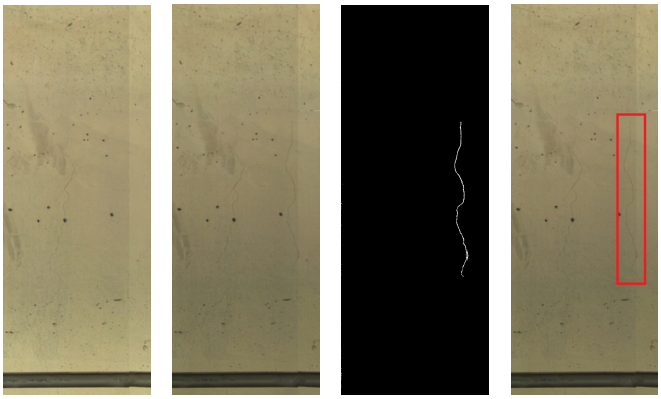
Applying Eq. 3 on the C_1 of the original and C_0 of the pre-processed images generated the CMs in Fig. 9.

6.3 Structural similarity (SSIM) index

SSIM [27] performs different similarity measurements of luminance, contrast and structure, and thereafter combines them to obtain a single value. Considering two image blocks x and y, the SSIM is given by:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x{}^2 + \mu_y{}^2 + c_1)(\sigma_x{}^2 + \sigma_y{}^2 + c_2)} \tag{4}$$

where μ_x, μ_y are the mean and $\sigma_x{}^2$, $\sigma_y{}^2$ are the variance of x and y while σ_{xy} is the covariance between x and y.



(a) I_r (b) I_s (c) CM (d) change

Fig. 6 Change detection between (a) reference and (b) survey images in an ideal-world scenario, generating the (c) ideal change map and (d) the corresponding bounding box



Fig. 7 Image differencing using Gaussian valley emphasis for automatic thresholding

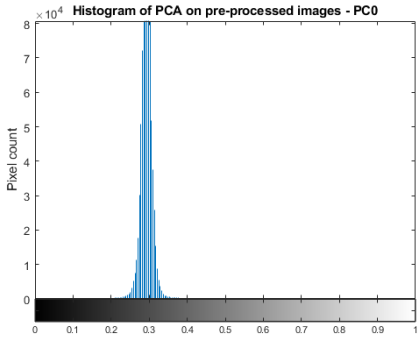
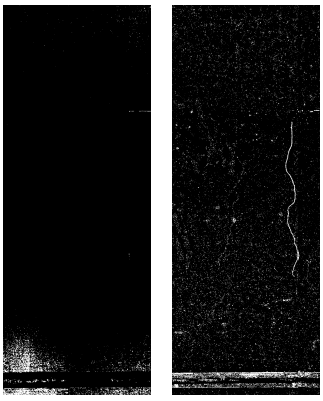


Fig. 8 Histogram of normalised C_0 from PCA on pre-processed images

Constants c_1, c_2, c_3 are calculated using:

$$c_1 = (K_1 L)^2,\ c_2 = (K_2 L)^2,\ c_3 = \frac{c_2}{2} \tag{5}$$



(a) C_1 (b) C_0

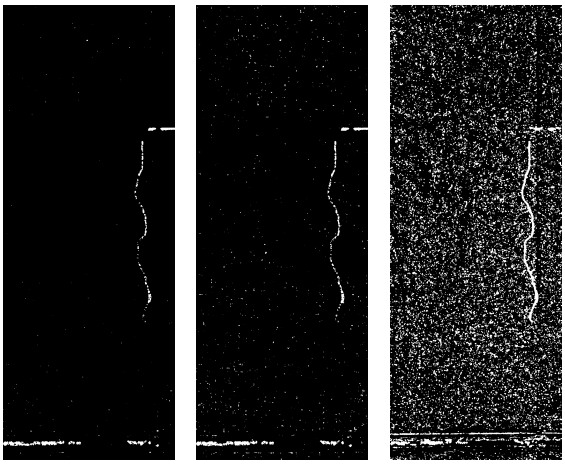
Fig. 9 CMs from PCA applied to (a) original images (C_1) (b) pre-processed images (C_0) where the 'crack change' is only identified when the pre-processed images are used

where $K_1, K_2 \ll 1$, generally $K_1 = 0.01, K_2 = 0.03$ and L is the dynamic range of the pixel values ($L = 255$ for 8-bit greyscale images). Here, SSIM is used as a PBCD method to generate a CM between a reference and survey image. The SSIM is normalised to a range of $[0, 255]$ and thresholded using:

$$\begin{aligned} D(x,y) &= 1 - \frac{SSIM(x,y)+1}{2} \\ CM(x,y) &= \begin{cases} 1 & \text{if D(x,y)} \geq \text{ T} \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{6}$$

where $D(x, y)$ represents the difference image and T is a constant. A fixed threshold value cannot satisfy all scenarios, thus the Gaussian VE automatic thresholding method is applied. An investigation of the performance in change detection is done using greyscale images, the

V channel in HSV images and pre-processed images corrected for uneven illumination. In general, the best results with minimum noise were obtained using greyscale images as shown in Fig. 10.



(a) greyscale (b) HSV (c) corrected

Fig. 10 Resulting change maps from SSIM applied to the (a) greyscale images (b) V channel in HSV images and (c) pre-processed images, corrected for uneven illumination

7 Decision-level fusion

Considering the complementary advantages of the implemented PBCD methods, the generated CMs from image differencing (CM_{diff}), PCA (CM_{PCA}) and SSIM (CM_{SSIM}) are fused into a single CM using decision-level fusion methods.

7.1 PCA-weighted sum

The PCA-based fusion algorithm is illustrated in Fig. 11. PCA is applied and the resulting components C_i are used as weights multiplied to each of the CMs. A summation of these weighted terms generates the fused CM using:

$$CM_{PCA} = CM_D \cdot C_0 + CM_{PCA} \cdot C_1 + CM_{SSIM} \cdot C_2 \quad (7)$$

As shown in Fig. 12, the PCA approach generates few noise pixels while retaining the actual changes, in this case those belonging to the crack.

7.2 Majority Voting

In the majority voting algorithm, the three different CMs; CM_D, CM_{PCA} and CM_{ssim} cast a unit vote

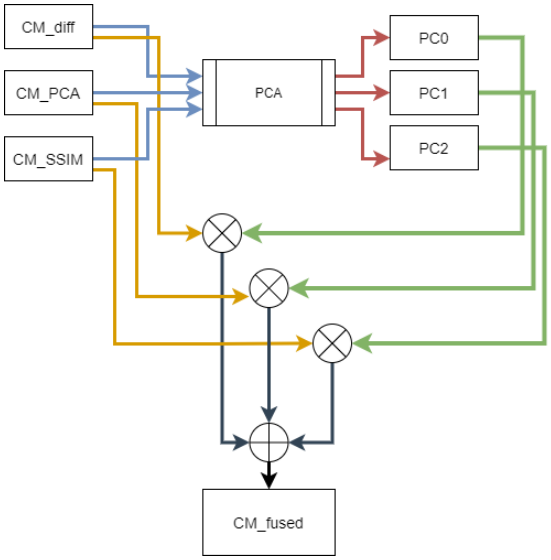


Fig. 11 Diagram of change map fusion by PCA

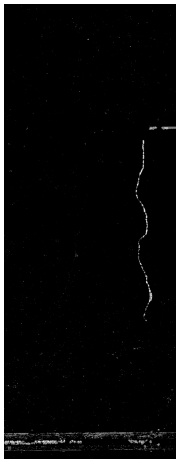


Fig. 12 Change map fusion by PCA

and if at least two of the CMs register a change, then the corresponding pixel in the fused CM is assigned '1' (change), otherwise '0' (no change). Similar to the previous method, this fusion approach generates only a few noise pixels while retaining the actual changes.

8 Change Map Analysis

At this point, the fused CM may still contain 'nuisance change' areas that should not be considered as 'changes'. Hence, the CM analysis process illustrated in Fig. 14 is developed.

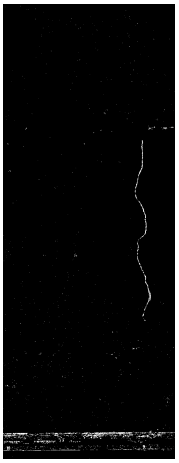


Fig. 13 Change map fusion by majority voting

8.1 Specular highlights filtering

Fusion between the inverse specular highlight mask image $SpecH(x, y)$ and the final $CM_{MV}(x, y)$ is done through an AND operation defined by:

$$CM_{filtered}(x, y) = CM_{MV}(x, y) \wedge SpecH(x, y) \tag{8}$$

8.2 Morphological operations

The filtered fused $CM_{filtered}(x, y)$ may contain some small 'change areas' coming from image noise and minor registration errors. Here, a morphological closing operation which uses dilation and erosion sequentially, is applied to the fused CM. This joins any change segments by filling gaps, such as in 'crack changes' while at the same time ignores the 'noise changes'.

8.3 Connected components labelling

Next, connected components labelling with 8-connectivity is used to identify and group neighbouring pixels into 'change components'.

8.4 Dimension Filtering

The components are now filtered by their size. A 'change component' is only retained if its width and/or height satisfies the corresponding thresholds T_W, T_H. Using the GDAL library [12], the orthophoto raster scale is obtained and using the simple proportion principle, the physical dimensions of the segment's field of view (FoV) are calculated. Using the configurable parameter d_{min} representing the minimum dimension for a detected change together with the corresponding image dimension and FoV, the thresholds are calculated using:

$$\begin{aligned} T_W &= \frac{d_{min} \times I_W}{FoV_W} \\ T_H &= \frac{d_{min} \times I_H}{FoV_H} \end{aligned} \tag{9}$$

If a candidate 'change component' has a width larger than T_W and/or a height larger than T_H then the component is confirmed as a 'change component'.

8.5 Binary Comparison

A further analysis is done to reduce false changes due to reflections, shadows and parallax errors. The images consist of a white background and darker areas where cracks, marks etc. appear. First the images are inverted, then the bounding rectangle of each 'change candidate' is masked out of both the reference and survey images using the corresponding area in the CM as a mask. The difference in number of pixels is divided by the total number of mask pixels.

Considering the same example, the difference ratios observed in Fig. 15 corresponding to the 'change candidates' in Fig. 16, whose image patches are displayed in Fig. 17 . This shows that the difference ratio for component '0' which is the 'actual change', is much larger than for the others. Thus a threshold is empirically set to filter out the 'false changes'. If the ratio is higher than a threshold, this is considered as a 'change', otherwise ignored such that in this case for example, only 'change candidate 0' is considered as a change.

9 Performance Evaluation

To demonstrate the effectiveness of the proposed change detection module, a set of experiments were conducted by simulating different changes such as cracks and other markings on the walls. In addition, some markings were also made on the images during post-processing, using a graphical editing software.

For each test scenario, the changed areas are manually marked with a red dot. The change detection output marked with green boxes and indices, is analysed and manually compared to the corresponding reference-survey image pair. An actual 'change component' is marked as a true positive (TP). Each actual 'change component' that is not detected by the algorithm is added to the false negative (FN) list. On the other hand, an area which is falsely detected as a change as it does not correspond to any of the actual changes, is noted

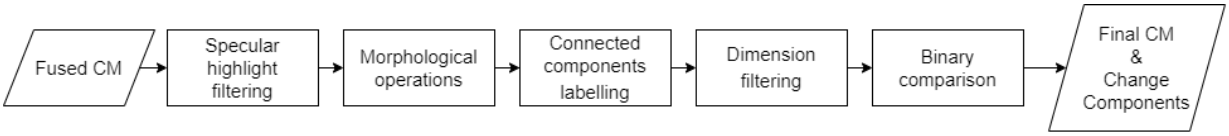


Fig. 14 Change map analysis process

```
Difference ratio [0]: 0.57
Difference ratio [1]: 0.03
Difference ratio [2]: 0.04
Difference ratio [3]: 0.05
```

Fig. 15 Difference ratios of 'change candidates'

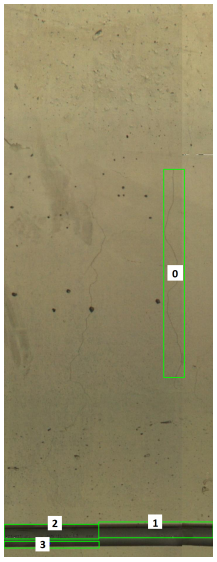


Fig. 16 Change candidates

as a false positive (FP). To quantitatively evaluate the performance of the change detection algorithm, the following metrics are used.

9.1 Evaluation Metrics

The recall is calculated using the true positive rate (TPR), implying the system's ability to find the changes. The precision is calculated using the positive detection rate (PDR) implying the system's ability to identify only the actual changes. The $F1{-}score$ is also calculated to find an optimal blend of both.

$$\begin{aligned} TPR(Recall) &= \frac{TP}{TP+FN} \times 100\% \\ PDR(Precision) &= \frac{TP}{TP+FP} \times 100\% \\ F1{-}score &= 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100\% \end{aligned} \tag{10}$$

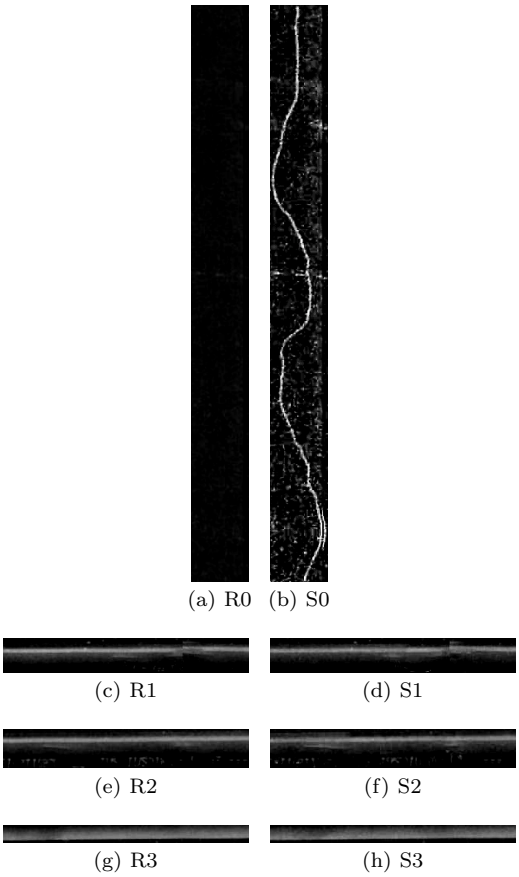


Fig. 17 Change candidates (a)-(b) reference and survey patch '0 (c)-(d) reference and survey patch '1' (e)-(f) reference and survey patch '2' (g)-(h) reference and survey patch '3'

9.2 Quantitative Analysis

The quantitative results recorded in Table 1 show that the decision-level fusion by PCA generated a higher precision rate. As the threshold of the final binary comparison was increased from 0.1 to 0.2, the precision value increased from 83.0% to 94.5%. When the majority voting approach was used, a precision of 78.8% and 93% was achieved at the same thresholds of 0.1 and 0.2 in the final comparison stage. This implies that, the PCA approach distinguished better between actual and nuisance changes.

However, it is also important to evaluate the effectiveness of the algorithm with respect to its ability to find all the data points of interest, in this case the identified changes. This is given by the recall rate, which had higher values of 83.71% and 81.11% for the majority voting approach with binary comparison stage threshold values of 0.1 and 0.2 respectively. This implies that the majority voting approach could identify more actual changes with fewer misses.

It is beneficial if the algorithm can correctly classify the changes, to avoid false alarms, however, it is important that changes due to defects on the tunnel lining are not missed. Hence, a trade-off between precision and recall is essential. This is found by analysing the $F1-score$ which combines both metrics. As observed in the Table 1, the fusion using a majority voting approach achieved a better general performance with respect to the $F1-score$.

Table 1 Quantitative results from the change detection algorithm using different decision-level fusion methods (majority voting and PCA) and different threshold values for the binary comparison in the change component analysis stage

Fusion Method	TH	TP	FP	FN	TPR %	PDR %	F1-score %
MV	0.1	149	40	29	83.7	78.8	81.0
MV	0.2	146	11	34	81.1	93.0	**86.7**
PCA	0.1	137	28	39	77.8	83.0	80.4
PCA	0.2	103	6	73	58.5	94.5	72.3

9.3 Qualitative Analysis

Further to the quantitative results, a qualitative analysis was made on different scenarios with 'crack changes', other defects and also 'nuisance changes' caused by varying light conditions and shadows.

In the example presented in Fig. 18, both of the fusion approaches identified the actual changes correctly. However, the majority voting approach gave a more confined bounding box around the 'crack change' labelled '1'.

Using the reference and survey images in Fig. 19, the change detection algorithm using majority voting correctly identified both of the 'crack changes', however the connectivity and binary comparison stages following the PCA method incorrectly identified this as a 'nuisance change' and thus discarded it.

In Fig. 20, another 'defect' was simulated on the wall. In this case, both methods correctly identified the change. The final example in Fig. 21 only exhibits 'nuisance changes' with respect to the light. Both CMs show white pixels in different areas in the image, implying possible change due to specular highlights, shadows and light changes. However, the CM analysis stage ignored most of these regions except for the small shadow area at the bottom of the image when using PCA-based fusion, generating a 'false change'.

Considering both the quantitative and qualitative results, the final implementation of the proposed solution uses the majority voting approach for the decision-level fusion and a threshold of 0.2 for the final binary comparison stage.

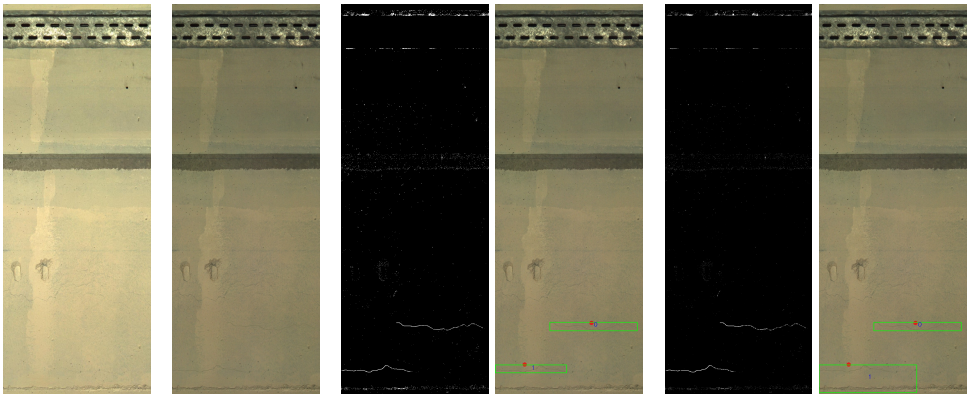
10 Conclusion and Future Work

Periodic tunnel structural inspections are a necessity. Inspections are predominantly performed through visual observations which involve looking for structural defects and making sketches for civil engineers to assess them and in turn suggest the required maintenance and/or repairs. Associated with this, there are several drawbacks including personnel exposure to hazardous conditions and outcome subjectivity that is highly dependent on human intervention which may lead to inaccuracies or misinterpretations. Considering this, a tunnel inspection solution to monitor for changes on tunnel linings was proposed. This work advances the state of the art by contributing to the fields of machine vision applications and structural inspections. An automatic image data acquisition integrated on a robotic platform is used to capture tunnel wall images. To alleviate the effects of different light conditions on change detection, pre-processing stages were also implemented. These include a shading correction to adjust uneven illumination and highlights localisation to reduce false changes due to flash light reflections. Subsequently, a change detection algorithm was developed through a combination of different bi-temporal pixel-based fusion methods and decision-level fusion of change maps. The proposed solution aids structural health monitoring and provides a better means of tunnel surface documentation.

Acknowledgements We would like to thank SITES for providing ScanTubes® camera system for a demo test in the LHC tunnel. The data collected during this test allowed us to appropriately develop and test our proposed solution.

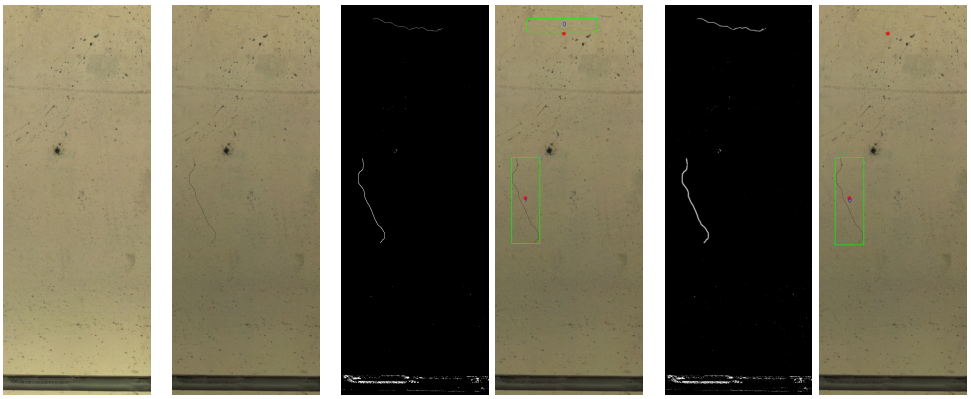
References

1. Adali, T., Jutten, C., Hansen, L.K.: Multimodal data fusion [scanning the issue]. Proc. IEEE **103**(9), 1445–1448 (2015)



(a) reference (b) survey (c) majority voting (d) PCA

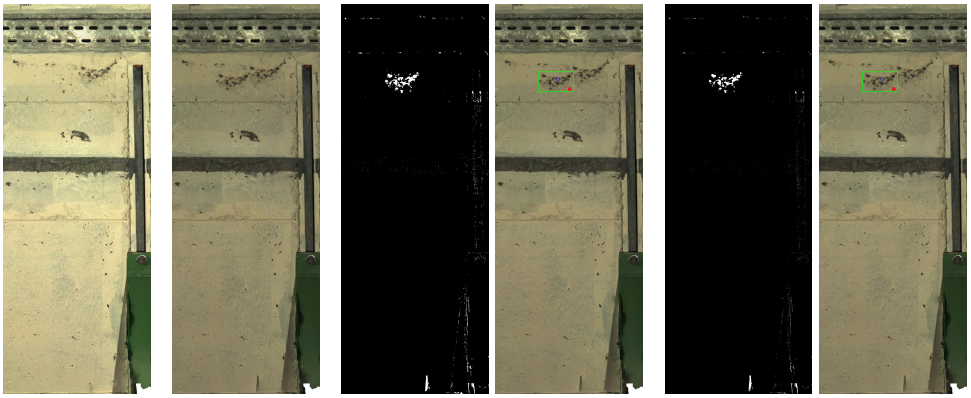
Fig. 18 An example showing similar results for both majority voting and PCA



(a) reference (b) survey (c) majority voting (d) PCA

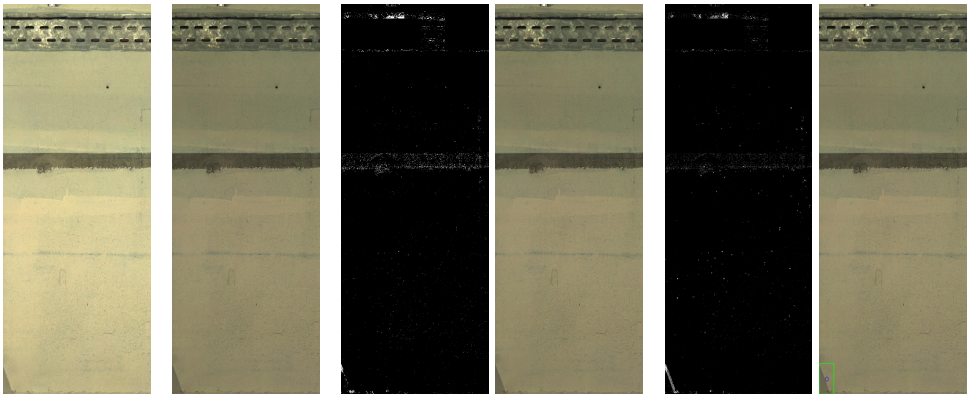
Fig. 19 An example showing different detection results from majority voting and PCA

2. Attard, L., Debono, C.J., Valentino, G., Castro, M.D.: Image mosaicing of tunnel wall images using high level features. In: Proc. 10th Int. Symposium on Image and Signal Processing and Anal., pp. 141–146 (2017)
3. Attard, L., Debono, C.J., Valentino, G., Castro, M.D.: Vis.-based change detection for inspection of tunnel liners. Automation in Construction **91**, 142–154 (2018)
4. Attard, L., Debono, C.J., Valentino, G., Castro, M.D.: Specular highlights detection using a u-net based deep learning architecture. In: submitted to the 27th Int. Conf. on Image Processing (2020)
5. Attard, L., Debono, C.J., Valentino, G., Di Castro, M.: Tunnel inspection using photogrammetric techniques and image processing: A review. ISPRS Journal of Photogrammetry and Remote Sensing **144**, 180 – 188 (2018)
6. Ayed, S.B., Trichili, H., Alimi, A.M.: Data fusion architectures: A survey and comparison. In: Proc. 15th Int. Conf. on Intelligent Systems Design and Applications (ISDA), pp. 277–282 (2015)
7. Azimirad, E., Haddadnia, J., Izadipour, A.: A comprehensive review of the multi-sensor data fusion architectures. Journal of Theoretical and Applied Information Technology **71**(1), 33–42 (2015)
8. Balaguer, C., Montero, R., Victores, J.G., Martínez, S., Jardón, A.: Towards fully automated tunnel inspection : A survey and future trends. In: Proc. 31st ISARC, Sydney, Australia, pp. 19–33 (2014)
9. Bovolo, F., Bruzzone, L.: The time variable in data fusion: A change detection perspective. IEEE Geoscience and Remote Sensing Magazine **3**(3), 8–26 (2015)
10. Crosilla, F.: Procrustes Anal. and Geodetic Sciences, pp. 287–292. Springer Berlin Heidelberg, Berlin, Heidelberg (2003)
11. Di Castro, M., Buonocore, L.R., Ferre, M., Gilardoni, S., Losito, R., Lunghi, G., Masi, A.: A dual arms robotic platform control for navigation, inspection and telemanipulation. In: Proc. 16th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS 2017) (2017)
12. GDAL/OGR contributors: GDAL/OGR Geospatial Data Abstraction software Library. Open Source Geospatial Foundation (2020). https://gdal.org
13. Jan, J.: Medical Image Processing, Reconstruction and Restoration: Concepts and Methods, pp. 481–482 (2006)
14. Jenkins, M.D., Buggy, T., Morison, G.: An imaging system for visual inspection and structural condition monitoring of railway tunnels. In: Proc. IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS), pp. 1–6 (2017)



(a) reference (b) survey (c) majority voting (d) PCA

Fig. 20 An example showing similar performance of majority voting and PCA solutions on a different simulated defect on the wall



(a) reference (b) survey (c) majority voting (d) PCA

Fig. 21 An example exhibiting lighting changes, that are correctly identified as a nuisance and not detected as a change

15. Lahat, D., Adali, T., Jutten, C.: Multimodal data fusion: An overview of methods, challenges, and prospects. Proc. IEEE **103**(9), 1449–1477 (2015)
16. Lu, D., Mausel, P., Brondízio, E., Moran, E.: Change detection techniques. Int. Journal of Remote Sensing **25**(12), 2365–2401 (2004)
17. Montero, R., Victores, J., Martanez, S., Jardon, A., Balaguer, C.: Past, present and future of robotic tunnel inspection. Automation in Construction **59**, 99–112 (2015)
18. Ng, H., Jargalsaikhan, D., Tsai, H., Lin, C.: An improved method for image thresholding based on the valley-emphasis method. In: Proc. 2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conf., pp. 1–4 (2013)
19. Radke, R.J., Andra, S., Al-Kofahi, O., Roysam, B.: Image change detection algorithms: a systematic survey. IEEE Transactions on Image Processing **14**(3), 294–307 (2005)
20. Rajini, K.C., Roopa, S.: A multi-view super-resolution method with joint-optimization of image fusion and blind deblurring. In: KSII Transactions on Internet and Information Systems, vol. 12, pp. 2366–2395 (2018)
21. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: N. Navab, J. Hornegger, W.M. Wells, A.F. Frangi (eds.) Medical Image Comput. and Comput.-Assisted Intervention - MICCAI 2015, pp. 234–241. Springer Int. Publishing, Cham (2015)
22. SITES: Scantubes. URL https://www.sites.fr/cas-pratique/inspection-scantubes/
23. Stent, S., Gherardi, R., Stenger, B., Cipolla, R.: Detecting change for multi-view, long-term surface inspection. In: Proc. British Mach. Vis. Conf. (BMVC), pp. 127–139 (2015)
24. Stent, S., Gherardi, R., Stenger, B., Soga, K., Cipolla, R.: An Image-Based System for Change Detection on Tunnel Linings. pp. 2–5 (2013)
25. Tsai, D.M., Huang, T.Y.: Automated surface inspection for statistical textures. Image and Vis. Comput. **21**(4), 307 – 323 (2003)
26. Yan, L., Fei, L., Chen, C., Ye, Z., Zhu, R.: A multi-view dense image matching method for high-resolution aerial imagery based on a graph network. Remote Sensing **8**(10) (2016)
27. Z. Wang, Bovik, A.C.: A universal image quality index. IEEE Signal Processing Letters **9**(3), 81–84 (2002)