Crowdsensing-based Monitoring of Transportation Infrastructure using Moving Vehicles

by

Qipei Mei

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

Doctor of Philosophy

in

Structural Engineering

Department of Civil and Environmental Engineering

University of Alberta

Abstract

Transportation infrastructure system is a key component of smart and sustainable cities. To support the development of future smart cities, the efficiency and sustainability of transportation infrastructure systems must be improved. In this context, there is an increasing demand for efficient and effective monitoring and sensing of such infrastructures. This thesis proposes a novel framework to monitor and sense the transportation infrastructures using crowdsensing data in vehicles. With the participation of passengers and drivers, the framework has the potential to monitor a large number of transportation infrastructures with reduced costs in a timely manner. Under this framework, this thesis first develops methods to identify damage and mode shapes of bridges using the vibration data collected from vehicles crossing these bridges. Numerical analysis and laboratory experiments are conducted to verify the proposed methods. Second, a deep learning-based algorithm is developed to automatically identify cracks in pavements using in-vehicle camera. At the end of this thesis, the conclusions and recommendations for future research are presented.

The general contribution of this thesis is the development and investigation of a crowdsensing-based framework for infrastructure monitoring using vehicles. The detailed contributions can be summarized as below: 1) A methodology is proposed for bridge condition screening using a large number of vehicles. This method identifies the damage in the bridge by introducing Mel-frequency cepstral coefficients and comparing the distributions of the extracted features. The method is verified by numerical analysis and laboratory experiments with professional sensors and smartphones. 2) A methodology is developed to identify bridge mode shapes from moving vehicles. In this method, the problem is first converted into matrix completion problem through a mapping process. Then, a “soft-imputing” algorithm is implemented to fill the matrix for system identification. The numerical results demonstrate that the methodology can find mode shapes accurately with multiple mobile sensors moving at traffic speed. 3) A methodology is proposed for pavement crack detection using in-vehicle camera and deep learning algorithm. With the help of connectivity maps and generative adversarial networks, the proposed method can achieve better performance than traditional image processing methods, and also has the potential to scan the roads quickly with reduced costs and high efficiency.

Preface

This thesis includes original research conducted by Qipei Mei. Five journal papers have been published/submitted for publication on the basis of this thesis. The details of each chapter are summarized as below:

Some of the contents in Chapter 1 have been published as ***Mei, Q., Gül, M. and Shirzad-Ghaleroudkhani, N. (2020) "Towards Smart Cities: Crowdsensing-based Monitoring of Transportation Infrastructure using Moving Vehicles." Journal of Civil Structural Health Monitoring.*** For the consistency and coherence of this thesis, contents have been modified, removed, or added from the published paper. Qipei Mei was responsible for conceptualization, methodology development, conducting experiments, analysis implementation and paper composition. Mustafa Gül was in charge of conceptualization, supervision, funding acquisition and paper editing. Nima Shirzad-Ghaleroudkhani was involved in experiments and paper composition.

Chapter 2 presents modified and reorganized versions of the literature review parts of these five papers related to this thesis.

A version of Chapter 3 has been published as ***Mei, Q., Gül, M. and Boay, M. (2019) "Indirect Health Monitoring of Bridges using Mel-frequency Cepstral Coefficients and Principal Component Analysis." Mechanical Systems and Signal Processing, 119, pp.523-546.*** For the consistency and coherence of this thesis, contents have been modified, removed, or added from the published paper. Qipei Mei was responsible for conceptualization, methodology development, conducting experiments, analysis implementation and paper composition. Mustafa Gül was in charge of conceptualization, supervision, funding acquisition and paper editing. Marcus Boay was involved in experiments.

A version of Chapter 4 has been published as ***Mei, Q., and Gül, M. (2019) "A Crowdsourcing-based Methodology using Smartphones for Bridge Health Monitoring." Structural Health Monitoring, 18(5-6), pp.1602-1619.*** For the consistency and coherence of this thesis, contents have been modified, removed, or added from the published paper. Qipei Mei was responsible for conceptualization, methodology development, conducting experiments, analysis implementation and paper composition. Mustafa Gül was in charge of conceptualization, supervision, funding acquisition and paper editing.

A version of Chapter 5 is under review for publication as ***Mei. Q., Shirzad-Ghaleroudkhani, N., Gül, M., Ghahari, F., and Taciroglu, E. (2020) "Bridge Mode Shape Identification from Traffic Speed Moving Vehicles using Non-parametric Sparse Matrix Completion." Structural Control and Health Monitoring.*** For the consistency and coherence of this thesis, contents have been modified, removed, or added from the published paper. Qipei Mei was responsible for conceptualization, methodology development, conducting experiments, analysis implementation and paper composition. Nima Shirzad-Ghaleroudkhani and Farid Ghahari were involved in paper composition and editing. Mustafa Gül and Ertugrul Taciroglu were in charge of conceptualization, supervision, funding acquisition and paper editing.

A version of Chapter 6 has been published as ***Mei, Q., and Gül, M. (2020) "A Cost Effective Solution for Road Crack Inspection using Cameras and Deep Neural Networks." Construction and Building Materials, 156, pp.119397.*** For the consistency and coherence of this thesis, contents have been modified, removed, or added from the published paper. Qipei Mei was responsible for conceptualization, methodology development, conducting experiments, analysis implementation and paper composition. Mustafa Gül was in charge of conceptualization, supervision, funding acquisition and paper editing.

Some of the contents in Chapter 7 have been published as ***Mei, Q., Gül, M. and Shirzad-Ghaleroudkhani, N. (2020) "Towards Smart Cities: Crowdsensing-based Monitoring of Transportation Infrastructure using Moving Vehicles." Journal of Civil Structural Health Monitoring.*** For the consistency and coherence of this thesis, contents have been modified, removed, or added from the published paper. Qipei Mei was responsible for conceptualization, methodology development, conducting experiments, analysis implementation and paper composition. Mustafa Gül was in charge of conceptualization, supervision, funding acquisition and paper editing. Nima Shirzad-Ghaleroudkhani was involved in experiments and paper composition.

Acknowledgements

First of all, I would like to express my deepest gratitude to my supervisor, Dr. Mustafa Gül, for his encouragement and guidance throughout my study at the University of Alberta. I have been working with him for more than eight years. He is the best supervisor I have ever met. This thesis could never be completed without his support and mentorship. He taught me how to do research, write papers, come up with new ideas, and present my work, etc. All of these have prepared me to be an independent researcher.

I would also like to thank my supervisory committee, Dr. Samer Adeeb and Dr. Yong Li, and my PhD defense examiners and chair, Dr. Karim El-Basyouny, Dr. Devin K. Harris and Dr. Robert Driver, for their time and great feedback. Your constructive comments have inspired me to think and dig deeper in my research.

During my journey, I have met many great professors who has taught and helped me a lot of things. Dr. Roger Cheng taught me how to be a dedicated structural engineer. Dr. Samer Adeeb and Dr. Lindsey Westover introduced me to the research in biomechanics. Dr. Daniel Aalto helped me transform a course project into a journal paper.

Also, I am grateful to the members of our structural health monitoring research group, Md Riasat Azim, Haiyang Zhang, Ngoan Do, and Nima Shirzad-Ghaleroudkhani, for providing valuable feedback and assistance during my PhD journey. Many thanks to my friends, Fei Han, Ran Ding, Jian Jiang, Jun Hu, Baokui Zhang, Zheyuan Liu, Ran Li, who are very patient to me and makes my whole life enjoyable.

And last but not least, I wish to give my thanks to my family, especially my wife, Jiarui Wang and my two lovely sons, Jasper and Joshua. It is you that motivate me to become better. It was your encouragement and support that helped me through the most difficult times.

Table of Contents

CHAPTER 1: INTRODUCTION ... 1

1.1 Motivation ... 1

1.2 Problem Statement ... 2

1.2.1 Challenges in Bridge Condition Assessment ... 2

1.2.2 Challenges in Road Condition Assessment ... 2

1.3 Crowdsensing-based Transportation Monitoring using Moving Vehicles ... 3

1.4 Objectives and Scope ... 5

1.5 Organization of the Thesis ... 6

1.6 Research Contribution ... 7

CHAPTER 2: LITERATURE REVIEW ... 9

2.1 Literature Review on Crowdsensing-based Civil Infrastructure Monitoring ... 9

2.2 Literature Review on Drive-By Bridge Health Monitoring ... 9

2.3 Literature Review on Imaged Based Pavement Crack Detection ... 12

2.3.1 Rule-based Techniques ... 12

2.3.2 Machine Learning-based Techniques ... 12

2.3.3 Deep Learning-based Techniques ... 13

CHAPTER 3: DRIVE-BY BRIDGE DAMAGE IDENTIFICATION USING PROFESSIONAL ACCELEROMETERS ... 16

3.1 Overview ... 16

3.2 Part I: Methodology based on adapted MFCCs ... 17

3.2.1 Mel-Frequency Cepstral Coefficients (MFCCs) ... 17

3.2.2 Numerical Simulation ... 23

3.2.3 Results and Analysis ... 24

3.3 Part II: An improved methodology taking advantage of a large number of vehicles ... 27

3.3.1 Mobile Sensor Network ... 27

3.3.2 Principal Component Analysis ... 27

3.3.3 Damage Detection Procedure ... 29

3.3.4 Numerical Verification ... 31

3.3.5 Distribution of Transformed Features ... 31

3.3.6 Results and Analysis ... 33

3.4 Experimental Verification ... 41

3.4.1 Experimental Setup ... 41

3.4.2 Interpretation of Results ... 44

3.5 CONCLUSIONS ... 48

CHAPTER 4: DRIVE-BY BRIDGE DAMAGE DETECTION USING SMARTPHONES ... 50

4.1 OVERVIEW ... 50

4.2 METHODOLOGY ... 51

4.2.1 Overview ... 51

4.2.2 Vehicle Bridge Interaction ... 52

4.2.3 Crowdsensing with Smartphones in Passing by Vehicles ... 53

4.2.4 Mel-frequency Cepstral Coefficients (MFCCs) ... 54

4.2.5 Kullback–Leibler (KL) Divergence ... 55

4.3 NUMERICAL ANALYSIS ... 56

4.3.1 DC1a_N and DC1b_N:15% and 30% Reduction of Stiffness at the Mid-Span ... 58

4.3.2 DC2a_N and DC2b_N:15% and 30% Reduction of Stiffness at the ¼ Span ... 59

4.3.3 DC3_N: Boundary Condition Change at Both Ends ... 60

4.3.4 Influence of Number of Vehicles ... 60

4.4 EXPERIMENTAL STUDIES ... 61

4.4.1 Data Collection App ... 61

4.4.2 Global Coordinates ... 62

4.4.3 Sampling Frequency Correction ... 62

4.4.4 Experimental Setup ... 64

4.4.5 Experimental Results ... 69

4.5 DISCUSSION ... 72

4.6 CONCLUSIONS ... 73

CHAPTER 5: DRIVE-BY BRIDGE MODE SHAPE IDENTIFICATION ... 75

5.1 OVERVIEW ... 75

5.2 METHODOLOGY ... 76

5.2.1 Assumptions ... 76

5.2.2 Mapping Moving Observations to Fixed Observations ... 76

5.2.3 Valid Data Points ... 79

5.2.4 Matrix Completion using Soft Imputing ... 80

5.2.5 System Identification ... 83

5.3 NUMERICAL VERIFICATION ... 84

5.3.1 Numerical Setup ... 84

5.3.2 Results and Analysis ... 85
5.4 Discussion ... 88
5.4.1 Influence of the Number of MMPs ... 88
5.4.2 Influence of Car Speed ... 92
5.4.3 Influence of Car Weight ... 93
5.4.4 Linear Interpolation Versus High-order Lagrange Interpolation ... 93
5.4.5 Influence of Road Roughness ... 95
5.4.6 Influence of Noise ... 97
5.5 Conclusions ... 97
CHAPTER 6: PAVEMENT CRACK DETECTION USING IN-VEHICLE CAMERAS ... 99
6.1 Overview ... 99
6.2 Experimental Setup and Data Collection ... 99
6.3 Methodology ... 105
6.3.1 Overall Procedure ... 105
6.3.2 Connectivity Maps ... 106
6.3.3 Conditional Wasserstein Generative Adversarial Network (cWGAN) ... 109
6.3.4 Evaluation ... 113
6.4 Analysis and Results ... 113
6.4.1 Pretraining on ImageNet and CFD datasets ... 113
6.4.2 Performance on EdmCrack600 dataset ... 117
6.5 Conclusions ... 125
CHAPTER 7: A PLATFORM TO MANAGE CROWDSENSING DATA FOR CIVIL INFRASTRUCTURE MONITORING ... 126
7.1 Overview ... 126
7.2 Data Acquisition System for Vibration Data in In-vehicle Smartphones ... 127
7.3 Data Acquisition System for Image Data in In-vehicle Cameras ... 127
7.4 Web-based System for Analysis and Visualization ... 128
7.4.1 Visualization of Results from Cameras ... 128
7.4.2 Visualization of Results from Smartphone Accelerometers ... 130
7.5 Conclusions ... 131
CHAPTER 8: CONCLUSIONS AND RECOMMONDATIONS FOR FUTURE RESEARCH ... 132
8.1 Summary and Conclusions ... 132
8.2 Recommendations for Future Research ... 133

REFERENCES 135

List of Figures

Figure 1.1 - Possible sensors in vehicles (modified from [32]) 4

Figure 1.2 - General procedure of the crowdsourcing based civil infrastructure monitoring 5

Figure 3.1 - The method to utilize a large number of mobile sensors for indirect health monitoring of populations of bridges 16

Figure 3.2 - Procedure of MFC 19

Figure 3.3 - Transformation between Hertz Scale and Mel Scale 20

Figure 3.4 - Transformation between Hertz Scale and Adapted Mel Scale 21

Figure 3.5 - Filter bank for frequency ranging from 0 to 50 Hz 22

Figure 3.6 - Spring mass vehicle bridge interaction model (adapted from [43]) 24

Figure 3.7 - Mesh grid of the bridge 24

Figure 3.8 - DFs for DC1-DC3 25

Figure 3.9 - DFs for DC4-DC6 26

Figure 3.10 - DFs for DC7 26

Figure 3.11 - Procedure to obtain transformed features 29

Figure 3.12 - Procedure to detect damage 30

Figure 3.13 - Pairwise plots of transformed features for baseline and validation cases ... 32

Figure 3.14 - Boxplot of transformed features for baseline and validation cases 33

Figure 3.15 - Pairwise plots of transformed features for baseline case, DC1 and DC3 ... 34

Figure 3.16 - Boxplot of transformed features for baseline case, DC1 and DC3 35

Figure 3.17 - DFs for validation case, DC1, DC2 and DC3 35

Figure 3.18 - Pairwise plots of transformed features for baseline case, DC4 and DC6 ... 36

Figure 3.19 - Boxplot of transformed features for baseline case, DC4 and DC6 37

Figure 3.20 - DFs for DC4-DC6 37

Figure 3.21 - Pairwise plots of transformed features for baseline case and DC7 38

Figure 3.22 - Boxplot of transformed features for baseline case and DC7 39

Figure 3.23 - DFs for DC7 39

Figure 3.24 - Relationship between CoV of DFs and number of vehicles 40

Figure 3.25 - Ratio over average DF of DC0 for different number of transformed features 41
Figure 3.26 - Setup of the lab experiment........ 41
Figure 3.27 - Components of the robot car........ 42
Figure 3.28 - Springs with different spring constants to simulate different suspension systems........ 43
Figure 3.29 - Damage applied to the bridge........ 44
Figure 3.30 - Average acceleration taken from two PCB 393A03 accelerometers......... 45
Figure 3.31 - Boxplot of transformed features for baseline case, validation case, DC1 and DC2........ 47
Figure 3.32 - DFs for different damage cases........ 48
Figure 4.1 - Damage detection procedure........ 52
Figure 4.2 - Spring mass vehicle bridge interaction model........ 53
Figure 4.3 - Mesh grid of the bridge........ 57
Figure 4.4 - DFs for DC1a_N and DC1b........ 59
Figure 4.5 - DFs for DC2a_N to DC2b_N........ 59
Figure 4.6 - DFs for DC3_N........ 60
Figure 4.7 - Relationship between coefficient of variation of DFs and number of vehicles 61
Figure 4.8 - Data collection app........ 62
Figure 4.9 - Time intervals versus time........ 63
Figure 4.10 - The original and corrected acceleration data........ 64
Figure 4.11 - Setup of the lab experiment........ 65
Figure 4.12 - Artificial damage applied for DC1a_E and DC2a_E........ 65
Figure 4.13 - Artificial damage applied for DC1b_E and DC2b_E........ 66
Figure 4.14 - Boundary condition change for DC3_E........ 66
Figure 4.15 - Setup of the model vehicle........ 67
Figure 4.16 - Placement of smartphone and wireless accelerometer for impact test........ 67
Figure 4.17 - Comparison of smartphone and wireless accelerometer........ 68
Figure 4.18 - DFs obtained using the wireless accelerometers........ 71
Figure 4.19 - DFs obtained using the smartphone........ 72

Figure 5.1 - Mapping moving measurement points to fixed points 76

Figure 5.2 - Illustration of the valid and invalid data points... 80

Figure 5.3 - Visualization of a typical **D** matrix 80

Figure 5.4 - Soft imputing algorithm for matrix completion 83

Figure 5.5 - Setup of the numerical analysis.. 85

Figure 5.6 - Measured acceleration and integrated velocity and displacement for 2 MMPs: (a) measured acceleration; (b) integrated velocity; (c) integrated displacement 85

Figure 5.7 - Comparison between reconstructed displacements for 4 VFPs and ground truth (2 MMPs) 87

Figure 5.8 - Errors relative to the amplitude of each VFP (2 MMPs) 87

Figure 5.9 - Mapping from 2 MMPs.. 87

Figure 5.10 - The first three normalized mode shapes identified using 2 MMPs............ 88

Figure 5.11 - Integrated displacement for 4, 6 and 8 MMPs 89

Figure 5.12 - Comparison between reconstructed displacements for 4 VFPs and ground truth (4 MMPs) 89

Figure 5.13 - Comparison between reconstructed displacements for 4 VFPs and ground truth (6 MMPs) 90

Figure 5.14 - Comparison between reconstructed displacements for 4 VFPs and ground truth (8 MMPs) 90

Figure 5.15 - Errors relative to the amplitude of each VFP... 90

Figure 5.16 - The first 3 mode shapes identified from the proposed method.................. 91

Figure 5.17 – MAC values for the first three modes at different speeds 92

Figure 5.18 - MAC values for the first three modes for different car weights 93

Figure 5.19 – Comparison between Lagrange mapping proposed in Oshima et al.[58] and ground truth.. 94

Figure 5.20 - The first three normalized mode shapes identified from matrix **D** obtained by Lagrange mapping 95

Figure 5.21 - Generated road roughness 96

Figure 5.22 - MAC values for different road profiles.. 96

Figure 5.23 - MAC values of the proposed method at different noise levels 97

Figure 6.1 - GoPro Hero 7 Black mounted at the rear of Honda Pilot 2017 100

Figure 6.2 - Comparison of two configurations (modified from [123]) 101

Figure 6.3 - Sample images from EdmCrack600 dataset 103

Figure 6.4 - Overview of the identification process 106

Figure 6.5 - Overview of the training process 106

Figure 6.6 - Issues with deconvolution layer output (a) original patch; (b) raw output; (c) after 3×3 morphological operations; (d) after 15×15 morphological operations........... 108

Figure 6.7 - An example of crack detection... 108

Figure 6.8 - Connectivity maps for crack annotation 108

Figure 6.9 - Comparison between binary mask and connectivity maps 108

Figure 6.10 - Details of the generator 111

Figure 6.11 - Details of the discriminator 111

Figure 6.12 - Losses of the proposed method 115

Figure 6.13 - Sample results for the CFD dataset 117

Figure 6.14 - Losses for EdmCrack600 dataset 119

Figure 6.15 - Sample images and corresponding results for EdmCrack600 dataset 122

Figure 6.16 - Two wrongly identified images 123

Figure 6.17 - Performance of different regions of EdmCrack600 dataset..................... 125

Figure 7.1 - Layer architecture of the crowdsensing based transportation infrastructure monitoring system.. 126

Figure 7.2 – GUI of the data acquisiton system for smartphone data............................ 127

Figure 7.3 - Screen shot of the system for road condition monitoring 129

Figure 7.4 - Adminstration webpage to add and change road data points...................... 129

Figure 7.5 – Screen shot of the system for bridge condition monitoring 130

Figure 7.6 - Adminstration webpage to add and change bridge data points.................. 131

List of Tables

Table 3.1 - Variation of car components 43

Table 4.1 - Variation of vehicle configurations 57

Table 5.1 - MAC values of the first 3 modes for 4, 6 and 8 MMPs 91

Table 6.1 - Comparison between different mounting strategies 101

Table 6.2 - Comparison among different datasets 104

Table 6.3 - Comparison of performance for different methods on the CFD dataset 116

Table 6.4 - Comparison of performance for different methods on the EdmCrack600 dataset 120

List of Abbreviations

Abbreviation	Meaning
AWS	Amazon web services
CNN	Convolutional neural network
CoV	Coefficient of variance
cGAN	Conditional Generative adversarial network
cWGAN	Conditional Wasserstein generative adversarial network
DAQ	Data acquisition
DC	Damage case
DF	Damage features
DFS	Depth first search
DFT	Discrete Fourier transform
FCN	Fully convolutional neural network
FOV	Field of view
GAN	Generative adversarial network
GUI	Graphical user interface
IRI	International roughness index
KL	Kullback-Leibler divergence
MAC	Modal assurance criterion
MFC	Mel-frequency cepstrum
MFCC	Mel-frequency cepstral coefficients
MMP	Moving measurement point
PCA	Principal component analysis
RDI	Road deterioration index
SHM	Structural health monitoring
SVD	Singular value decomposition
VFP	Virtual fixed point

AWS	Amazon web services
CNN	Convolutional neural network
CoV	Coefficient of variance
cGAN	Conditional Generative adversarial network
cWGAN	Conditional Wasserstein generative adversarial network
DAQ	Data acquisition
DC	Damage case
DF	Damage features
DFS	Depth first search
DFT	Discrete Fourier transform
FCN	Fully convolutional neural network
FOV	Field of view
GAN	Generative adversarial network
GUI	Graphical user interface
IRI	International roughness index
KL	Kullback-Leibler divergence
MAC	Modal assurance criterion
MFC	Mel-frequency cepstrum
MFCC	Mel-frequency cepstral coefficients
MMP	Moving measurement point
PCA	Principal component analysis
RDI	Road deterioration index
SHM	Structural health monitoring
SVD	Singular value decomposition
VFP	Virtual fixed point

CHAPTER 1: INTRODUCTION

1.1 Motivation

Improving the efficiency and sustainability of transportation infrastructure systems is a major undertaking in the future development of smart cities [1-3]. In this context, advanced sensing and data analytics offer unique capabilities for improving various components of transportation infrastructure systems. Currently, transportation infrastructure systems in developed countries, as key components of smart cities, are mostly outdated and are vulnerable to various risks [4-6]. According to the latest Canadian infrastructure report card [5], approximately 16% of the roads and 12% of the bridges and tunnels are identified as poor or very poor condition. As these transportation infrastructure systems age, there is an increasing demand for cost effective and efficient tools to monitor and sense the systems due to the limited budget of municipal jurisdictions.

Currently, there are two groups of common practices to monitor and sense existing transportation infrastructure systems. The first group involves pre-installing sensors for the infrastructure to continuously collect and analyze data. The second group involves occasionally dispatching engineers or technicians to the site to record the measurements and bring them back for analysis. In spite of the rapid development of these technologies, there are still challenges in terms of scalability, i.e., applying these techniques to all of the existing transportation infrastructure systems. For the first group described above, a large number of sensors (accelerometers, strain gauges, cameras, etc.) have to be pre-installed for the infrastructure. The costs for sensors, installation and maintenance are high, and it is challenging to provide stable power supply for those sensors. For the second group, the inspection intervals are usually long due to the high costs of labor and inspection equipment. These issues have to be resolved to meet the high demands in infrastructure monitoring and assessment in developed countries.

1.2 Problem Statement

1.2.1 Challenges in Bridge Condition Assessment

Many studies show that any disruption in the performance of bridges, which are key components of developed transportation networks, could result in irreversible economic and social losses [7-10]. This vulnerability is, in fact, more acute in many developed countries, because a substantial portion of their bridge stock has reached their design lives, and their serviceability relies on a regular monitoring and maintenance operations.

In current practice, the condition assessment of bridges is usually implemented by either installing sensors onsite for long term monitoring or dispatch engineers to the site for occasional inspection. For these practices, the work has to be conducted by professionals, which requires professional equipment and engineers and/or technicians with advanced knowledge about structural design and sensing technologies. The installation or operations of professional devices are costly and time consuming. Therefore, the scalability of such technologies remains an open challenge. In fact, most of the bridges in developed countries cannot be assessed in a timely manner [11].

1.2.2 Challenges in Road Condition Assessment

The cracks on road surfaces are early signs for potential damage in the pavements and in the supporting structures [12, 13]. They serve as a good indicator to assess the current condition of the transportation infrastructure [14]. Defects in road surfaces may delay traffic and even cause safety issues if they are severe [15, 16]. In addition, our road infrastructure must be improved significantly to support the autonomous vehicles of the future in the scope of smart cities [1-3].

The common practice in road surface survey is based on manual inspection which has limitations like high costs and low efficiency, and therefore such cracks may be present for a considerable amount of time before they are repaired [17, 18]. In this context, the automation of crack detection on road surface is invaluable and a vast amount of research has been conducted in this field [19-24].

Efforts have been made to apply professional line scan cameras or laser scanners for automated pavement crack detection [25-27]. However, not only these professional sensors are expensive, but they also need to be mounted on specialized vehicles [28]. As a result, the inspection intervals could be long due to the limit of budget. As an example, the Ohio Department of Transportation spent 1,179,000 USD on purchasing such a specialized vehicle with laser sensors and the yearly maintenance fee for this vehicle is 70,000 USD [29, 30].

1.3 Crowdsensing-based Transportation Monitoring using Moving Vehicles

To overcome the abovementioned issues, this thesis proposes a novel framework for crowdsensing based civil infrastructure monitoring which uses data collected from smart devices owned by general public for civil infrastructure monitoring. Nowadays, smart devices are widely used in people's daily lives. Compared with using traditional professional sensors, there are a number of advantages of getting general public involved into the civil infrastructure monitoring. First, sensors in smart devices are more accessible than professional sensors. Second, the operating systems of smart devices are usually more sophisticated than professional sensor platforms, and therefore data collection, storage and transmission can be easily implemented. Third, various sensors are naturally integrated in such devices, so fusion of information from different sensors are easier than professional sensors. The goal of this framework is to develop a set of pre-screening tools to determine which infrastructures are at more critical condition prior to conducting the site inspection and installing fixed sensors. With the participation of general public, smart devices can provide valuable and continuous information about the civil infrastructure with reduced costs and higher efficiency. The framework can complement current practices for infrastructure monitoring and maintenance, and also can benefit the development of sustainable infrastructure systems.

This thesis focuses on a specific application under the crowdsensing based framework, i.e., using crowdsensing data from smart devices in moving vehicles for transportation infrastructure monitoring. In recent years, a number of researchers have proposed to use vehicles that are equipped with different kinds of sensors to monitor the transportation infrastructure systems [31, 32] with the involvement of general public to increase efficiency and reduce the costs for the authorities. Owing to the high mobility of vehicles, they can efficiently monitor the health condition of a population of the existing infrastructure. Figure 1.1 gives a list of widely used sensors in vehicles. This thesis will focus on two of these sensors, i.e. accelerometer and camera, to monitor bridges and roads.

The general procedure of transportation infrastructure monitoring using moving vehicles is presented in Figure 1.2. The sensors equipped in smart devices in vehicles can collect data while they pass across the infrastructure. Then, the data are transmitted to the cloud and remote computers for further processing, and the extracted information is provided to the decision makers to support their decision making. In this strategy, there is no need to install sensors locally, and the condition of a population of existing infrastructure can be monitored efficiently with the involvement of a large number of vehicles.

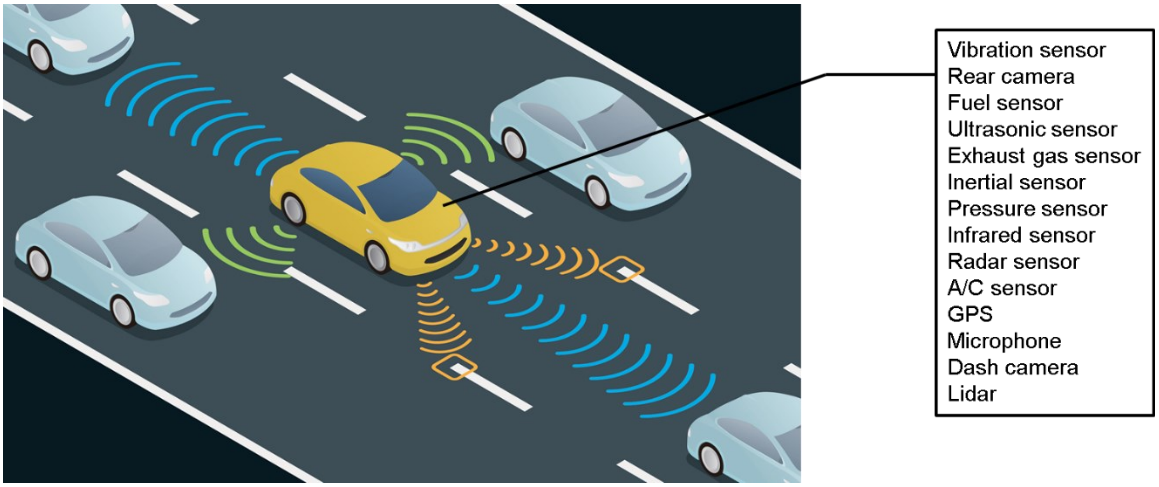


Figure 1.1 - Possible sensors in vehicles (modified from [33])

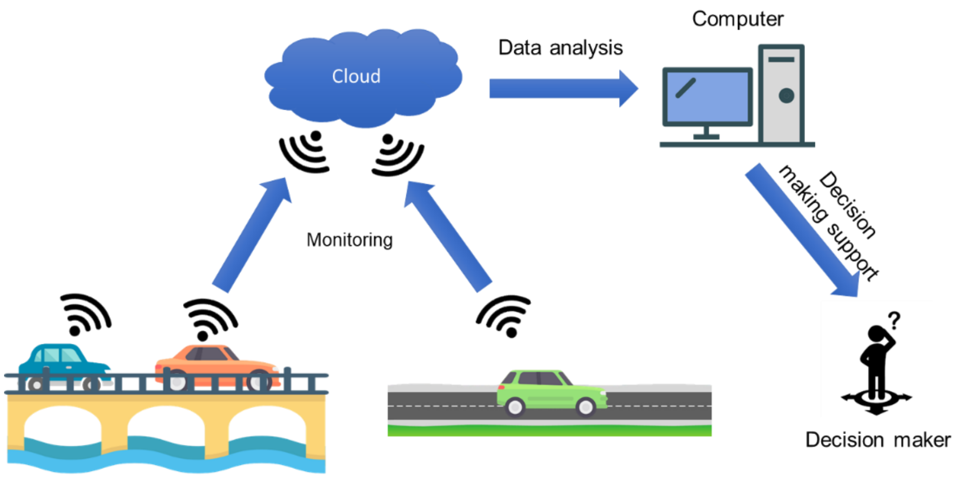


Figure 1.2 - General procedure of the crowdsourcing based civil infrastructure monitoring

There are several advantages of the proposed crowdsensing-based civil infrastructure monitoring framework: First, it can significantly reduce the cost for monitoring due to the use of commercial grade devices and voluntarily involvement of general public. Second, it has a potential to monitor a population of civil infrastructure systems in real time. Third, owing to the big data, the technology is robust to operational effects. Fourth, the technology can be fully automated after the system is established.

1.4 Objectives and Scope

In order to overcome scalability and cost-efficiency issues in existing monitoring and sensing technologies, a framework of crowdsensing based transportation infrastructure monitoring is developed to complement current practices for infrastructure monitoring and maintenance. Under this framework, two specific applications using two sensors from smart devices in vehicles, i.e., accelerometer and camera, are investigated. Briefly, two objectives of this thesis can be summarized as below:

1) Develop methods that use vibration data in multiple moving vehicles for bridge damage detection and mode shape identification. The potential challenges related to these methods are also investigated in this context.

2) Develop a method that can process images taken from in-vehicle cameras to automatically identify cracks in pavements. The potential challenges related to this method are also investigated in this context.

1.5 Organization of the Thesis

The organization of the thesis is as follows.

Chapter 2 presents literature reviews on three topics regarding this thesis. First, crowdsensing based civil infrastructure monitoring methodologies are reviewed. Second, the methodologies for drive-by bridge health monitoring are reviewed. Third, recent progresses regarding image-based pavement crack detection are reviewed. The contribution of this thesis is described in this chapter as well.

Chapter 3 presents a novel methodology for damage detection in bridges from moving vehicles. Mel-frequency analysis and principal component analysis are introduced to extract features and compare the distributions to identify damage. Numerical analysis and laboratory experiments are conducted in this chapter to verify the single-vehicle and multi-vehicle versions of this method.

Chapter 4 presents an improved version of the methodology introduced in Chapter 3 for bridge damage detection using smartphones in vehicles. Kullback–Leibler is introduced in this chapter for the comparison of features distribution. Smartphone app and laboratory experiments using smartphones are presented to verify this method.

Chapter 5 presents a novel methodology for mode shape identification using moving vehicles. In this study, the vibration data from the moving measurement point (MMP) to virtual fixed points (VFP) using linear interpolation in order to convert the problem into a matrix completion process. Then, a method called “soft-imputing” is implemented to fill the matrix. At last, singular value decomposition (SVD) is applied for system identification. Numerical analysis is conducted to verify this method.

Chapter 6 presents a cost-effective solution for road crack detection using in-vehicle cameras. A deep learning algorithm with novel architecture and loss function is developed and verified on the dataset collected from the in-vehicle cameras.

Chapter 7 introduces a platform to efficiently manage and analyze the crowdsensing data collected from the moving vehicles. In this platform, the bridge condition assessment and road crack detection results are fused with GPS data which can support the decision making of the management of existing transportation infrastructure at city level.

Chapter 8 includes the summary and conclusions of this thesis. Recommendations and possible future directions are also presented.

1.6 Research Contribution

The main contribution of this thesis is that it presents a novel framework to use crowdsensing data for civil infrastructure monitoring. Under this framework, methodologies are developed that can utilize sensors from smart devices in moving vehicles for transportation infrastructure monitoring. The detailed contributions of each chapter are summarized as below:

In chapter 3, a novel methodology is developed to assess bridge condition from multiple moving vehicles. In this study, the Mel-frequency cepstral analysis is introduced to bridge health monitoring to overcome the limitations of previous methods that are based on peak finding of frequency spectra. In addition, a large number of vehicles instead of a single vehicle are used in this method, and therefore uncertainty from vehicle configurations can be significantly reduced.

In chapter 4, instead of professional sensors, smartphones are introduced in drive-by bridge health monitoring. Android smartphone app for data collection is developed. The data collected from a large number of smartphones can be used to reduce operational and environmental effects. In this study, attempts are made to identify damage in the bridge using smartphones in vehicles with varying properties among different tests. This opens a promising path to real-life crowdsensing applications.

In chapter 5, a novel method for mode shape identification using moving vehicles is developed. This method can scan the bridge and find its mode shapes using as few as 2 mobile sensors, which is more efficient and cost-effective than the methods using fixed sensors. In addition, this method is pure data-driven, so no properties regarding the bridges need to be known as prior knowledge. Compared with other studies for drive-by mode shape identification, this method converts the mode shape identification problem into a matrix completion problem, so the resolution of the method is not limited to Nyquist sampling theorem and the method can work at traffic speed.

In chapter 6, a cost-effective solution is proposed for pavement crack detection using in-vehicle camera. A new deep learning algorithm is developed for automatic crack detection. Compared with other studies, densely connected layers are introduced to better reserve the detailed information of the cracks. In addition, connectivity maps are used to assign different weights to different pixels to overcome scattered output of the cracks. The performance of the proposed method is compared with other state-of-the-art methods.

Chapter 7 presents a novel platform that has the potential to manage and analyze the crowdsensing data from users. The platform can be used to support decision making for infrastructure management.

CHAPTER 2: LITERATURE REVIEW

2.1 Literature Review on Crowdsensing-based Civil Infrastructure Monitoring

As described in the previous chapter, crowdsourcing has become increasingly popular in recent years [34-39]. Crowdsourcing, by definition, is a sourcing model to solve problems with the help of a large number of participants [40]. Specifically, the terminology crowdsensing is used while sensing the environment and infrastructure using with the participation of a large group of people.

There have been studies that use smartphones in a crowdsensing way for civil infrastructure monitoring. Ozer et al. [41] proposed to use sensor networks formed by multiple smartphones as a crowdsensing platform for modal analysis. Two sensors in smartphones, i.e., camera and accelerometer, were used synchronized in a hybrid mode. It was shown in their paper that the data collected from camera and accelerometer were fused successfully on a multi-story laboratory model for frequency and mode shape identification. Zeng et al. [42] used Android tablet to collect acceleration and to estimate the road roughness profile. The road tests in Virginia showed that the collected datasets were highly correlated with International Roughness Index (IRI) collected using profiler vans. Matarazzo et al. [32] applied smartphones in moving vehicles on a bridge in Boston, and proved that the collected data contained consistent and significant indictors of the first 3 modes of the bridge. Maeda et al. [43] utilized smartphone as dash camera in vehicles and collected images for road condition assessment. Using deep learning algorithms, the road conditions were successfully classified into eight categories.

2.2 Literature Review on Drive-By Bridge Health Monitoring

In recent years, there have been studies about bridge health monitoring using sensors in moving vehicles. This research area is usually referred to as indirect bridge health monitoring. The first paper that utilized this concept to identify bridge information from a passing-by vehicle was written by Yang et al. [44]. In their paper, the equation of motion integrating dynamic properties of both vehicle and bridge was first studied, and then the acceleration of passing-by vehicle was represented as a function of dynamic characteristics of the bridge. After applying Fourier transform to the vehicle acceleration, the frequencies of the bridge can be identified. Afterwards, the indirect health monitoring became an active research field [45-54]. Bu et al. [55] proposed an approach based on dynamic response sensitivity analysis using acceleration measurement on the vehicle to identify the damage in the bridge. In their paper, damage, in terms of flexural stiffness reduction, is exposed in an iterative procedure using both 3-parameter and 5-parameter vehicle models. In their numerical analysis, they showed that their approach can detect the damage when measurement noise and road surface roughness exist. Keenahan et al. [49] utilized an instrumented truck-trailer vehicle to monitor the change of damping in a bridge. In their method, the influence of road profile roughness is removed by subtracting one axle acceleration from another, and the change of bridge properties are assessed. Mcgetrick et al. [56] implemented an experimental validation of a novel passing-by global stiffness identification method. The method proposed in their paper consisted of 6 steps. They first calculated contact force, total displacement and estimated road profile using only acceleration data from the vehicle, and then identified global stiffness of the bridge using those variables. In their paper, they found that the method can identify the stiffness correctly for 8 of 9 vehicle-speed combinations, which proves that their method is insensitive to the speed of the vehicle. Sitton et al. [57] applied discrete Fourier transform and multiple signal classification to estimate bridge frequencies from smartphone acceleration data. In their paper, the user interaction with the smartphone was studied.

The mode shapes of the bridge are important if the location of the damage is of interest [58]. There are a couple of past studies that focused on the mode shape identification from moving vehicles. Oshima et al. [59] proposed a method to detect support damages based on the mode shapes identified after mapping moving sensor data to fixed sensors. In another study, Malekjafarian and O’Brien [60] presented an algorithm for bridge damage detection based on the mode shapes calculated through applying Short Time Frequency Domain Decomposition method. Later on [61], they improved their method using laser vibrometers and accelerometers to achieve high resolution mode shapes. Empirical Mode Decomposition, as another mode shape detection method, was investigated in another study

[52]. Matarazzo et al. [62] introduced a method called structural identification using expectation maximization for mode shape identification from mobile sensors. Their method consists of two steps, where the first step (expectation step) is updating the missing observation due to mobile sensing, and the second step (maximization step) is used to update the parameters of the state-space equations of the system. Furthermore, Hilbert Transform was shown as an effective signal processing technique in identifying mode shapes of the bridge [63, 64]. Yang et al. [64] found that the instantaneous amplitude of the vehicle includes mode shapes information and developed a method based on Hilbert transform to reconstruct mode shapes from the measured response of the moving vehicle. In that study, they identified the vehicle speed, random traffic and road roughness as important factors in the accuracy of the method. Malekjafarian and Obrien [63] related the energy calculated from Hilbert Huang Transform to the mode shapes. Using the data collected from two adjacent axles of a vehicle and empirical mode decomposition, the first mode shape was reconstructed. One limitation of the above two studies was that engineering judgement had to be applied to determine the signs of the mode shapes. Eshkevari et al. [65-67] treated the mobile sensing data as a sparse matrix with missing values. In their work, alternating least-square was used to complete the matrix. Then, system identification techniques such as principal component analysis (PCA) and structured optimization analysis are used to identify modal properties.

More research about the indirect health monitoring can be found in a comprehensive review written by Malekjafarian et al. [58]. Although promising progresses have been achieved in the area of indirect bridge health monitoring using moving vehicles, most of the studies have focused on the health monitoring through a single vehicle which is sensitive to the operational and environmental effects and is limited by the short vehicle-bridge interaction time [58]. Among these studies, damage can only be detected when the car configurations (natural frequency, speed, etc.) are known or the same car is used for both baseline and damaged cases. In reality, it is difficult to measure the properties of the car or keep the vehicle's properties the same for every test. Besides, it is costly to have a specifically designed vehicle only for test purpose, and it cannot assess the condition of bridges in real time since due to the large intervals between different tests.

2.3 Literature Review on Imaged Based Pavement Crack Detection

2.3.1 Rule-based Techniques

In general, there are three major paths for crack detection utilizing images, rule based, machine learning based and deep learning-based methods. In rule-based methods, different combinations of filters and image processing techniques are applied to identify the cracks in images.

Gavilán et al. [68] proposed an approach combining a series of image processing techniques. First, the image was preprocessed to enhance the linear features, and non-crack feature detection was conducted to eliminate confusing area like joints or filled cracks on pavements. Then, a seed-based approach combining multiple directional non-minimum suppression with symmetry check was proposed. Zou et al. [69] developed a three step method called CrackTree. In their method, the shadow was first removed using a geodesic based algorithm. Then, a probability map was created based on tensor voting. Finally, recursive tree-edge pruning was conducted on the minimum spanning tree generated on the probability map to identify cracks. Amhaz et al. [70] introduced an improved minimal path selection algorithms with a refined artifact filtering step so that the thickness of the crack pattern can be estimated. Their approach showed superior performance than another 5 existing methods in their paper.

Overall, the major advantage of these methods is that neither annotation nor training process is required, so it is easy to implement the methods and verify the performance. Also, these methods are computationally efficient. The biggest disadvantage of this kind of methods is that most of the features are handcrafted on some given datasets. In general, they cannot consider all the variation in real life images such as illumination changes or irregular shape of cracks. Therefore, the methods that are developed in a controlled environment cannot work well when the situation changes.

2.3.2 Machine Learning-based Techniques

Realizing the complexity in texture of pavement surfaces, variation in the illumination and the irregularity in shapes of the cracks, researchers tend to seek machine learning based algorithms for crack detection starting from the last decade [71]. Comparing with traditional rule-based techniques, machine learning based algorithms can implicitly consider a variety of the factors that could affect the appearance of cracks in the training process, but usually require more computational resources.

Hu et al. [72] treated the pavement as texture surface and cracks as inhomogeneity, and used texture analysis and shape descriptors to extract features. Support vector machines were used to classify whether a sub-region was crack or non-crack. Mathavan et al. [73] applied an unsupervised learning algorithm called self-organizing map to the crack images. Texture and color properties were integrated within the self-organizing map to distinguish cracks from background. Shi et al. [71] proposed a crack detection method based on random structured forests. In their method, integral channel features were introduced to learn the crack tokens with structured information. Then, random structured forest was applied to process the tokens and find the cracks.

Although machine learning-based methods can achieve better accuracy than rule-based methods due to the training process, the input to these methods are still features handcrafted according to the researchers' knowledge. The fact that prior knowledge must be applied has limited the performance of these crack detection methods.

2.3.3 Deep Learning-based Techniques

Deep learning, as a branch of machine learning, has drawn much attention in last few years due to its superior performance in object detection and semantic segmentation [74, 75]. They were first time applied to crack detection task in 2016 [76]. In general, deep learning-based crack detection methods can be categorized into two groups, i.e., region based and pixel-based methods.

The region-based method is less computationally intensive and has been studied by a number of researchers. Cha et al. [77] developed a convolutional neural network (CNN) and applied it to 40,000 regions with a resolution of 256×256 pixels for training. The algorithm can detect cracks by classifying each region separately. Gopalakrishnan et al. [78] utilized a per-trained deep CNN model and applied transfer learning to hot-mix asphalt and Portland cement concrete pavement images. Their algorithm can identify whether an image has crack or not in it. Alipour and Harris [79] studied the generalization of deep learning models across different materials. They noticed that the change of materials could adversely affect the performance of models. To resolve this issue, they proposed three domain adaption techniques for training, and discussed the advantages and disadvantages of each technique. Hoang et al. [80] compared a CNN model with metaheuristic optimized edge detection algorithm. They showed that the performance of CNN was significantly better than edge detector.

However, the region-based methods can only provide information about the existence of cracks and rough shape and location depending on the size of regions. The value of crack detection decreases if the accurate pattern and location of the cracks cannot be given. To resolve this issue, pixel-level crack detection methods are studied. Ni et al. [81] developed a method comprising two deep neural networks. The first neural network was called GoogLeNet which served as a feature extractor. Then, a second neural network including bilinear deconvolution layer and eltwise operation layer were used for pixel-level crack detection. Fei et al. [21] designed a deep neural network consisting of a preprocess layer, eight convolutional layers, and one output layer. With invariant spatial size through all layers, the method can achieve pixel level crack detection. Yang et al. [82] utilized a fully convolutional neural network (FCN) to realize the pixel level detection. Through the encoder and decoder process, the output was guaranteed to be the same size as input. Therefore, the prediction was included in the output probability map. Alipour et al. [83] also used FCN to identify cracks in concrete. They showed that the performance of the method was better than patchwise models and traditional image processing techniques. Zhang [84] developed a context-aware deep semantic segmentation network for crack detection. Their method consisted of three steps. First, an adaptive sliding-window method was used for patch proposal. Second, an encoder-decoder deep neural network was applied to the patches to generate binary masks for cracks. Third, the masks were integrated to the size of the original image using contextual information through cross-state and cross-space potential functions. Liu et al. [85] introduced a neural network called U-Net for crack detection in concrete. Focal loss and Adam optimizer were used for training.

There have been some studies that applied deep learning algorithms on images collected from cameras in moving vehicles for road crack detection. Maeda et al. [43] used smartphone in a vehicle to collect road images in Japan, and deployed a deep neural network on it for road defect detection. However, their study focused on multiple defect types. Bounding boxes were drawn around the defects, but no detailed information could be provided regarding the exact location, shape or orientation of the cracks. Bang et al. [30] collected images from black-box cameras in vehicles and applied an encoder-decoder deep neural network for pixel-level crack detection. These studies have shown great potential in solving crack detection problems on pavement surface using moving vehicles. However, there are still remaining challenges due to various issues such as inhomogeneity of cracks, complexity of illumination conditions, and connectivity of identified cracks.

CHAPTER 3: DRIVE-BY BRIDGE DAMAGE IDENTIFICATION USING PROFESSIONAL ACCELEROMETERS

3.1 Overview

To overcome the issues described in Chapter 1, a methodology is proposed to investigate the potential of employing a large number of mobile sensors for indirect monitoring of population of bridges. As shown in Figure 3.1, in this methodology, data collected from a large number of vehicles passing across a bridge within a certain period are used systematically for damage detection. Assuming two cases of the bridge, baseline and unknown cases, the acceleration data from vehicles passing across the bridge within two different periods are collected, and the designed features are extracted from the data before transmitting to the remote computer. Then, the distributions of the extracted features for data collected from two periods are estimated and compared. It should be noted that for the proposed method, types of vehicles during different periods may be different but they are expected to follow the similar distributions. Since the influence of the types of vehicles is eliminated from the data from a large number of vehicles using proposed data analysis method, the difference in distributions of extracted features is deemed to be related to the damage in the bridge. Using this method, populations of bridges can be monitored continuously and simultaneously with the help of a large number of mobile sensors.

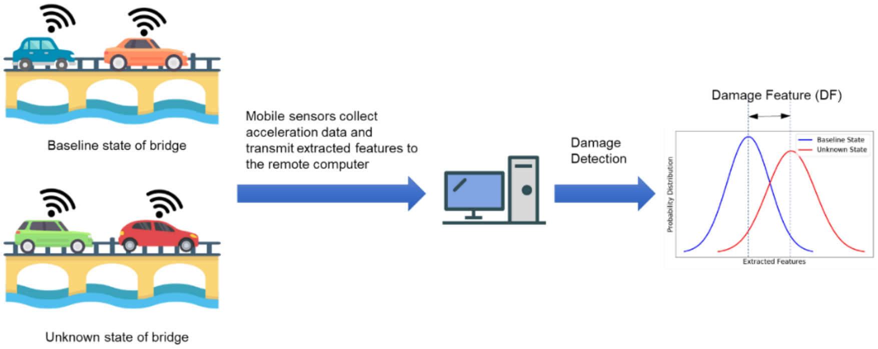


Figure 3.1 - The method to utilize a large number of mobile sensors for indirect health monitoring of populations of bridges

Within this method, features are first extracted using Mel-frequency cepstrum (MFC) and Principal Component Analysis (PCA) is developed in this chapter. The distributions of transformed features are then compared for damage identification. This chapter consists of two parts.

In the first part, a single-vehicle version of the method based on MFC only is introduced to study the feasibility of using MFC for damage detection using indirect monitoring. Cepstrum is obtained by taking the inverse Fourier transform of the logarithm of spectrum from a signal [86]. It was first developed to study seismic echoes from earthquake and explosions. Currently, the power cepstrum are mainly used to extract characteristics of voice signal. Among various cepstrum-based methods for voice signal processing, MFC is one of the most popular ones. It has been proven as useful in the area of speech recognition [87-89] due to its efficiency and compactness. The main advantages of MFC are that it scans a range of frequencies instead of peak frequencies, and it applies Mel scale instead of Hertz scale to the spectrum to mimic the human auditory system's response. The features extracted by MFC is called Mel-frequency cepstral coefficients (MFCCs). In the first part, the Euclidean distance of the MFCCs are used as damage features (DFs).

In the second part, this method is extended to multi-vehicle version with the help of PCA so that data from a large number of vehicles can be used systematically for damage detection. In the improved method, for baseline and unknown cases, the acceleration data of every single vehicle passing through the bridge are collected and processed using MFC. Afterwards, the extracted features are transformed and uncorrelated by PCA. Similar to single-vehicle version, the Euclidean distance of the transformed features are used for damage detection.

3.2 Part I: Methodology based on adapted MFCCs

3.2.1 Mel-Frequency Cepstral Coefficients (MFCCs)

In SHM, there are only a few research studies about applying cepstrum for damage detection in recent years, and all of them are applied to non-destructive evaluation or traditional health monitoring using sensors installed on the bridges. In order to detect damage in layered carbon fiber reinforced polymer, Bochud et al. [90] designed a classification system according to cepstral distance to extract features, and applied them in an analysis-by-synthesis scheme for damage detection. Dackermann et al. [91] used cepstrum analysis to generate frequency response functions based on response-only measurements. Then, the difference between baseline frequency response functions and damaged frequency response functions were calculated and passed to PCA to compress the data. At last, neural networks were trained for damage detection. In their paper, the approach was verified both numerically and experimentally.

MFC is a special kind of cepstrum analysis, which has also attracted the interests from researchers of structural health monitoring. Considering that the performance of traditional delamination detection methods is easily affected by environmental noise and subjectivity of the inspector, Zhang et al. [92] introduced independent component analysis to cancel noise, and used MFCCs to remove subjectivity. Then, a radial basis function network was selected for following delamination detection. The experiments and field tests showed that the proposed method had better robustness to noise and still worked even with limited training data. Balsamo et al. [93] proposed an adaption of MFCCs as damage features. Then, the Mahalanobis distance is calculated on those damage features for damage detection. They tested their method first on a numerical 10-DOF shear type system and then a lab structure that behaves nonlinearly. For comparison, damage detection based on auto-regressive coefficients were also conducted as a reference. They concluded that the method based on MFCCs works better than the one based on auto-regressive coefficients. MFCCs has gained more attention recently in the area of SHM, but to the best of the authors' knowledge, there is still no research applying MFCCs to indirect health monitoring for damage detection.

The reason why cepstrum analysis is used for our proposed method is that cepstrum analysis scans and extract information for a range of frequencies instead of just looking for peaks. However, traditional cepstrum analysis assigns equal weights on different frequency ranges while MFC analysis assigns more weights on lower frequency, which is a very important property and more appropriate for bridge monitoring. MFC is originally designed to mimic how human beings respond with their auditory system. For instance, in human auditory system, the conceptual distance between 100 Hz and 200 Hz is much more significant than between 10000 Hz and 10100 Hz, even though their linear distances are the same. The situation is similar in bridge frequency domain, the difference in lower frequency range that covers most of the significant modes is usually more significant than the difference in higher frequency ranges. The procedure of MFC is summarized in Figure 3.2.

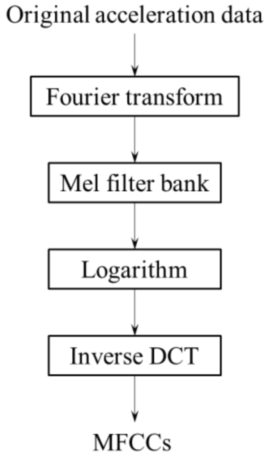


Figure 3.2 - Procedure of MFC

Although MFC is popular for many applications, there is no single Mel-scale formula for the transformation. One of the most widely used formula to transform between linear frequency, *f*, and Mel-scale frequency, *m*, is presented in Eq. (3.1) and Eq. (3.2) [94]. The transformation using this formula is presented in Figure 3.3.

$$m = M(f) = 1127\ln(1+\frac{f}{700}) \tag{3.1}$$

$$f = M^{-1}(m) = 700(e^{m/1127} - 1) \tag{3.2}$$

where *f* is the Hertz-scale frequency and *m* is the Mel-scale frequency.

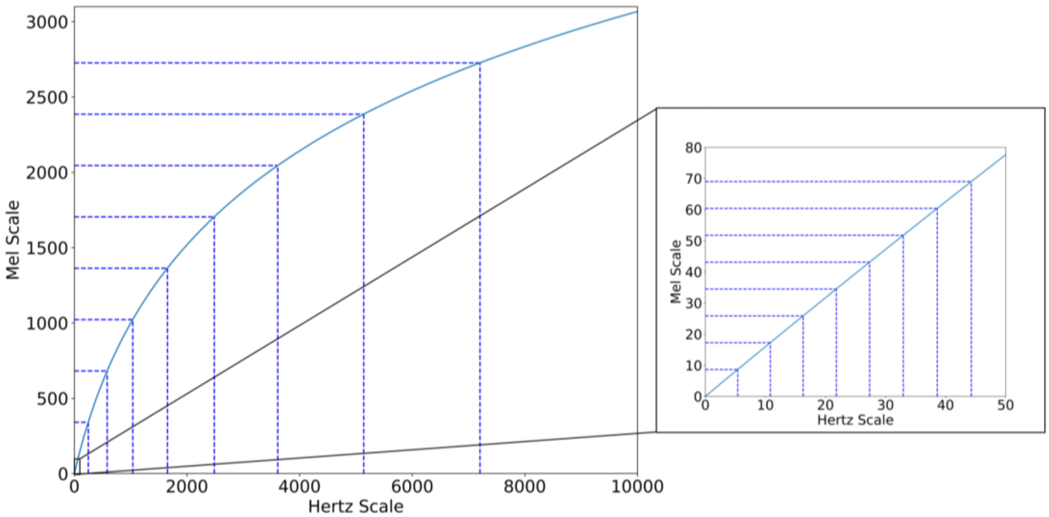


Figure 3.3 - Transformation between Hertz Scale and Mel Scale

However, the traditional Mel-scale was developed to mimic human auditory system which is most sensitive to frequencies between 2000 and 5000 Hz. This is much higher than the range of natural frequencies in bridges (0-100 Hz). Using this formula, the low frequencies are still linear after the transformation (as shown in Figure 3.3).

Therefore, the original Mel scale that was designed for speech signal is not applicable to bridges. Such conclusion was also given in [93]. Considering that lower frequency of bridges is always more important, we propose an adapted formula for the transformation to mimic the trending of Mel-scale in Eq. (3.3) and Eq. (3.4). The relationship between adapted Mel scale and Hertz scale is shown in Figure 3.4.

$$m = M(f) = 5\ln(1+\frac{f}{5}) \tag{3.3}$$

$$f = M^{-1}(m) = 5(e^{m/5} - 1) \tag{3.4}$$

where *f* is the Hertz-scale frequency and *m* is the Mel-scale frequency.

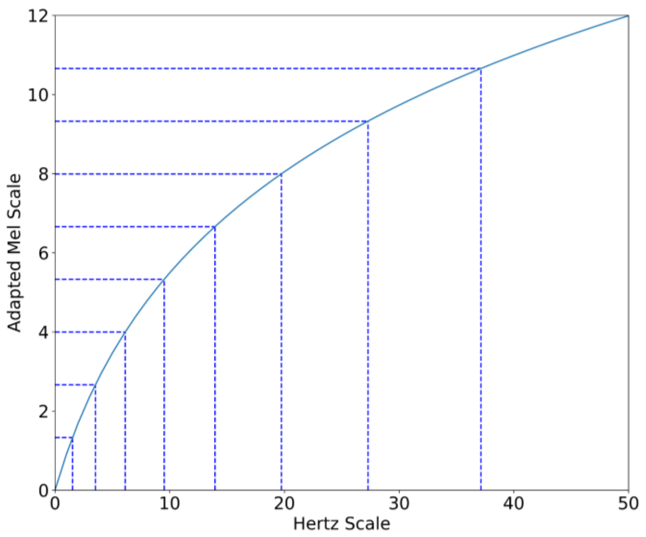


Figure 3.4 - Transformation between Hertz Scale and Adapted Mel Scale

Adapted Mel-scale frequency can be linearly separated into *n* segments. The *i*th Hertz-scale frequency corresponding to *i*th adapted Mel-scale frequency is presented in Eq. (3.5). Generally, larger *n* leads to smaller triangles in following analysis, which focuses more on details of the power spectrum. However, if *n* is too large, the obtained MFCCs may not capture the generalization of the spectra and result in poor performance. It is seen in Figure 3.4 that linearly separated adapted Mel-scale frequencies correspond to in Hertz-scale frequencies with exponential growing intervals.

$$f_i = M^{-1}(M(f_1) + (i-1) \times \frac{M(f_n) - M(f_1)}{n-1}) \tag{3.5}$$

A set of triangular filters called filter bank are applied to the spectrum at each of the Mel-frequency. In speech recognition, the number of triangular filters is usually between 20 and 40. The shape of the filter bank is demonstrated in Figure 3.5. The function of the filter bank is represented in Eq. (3.6):

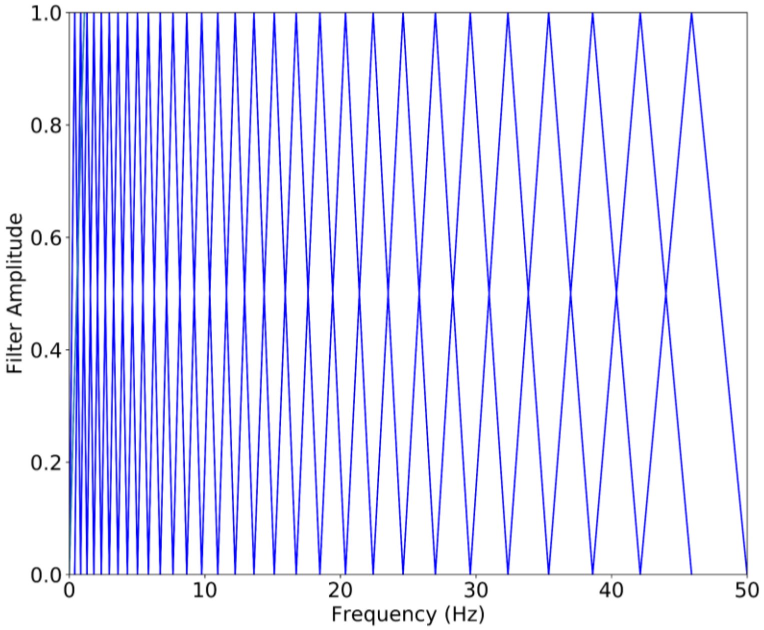


Figure 3.5 - Filter bank for frequency ranging from 0 to 50 Hz

$$H_i(k)=\begin{cases} 0 & k<f_{i-1} \\ \dfrac{k-f_{i-1}}{f_i-f_{i-1}} & f_{i-1}\le k<f_i \\ \dfrac{f_{i+1}-k}{f_{i+1}-f_i} & f_i\le k<f_{i+1} \\ 0 & k\ge f_{i+1} \end{cases},i=2,3,\cdots,n\text{-}1 \tag{3.6}$$

where $H_i(k)$ is the i^{th} filter bank that is applied to the power spectrum.

Since both power spectrum and filter are represented in frequency domain, applying filter to the original signal is to simply multiply the power spectrum by the filter. Then, the summation of the energy after applying all the filters is presented below:

$$E(i)=\sum_k H_i(k)\times F(k),i=2,3,\cdots,n\text{-}1 \tag{3.7}$$

where $F(k)$ is the Discrete Fourier Transform (DFT) of the original acceleration signal. Then, the logarithms of the powers are taken at each of the Mel frequencies:

$$\log E(i) = \log(\sum_k H_i(k) \times F(k)), i = 2,3,\cdots,n\text{-}1 \tag{3.8}$$

The last step of the cepstrum analysis is to take the Discrete Cosine Transform (DCT) of the logged powers as they are signals.

$$X_j = \sum_{i=2}^{n-1} \log E(i) \cos\left[\frac{\pi}{n\text{-}2}\left(i-\frac{3}{2}\right)j\right], j = 1,2,3,...,n-2 \tag{3.9}$$

where X_j is denoted as Mel-frequency cepstral coefficients or MFCCs. Usually, not all MFCCs are used as inputs for further analysis.

For damage detection, the Euclidean distance of the MFCCs are used as damage features (DFs):

$$\mathrm{DF} = \sqrt{\sum_{\mathrm{j}=n_1}^{n_2} (X_{j0} - X_{jk})^2} \tag{3.10}$$

in which X_{j0} stands for j^{th} MFCC for baseline case and X_{jk} stands for j^{th} MFCC for unknown case k (e.g. damaged case). Only MFCCs ranging from n_1 and n_2 are utilized in the DF calculation. According to our analysis, the first four coefficients are more related to the global vibration of the damage and not very sensitive to the damage, and therefore features 4-9 (6 features) are utilized for DF calculation.

3.2.2 Numerical Simulation

In order to verify the above-mentioned approach, first numerical analysis for a single span beam-type bridge is conducted in Abaqus. The bridge is made of steel, and is 2000 mm long, 304 mm wide and 12.7 mm thick. As shown in Figure 3.6, a spring mass model is introduced to simulate the vehicle passing over the bridge. The bridge itself is meshed into 16 elements as presented in Figure 3.7. The mass of the vehicle is set to 3 kg, and the spring constant of the spring is 266.7 N/m. Using the configuration, the natural frequency of the vehicle is 1.5 Hz. The speed of the car is set to 1 m/s.

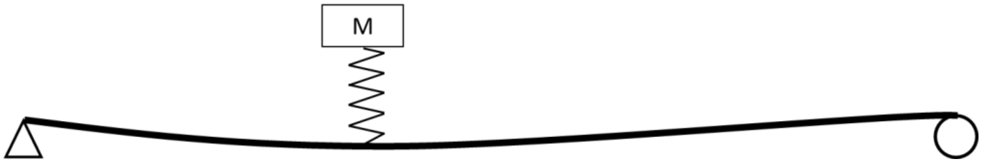


Figure 3.6 - Spring mass vehicle bridge interaction model (adapted from [44])

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Figure 3.7 - Mesh grid of the bridge

Except baseline case, 7 damage cases are introduced. DC1 to DC3 are related to stiffness reduction of elements 7 and 8 (mid-span), DC4 to DC6 are created by changing stiffness of elements 4 and 5 (1/4 span), and DC7 is about boundary condition changes at both ends. The stiffness changes for DC1-DC6 are simulated by reducing elastic modulus. To simulate the measurement error, all damage cases are corrupted with 5% artificial noise. Except baseline case and 7 damage cases, a validation case is introduced with the same setup as baseline but different noise and referred as DC0. The validation case and eight damage cases are summarized as below:

1) DC0: No damage (validation case)
2) DC1: 10% reduction of stiffness at mid-span
3) DC2: 20% reduction of stiffness at mid-span
4) DC3: 30% reduction of stiffness at mid-span
5) DC4: 10% reduction of stiffness at ¼ span
6) DC5: 20% reduction of stiffness at ¼ span
7) DC6: 30% reduction of stiffness at ¼ span
8) DC7: Both ends are changed to fixed supports.

3.2.3 Results and Analysis

3.2.3.1 DC1-DC3: stiffness reduction at mid-span

For DC1 to DC3, the numerical simulation for each damage case is repeated 20 times with noise added. From Figure 3.8, it is seen that the existence of stiffness reduction can be identified accurately with DFs derived from MFCCs. The DFs from validation case are close to zero even though 5% noise is added to the data. 10% stiffness reduction at mid-span leads to DFs of around 2. As the damage becomes more severe, the magnitude of the DFs increases accordingly. For different trials, the DFs fluctuate within an acceptable range, which means the DFs can distinguish real damage from noise.

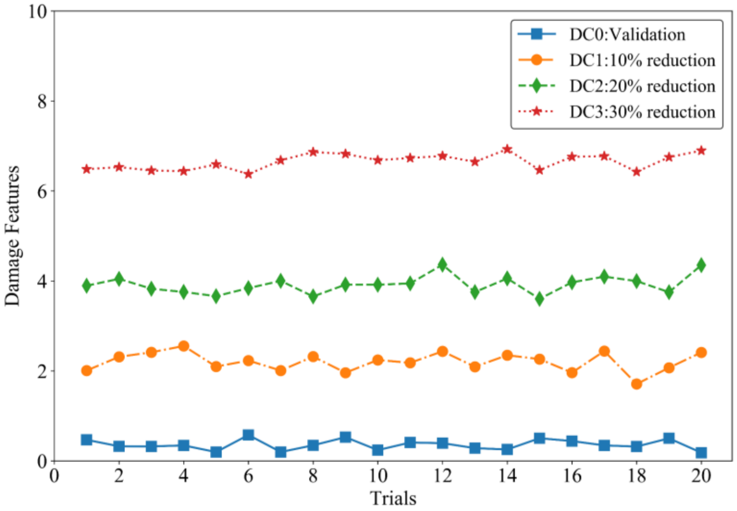


Figure 3.8 - DFs for DC1-DC3

3.2.3.2 DC4-DC6: stiffness reduction at ¼ span

Similar to DC1 to DC3, DC4 to DC6 introduce stiffness reduction, but now at the ¼ span instead of mid-span. From Figure 3.9, we can see that the DFs shows the existence of damage very well with high robustness. In addition, the values of DFs are also related to the severity of damage. When comparing Figure 3.8 and Figure 3.9, it is seen that at the same percentage of stiffness reduction, the DFs for mid-span cases are larger to ¼ span cases. From our modal analysis, the first natural frequency of the bridge drops from 7.3365 Hz to 7.0047 Hz if the damage occurs at mid-span, but for ¼ span damage, the first frequency only drops to 7.1755 Hz. We can see that damage at mid-span has larger impact on the overall performance of the bridge.

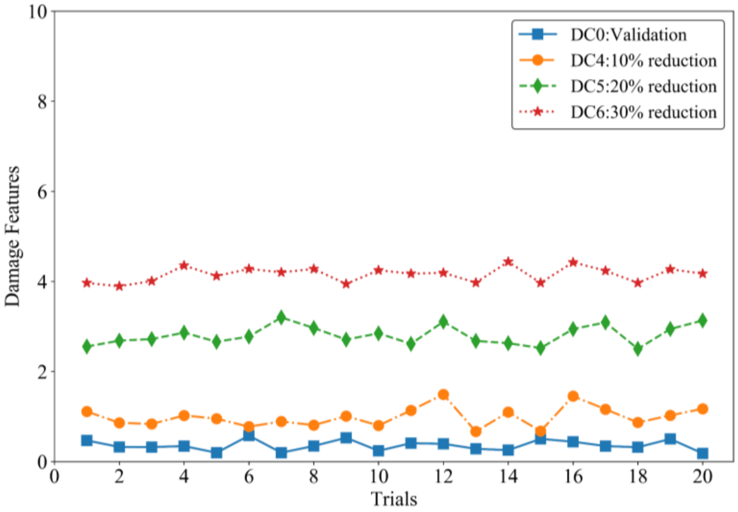


Figure 3.9 - DFs for DC4-DC6

3.2.3.3 DC7: boundary condition changes

Figure 3.10 illustrates the most severe damage case introduced in the simulation, change of boundary conditions at both ends. We can see from the figure that the DFs for this case is around 11, which is almost twice as DFs for 30% stiffness reduction at mid-span and thrice as DFs for 30% stiffness reduction at ¼ span. This concludes that the DFs derived from MFCCs are sensitive to not only the existence but also the severity of the damage.

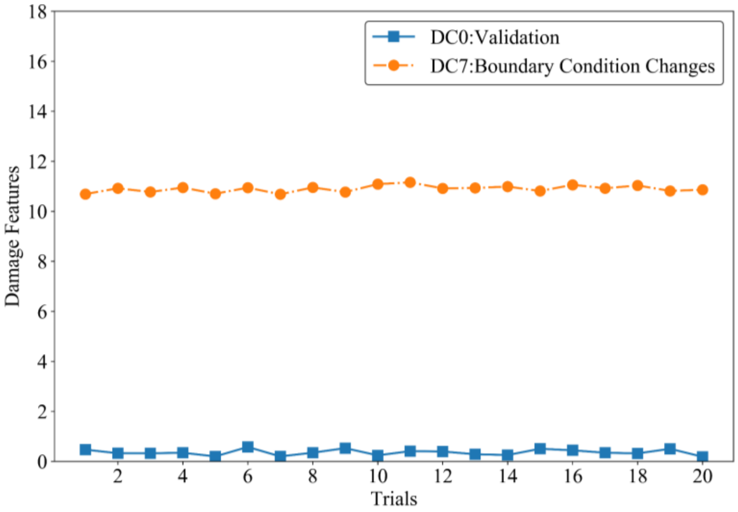


Figure 3.10 - DFs for DC7

3.3 Part II: An improved methodology taking advantage of a large number of vehicles

3.3.1 Mobile Sensor Network

In last section, MFCCs have been proven sensitive to damage and robust to the noise. However, in such damage detection approach, the parameters of the vehicles, such as natural frequency or speed, must be kept the same or very similar for different tests so that the variation of vehicle configurations would not mask the real damage on bridge. Also, the tests have to be scheduled on a single bridge at a time in order to detect damage.

In recent years, Internet of Things technologies have been developing significantly [95, 96]. Sensors can easily be installed on smart devices, such as electric vehicles and cell phones, and they usually have the ability to connect to Internet. Therefore, large amount of data (i.e. 'big data') can be collected in real time for analysis use [38, 97]. In this section, an improved methodology is proposed for damage detection of bridges by conducting a PCA based statistical analysis of the MFCCs extracted from data collected from large number of drive-by vehicles. Using this approach, although the type of vehicles may vary for different tests, the statistical characteristics of all the vehicles can be used for damage detection. Under the concept of mobile sensor network, the monitoring process could be distributed over a population of bridges and be real time with the help of several thousands or tens of thousands of vehicles, if not more.

3.3.2 Principal Component Analysis

Unlike MFC, PCA has been applied for SHM extensively for years [98-100]. Zang and Imregun [101] used PCA as a pre-process step for frequency response functions based damage detection. Using the frequency response functions with reduced dimensionality as input, artificial neural network provides good performance for damage detection. Tibaduiza et al. [102] proposed an approach that integrates a multiactuator system, PCA and self-organizing maps, and experimentally showed that their approach can successfully classify baseline cases and six damage cases. Park et al. [103] invented an approach based on PCA and k-means. In their approach, an on-board active sensor system was utilized to collect impedance data. PCA was applied to eliminate unwanted noise in impedance data, and then k-means unsupervised learning algorithm was employed for damage detection. The approach was verified through a bolt-jointed aluminum structure with loosing bolts.

In other studies, PCA is usually used for dimensionality reduction. In our approach, after the MFCCs are extracted, PCA is used to transform them into a space that the features are uncorrelated so that each transformed feature can be compared separately. Assuming we get q features from MFC, the original features are stored in a matrix called $\mathbf{X_k}$, and $X_{ij,k}$ represents the j^{th} feature for i^{th} vehicle for damage case k. The first step of PCA is to subtract the column-wise mean from each row:

$$\mathrm{X}'_{ij,k} = X_{ij,k} - \frac{1}{m}\sum_{i=1}^{m} X_{ij,k}, j = 1,2,3,\cdots,q \tag{3.11}$$

where m is the number of vehicles for each damage case, and q is the number of features we extracted from MFC.

Then, the covariance matrix can be derived as below,

$$\mathbf{C_k} = \begin{bmatrix} \mathrm{cov}(X'_{1,k}, X'_{1,k}) & \mathrm{cov}(X'_{1,k}, X'_{2,k}) & \cdots & \mathrm{cov}(X'_{1,k}, X'_{q,k}) \\ \mathrm{cov}(X'_{2,k}, X'_{n_1,k}) & \mathrm{cov}(X'_{2,k}, X'_{2,k}) & & \mathrm{cov}(X'_{2,k}, X'_{q,k}) \\ \vdots & & \ddots & \vdots \\ \mathrm{cov}(X'_{q,k}, X'_{q,k}) & \mathrm{cov}(X'_{q,k}, X'_{2,k}) & \ldots & \mathrm{cov}(X'_{q,k}, X'_{q,k}) \end{bmatrix} \tag{3.12}$$

$$\mathrm{cov}(X'_{u,k}, X'_{v,k}) = \frac{1}{m-1}\sum_{i=1}^{m} X'_{iu,k} X'_{iv,k} \quad u,v = 1,2,3,\cdots,q \tag{3.13}$$

The covariance matrix is q by q. The eigen-values and eigen-vectors of the covariance matrix must satisfy the following equation:

$$\mathbf{C_k} \times \mathbf{V_k} = \boldsymbol{\lambda}_{\mathbf{k}} \times \mathbf{V_k} \tag{3.14}$$

where **V** is the eigen-vector matrix and **λ** is the eigen-value matrix. The transformation of features is implemented by multiplying original features by eigen-vectors [104].

$$\mathbf{Y_k} = \mathbf{X_k} \times \mathbf{V_k} \tag{3.15}$$

where $\mathbf{Y_k}$ is the transformed feature matrix, in which $Y_{ij,k}$ is the j^{th} transformed feature for i^{th} vehicle in damage case k.

3.3.3 Damage Detection Procedure

The procedure to obtain the transformed features is summarized in Figure 3.11. For the baseline case of the bridge, acceleration data from m_1 passing-by vehicles are collected within a certain period of time. MFCCs are calculated using the data from every single vehicle. Then, PCA is implemented for all m_1 vehicles passing across the bridge at baseline case. In this way, the average of each transformed feature is set to zero since PCA normalizes the data. For the unknown case, acceleration data from m_2 passing-by vehicles are collected within another period of time. MFCCs are calculated for the unknown case and then, the MFCCs are transformed using the eigenvectors obtained by the PCA conducted with the baseline case data. In this way, the transformed features for the data from both baseline and unknown cases can be compared in the same space.

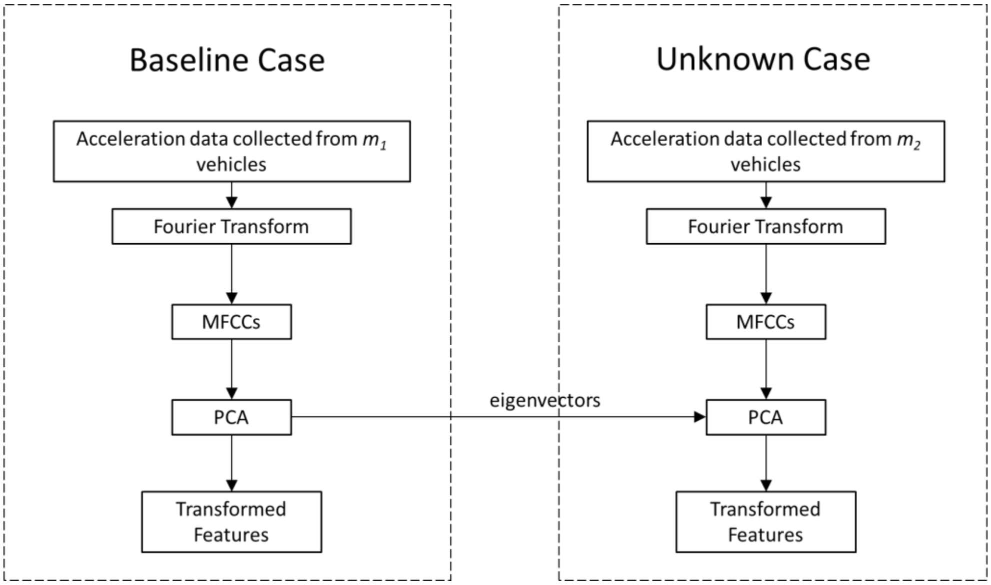


Figure 3.11 - Procedure to obtain transformed features

Following the above procedure, each vehicle has q transformed features. When a large volume of acceleration data are collected, each transformed feature would have m_1 samples for baseline case and m_2 samples for unknown case where m_1 and m_2 are the number of vehicles for both baseline and unknown cases. Although the numbers of samples for each transformed feature are different for baseline and unknown cases, the statistical characteristics are still comparable assuming the types of vehicles follow the similar distributions. The damage detection procedure using transformed features is described in Figure 3.12.

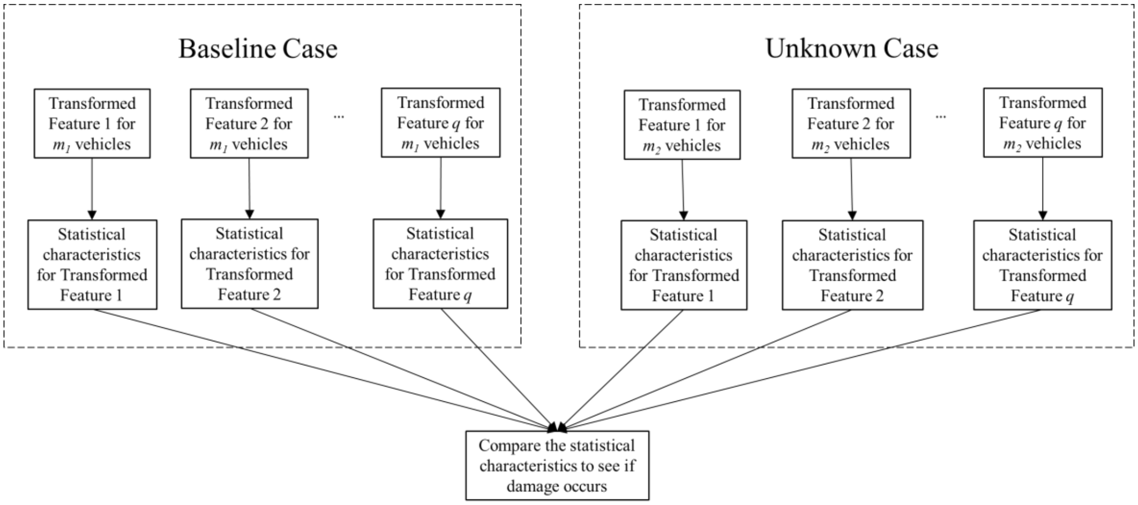

Baseline Case
Transformed Feature 1 for m_1 vehicles
Transformed Feature 2 for m_1 vehicles
...
Transformed Feature q for m_1 vehicles
Statistical characteristics for Transformed Feature 1
Statistical characteristics for Transformed Feature 2
Statistical characteristics for Transformed Feature q
Unknown Case
Transformed Feature 1 for m_2 vehicles
Transformed Feature 2 for m_2 vehicles
...
Transformed Feature q for m_2 vehicles
Statistical characteristics for Transformed Feature 1
Statistical characteristics for Transformed Feature 2
Statistical characteristics for Transformed Feature q
Compare the statistical characteristics to see if damage occurs

Figure 3.12 - Procedure to detect damage

The damage detection procedure in this section is based on the premise that if the bridge is damaged, the statistical characteristics of the transformed features should vary when the distributions of vehicle configurations are similar. In this chapter, the Euclidean distance of the mean of the distributions of all transformed features for baseline and unknown cases are considered as an indicator of damage, i.e. Damage Feature (DF). The DFs are calculated as below:

$$\mathrm{DF} = \sqrt{\sum_{j=1}^{q} \left(\frac{1}{m_1}\sum_{i=1}^{m_1} Y_{ij,0} - \frac{1}{m_2}\sum_{i=1}^{m_2} Y_{ij,k}\right)^2} \tag{3.16}$$

where $Y_{ij,0}$ is the j^{th} transformed feature of i^{th} vehicle for baseline case, and $Y_{ij,k}$ is the j^{th} transformed feature for i^{th} vehicle in unknown case k.

3.3.4 Numerical Verification

In this section, the numerical setup that is the same as the one in section 3.2 is used. Since the method proposed in this part requires a large number of tests, the parameters are changed in spring mass model to simulate different vehicles. To be specific, for each damage case, the following changes are made for the vehicle:

1) The spring constant of the vehicle is changed to 50%, 75%, 100%, 125%, and 150% of its original value
2) The mass of the vehicle is changed to 50%, 75%, 100%, 125%, and 150% of its original value
3) The speed of the vehicle is changed to 50%, 75%, 100%, 125%, and 150% of its original value

Therefore, there are 5×5×5＝125 tests for each bridge configuration. For the healthy state of the bridge, we randomly pick 60% of the tests (60%×125=75) and reserved them for baseline case. The remaining 40% healthy data are used for validation (40%×125=50). All damaged cases with 125 data entries are compared with baseline case built with 75 randomly picked data entries. In this way, we can verify the approach even when the distributions of the car configurations are not exactly the same for baseline and unknown cases. The whole process is repeated 10 times. In each time, different data entries in baseline case are picked.

3.3.5 Distribution of Transformed Features

The distributions of all 6 transformed features for baseline and validation cases calculated by the approach proposed in this part are shown in Figure 3.13 and. In Figure 3.13, the diagonal plots demonstrate the probability density curves of transformed features in baseline and validation cases. For instance, the plot at the first row and the first column represents the probability density curves of transformed feature 1. It is seen that the curves for all transformed features are very close to each other even though they have different number of samples and configurations for vehicles. Since the transformed features are distributed in 6 dimensional space (6 transformed features), pairwise plots for any of two transformed features are given in order to visualize the patterns better. The off-diagonal pairwise scatter plots show that the transformed features vary even when no damage exists. The probabilistic approach proposed in our study provides a tool to identify whether the change is caused by the real damage or just the variation of vehicle configurations. If only the approach in part I is utilized, there will be a big chance to obtain false positive or negative results due to changes in vehicle configurations. In contrast, it is reasonable to compare statistical characteristics of two clusters constructed by a number of data points. In this way, the impact of variation of vehicle configurations or noise from single measurement can be significantly reduced. Figure 3.14 shows the boxplot of all 6 transformed features as another way to represent the distributions. In such plot, the band inside the box represents the median of the transformed feature, while the top and bottom of the box indicates the first and third quartiles. The maximum and minimum values of each transformed feature are also marked on the boxplot. Using boxplot, we can see the distributions of all 6 transformed features for baseline and validation cases are indeed very similar.

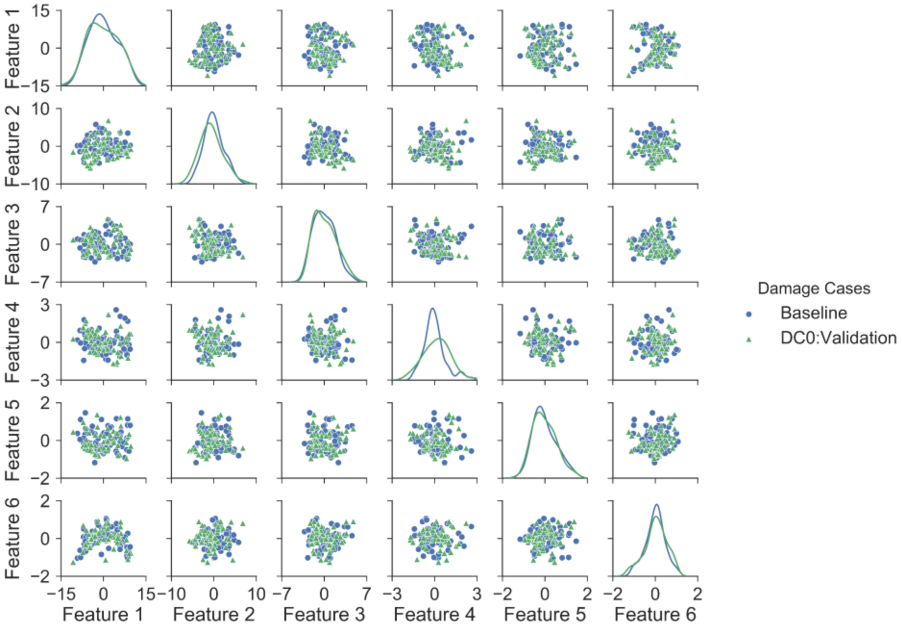


Figure 3.13 - Pairwise plots of transformed features for baseline and validation cases

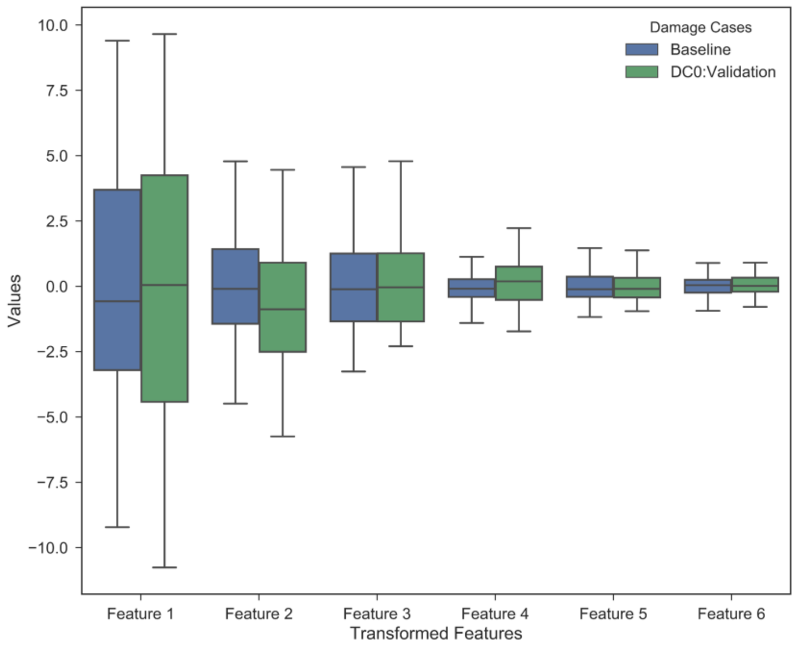


Figure 3.14 - Boxplot of transformed features for baseline and validation cases

3.3.6 Results and Analysis

3.3.6.1 DC1-DC3: Stiffness reduction at mid-span

The results for cases of stiffness reduction at mid-span are shown in Figure 3.15, Figure 3.16 and Figure 3.17. The probability density curves and pairwise relationships of all 6 transformed features for baseline case, DC1 and DC3 are presented in Figure 3.15. The blue circles represent the features for baseline case, while green triangles and red squares are for DC1 and DC3, respectively. In the off-diagonal plots of Figure 3.15, it is seen that the transformed features for each case tend to form a cluster. In other words, when the vehicle configurations change, the transformed features tend to vary around a center. The center of the cluster moves when the state of the bridge changes. In the plots, it is seen that the center of the transformed features for DC3 has larger shift from the baseline than the center from of the transformed features for DC1, which is a sign that the center of the clusters can be used as an indicator for relative severity assessment. Also from the pairwise scatter plots, since the points with different colors have some overlaps with each other, which indicates that changes in those transformed features caused by the variation of vehicle configurations, such as frequency and speed, may be at the same level as those caused by damage. Therefore, it is deemed that employing the MFC approach proposed in part I using single vehicle measurement may not reliable when the vehicle changes, because the change of transformed features caused by the variation of vehicle configuration could mask the ones caused by the damage in bridge. When looking at the probability density curves in Figure 3.15 and box plots in Figure 3.16 of those transformed features for various vehicles, it is also seen the deviation has become larger as the damage becomes more severe. This demonstrates that the statistical characteristics of the transformed features using MFC and PCA are more reliable for indirect health monitoring of bridges using data collected from a large number of vehicles.

The DFs calculated from transformed features are presented in Figure 3.17. They are grouped by damage cases, and each group includes ten trials (with different artificial noise and sampling). Compared with the validation case, the DFs for DC1 to DC3 are larger. For 10% stiffness reduction at mid-span, the DFs are around 1.5, while the DFs are about 5 for 30% stiffness reduction cases. The DFs gradually increase as the damage become more severe, which means that the improved approach can still show the existence and severity of the damage without any issues. In addition, for 10 trials within each damage case with added artificial noise, the DFs values are very stable, which shows that the DFs are robust to randomness in the sampling process.

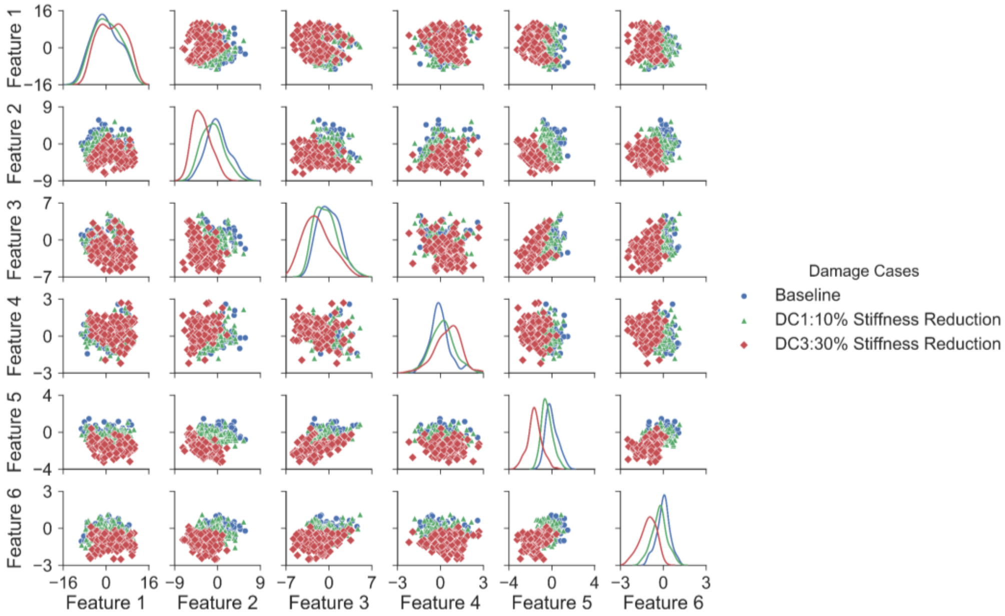


Figure 3.15 - Pairwise plots of transformed features for baseline case, DC1 and DC3

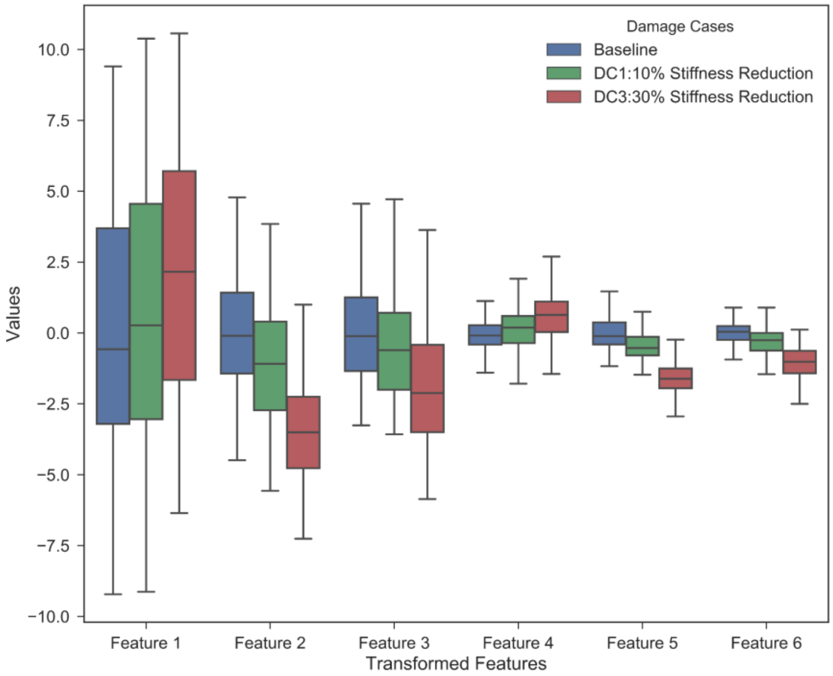


Figure 3.16 - Boxplot of transformed features for baseline case, DC1 and DC3

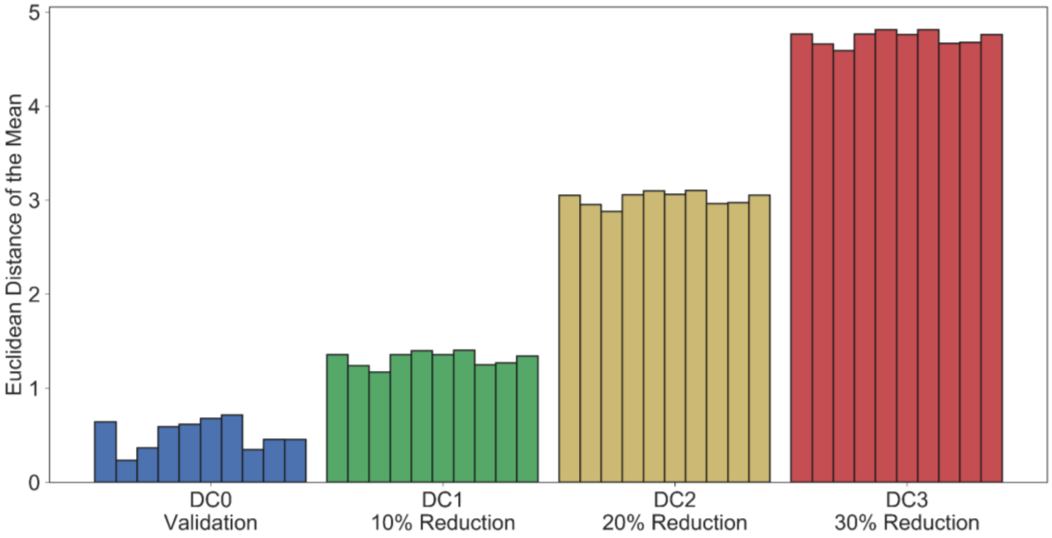


Figure 3.17 - DFs for validation case, DC1, DC2 and DC3

3.3.6.2 DC4-DC6: Stiffness reduction at ¼ span

In Figure 3.18 and Figure 3.19, the distributions of transformed features related to stiffness reduction at ¼ span are demonstrated. It is shown that the patterns for transformed features in DC4 to DC6 are very similar to patterns in DC1 to DC3 since they are all damage caused by stiffness loss. Like in DC1-DC3, the centers of the clusters for DC4 to DC6 are still related to the state of the bridge. In Figure 3.19, it is seen that as damage increases, transformed features 1, 2, 3 and 5 decrease, and transformed features 4 and 5 increase. In the multi-dimensional transformed feature space, we are measuring the absolute shift of cluster centers for damage detection, and the direction of deviation should not affect the results. Therefore, Euclidean distance is used to define the DFs.

As mentioned above, the DFs derived from transformed features as Euclidean distance of the mean values are presented in Figure 3.20. In this figure, we can still see that the DFs are higher as the damage increases. Comparing Figure 3.17 and Figure 3.20, we can see that DFs for mid-span cases is larger than for ¼ cases when the same percentage of stiffness loss is applied. This is because that stiffness loss at mid-span has more impact on the global performance of the bridge.

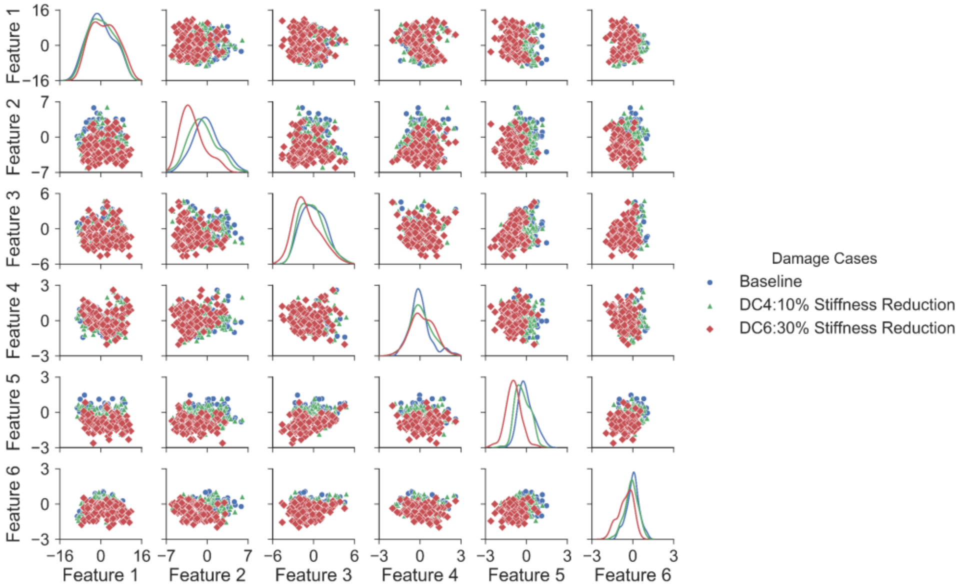


Figure 3.18 - Pairwise plots of transformed features for baseline case, DC4 and DC6

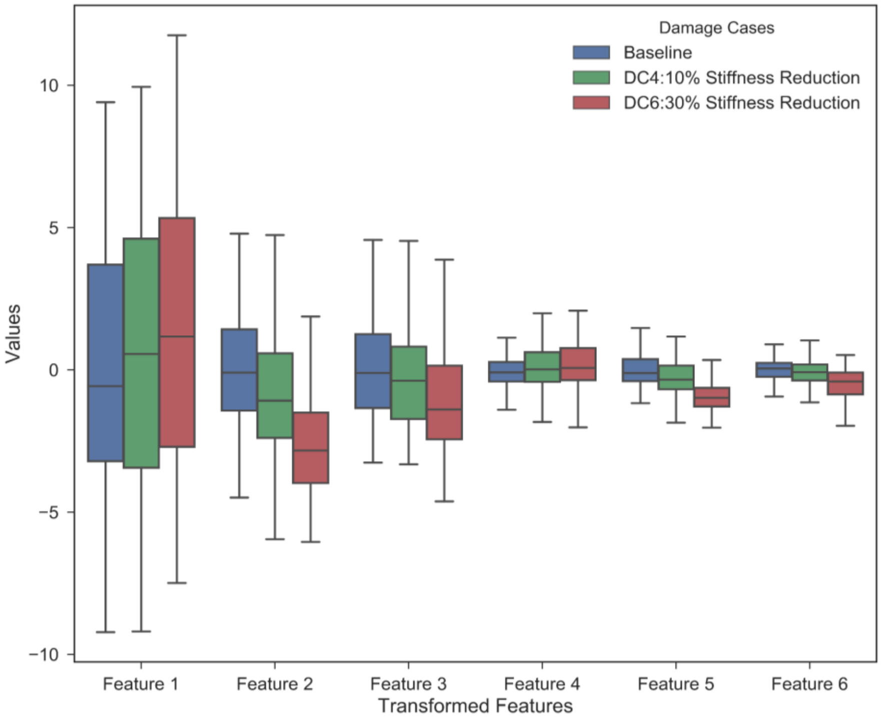


Figure 3.19 - Boxplot of transformed features for baseline case, DC4 and DC6

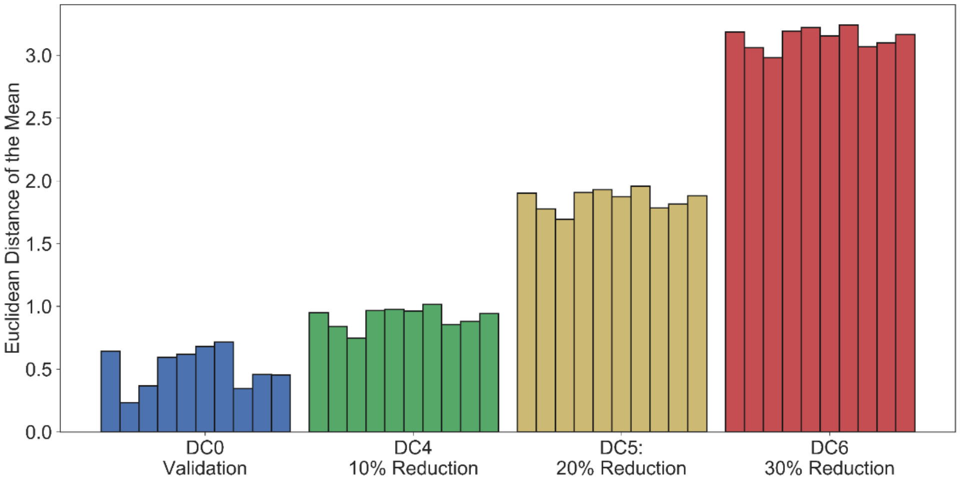


Figure 3.20 - DFs for DC4-DC6

3.3.6.3 DC7: boundary condition changes

DC7, boundary condition changes at both ends, is very severe damage compared to other local damage simulated. The pairwise scatter plots and probability density curves for 6 transformed features are presented in Figure 3.21. From both scatter points and probability density curves, we can tell that the difference of the distributions of transformed features between baseline and DC7 is much larger than other cases. From Figure 3.22, it is seen that the means of all 6 features deviate from baseline significantly. Like all other damage cases, the DFs derived from Euclidean distance of the transformed features are presented in Figure 3.23. The DFs for DC7 are around 9 for this case, which is almost twice as DFs for 30% stiffness reduction at mid-span in Figure 3.17. The DFs are also stable among 10 trials with different sampling and artificial noise.

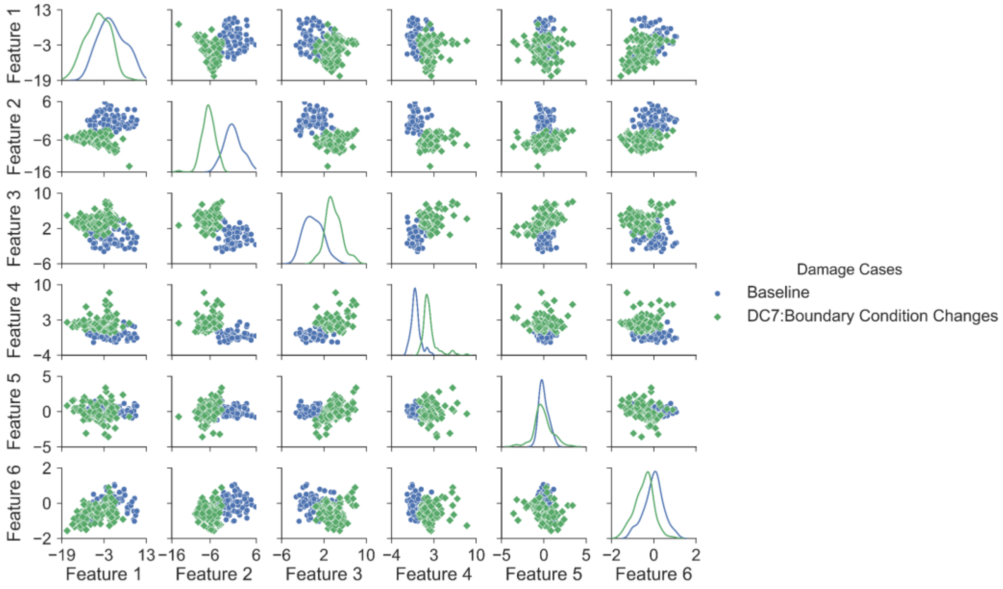


Figure 3.21 - Pairwise plots of transformed features for baseline case and DC7

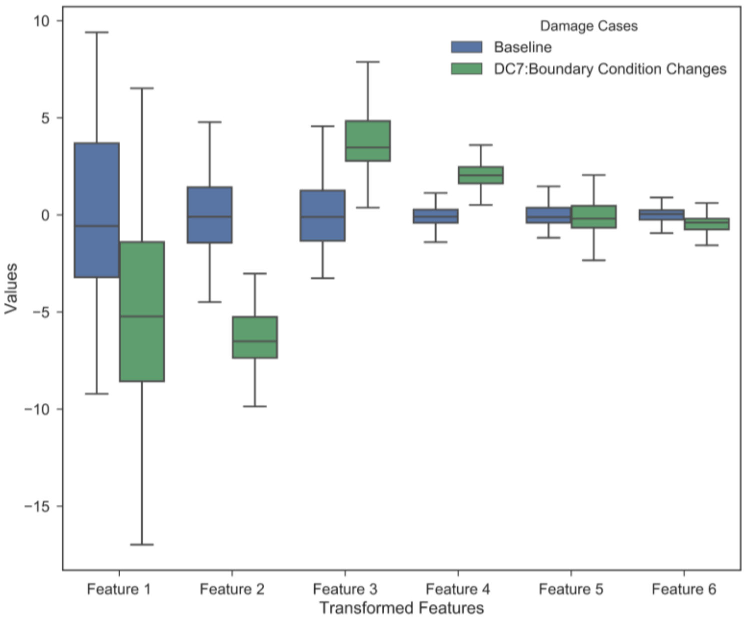


Figure 3.22 - Boxplot of transformed features for baseline case and DC7

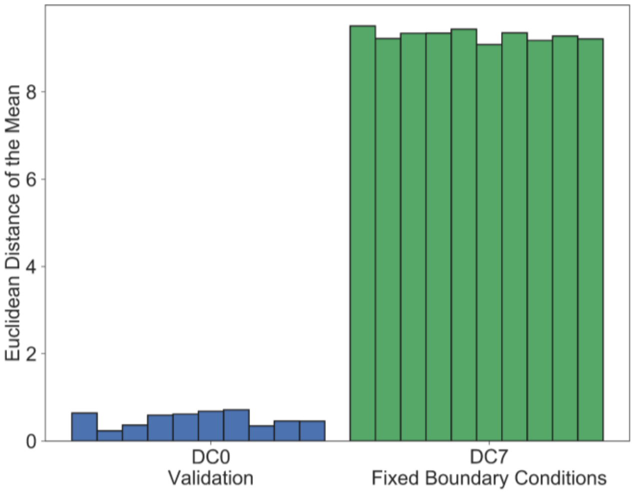


Figure 3.23 - DFs for DC7

3.3.6.4 Influence of number of vehicles

In the previous analysis, 75 vehicles are randomly selected for baseline case, while 125 vehicles are used for other cases. As discussed in previous sections, the DFs fluctuate because different sets of vehicles are used for baseline and other cases. The artificial noise added to the acceleration data could also lead to fluctuation. In this section, the influence of the number of vehicles on the DFs is investigated. In Figure 3.24, the coefficient of variance (CoV) of DFs is plotted against 25 to 100 vehicles for baseline case. It is seen that the CoV of DFs is higher when a small number of vehicles are used for baseline case. The reason is that when the data set is not big enough, the estimate of distribution is not as accurate. When the number of vehicles increases, the CoV for DFs decreases gradually. Therefore, it is safe to say that the method is more robust when larger number of vehicles are used which is very promising for real-life applications with big-data extracted from large number of vehicles.

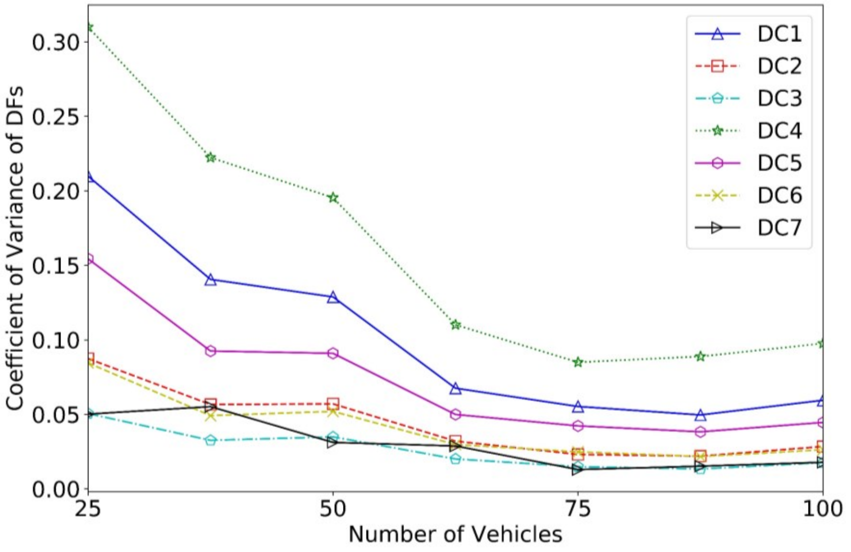


Figure 3.24 - Relationship between CoV of DFs and number of vehicles

3.3.6.5 Influence of number of transformed features

In previous analysis, 6 transformed features calculated through MFC and PCA are used for damage detection. In order to investigate the effect of number of transformed features on the performance of the method, the number of features is changed between 4-10 and a sensitivity analysis is conducted. The ratio of the average DF between damaged cases and DC0 are used as an indicator to show the distinguishability of the DF. As can be seen in Figure 3.25, the changes in the ratios for DC1-DC6 as the number of transformed features increase are negligible. The performance of DF is considered quite stable for different number of transformed features. Therefore, in practice, this parameter is not expected to have a significant impact on the results.

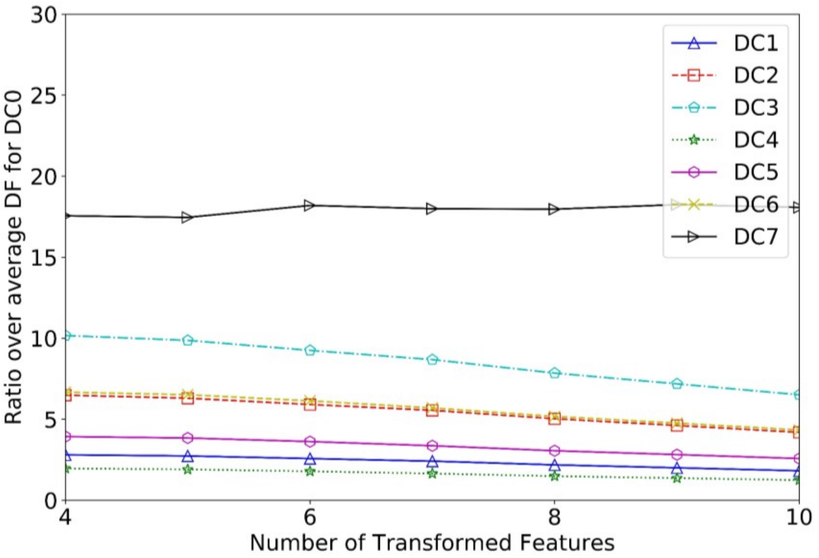


Figure 3.25 - Ratio over average DF of DC0 for different number of transformed features

3.4 Experimental Verification

3.4.1 Experimental Setup

In addition to numerical simulations, lab experiments are conducted to verify the improved approach proposed in part II (see Figure 3.26). In the experiments, a simply supported bridge is used. The bridge deck is made of steel. The length of the bridge is 2000 mm, the width is 304 mm, and the thickness is 12.7 mm.

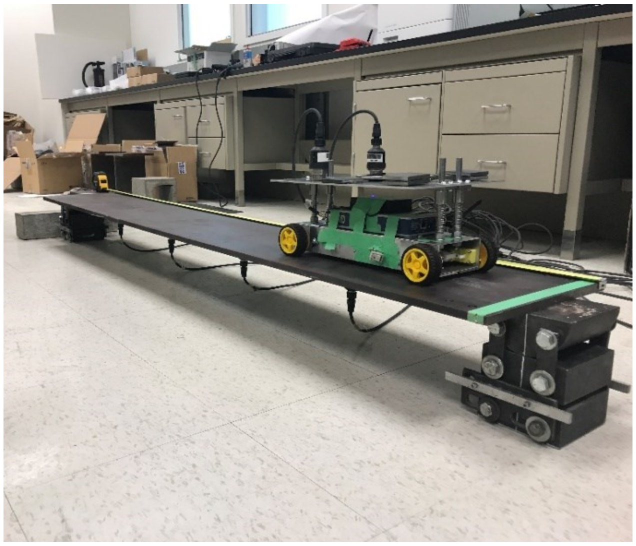


Figure 3.26 - Setup of the lab experiment

A robot car is designed to simulate the spring mass vehicle in terms of frequency. The sensors and added masses are mounted on the top plate of the car, and the data acquisition system is installed on the chassis of the car. To simulate the suspension system, four rods with springs are installed on the car. The top plate is connected to the chassis of the car through linear bearings. The chassis of the car is 2.710 kg. The configuration of the robot car is demonstrated in Figure 3.27. It is powered by 5 AA batteries and controlled by Arduino UNO board, which are mounted inside the chassis of the car. The vertical acceleration is measured through two PCB 393A03 accelerometers and a NI9234 data acquisition (DAQ) system with sampling frequency of 1652 Hz. NI LabVIEW is the software for data acquisition.

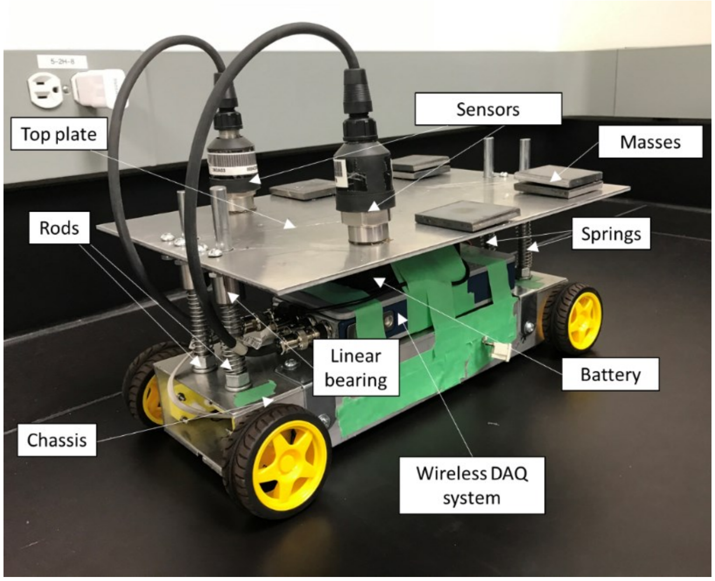


Figure 3.27 - Components of the robot car

Similar to numerical analysis in section 3, the configuration of the vehicle is changed to simulate the variety of vehicles passing across the bridge. The change in the properties of three components, i.e. weights of the top plate, speed of the car, and spring constants of the suspension system, are considered in the experiments, since these three parameters could affect the dynamic behavior of the vehicles significantly. As shown in Table 3.1, for each component, three values are used. The weight of the top plate is changed by putting additional mass on the middle of the top plate. The speed is changed by adjusting the voltage on Arduino board. The spring constant of the suspension system is changed by replacing the springs (see Figure 3.28).

Table 3.1 - Variation of car components

	1	2	3
Weights of top plate (kg)	2.094	2.350	2.604
Speed (m/s)	0.09	0.185	0.250
Spring Constants (N/m)	368.4	420	580

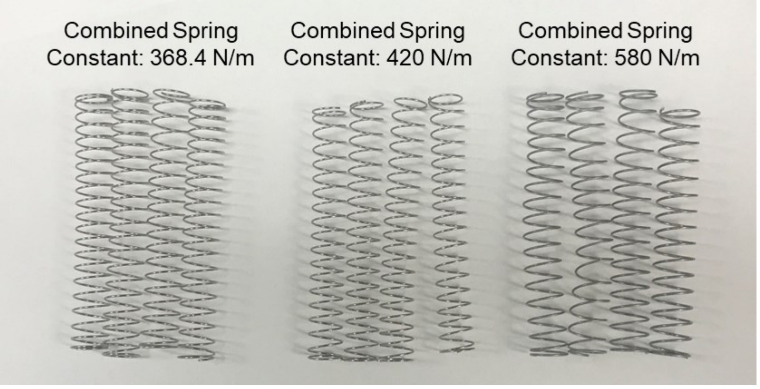
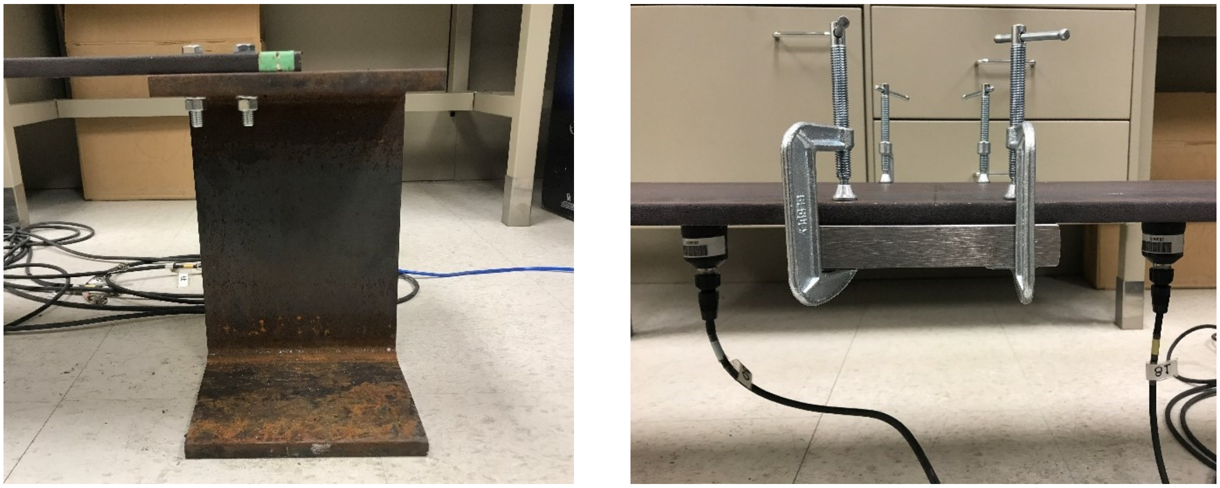

Combined Spring Constant: 368.4 N/m
Combined Spring Constant: 420 N/m
Combined Spring Constant: 580 N/m

Figure 3.28 - Springs with different spring constants to simulate different suspension systems

As shown in Figure 3.29, two damage cases as well as baseline case are applied to the bridge:

1) DC1: Boundary condition change at both ends (Figure 3.29(a)).
2) DC2: Added mass at mid-span (Figure 3.29(b)).

In DC1, the boundary condition change is experimentally simulated by replacing rollers with short I-beams and connect the bridge deck to the short beam through four M12 bolts at each side. In DC2, two 5 kg steel blocks are softly attached to the bridge deck through clamps at mid-span.



(a) DC1: Change of boundary conditions

(b) DC2: Added mass at mid-span

Figure 3.29 - Damage applied to the bridge

For each vehicle configuration, the test is repeated three times. Therefore, there are in total 3×3×3×3=81 tests for each damage case. Similar to numerical analysis, 60% of data in the healthy case are reserved for baseline, and 40% are for validation purpose.

Since the vehicle requires time to start and stop, the acceleration data near the ends are not reliable and are therefore truncated. The speed of the robot car is pre-set to the values in Table 3.1, but it is not precisely controlled by the program. This is to mimic the fact that vehicles on the bridge may not have constant speed. It should also be noted that only a single car is considered on the bridge at a time. The road roughness is not simulated explicitly in the experiments, but the existence of rust on the bridge deck is expected to simulate the road roughness.

3.4.2 Interpretation of Results

Since there are two accelerometers installed on the car, the average of these two sensors is taken as input for our analysis to reduce the uneven movement of the top plate. Figure 3.30 presents the average acceleration taken from the robot car with 2.350 kg top plate, 0.185 m/s speed, and 420 N/m spring constant passing over the baseline bridge. It should be noted that the first two seconds of the acceleration data is not used to avoid the disturbance of the engine startup.

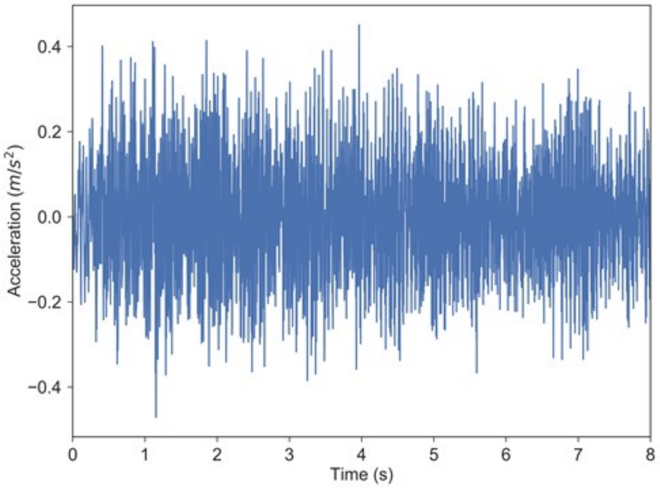
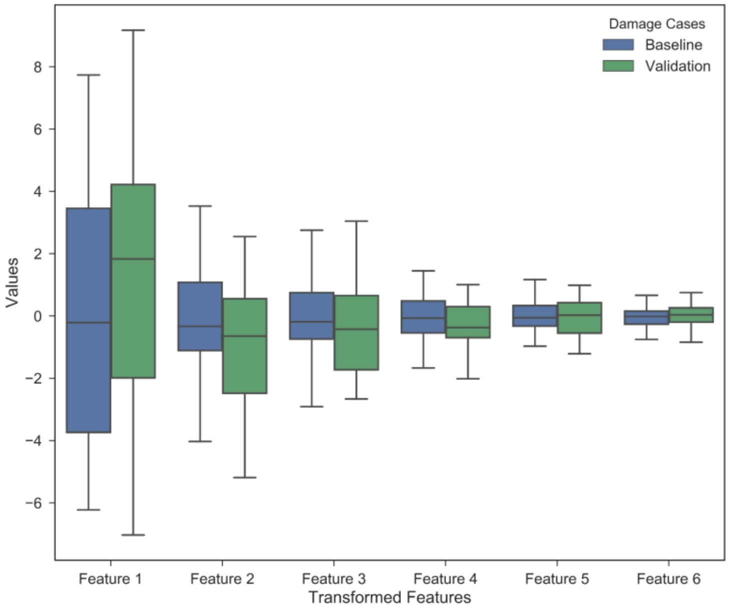
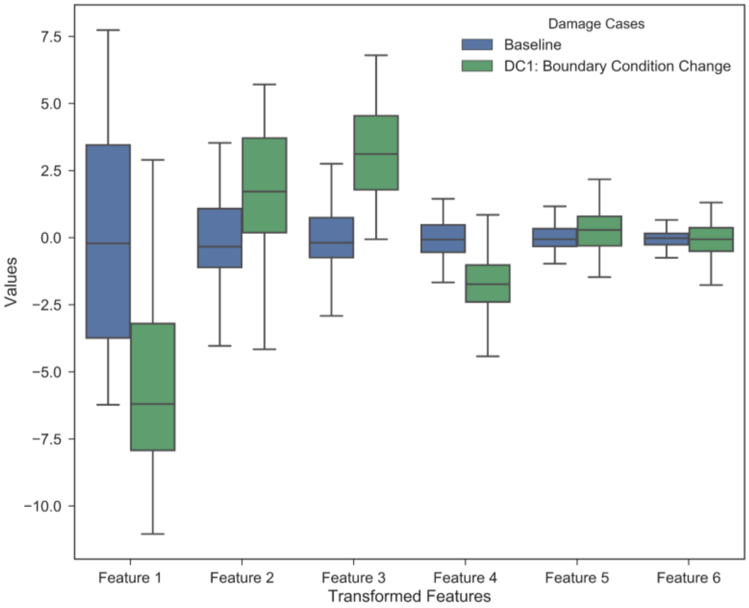


Figure 3.30 - Average acceleration taken from two PCB 393A03 accelerometers.

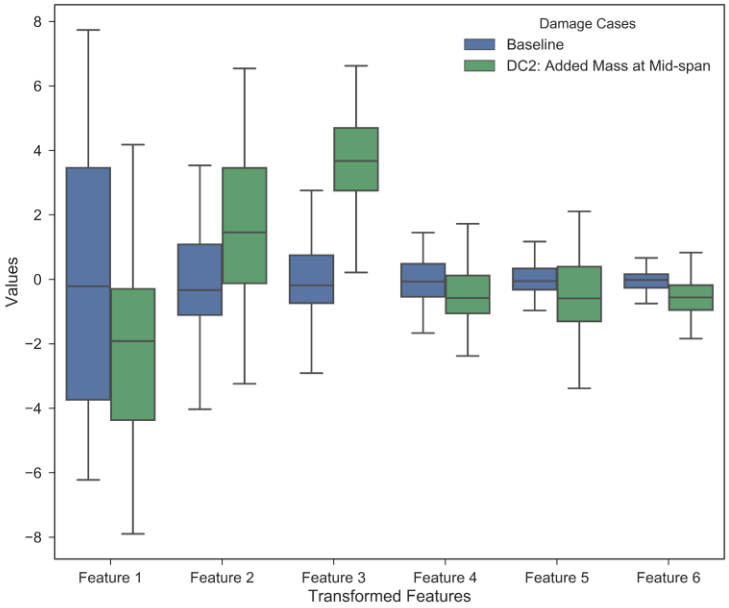
Similar to numerical analysis, for different damage cases, the distributions of the transformed features extracted from MFC and PCA are shown in Figure 3.31 in terms of boxplots. Regarding damage, DC1 is boundary condition change to simulate global damage, while DC2 is to put additional mass at mid-span to simulate local damage. Figure 3.31(a) shows the distribution of transformed features for both baseline and validation data. Since both baseline and validation cases represent the same structural configuration (healthy), it is seen that all 6 transformed features have very similar distributions in terms of statistical characteristics such as median, quantiles and mean. However, the difference is not as small as in numerical analysis, this is because the experimental data are noisier and corrupted by other factors, such as vibration of the motors and road surface roughness. In Figure 3.31(b), the distributions of transformed features for boundary condition change case (DC1) is compared with baseline case. It is seen that the difference between these distributions are much larger than in Figure 3.31(a), which indicates the existence of structural change, i.e. damage. It should be noted that the variation for transformed features 1-4 is larger than for transformed features 5-6, which means transformed features 1-4 are more sensitive to the damage in this case. More detailed investigations about the sensitivity of transformed features to damage will be conducted in future. Figure 3.31(c) shows the distributions of those transformed features for added mass case (DC2). The box plots for DC2 also deviate significantly from baseline case, but the difference is smaller than DC1, which indicates that DC2 is less severe than DC1.



(a) baseline case and validation case



(b) baseline case and DC1



(c) baseline case and DC2

Figure 3.31 - Boxplot of transformed features for baseline case, validation case, DC1 and DC2

As shown in Figure 3.32, the DFs which are the Euclidean distances of the mean of transformed features as specified in Eq. (3.16) are calculated for both two damage cases. It is shown that DFs for both DC1 and DC2 are higher than validation case, which implies the existence of damage. In addition, DFs for DC1 is higher than DC2, which proves that boundary conditions change is more severe than putting additional mass locally. It should also be noted that the DFs are considered stable in the experiments when the data from vehicles are sampled differently even though the disparity among different samplings are higher than in numerical analysis.

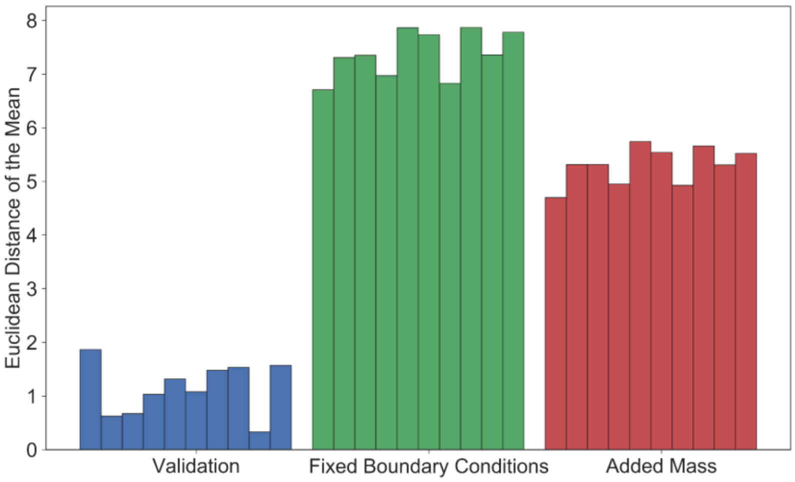


Figure 3.32 - DFs for different damage cases

3.5 Conclusions

This chapter consists of two parts. In the first part, a novel damage feature derived from adapted MFCCs is developed for damage detection of bridges using drive-by data. To overcome the deficiencies from a single drive-by measurement, the concept of mobile sensor network for structural health monitoring utilizing large amount of data is introduced in the second part. With this concept, an improved version of the approach based on adapted MFCCs and PCA is proposed. In this approach, the statistical characteristics of the transformed features from a large number of vehicles are extracted and compared. Both numerical analysis and experiments are conducted to verify the approach. The following major conclusions are obtained from our analysis:

1) DFs based on MFCCs are very sensitive to the damage and robust to the noise. The damage, including stiffness change and boundary condition changes, can be successfully detected using these DFs.
2) The values of DFs are related to the relative severity of the damage.
3) In the proposed approach, the damage can be detected and quantified using data from a large number of vehicles. No specific parameters from a single vehicle are required.
4) The potential of applying mobile sensor network for structural health monitoring is investigated. The research finds that usage of data from a large number of vehicular

sensors for damage detection is feasible, and has the potential to monitor a population of bridges simultaneously and in almost real time.

CHAPTER 4: DRIVE-BY BRIDGE DAMAGE DETECTION USING SMARTPHONES

4.1 Overview

This chapter presents a crowdsensing based methodology for indirect bridge health monitoring by utilizing data from smartphones in a large number of vehicles. The proposed method is an improved version of the method in chapter 2. Also, the smartphones are introduced for bridge condition assessment in this chapter. Similar to the previous chapter, the method is developed based on Mel-frequency cepstral coefficients (MFCCs) and Kullback-Leibler (KL) divergence which will be discussed in the next sections. Afterwards, numerical analysis on a bridge with parameters similar to Yang et al [44] is conducted to verify the proposed methodology. Finally, the smartphone app for data collection is developed and the laboratory experiments are implemented and discussed, which are followed by discussions and conclusions.

It should be noted that the method can easily be extended to other smart devices including smart vehicles. By definition, indirect health monitoring is to place one or more sensors on a moving vehicle and assess the condition of bridge by analyzing the data collected when vehicle is passing across the bridge. Researchers in this area have been able to identify frequency, mode shapes, damping, or the damage on bridges successfully using a single vehicle [44, 46-54, 56, 105]. However, those methods are still considered as active monitoring since specific vehicles must be dispatched for monitoring purpose. Usually, the same vehicle has to be used for all the measurement in order to make the results valid.[58] The conclusions can vary when using different vehicles or at different speeds.

In the proposed method, the smartphones in vehicles act as sensors. The data collected from smartphones in a large number of vehicles can be used to detect damage. The variation of vehicle configurations is taken into consideration in our research. When people drive across the bridges, their smartphones would automatically collect the vibration data and process the vibration data to extract features. Then, the extracted features will be transmitted to the remote database termed as cloud. Finally, the gathered data are analyzed systematically from the remote computers to assess the condition of bridges.

There are several advantages when smartphones in a large number of vehicles (i.e., crowdsensing) that connect to the internet working as mobile sensors instead of a single one is used. First, it can significantly reduce the cost for maintenance since no specific arrangements have to be made for monitoring. The vibration data of bridges can be collected by the smartphones automatically when vehicles cross the bridge. Second, this system has the potential to monitor a population of bridges regularly as long as there are vehicles passing through the bridges. Third, owing to crowdsensing, this technique is more robust to operational effects and human errors.

4.2 Methodology

4.2.1 Overview

In this section, the data analysis methodology within the proposed crowd-sensing based monitoring framework which utilizes smartphones in a large number of vehicles for indirect health monitoring is presented. In our method, it is assumed that there is one smartphone in each vehicle crossing a bridge. As different vehicles pass through the same bridge at different times. MFCCs are extracted for each vehicle (discussed in detail in section 3.2.1), and the extracted features are transmitted to a feature pool. This feature pool accumulates the features from a large number of vehicles, and an estimate of multi-variate distribution of the features can be made from the feature pool. Considering two different periods, one for baseline state of the bridge and the other for unknown state, it is assumed that there are s_1 vehicles passing across bridge in baseline period and s_2 vehicles for unknown period. Estimates of two distributions (baseline and unknown) can be compared using KL divergence (described in section 2.5). The damage detection procedure described in this paragraph is presented in Figure 4.1.

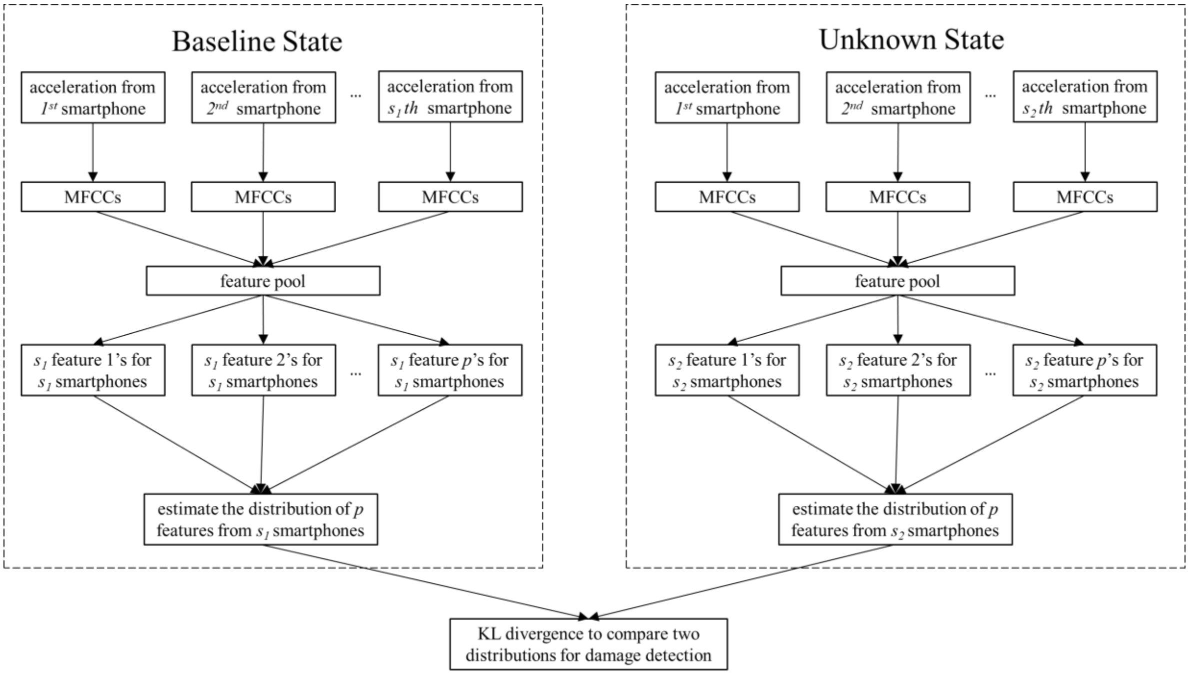

Baseline State
acceleration from 1st smartphone
acceleration from 2nd smartphone
...
acceleration from s1th smartphone
MFCCs
MFCCs
MFCCs
feature pool
s1 feature 1's for s1 smartphones
s1 feature 2's for s1 smartphones
...
s1 feature p's for s1 smartphones
estimate the distribution of p features from s1 smartphones
Unknown State
acceleration from 1st smartphone
acceleration from 2nd smartphone
...
acceleration from s2th smartphone
MFCCs
MFCCs
MFCCs
feature pool
s2 feature 1's for s2 smartphones
s2 feature 2's for s2 smartphones
...
s2 feature p's for s2 smartphones
estimate the distribution of p features from s2 smartphones
KL divergence to compare two distributions for damage detection

Figure 4.1 - Damage detection procedure

4.2.2 Vehicle Bridge Interaction

Vehicle bridge interaction is an important concept for bridge design and maintenance. Due to the dynamic interaction, vibration data collected from either vehicle or bridge would have each other's information in a coupled form [106]. The concept of indirect health monitoring is to indirectly monitor the condition of the bridge utilizing the response of the vehicles. The very early work that extracts the frequencies of the bridge from vibrational data of a passing-by vehicle is from Yang et al. [44, 107]. They provided a theoretical solution to represent vehicle's acceleration as function of the dynamic properties of both bridge and vehicle. Following that, researchers have successfully identified the damping [49, 108, 109] and mode shapes [63, 64, 110] of the bridge indirectly.

To briefly describe the vehicle bridge interaction process, a simply supported bridge is presented in Figure 4.2 and a spring mass model is used to simulate vehicle, the VBI equations can be written in Eq. (4.1) [107]. The first equation is the dynamic equilibrium of the bridge, and the second one is for the vehicle.

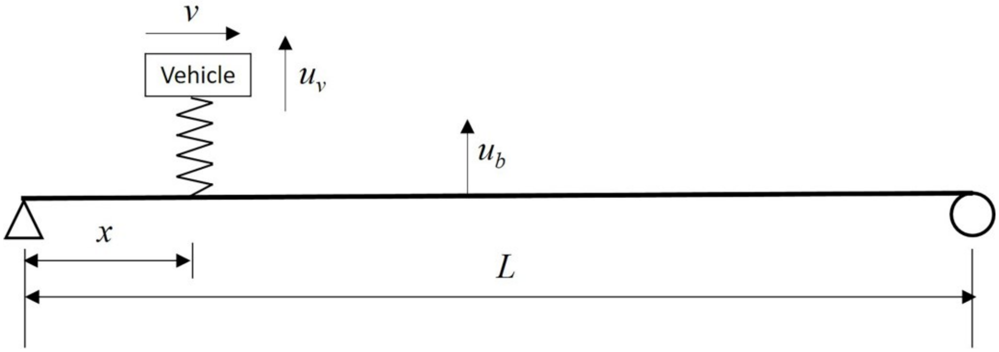


Figure 4.2 - Spring mass vehicle bridge interaction model

$$\begin{cases} \bar{m}(x)u_b^{''}(x,t)+E(x)I(x)u_b^{'''}(x,t)=c(x,t) \\ m_v u_v^{''}(t)+k_v[u_v(t)-u_b(vt,t)]=0 \end{cases} \tag{4.1}$$

where x is the distance to an end of the bridge, v is the speed of the vehicle*,* t is the current time, $\bar{m}(x)$ is the mass of bridge per unit length, $E(x)$ and $I(x)$ are elastic modulus and moment of inertia of the bridge section, m_v is the mass of the vehicle, k_v is the stiffness of the vehicle, and $c(x, t)$ is the contact force. In addition, u_b and u_v are the displacements of bridge and vehicle relative to the equilibrium position. It should be noted that u_b is related to the location x and time $t,$ and u_v is only related to time $t.$ Also note that Eq (4.1) is a general form which suits all beam-like structures, and $\bar{m}$, E and I will be constants if cross-sectional and material properties are uniform.

The detailed derivation and solution of the equations can be found in Yang and Lin [107]. It could be seen in their paper that the displacement, velocity and acceleration from vehicle include dynamic characteristics of both bridge and vehicle. Standard signal techniques such as Fourier transform is a reasonable way to separate the dynamic information. However, with the existence of road surface roughness, operational effects and measurement noise, the direct use of Fourier transform usually does not work.

4.2.3 Crowdsensing with Smartphones in Passing by Vehicles

In traditional indirect health monitoring methods, damage can only be detected when the properties of the vehicle (e.g. natural frequency, speed, etc.) are known or the same vehicle is used for both baseline and unknown cases. In reality, it is impractical to measure all the properties of the vehicle. Also, the damage could propagate slowly, and it is hard to keep the vehicle's properties the same for every test. Any inconsistency in the vehicle configurations would lead to increased uncertainties in the results.

In recent years, thanks to the fast development of the Internet of Things technologies, we can easily access a variety of sensors that are embedded in smartphones or smart connected vehicles, which have access to internet. In our method, instead of using a single vehicle, crowdsensing data collected from smartphones in a large number of vehicles are processed to extract features. Unlike traditional indirect health monitoring where features are directly compared to identify damage, our proposed method compares the distributions of features estimated from data collected from smartphones in a large number of vehicles. In this way, although the configurations of vehicles may vary for different tests, the statistical characteristics of all the vehicles are expected to be stable and thus changes in the distributions of the features can be used for damage detection. Using crowdsensing data from smartphones, a population of bridges could continuously be monitored in near real time with the help of several thousands or tens of thousands of vehicles, if not more.

4.2.4 Mel-frequency Cepstral Coefficients (MFCCs)

Cepstral analysis is the inverse Fourier transform of the logarithm of the spectrum for a signal, which is a powerful tool for speech recognition and natural language processing. There are a number of varied cepstral analysis techniques, among which Mel-frequency cepstral analysis is one of the most popular ones. The features extracted from Mel-frequency cepstral analysis is called MFCCs, i.e., Mel-frequency cepstral coefficients. The major advantages of MFCCs compared to other feature extraction methods include: 1) Mel-cepstral analysis looks for patterns in a frequency range instead of searching peaks. 2) It gives higher weights for lower frequencies and lower weights for higher frequencies. MFCC has gained great success in the area of voice recognition and natural language processing [87, 111, 112], but has only started to attract a little attention from structural health monitoring researchers in recent years [91, 92].

The procedure of calculating MFCCs from acceleration data collected from the smartphones can be summarized in four main steps: 1) the Fourier transform of the acceleration signal is computed; 2) the power spectrum in Hertz scale is mapped to Mel scale using Mel filter bank; 3) the logarithms of the mapped powers at each Mel-frequency is calculated; 4) the discrete cosine transforms of the log powers are calculated. After implementing these four steps, the amplitudes of the resulting spectrum will be MFCCs. In this way, all the information in the power spectrum would be integrated in MFCCs with higher weights for lower frequencies.

For the simplicity of the thesis, the detailed procedure the MFCCs can be found in section 3.2.1, and will not be repeated here.

4.2.5 Kullback–Leibler (KL) Divergence

For baseline case, extracting MFCCs from s_1 vehicles, an $s_1 \times p$ matrix can be formed, in which p is the number of coefficients kept from MFCCs. It should be acknowledged that this $s_1 \times p$ matrix should follow a multi-variate distribution. For an unknown state, if there are s_2 vehicles passing through the bridge, the resulting $s_2 \times p$ matrix should follow another multi-variate distribution. The damage can be identified by comparing these two distributions. In this chapter, KL divergence, which is deemed as one of the few powerful methods for comparing multi-variate distribution, is utilized to compare two probability distributions, i.e., distributions of features from baseline and unknown cases.[113] The general form of KL divergence is presented in Eq. (4.2).

$$D_{KL}(N_0 \| N_1) = \int_X \log \frac{dN_0}{dN_1} dN_0 \tag{4.2}$$

where N_0 and N_1 are the probabilities over a set X, and $D_{KL}(N_0\|N_1)$ stands for the KL divergence from N_1 to N_0. The KL divergence comes from information theory, and measures the information loss when N_1 is used to approximate N_0. KL divergence is always non-negative, and is well studied for multi-variate distribution comparison, which makes it a suitable mathematical tool for designing damage features (DFs).

In this chapter, the features collected from vehicles within a certain period are assumed to follow multi-dimensional Gaussian distribution [114, 115], the KL divergence can be written as below.[116]

$$D_{KL}(N_0 \| N_1) = \frac{1}{2}\left[\ln\frac{|\Sigma_1|}{|\Sigma_0|} + trace(\Sigma_1^{-1}\Sigma_0) + (\mu_1 - \mu_0)^T \Sigma_1^{-1}(\mu_1 - \mu_0) - k \right] \tag{4.3}$$

where μ_0 and μ_1 are the mean matrices and Σ_0 and Σ_1 are the covariance matrices for baseline and unknown cases, *trace*() stands for the trace of matrix, and k is the number of features.

The range of KL divergence is [0, ∞). Due to its mathematical form, the KL divergence is exponentially related to the distance of distributions. In order to detect the damage more robustly, it is mapped to a linear relationship with range of [0, ∞) as well. The following transformation is applied and the output is defined as damage feature (DF).

$$D\mathrm{F}(N_0, N_1)=\ln\left[D_{KL}(N_0 \| N_1) + e\right] - 1 \tag{4.4}$$

where *e* is the Euler's number.

4.3 Numerical Analysis

In this section, a series of numerical simulations conducted in Abaqus on a simply supported bridge with the same dimensions as in Yang et al. is presented [44]. The length of the bridge is 25 m. The bridge is made of reinforced concrete, which has density of 2400 kg/m^3 and elastic modulus of 27.5 GPa. The cross section of the bridge has an area of 2.0 m^2 and moment of inertia of 0.12 m^4. The first three frequencies of the bridge are 2.08 Hz, 8.33 Hz and 18.75 Hz. The bridge is modeled by beam elements and is divided into 16 elements as numbered in Figure 4.3.

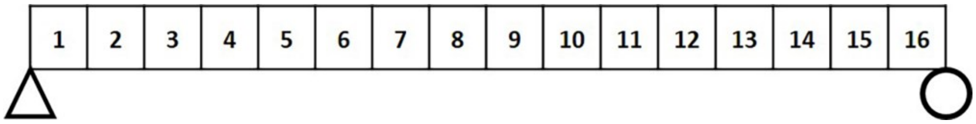


Figure 4.3 - Mesh grid of the bridge

The vehicle is simply modeled as a spring mass moving over the bridge. The spring constant, mass and speed are defined according to Yang et al. [44] and are varied to model different vehicles. The configurations are summarized in Table 4.1.

Table 4.1 - Variation of vehicle configurations

	Vehicle configurations
Mass (kg)	960, 1200, 1440, 1680, 1920, 2160, 2400
Spring constant (kN/m)	200, 250, 300, 350, 400, 450, 500
Speed (m/s)	8, 10, 12, 14, 16, 18, 20

Combing all the possible values for three parameters, there are 7×7×7=343 simulations with different vehicles. If each vehicle is repeated 5 times (artificial random noise is added to each simulation), the vehicle test pool is extended to 343×5=1715 vehicles. In Abaqus, all the vehicles passing through the bridge are simulated. The sampling frequency of the acceleration data is 100 Hz. To consider the measurement noise, all the data are corrupted with 5% artificial noise.

In order to consider the fact that the vehicles passing across the bridge during different periods are not the same, a randomly sampling process is taken for each state of the bridge. For the intact bridge, 50% of 1715 data records are sampled randomly and reserved as baseline case, and the remaining 50% data records are taken for validation (called DC0_N).

In addition to baseline and validation cases, 5 damage cases that are grouped into 3 categories are introduced to verify the proposed method. Similarly, 50% data records for each damage case are randomly sampled as well. To verify the robustness of the method, the random sampling process is repeated 30 times, and at each time the data for the baseline, validation and damage cases are selected differently. The list of numerically introduced damage cases is shown as below:

1) DC1a_N: 15% reduction of stiffness at elements 8 and 9
2) DC1b_N: 30% reduction of stiffness at elements 8 and 9
3) DC2a_N: 15% reduction of stiffness at elements 4 and 5
4) DC2b_N: 30% reduction of stiffness at elements 4 and 5
5) DC3_N: Both ends are changed to fixed supports.

Among the above damage cases, DC1_N and DC2_N are related to stiffness reduction of elements, while DC3_N is the boundary condition change at both ends. DC1_N and DC2_N introduce damage at different locations, and *a* and *b* stand for 15% and 30% stiffness reduction. The stiffness reduction cases are simulated by reducing elasticity modulus of elements.

4.3.1 DC1a_N and DC1b_N:15% and 30% Reduction of Stiffness at the Mid-Span

The DFs for the cases of stiffness reduction at the mid-span are presented in Figure 4.4. Each point on the figure represent one complete implementation of the algorithm for a single damage case. As shown in the figure, the DFs for DC1a_N and DC1b_N are about 0.3 and 0.45, which are both higher than DC0_N, i.e., validation case. Also, as the damage becomes more severe, the DFs become higher. This indicates that the severity of the damage is also successfully identified. Due to the randomness of the sampling process, the results for 30 trials are different. However, as can be seen in the figure, the DFs for each damage case fluctuate about the average within a reasonable range. The coefficient of variation of DFs for DC0_N, DC1a_N and DC1b_N are 0.128, 0.034 and 0.038, respectively. The difference of DFs among different damage cases are larger than fluctuation for a single damage case.

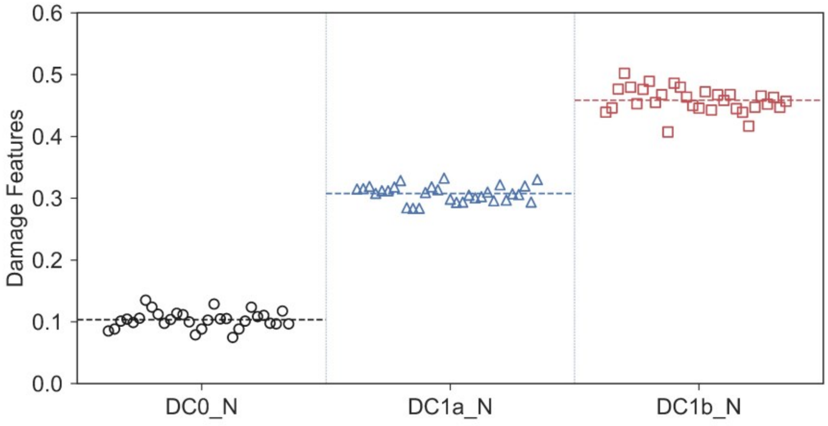


Figure 4.4 - DFs for DC1a_N and DC1b

4.3.2 DC2a_N and DC2b_N:15% and 30% Reduction of Stiffness at the ¼ Span

DC2a_N and DC2b_N are also related to stiffness reduction, but at different location than DC1a_N and DC1b_N. In Figure 4.5, it is seen that both two damage cases are successfully identified as well. The DFs also increases as the damage become more severe. The average DFs for DC2a_N and DC2b_N are about 0.3 and 0.4, and the coefficient of variation for them are 0.050 and 0.037. Comparing Figure 4.4 and Figure 4.5, the DFs for the cases with the same extent of stiffness reduction, i.e., DC1a_N and DC2a_N, DC1b_N and DC2b_N, are at about the same level, which in a way shows that the values of DFs are mainly related to the severity of damage.

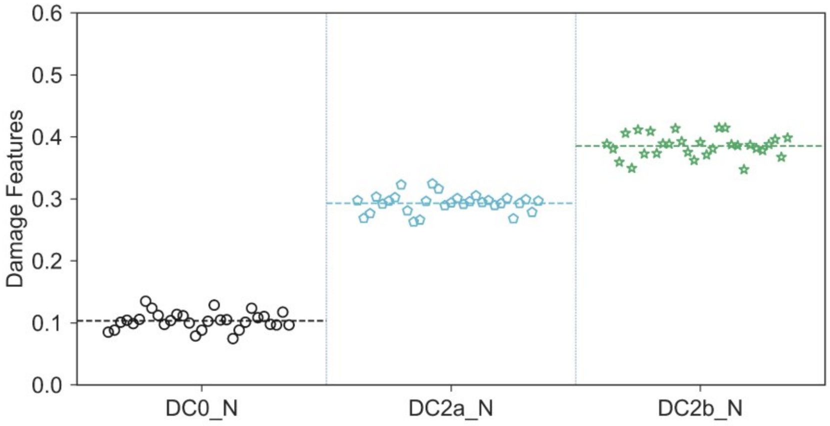


Figure 4.5 - DFs for DC2a_N to DC2b_N

4.3.3 DC3_N: Boundary Condition Change at Both Ends

Comparing with the local damage cases, DC3_N, boundary condition change at both ends, is the most severe damage. Figure 4.6 shows the DFs for both DC0_N and DC3_N. It is seen that the average of DFs for DC3_N are around 2.8, which is significantly larger than validation case and the local damage cases. This again shows that the level of DFs is a very useful indicator for assessing the severity of the damage. The coefficient of variation of DFs for DC3_N is 0.011.

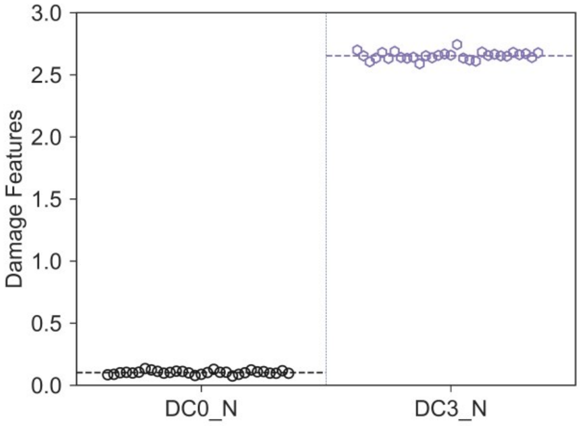


Figure 4.6 - DFs for DC3_N

4.3.4 Influence of Number of Vehicles

As described in previous sections, the DFs fluctuate due to the fact that data from different sets of vehicles are used for baseline and unknown states as well as the influence of operational effect. In this section, the relationship between number of vehicles and the dispersion of DFs is discussed. As shown in Figure 4.7, the coefficient of variation of DFs is plotted against different number of vehicles. It is seen that the coefficient of variation of DFs are higher for all 5 damaged cases when small number of vehicles are used. As the number of vehicles increases, they all decrease gradually. Therefore, it is safe to say that the approach is more robust when larger number of vehicles are used which is very promising for real-life applications with big-data extracted from large number of vehicles. It should also be noted that curves for coefficient of variation of DC1a_N, DC1b_N, DC2a_N and DC2b_N are very close to each other, which are all far from the curve for DC3_N. This is because the first 4 damage cases are all related to stiffness reduction but

DC3_N is about boundary condition change, which shows that the dispersion of DFs is affected by the type of damage.

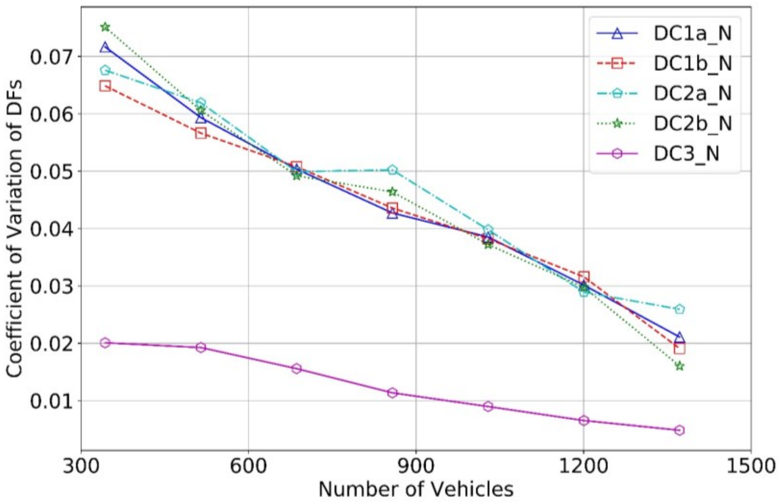
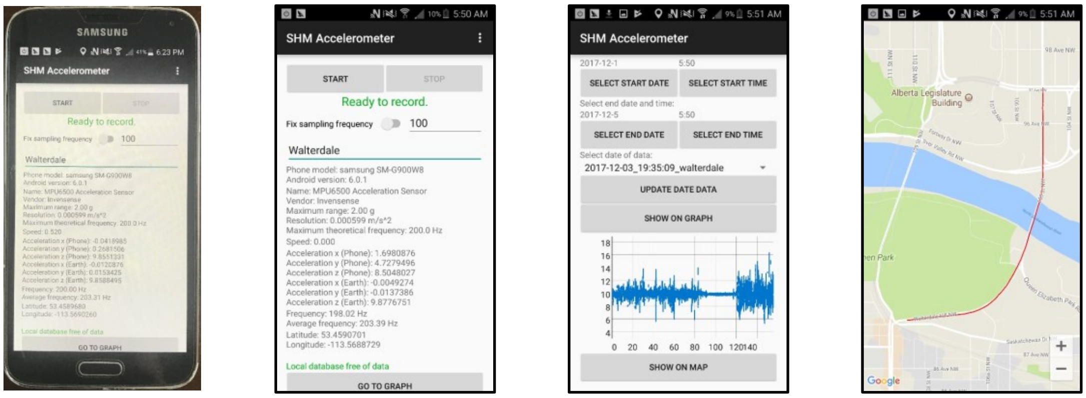


Figure 4.7 - Relationship between coefficient of variation of DFs and number of vehicles

4.4 Experimental Studies

4.4.1 Data Collection App

In order to collect the data smoothly, our research team developed an android app that can collect acceleration data and GPS information simultaneously. The screens of the app are presented in Figure 4.8. The data is collected from the sensors on smartphone and stored remotely on Amazon web service remote database. A Matlab based post-processing program compatible to this app is also developed. This app has the flexibility to set the sampling frequency, collect acceleration data along different axes and record the GPS information along with the vibration information. It also has the ability to visualize the vibration data and GPS information. This app is our first step to apply our techniques onto experiments or real bridges.



Ready to record.

Fix sampling frequency 100

Walterdale

Phone model: samsung SM-G900W8
Android version: 6.0.1
Name: MPU6500 Acceleration Sensor
Vendor: Invensense
Maximum range: 2.00 g
Resolution: 0.000599 m/s^2
Maximum theoretical frequency: 200.0 Hz
Speed: 0.000
Acceleration x (Phone): 1.6980876
Acceleration y (Phone): 4.7279496
Acceleration z (Phone): 8.5048027
Acceleration x (Earth): -0.0049274
Acceleration y (Earth): -0.0137386
Acceleration z (Earth): 9.8776751
Frequency: 198.02 Hz
Average frequency: 203.39 Hz
Latitude: 53.4590701
Longitude: -113.5688729

Local database free of data

2017-12-1 5:50

Select end date and time:
2017-12-5 5:50

Select date of data:
2017-12-03_19:35:09_walterdale

(a) Android app (b) setup screen (c) acceleration data visualization (d) GPS data visualization

Figure 4.8 - Data collection app

4.4.2 Global Coordinates

In practice, the smartphone is likely to be placed in arbitrary orientation in the vehicle. The acceleration data collected from the built-in triaxial accelerometer is relative acceleration regarding to smartphone's local coordinate system. However, in order to identify damage in a bridge, the global vertical acceleration is used. To resolve this issue, the orientation of the smartphone is measured through the gyroscope and magnetometer in combination with accelerometer, and a rotation matrix, R, can be extracted from Android API. The global acceleration can be simply calculated by multiplying the inverse of R to the relative acceleration vector.[117]

4.4.3 Sampling Frequency Correction

It is observed that Android smartphones usually have imperfect sampling frequency, i.e. the sampling intervals would deviate from the preset values. This phenomenon is also reported by other researchers.[41] This not only affects performance but also causes problems to signal processing techniques such as Fourier transform.

In order to study this effect, a simple impact test is conducted on an intact simply supported bridge in the lab, and 10 s acceleration data is recorded using our app. The plot that shows the relationship between time intervals and time is presented in Figure 4.9. It is seen that the time interval fluctuates around the given value, 0.01 s, except for a couple of samples with very large or very small intervals. The average of all the intervals in this period is 0.00994 s and the standard deviation is 0.00055 s.

Since the time interval does not deviate from preset value significantly, to resolve this issue, a sampling frequency correction technique similar to the one proposed in Ozer et al.[41] is used. The corrected sampling frequency is calculated as the multiplication of original sampling frequency and correction coefficient, where correction coefficient is the ratio of the number of data points for target sampling frequency and original sampling frequency. After the correction, the acceleration data is illustrated in Figure 4.10. It is seen that the pattern of acceleration data does not change considerably, but now a variety of signal processing techniques can be used.

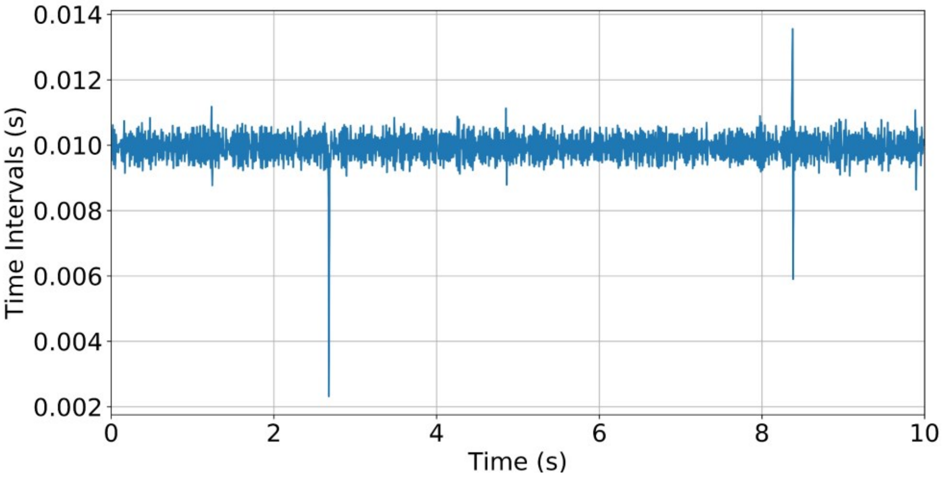


Figure 4.9 - Time intervals versus time

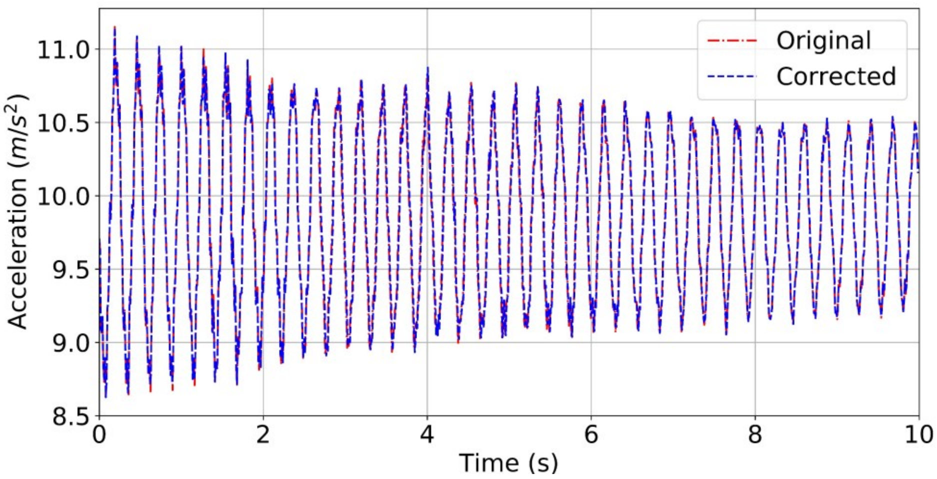


Figure 4.10 - The original and corrected acceleration data

4.4.4 Experimental Setup

In this section, a simply supported bridge as shown in Figure 4.11 is setup in our lab to verify the proposed method experimentally. The bridge deck is made of hot rolled steel W44, which has a yield strength of 250 MPa, and ultimate strength of 310 MPa. The modulus of elasticity of the steel is 200 GPa. The dimensions of the bridge are as follows: length 2 m, width 330 mm and thickness 6.35 mm. Similar to numerical analysis, 5 artificial damage cases grouped in 3 categories are applied to the experimental bridge model:

1) DC1a_E: 15% stiffness reduction at the mid-span; (See Figure 4.12)
2) DC1b_E: 30% stiffness reduction at the mid-span; (See Figure 4.13)
3) DC2a_E: 15% stiffness reduction at the ¼-span. (See Figure 4.12)
4) DC2b_E: 30% stiffness reduction at the ¼-span. (See Figure 4.13)
5) DC3_E: Boundary condition change at both ends (See Figure 4.14)

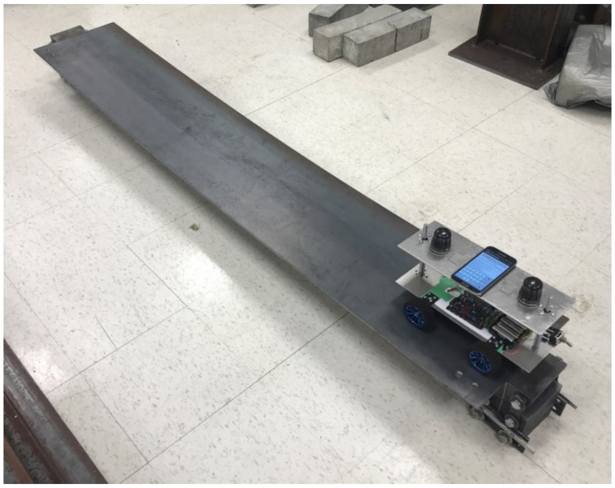


Figure 4.11 - Setup of the lab experiment

In above damage cases, DC1a_E has a 24.8 mm by 250 mm cut centered at mid-span at each side, while in DC2a_E the cut with the same size is centered at 0.5 m from an end. Similarly, a 49.5 mm by 250 mm cut is made at each side of steel bridge as well to simulate 30% stiffness reduction, where the damage at the mid-span is called DC1b_E and at the ¼ span is called DC2b_E. In order to eliminate the effect of mass reduction when area is cut off the bridge, for all stiffness reduction damage cases, steel flat bars with the same size as cut area (see Figure 4.12 and Figure 4.13) are loosely attached to the bridge using hot glue. DC3 is applied by mounting each end to a short I-beam through four bolts.

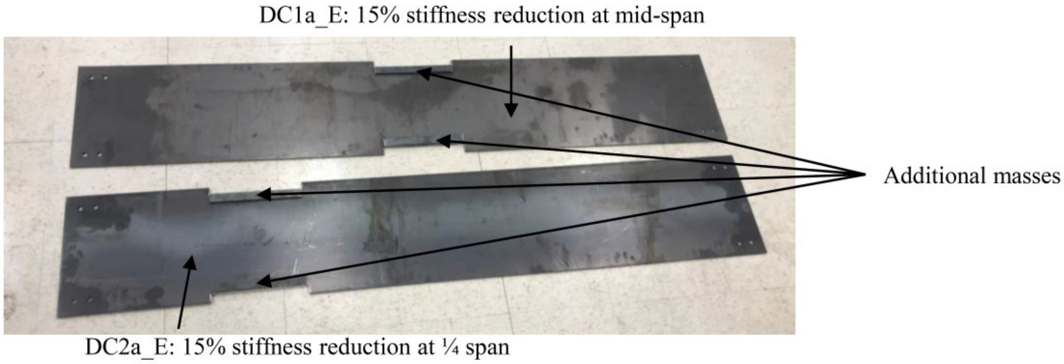

DC1a_E: 15% stiffness reduction at mid-span
Additional masses
DC2a_E: 15% stiffness reduction at ¼ span

Figure 4.12 - Artificial damage applied for DC1a_E and DC2a_E

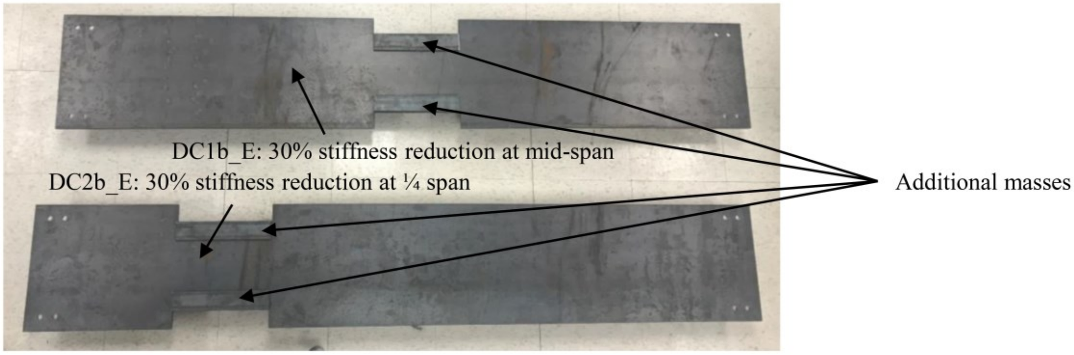

DC1b_E: 30% stiffness reduction at mid-span
DC2b_E: 30% stiffness reduction at ¼ span
Additional masses

Figure 4.13 - Artificial damage applied for DC1b_E and DC2b_E

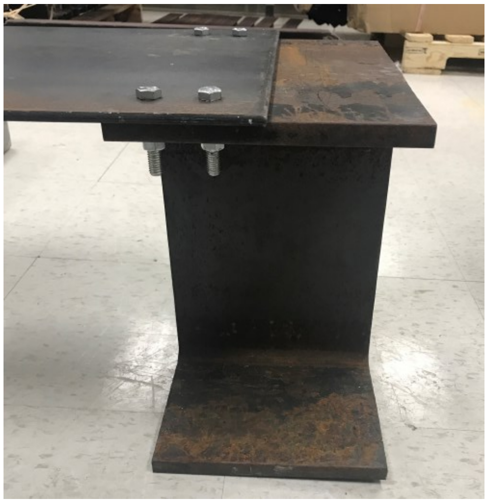


Figure 4.14 - Boundary condition change for DC3_E

A model vehicle is developed to mimic the behavior of real vehicles. This model vehicle can be used to simulate different speed, suspension, weight etc. As illustrated in Figure 4.15, the vehicle is mainly composed of two aluminum plates. The motors and wheels are connected to the bottom plate and controlled by an Arduino programmable board. The whole vehicle is powered by 5 AA batteries. The top plate is connected to the bottom plate through rods, linear bearings and springs to model the suspension system. Two G-Link®-200 wireless accelerometers are mounted at sides of the top plate. Galaxy S5 smartphone is mounted at the center of the top plate. The wireless accelerometers have a sampling frequency of 128 Hz, while smartphone has a sampling frequency of 100 Hz. The average is taken when data are collected from two wireless accelerometers.

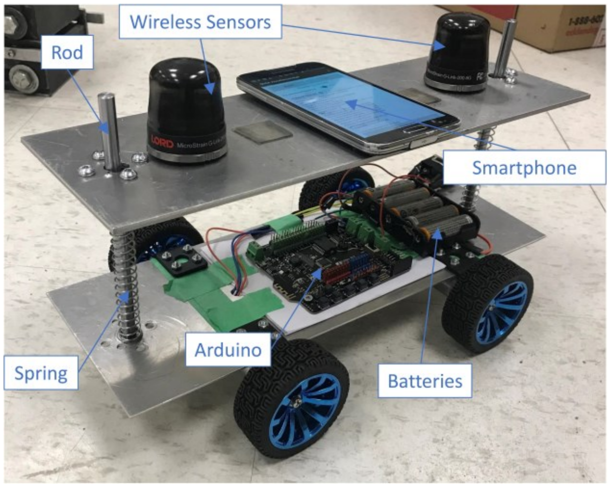


Figure 4.15 - Setup of the model vehicle

As presented in Figure 4.16, to verify the capacity of the smartphone to collect acceleration data, the same impact test as described in section 4.1.2 is conducted on the intact simply supported bridge. The smartphone as well as wireless accelerometer are mounted at the mid-span of the bridge for data collection. The acceleration data measured from both devices are presented in Figure 4.17(a), and Fourier transform are conducted and shown in Figure 4.17(b). It is seen that smartphone can measure the acceleration very accurately and can identify the modal frequencies of the bridge in the given range successfully, which are 3.71 Hz, 14.9 Hz, 33.4 Hz.

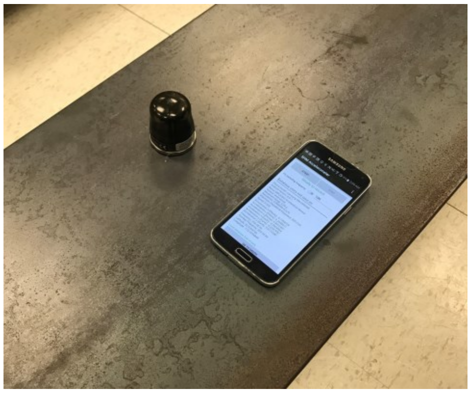
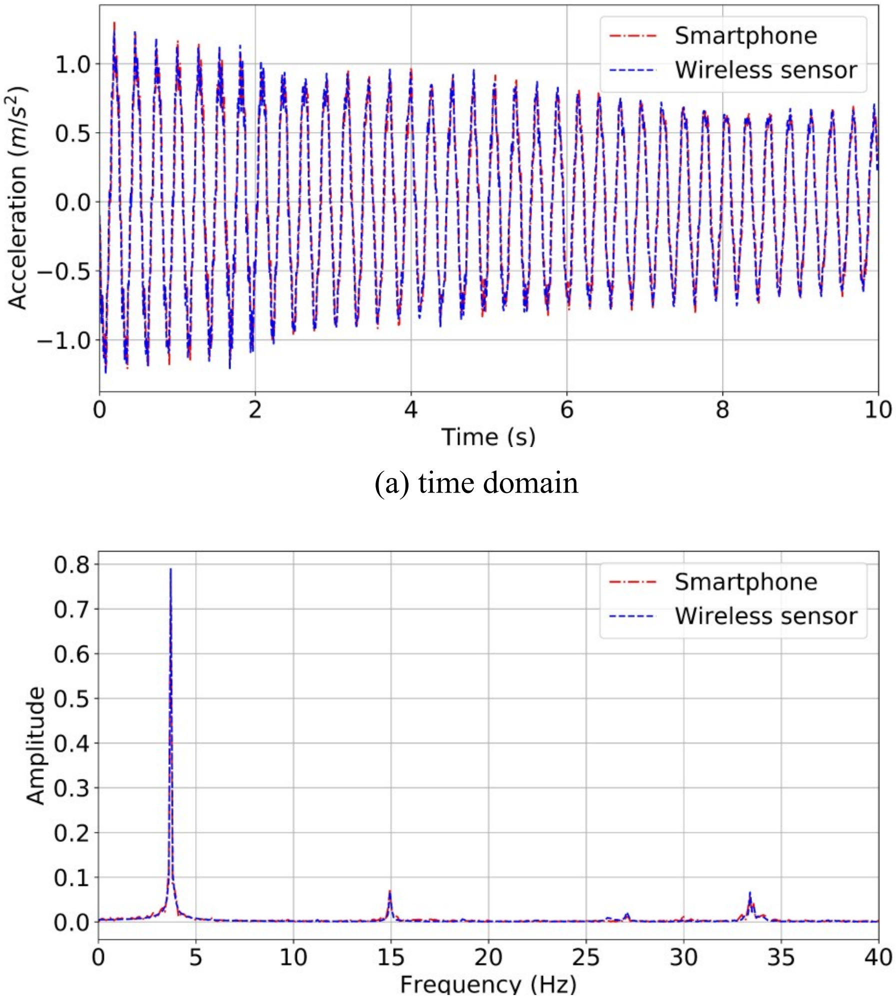


Figure 4.16 - Placement of smartphone and wireless accelerometer for impact test



(a) time domain

(b) frequency domain

Figure 4.17 - Comparison of smartphone and wireless accelerometer

In the model vehicle, the springs are replaceable, and the spring constant is changed among 5 values: 155 N/m, 288 N/m, 425 N/m, 615 N/m, and 726 N/m. The weight of the model vehicle can be changed by placing additional masses on the top plate, and it is changed among five levels: 0.898 kg, 0.988 kg, 1.084 kg, 1.170 kg, 1.270 kg. The speed can be changed by programming the Arduino board, and it is varied among 3 values: 0.25 m/s, 0.33 m/s, 0.40 m/s. In addition, each test is repeated 3 times for each model vehicle configuration.

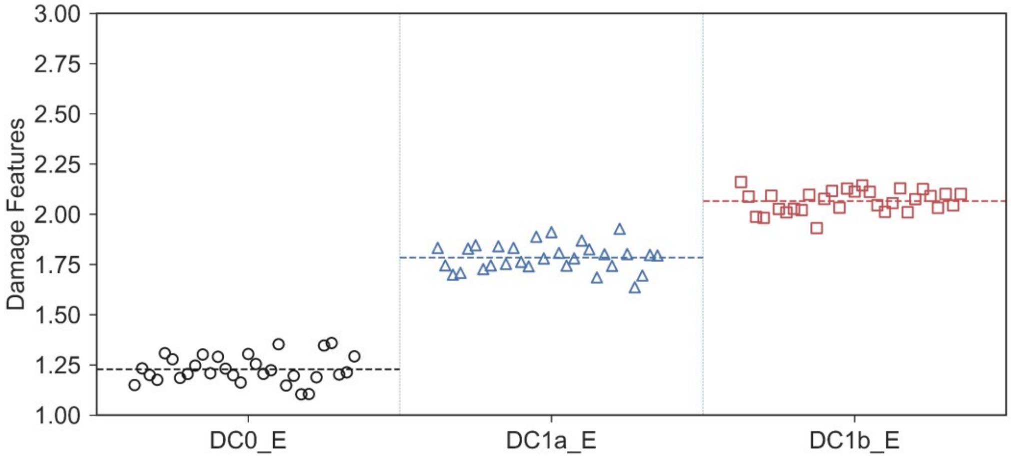
Combining all the parameter changes and repeated tests, there are in total 5×5×3×3 = 225 tests for each bridge state. To avoid using exactly the same set of vehicles for different damage cases, 50% data entries of the vehicle pool are randomly sampled. Similar to the numerical analysis, 50% of the tests on intact bridge are reserved as baseline case and the remaining 50% are for validation. For other damage cases, only 50% of the tests are selected for damage detection. In this way, the sets of vehicles would be different for different damage cases, which is close to real situation because it is impossible to have exactly the same set of vehicles passing across bridge during different periods. However, since they are sampled from the same set of vehicle configurations, they should follow similar distributions. The sampling process has randomness. Therefore, in order to verify the robustness, the method is implemented 30 times with different samples included.

4.4.5 Experimental Results

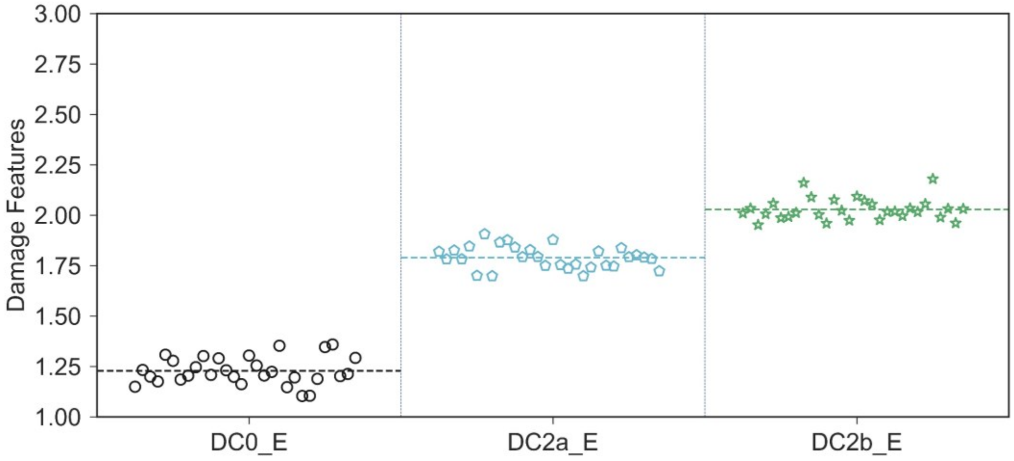
The results obtained using the wireless sensors and the smartphone are presented in Figure 4.18 and Figure 4.19, respectively. Each point in the figures represents the DF for a complete implementation of the algorithm from one sampling process, and therefore there are in total 30 points for each damage cases.

Similar to numerical analysis, the horizontal dashed lines stand for the average DF over all 30 trials. In Figure 4.18, it is shown that DFs for all damage cases are higher than DC0_E, which indicates that the damage is successfully identified. Among these damaged cases, DFs for DC1a_E and DC1b are about the same level as DC2a_E and DC2b_E. DFs for DC1b_E and DC2b_E are higher than DC1a_E and DC2a_E, because 30% stiffness reduction is more severe than 15% reduction. DC3_E has the highest DFs, which makes sense because boundary condition change is the most severe damage. It should be noted that the difference of DFs among different damage cases is not as significant as in numerical analysis. The reason is that the vibration of the vehicle and bridge in experiments is not as ideal as in numerical models. The existence of noise in frequency spectrums makes the extracted features harder to distinguish. In spite of that, the variation of DFs in a single damage case is still within an acceptable range so that different damage cases are separable. The observations above show that the approach is robust when different sets of vehicles following the same distribution are used for damage detection.

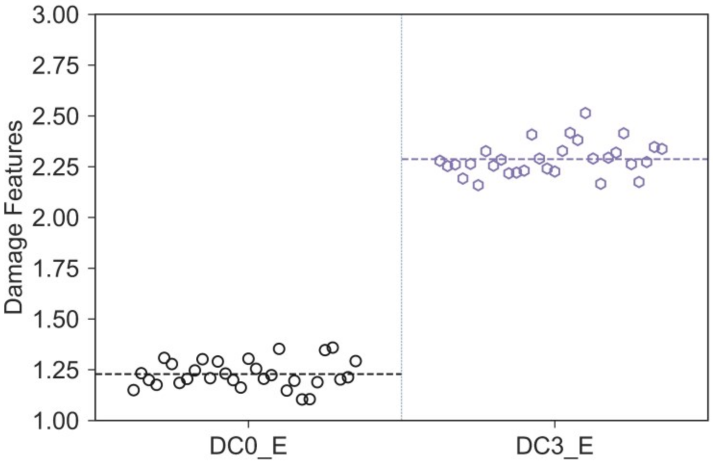
Comparing DFs obtained using wireless accelerometers and the smartphone, it is seen that the DFs from both devices are very similar, but the fluctuation of DFs from smartphone data is generally larger than from sensors. The reason for this is that the location of wireless accelerometers is different than the smartphone. It can be seen in Figure 4.15 that smartphone is placed at the center of the top plate, and therefore the vibration of the top plate is likely to be included in smartphone data.



(a) DC1a_E and DC1b_E

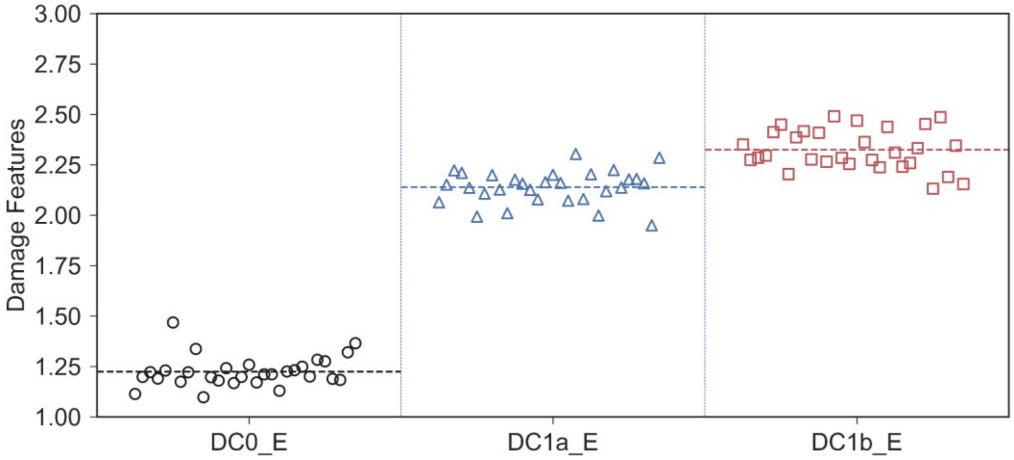


(b) DC2a_E and DC2b_E

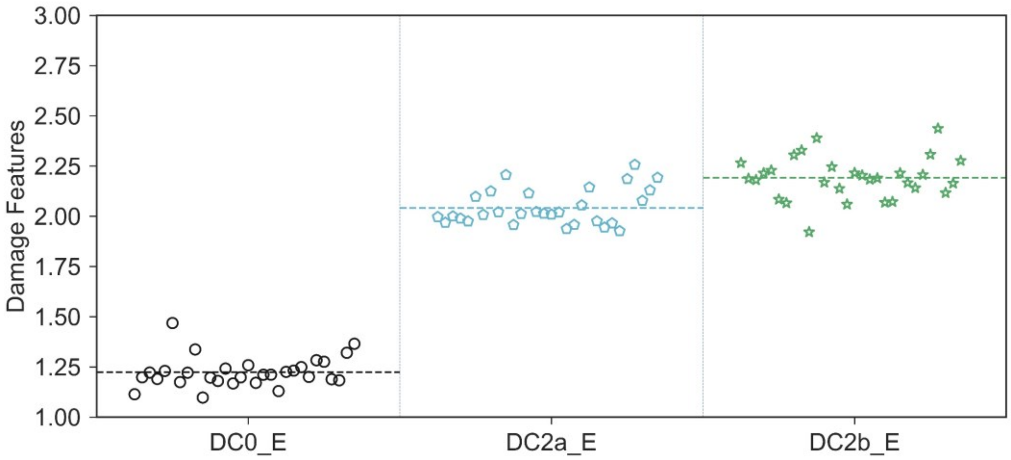


(c) DC3_E

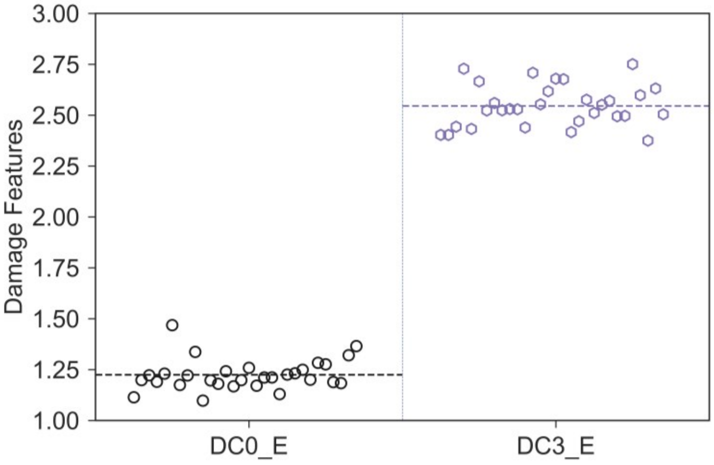
Figure 4.18 - DFs obtained using the wireless accelerometers



(a) DC1a_E and DC1b_E



(b) DC2a_E and DC2b_E



(c) DC3_E

Figure 4.19 - DFs obtained using the smartphone

4.5 Discussion

It is anticipated that the expected outcomes of the proposed research program will be useful for developing smart transportation infrastructure systems using connected vehicles and other mobile sensing technologies. Although the proposed methodology of utilizing crowdsensing data from smartphones in vehicles has shown promising results, there are still challenges that need to be resolved before routine real-life applications.

First, the increased uncertainties on the collected data introduced by the random placement of the smartphone in the vehicle is an important factor. The influence of location and direction of the smartphone should be investigated and eliminated. One possible solution is to use the built-in gyroscope and magnetometer data to aid the accelerometer and convert the local acceleration data to global vertical acceleration, which is already implemented in our Android application. Also, active filtering techniques could be applied to identify the validity of the data, and unstable data could be discarded automatically.

Besides, this proposed method requires the smartphone to measure the acceleration and transmit data to remote database. To minimize the battery and data usage in real-life applications, the method presented in this chapter only requires the transmission of a small number of features extracted from the data. Also, this issue can be overcome by using GPS data. For example, the data recording process will not be activated until the GPS tells that the vehicle is about to pass a bridge. The data accessibility is another issue that should be resolved for future real-life implementations. Currently, we rely on volunteers to install the app and ask their permission to collect and transmit data. In the future, the method can be integrated into commercial apps such as Google map to increase the volume of data or apps that are developed by municipalities and other local authorities for the use of the general public.

In addition, the challenges from traditional indirect health monitoring problem such as road profile roughness, limited vehicle-bridge interaction time and environmental effects, will affect the method as well and should be studied. In practice, to reduce the influence of environmental and operational, the method can also be used with other monitoring data such as weather information or temperature. Also, it should be noted that the method neither requires the collection window to be continuous nor restrict the length of it as long as a large number of vehicles passing through the bridge within that window. In practice, the collection window can be chosen as large as possible to average out the environmental and operational effects.

4.6 Conclusions

This chapter proposes a novel methodology for damage detection of bridges using crowdsensing data collected from a large number of vehicles passing through a bridge at different times. The method utilizes MFCCs and KL divergence to detect damage on the bridge. Both numerical analysis and experiments are conducted to verify this method. The crowdsensed vehicles are simulated by changing the configurations of a single vehicle (i.e. weight, suspension spring and the speed of the vehicle). In the experiments, data collection using both wireless sensors and smartphones are investigated. The following main conclusions are obtained from our studies:

1) The proposed DFs calculated based on crowdsensing data from smartphones in vehicles successfully identify the existence of damage and extract useful information about severity.
2) One of the main advantages of the proposed method is that the requirement of knowing vehicle configuration or using the same vehicle for damage detection is eliminated.
3) It is shown that smartphones can potentially be used to indirectly detect damage in bridges.

In this chapter, the smartphones are considered as sensors for data collection when dealing with a large number of vehicles. It has to be acknowledged that smartphones have several defects compared to wireless sensors, such as lower sampling frequency and lower resolution, but they are widely used and have access to internet. This chapter has demonstrated that the smartphone can give just as good results as wireless sensors for indirect bridge health monitoring. With the help of crowdsensing with smartphones, this method has the potential to monitor a population of bridges simultaneously and in almost real time.

CHAPTER 5: DRIVE-BY BRIDGE MODE SHAPE IDENTIFICATION

5.1 Overview

One way to assess the condition of a bridge through mobile sensing is based on modal characteristics. In this type of methods, the data collected from moving vehicles are used to estimate the frequencies and mode shapes of the bridge, which then can be used for further assessment of the bridge condition. The accuracy of mode shape identification is the key to proper condition assessment of bridges. A through review on the methods of mode shape identification from moving vehicles can be found in section 2.2.

As identified in several prior studies [58, 60, 63], the challenges of mode shape identification using moving vehicles mainly include the following: (*i*) the accuracy of mode shape identification is adversely affected by the limited vehicle bridge interaction time, as indicated in Malekjafarian and Obrien [63]; (*ii*) Since the sensor usually stays in one location for a very short period, high accuracy in mode shape identification is only achievable by carefully accounting for vehicle-bridge interaction, and by using multiple mobile sensors [62].

This chapter proposes a new method for mode shape identification from moving vehicles by converting the problem into a matrix completion problem, and by implementing a "soft imputing" algorithm introduced by Mazumder et al. [118] to iteratively fill/complete the matrix. There are several advantages with respect to previous methods: (*i*) Since the method does not explicitly use frequency domain information for system identification, it can work at traffic speeds with limited vehicle-bridge-interaction time. (*ii*) Owing to the mapping step in the method, the method can achieve high accuracy with a limited number of sensors. (*iii*) Since the soft imputing algorithm is non-parametric, no information about the dynamic system is required.

5.2 Methodology

5.2.1 Assumptions

In the present study, the mobile sensors are assumed as moving measurement point (MMP) on the bridge. The influence of the suspension system is ignored. In addition, the deformation of the wheels and tires are ignored for simplicity as well. In practice, the effect of the vehicle's dynamic system can be removed, for example, by using its empirical transfer function. It is assumed that recorded data are accelerations, which are integrated twice to obtain the displacements, and serve as inputs to the matrix completion algorithm. The speeds of traversing vehicles are assumed to be the identical, and constant. It is noted that the speed assumptions are made for simplicity, and the method devised in the present study can be generalized in a straightforward manner.

5.2.2 Mapping Moving Observations to Fixed Observations

Two terms are defined first—namely, moving measurement point (MMP) and virtual fixed points (VFP). Each MMP represents a mobile sensor on the moving vehicle; and each VFP is a virtual sensor placed at a fixed location. There is no direct measurement at VFPs. As shown in Figure 5.1, in the proposed method, the observations from MMPs are first mapped to the VFPs using linear interpolation.

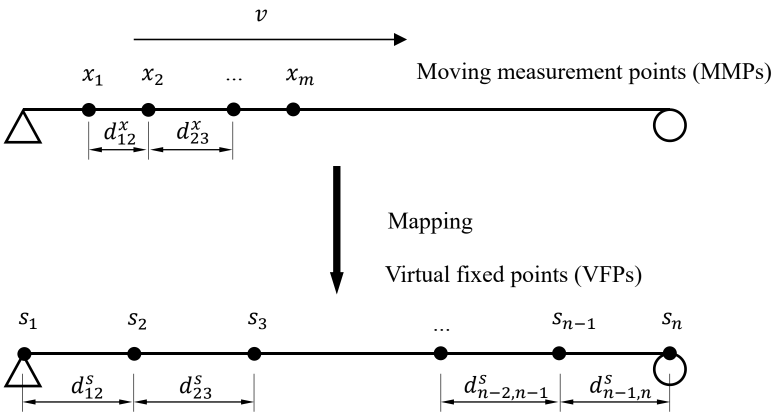


Figure 5.1 - Mapping moving measurement points to fixed points

Considering the modal superposition, the vertical displacement $y(x_i(t),t)$ of the i^{th} MMP can be expressed as

$$y(x_i(t),t) = \sum_{k=1}^{n_d} \phi_k(x_i(t)) q_k(t) \tag{5.1}$$

where $\phi_k(x_i(t))$ is the value of k^{th} mode shape at location $x_i(t)$, and $q_k(t)$ is the generalized coordinate for k^{th} mode. The Eq. (5.1) can be written in a matrix form by applying to all MMPs, as seen in Eqs. (5.2) and (5.3) below.

$$\begin{pmatrix} y(x_1(t),t) \\ y(x_2(t),t) \\ \vdots \\ y(x_m(t),t) \end{pmatrix} = \begin{bmatrix} \phi_1(x_1(t)) & \phi_2(x_1(t)) & \cdots & \phi_{n_d}(x_1(t)) \\ \phi_1(x_2(t)) & \phi_2(x_2(t)) & \cdots & \phi_{n_d}(x_2(t)) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(x_m(t)) & \phi_2(x_m(t)) & \cdots & \phi_{n_d}(x_m(t)) \end{bmatrix} \begin{pmatrix} q_1(t) \\ q_2(t) \\ \vdots \\ q_{n_d}(t) \end{pmatrix} \tag{5.2}$$

$$\mathbf{Y}(t) = \mathbf{\Phi}(t)\mathbf{Q}(t) \tag{5.3}$$

In the equation above, $\mathbf{Y}(t)$ represents the displacement vector for all MMPs, $\mathbf{Q}(t)$ is the generalized coordinates for all modes at time *t*, and $\mathbf{\Phi}(t)$ is the shape function at the location of MMPs at time *t*. It should be noted that $\mathbf{\Phi}(t)$ is not the mode shape matrix of the bridge since it is dependent on the time.

To obtain the mode shape, the shape function value at location *x* can be projected to *n* base functions through Eq. (5.4), where *n* is the number of VFPs.

$$\phi_k(x) = \sum_{j=1}^{n} a_{jk} l_j(x) \tag{5.4}$$

The proposed $l_j(x)$ is given in Eq. (5.5). Unlike in [59], this study uses linear interpolation from adjacent fixed points instead of all the points because the interpolation using all VFPs has a very high order, which can result in a large error when being inversed.

$$l_j(x)=\begin{cases} \dfrac{x-s_{j-1}}{s_j-s_{j-1}} & \mathrm{s}_{j-1}\leq x\leq s_j \\ \dfrac{x-s_{j+1}}{s_j-s_{j+1}} & \mathrm{s}_j< x\leq s_{j+1} \\ 0 & \text{otherwise} \end{cases} \tag{5.5}$$

Substituting the location of one VFP, s_p, into Eq. (5.4) results in the following equation,

$$\phi_k(s_p)=\sum_{j=1}^{n} a_{jk} l_j(s_p)=a_{pk}\frac{s_p-s_{p-1}}{s_p-s_{p-1}}=a_{pk} \tag{5.6}$$

Therefore, Eq. (5.4) can be written as in Eq. (5.7) for the location of MMP *i*,

$$\phi_l(x_i(t))=\sum_{j=1}^{n}\phi_k(s_j)l_j(x_i(t)) \tag{5.7}$$

Writing Eq. (5.7) in matrix form for all modes, the Eqs. (5.8) and (5.9) can be obtained as follows,

$$\begin{aligned}\begin{bmatrix} \phi_1(x_1(t)) & \phi_2(x_1(t)) & \cdots & \phi_{n_d}(x_1(t)) \\ \phi_1(x_2(t)) & \phi_2(x_2(t)) & \cdots & \phi_{n_d}(x_2(t)) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(x_m(t)) & \phi_2(x_m(t)) & \cdots & \phi_{n_d}(x_m(t)) \end{bmatrix} &= \begin{bmatrix} l_1(x_1(t)) & l_2(x_1(t)) & \cdots & l_n(x_1(t)) \\ l_1(x_2(t)) & l_2(x_2(t)) & \cdots & l_n(x_2(t)) \\ \vdots & \vdots & \ddots & \vdots \\ l_1(x_m(t)) & l_n(x_m(t)) & \cdots & l_n(x_m(t)) \end{bmatrix} \\ &\times \begin{bmatrix} \phi_1(s_1) & \phi_2(s_1) & \cdots & \phi_{n_d}(s_1) \\ \phi_1(s_2) & \phi_2(s_2) & \cdots & \phi_{n_d}(s_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(s_n) & \phi_2(s_n) & \cdots & \phi_{n_d}(s_n) \end{bmatrix}\end{aligned} \tag{5.8}$$

$$\mathbf{\Phi}(t)=\mathbf{L}(t)\mathbf{\Phi}^s \tag{5.9}$$

In Eq. (5.9), the right side is the multiplication of two matrices. The first one, $\mathbf{L}(t)$, is the mapping function converting the displacement from MMPs to VFPs, and the second one is the shape function at *n* VFPs. The second matrix will not change with time and is the mode shape of the bridge. Substituting Eq. (5.9) into Eq. (5.3) yields to the following relationship,

$$\mathbf{Y}(t) = \mathbf{\Phi}(t)\mathbf{Q}(t) = \mathbf{L}(t)\mathbf{\Phi}^{s}\mathbf{Q}(t) \tag{5.10}$$

Multiplying Eq. (5.10) by the inverse of $\mathbf{L}(t)$ produces

$$\mathbf{L}^{-1}(t)\mathbf{Y}(t) = \mathbf{\Phi}^{s}\mathbf{Q}(t) \tag{5.11}$$

It should be noted here that if $m \neq n$, i.e. $\mathbf{L}(t)$ is not a square matrix, then the pseudoinverse should be evaluated instead of the inverse. $\mathbf{L}^{-1}(t)\mathbf{Y}(t)$ is a vector with *m* elements, and the following matrix **D** in Eq. (5.12) is defined with columns representing time steps, where *K* is the total number of time steps, and rows representing VFPs.

$$\begin{aligned}\mathbf{D} &= \left[\mathbf{L}^{-1}(t_1)\mathbf{Y}(t_1) \quad \mathbf{L}^{-1}(t_2)\mathbf{Y}(t_2) \quad \cdots \quad \mathbf{L}^{-1}(t_K)\mathbf{Y}(t_K)\right] \\ &= \mathbf{\Phi}^{s}\left[\mathbf{Q}(t_1) \quad \mathbf{Q}(t_2) \quad \cdots \quad \mathbf{Q}(t_K)\right]\end{aligned} \tag{5.12}$$

In practice, the displacement is difficult to measure. Therefore, in this study, the acceleration data is numerically integrated twice to obtain the displacement.

5.2.3 Valid Data Points

Even though the inverse of the mapping function matrix $\mathbf{L}(t)$ is applied to all MMPs and VFPs, we observe the errors between the converted displacements at VFPs and the ground truth are small only when there are at least two MMPs in the region formed by two VFPs adjacent to the given VFP. We define the mapping within this range as valid and consider the other elements as invalid or missing. An example with 4 MMPs is presented in Figure 5.2. At certain time *t*, there are two MMPs between VFPs, s_2 and s_3, so these two VFPs have valid data at time *t*. Since all other VFPs do not have at least two MMPs in their adjacent segments, data for these VFPs are invalid and are considered as missing. The reason behind requiring at least two MMPs is that the linear interpolation is used to represent the displacement at an MMP using two VFPs. At least two MMPs are required to fully determine the inverse function of the mapping.

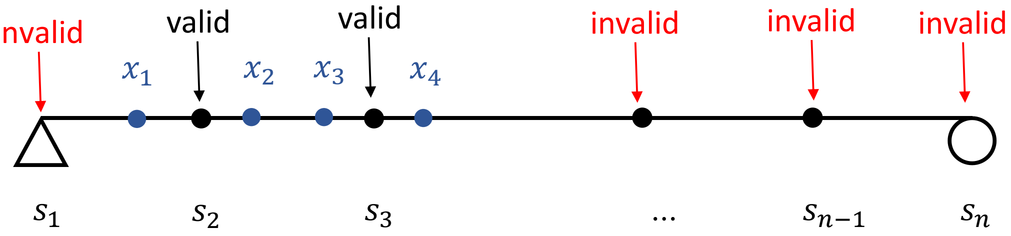


Figure 5.2 - Illustration of the valid and invalid data points

Since the invalid data cannot be used for system identification, the matrix **D** will have many missing values. A typical matrix **D** is presented in Figure 5.3 for visualization purpose. Figure 5.3 illustrates that as the time goes on and the MMPs move to the right, VFPs at the right side start to get valid data.

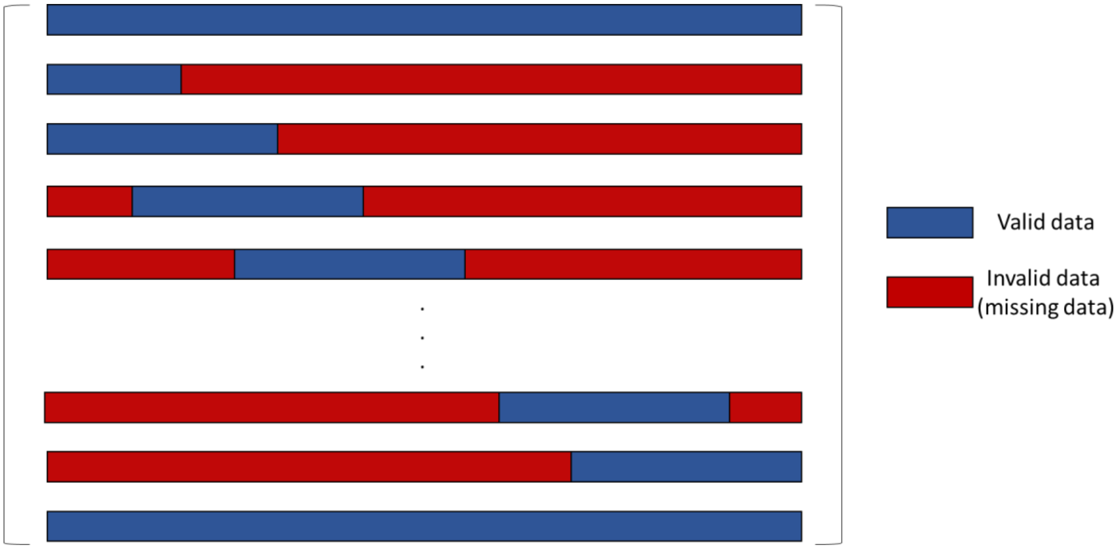


Figure 5.3 - Visualization of a typical **D** matrix

5.2.4 Matrix Completion using Soft Imputing

System identification techniques cannot be directly applied to a matrix with missing values. To overcome this issue, a technique called soft imputing is applied to look for patterns in the data and fill the matrix using an expectation maximization framework. The reason we choose soft imputing algorithm for matrix completion is that there is no need to assume the rank of the estimated matrix. Also, the Singular Value Decomposition (SVD) within this algorithm can guarantee the orthogonality of the factorized matrices, which is a property of mode shapes. This algorithm has a time complexity of O(n) which is also efficient in implementation [118].

In this framework, the missing data are assumed and updated in an iterative process. Consider a matrix $\mathbf{P}_{\mathbf{\Omega}}(\mathbf{D})$ with element defined as below,

$$\mathbf{P}_{\mathbf{\Omega}}(\mathbf{D})(i,j)=\begin{cases}\mathbf{D}(i,j) & (i,j)\in\Omega\\ 0 & (i,j)\notin\Omega\end{cases} \tag{5.13}$$

where Ω is the set including the indices of all valid data. This matrix keeps the values of all the valid data and set all the missing data to zero. Similarly, $\mathbf{P}_{\mathbf{\Omega}}^{\perp}(\mathbf{D})$ is defined as complementary projection of $\mathbf{P}_{\mathbf{\Omega}}(\mathbf{D})$ where $\mathbf{P}_{\mathbf{\Omega}}(\mathbf{D})+\mathbf{P}_{\mathbf{\Omega}}^{\perp}(\mathbf{D})=\mathbf{D}$.

Assuming the estimated matrix is **Z,** all the elements in **Z** are valid. There are two objectives for the optimization algorithm to fill the matrix **D**. First, $\mathbf{P}_{\mathbf{\Omega}}(\mathbf{Z})$ as defined as elements in **Z** corresponding to the valid elements in **D** should be as close to $\mathbf{P}_{\mathbf{\Omega}}(\mathbf{D})$ as possible. Second, the rank of the estimated matrix **Z** should be as low as possible. This is a regularization term based on the assumption that using fewer modes to represent the vibration is expected to be more accurate than using more modes. According to the descriptions above, the following objective function is defined,

$$\underset{Z}{\text{minimize}}\quad f_{\lambda}(\mathbf{Z})=\frac{1}{2}\left\|\mathbf{P}_{\mathbf{\Omega}}(\mathbf{D})-\mathbf{P}_{\mathbf{\Omega}}(\mathbf{Z})\right\|_{n}^{2}+\lambda_{i}\times\left\|\mathbf{Z}\right\|_{*} \tag{5.14}$$

where $\|\cdot\|_{n}$ represents the nuclear norm of a matrix and $\|\cdot\|_{*}$ is the rank of the matrix. λ_i is the regularization coefficient which plays an important role in the algorithm. This optimization problem can be solved iteratively, as presented in Figure 5.4. At the beginning, the $\mathbf{P}_{\mathbf{\Omega}}(\mathbf{Z}_{\text{old}})$ is set to $\mathbf{P}_{\mathbf{\Omega}}(\mathbf{D})$, and $\mathbf{P}_{\mathbf{\Omega}}^{\perp}(\mathbf{Z}_{\text{old}})$ is initialized randomly. λ_i are chosen in a descending order, and λ_1 should be smaller than the maximum singular value of $\mathbf{Z}_{\text{old}}$. As stated in Oshima et al. [118], for a given λ_i, the solution $\mathbf{Z}_{new}$ for the above optimization problem is given by the following iterative process.

$$\begin{gathered}\mathbf{Z}_{new} = S_{\lambda_i}(\mathbf{P}_{\boldsymbol{\Omega}}(\mathbf{D}) + \mathbf{P}_{\boldsymbol{\Omega}}^{\perp}(\mathbf{Z}_{old})) = \mathbf{U}\boldsymbol{\Sigma}_{\lambda_i}\mathbf{V} \\ \boldsymbol{\Sigma}_{\lambda_i} = \mathrm{diag}[(d_1 - \lambda_i)_+, (d_2 - \lambda_i)_+, \cdots, (d_r - \lambda_i)_+], r = \min(m, n)\end{gathered} \tag{5.15}$$

where $\mathbf{U}$, $\boldsymbol{\Sigma}$, $\mathbf{V}$ are determined by Singular Value Decomposition (SVD) of a new matrix $\mathbf{P}_{\boldsymbol{\Omega}}(\mathbf{D}) + \mathbf{P}_{\boldsymbol{\Omega}}^{\perp}(\mathbf{Z}_{old})$, and specifically $\boldsymbol{\Sigma}$=diag$(d_1, d_2, \cdots d_r)$ consists of the signular values. The matrix $\boldsymbol{\Sigma}_{\lambda_i}$ is the soft-thresholded matrix of $\boldsymbol{\Sigma}$, in which the notation $(\cdot)_+$ means the larger of the given value and 0. Repeating the process of replacing the missing value with the current guess and updating through Eq. (5.15), the matrix can be imputed after converging. As shown in Figure 5.4, the algorithm will generate a series of $\mathbf{Z}_{\lambda_i}$ and the optimal one determined by the objective function value, i.e. Eq. (5.14), as well as engineering judgement, can be used for further analysis. The optimal Z_{λ_i} is termed as $\mathbf{Z}_{opt}$.

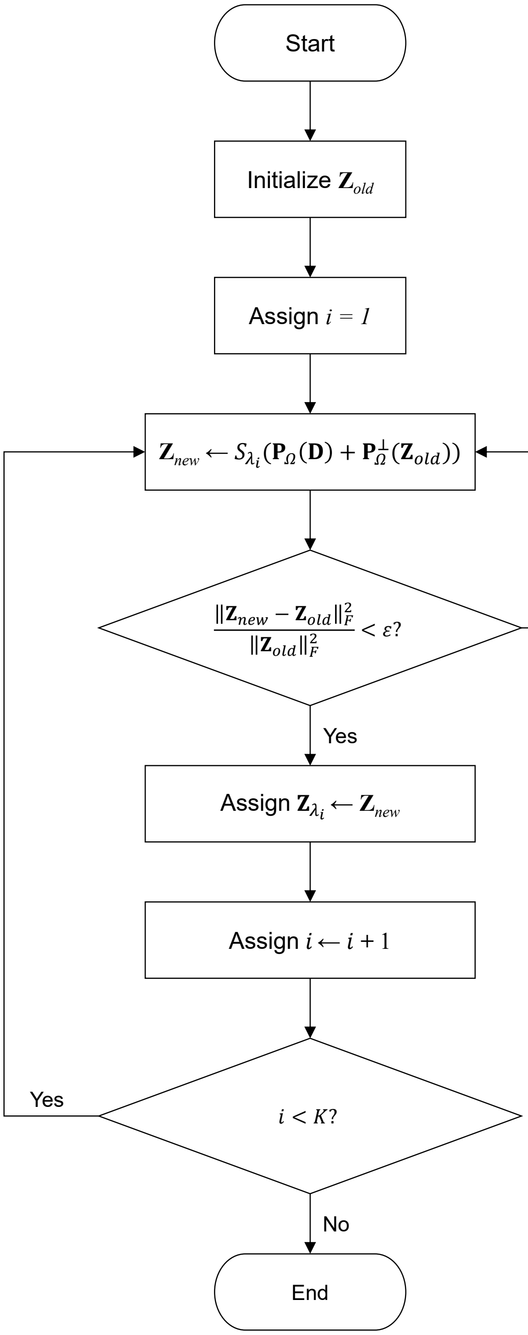

Start
Initialize $\mathbf{Z}_{old}$
Assign $i = 1$
$\mathbf{Z}_{new} \leftarrow S_{\lambda_i}(\mathbf{P}_{\Omega}(\mathbf{D}) + \mathbf{P}_{\Omega}^{\perp}(\mathbf{Z}_{old}))$
Assign $\mathbf{Z}_{new} \leftarrow \mathbf{Z}_{old}$
$\frac{\|\mathbf{Z}_{new} - \mathbf{Z}_{old}\|_F^2}{\|\mathbf{Z}_{old}\|_F^2} < \varepsilon?$
No
Yes
Assign $\mathbf{Z}_{\lambda_i} \leftarrow \mathbf{Z}_{new}$
Assign $i \leftarrow i + 1$
$i < K?$
Yes
No
End

Figure 5.4 - Soft imputing algorithm for matrix completion

5.2.5 System Identification

After obtaining the optimal estimated matrix $\mathbf{Z}_{\text{opt}}$, different system identification techniques can be applied to retrieve the mode shapes of the bridge. In this study, SVD is used as it reserves the orthogonality of mode shapes in the decomposition. The estimated matrix $\mathbf{Z}_{\text{opt}}$ has the same dimension as the matrix **D** which is $n \times K$. The SVD of $\mathbf{Z}_{\text{opt}}$ can be expressed as,

$$\mathbf{Z}_{opt} = \mathbf{U}_{opt}\mathbf{\Sigma}_{opt}\mathbf{V}_{opt}^{T} \tag{5.16}$$

Comparing Eq. (5.16) with Eq. (11), it is inferred that $\mathbf{U}_{opt}$ relates to the mode shapes of the bridge since they are both orthogonal. By normalizing the $\mathbf{U}_{opt}$, the mode shape of the bridge will be extracted.

5.3 Numerical Verification

5.3.1 Numerical Setup

To verify the proposed method, numerical analyses are conducted using Abaqus on a simply supported bridge, as illustrated in Figure 5.5. The length of the bridge is 40 m. The rectangular cross section of the bridge has a width of 3 m and a height of 1.5 m. The bridge is made of concrete, which has a density of 2400 kg/m^3 and elastic modulus of 27.5 GPa. The first three frequencies of the bridge are 1.44 Hz, 5.76 Hz and 12.95 Hz. Two MMPs are placed on the bridge with a distance of 2.5 m to represent the wheels of a single vehicle. A constant speed of 40 km/h is assigned to all the MMPs, and the linear implicit dynamic analysis is conducted with contacts among MMPs and the bridge. Therefore, the analysis ends when the foremost MMP reaches the right end of the bridge. The sampling frequency of acceleration is 200 Hz.

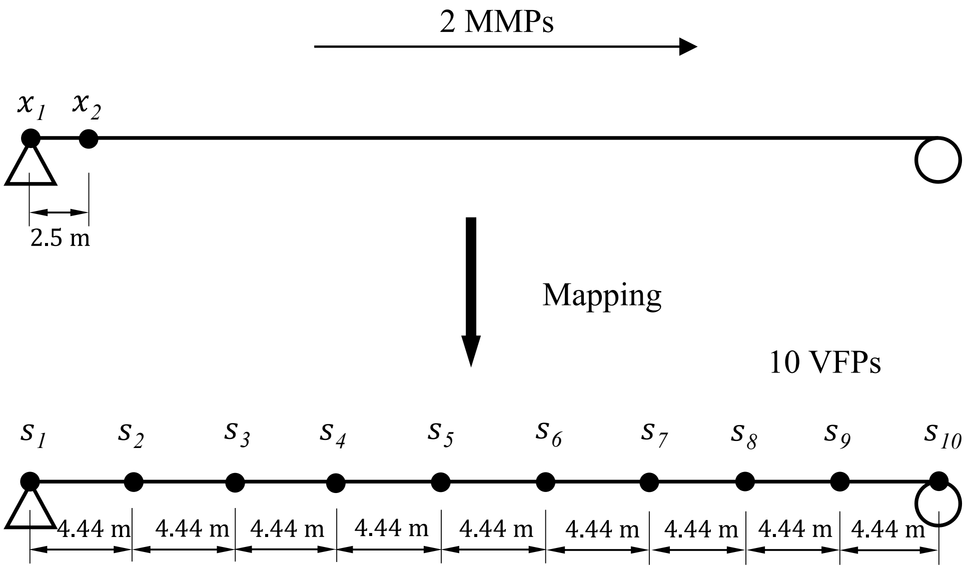


Figure 5.5 - Setup of the numerical analysis

5.3.2 Results and Analysis

Figure 5.6 presents the measured accelerations, integrated velocities, and integrated displacements for 2 MMPs. As described in section 5.2, acceleration is considered as the measurement from the analysis. In this study, trapezoidal numerical integrations are applied twice to obtain the displacements, which are the input to the mapping and matrix completion procedures. In Figure 5.6, it is seen that the integrated displacements are not stationary. They follow the pattern of the deflection of a simply supported bridge where the absolute displacement is the highest near mid-span, but they are in a vibration form due to the moving of the vehicle.

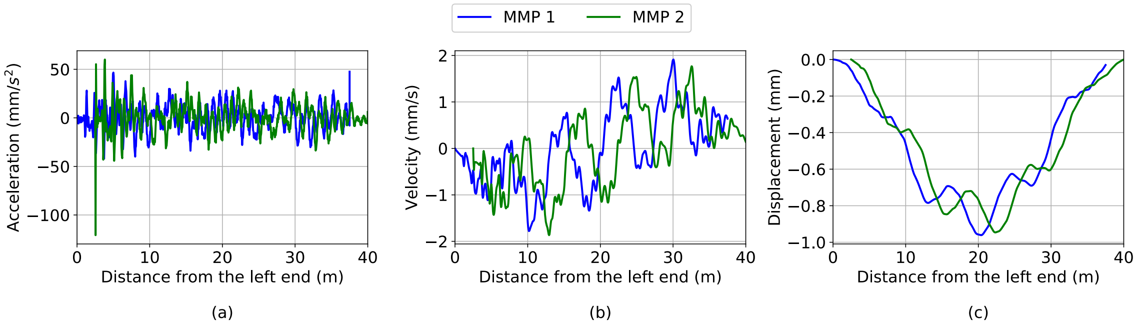
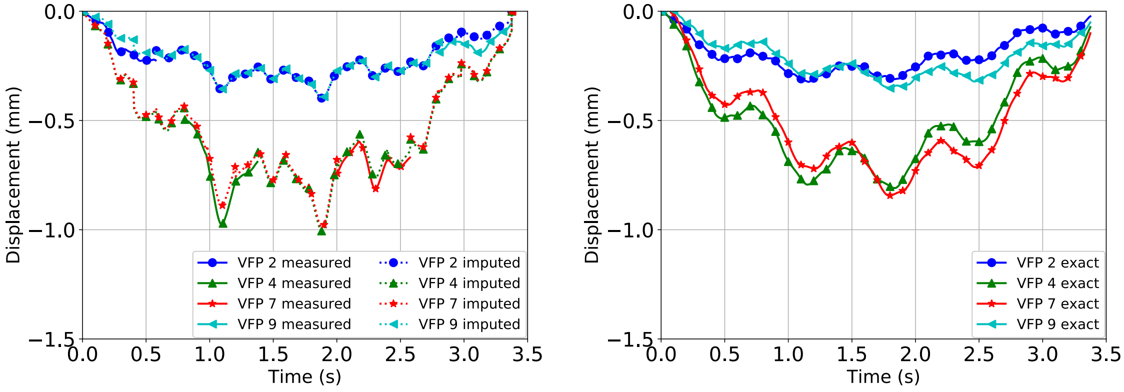


Figure 5.6 - Measured acceleration and integrated velocity and displacement for 2 MMPs: (a) measured acceleration; (b) integrated velocity; (c) integrated displacement

The integrated displacements are then mapped to 10 VFPs in order to construct displacements at those VFPs following the procedure described in section 2.2. As mentioned earlier, it is assumed that the mapping is only valid when there are more than two MMPs in a VFP's adjacent segments, as demonstrated in Figure 5.2. The invalid data in the matrix **D** are considered as missing data. Then, the matrix completion algorithm described in section 2.4 is applied to fill these *missing values.* The maximum singular value of the initialized ***Z*** is 0.02**.** A series of 11 λ values are given in descending order to the algorithm. λ values chosen are 1×10^{-2}, 5×10^{-3}, 1×10^{-3}, 5×10^{-4}, 1×10^{-4} 5×10^{-5}, 1×10^{-5}, 5×10^{-6}, 1×10^{-6}, 5×10^{-7} and 1×10^{-7}*,* respectively. The threshold $\mathcal{E}$ is set to 1×10^{-15}, and the maximum number of iterations for each λ *is* 10^5. The optimal λ *is determined by measuring* the error between valid elements within estimated matrix **Z** and matrix **D** and using engineering judgement. The optimal λ will be used to generate final output for further analysis.

Figure 5.7 compares the estimated **Z** matrix obtained by completing matrix **D** with ground truth displacements extracted at VFPs. Four out of ten VFPs are chosen for better visualization. In Figure 5.7(a), the solid portions of the curves are measured valid displacements mapped from the MMPs, and the dotted portions of the curves are estimated from the proposed algorithms. Figure 5.8 shows the errors relative to the amplitude of that specific VFP (minimum absolute displacement). It should be noted that even the measured data (solid portions of the curves) are not exactly equal to the ground truth, shown in Figure 5.7 (b), due to the mapping process. This phenomenon could be explained by Figure 5.9. The valid points are defined when there are more than 2 MMPs in a VFP's adjacent segments. When only 2 MMPs are used, there is a chance that 2 MMPs cross the valid points, i.e. s_i in Figure 5.9. In this case, taking the pseudoinverse is equivalent to calculating values at three VFPs, i.e. s_{i-1}, s_i, and s_{i+1}, from two MMPs even though only data from one VFP, i.e. s_i, is considered as valid. This problem is underdetermined, and the mapping could cause errors. In following sections, it will be shown that when more MMPs are used, these errors can be reduced.



(a) Matrix Z obtained by completing D

(b) Ground truth

Figure 5.7 - Comparison between reconstructed displacements for 4 VFPs and ground truth (2 MMPs)

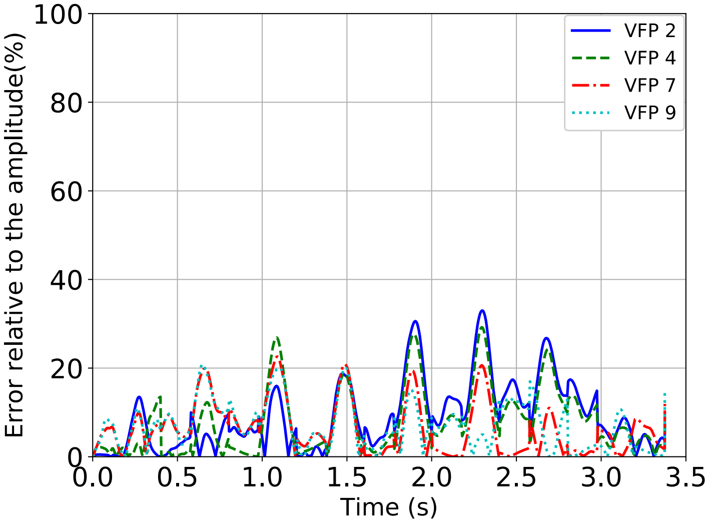


Figure 5.8 - Errors relative to the amplitude of each VFP (2 MMPs)

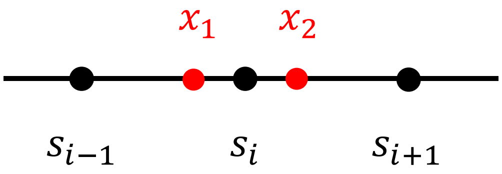


Figure 5.9 - Mapping from 2 MMPs

Following the mapping process, the matrix completion algorithm is implemented to find the internal relationship of displacements at different VFPs and to use this relationship to complete the matrix **D**. The matrix completion process is non-parametric which means no physical model is required for the process. The matrix **D** has a dimension of 10×677 with

4496 missing values. The missing rate for matrix **D** is 66.41%. It should be noted that the matrix **D** is constructed by mapping from 2 MMPs (2×677).

After obtaining the matrix **Z**, i.e. completed from of matrix **D**, SVD is applied to extract the mode shapes of the bridge. The identified mode shapes are summarized in Figure 5.10. In order to quantitatively measure the similarity between the identified and the exact mode shapes, the Modal Assurance Criterion (MAC) is calculated using Eq. (5.17),

$$\mathrm{MAC}(\phi_{\text{identified}},\phi_{\text{exact}})=\frac{\left(\phi_{\text{identified}}^{T}\phi_{\text{exact}}\right)^{2}}{\left(\phi_{\text{identified}}^{T}\phi_{\text{identified}}\right)\left(\phi_{\text{exact}}^{T}\phi_{exact}\right)} \tag{5.17}$$

where $\phi_{\text{identified}}$ and ϕ_{exact} are the vectors representing the identified and the exact mode shapes. The MAC values for these modes are 1.000, 0.998 and 0.977. It is seen that the system identification is very accurate even though the mapping is not exact.

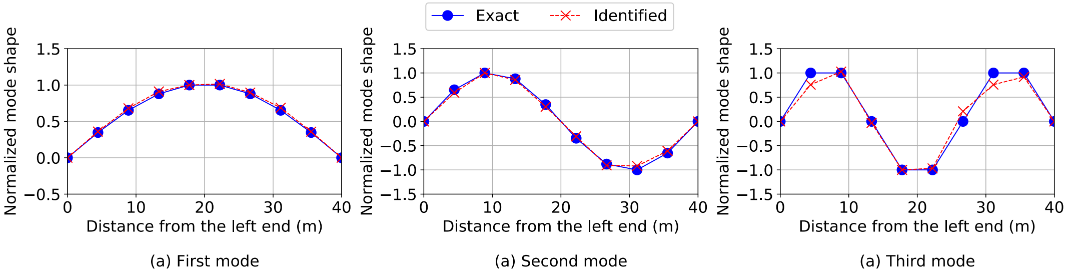
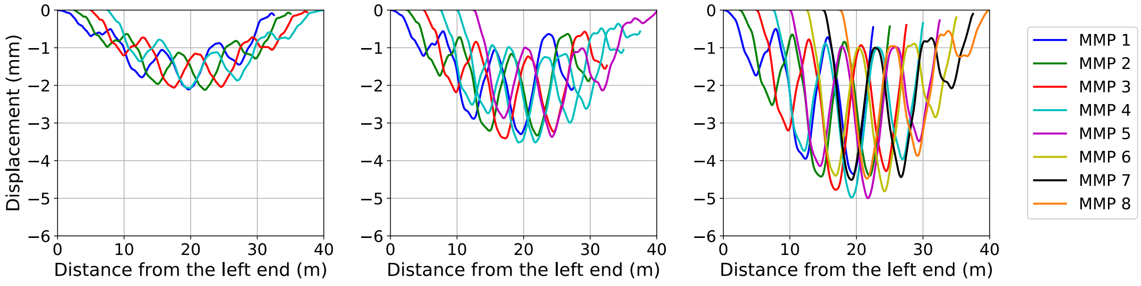


Figure 5.10 - The first three normalized mode shapes identified using 2 MMPs

5.4 Discussion

5.4.1 Influence of the Number of MMPs

As described before, in this study, two MMPs represent a car, and the number of cars is expected to have effect on the mode shape identification. In this section, the effect of the number of MMPs on the identification results is investigated. Figure 5.11 presents the displacement values integrated from acceleration data for 4, 6 and 8 MMPs. As seen, the displacements for different numbers of MMPs have similar patterns, but the maximum displacement is larger when more MMPs are on the bridge.



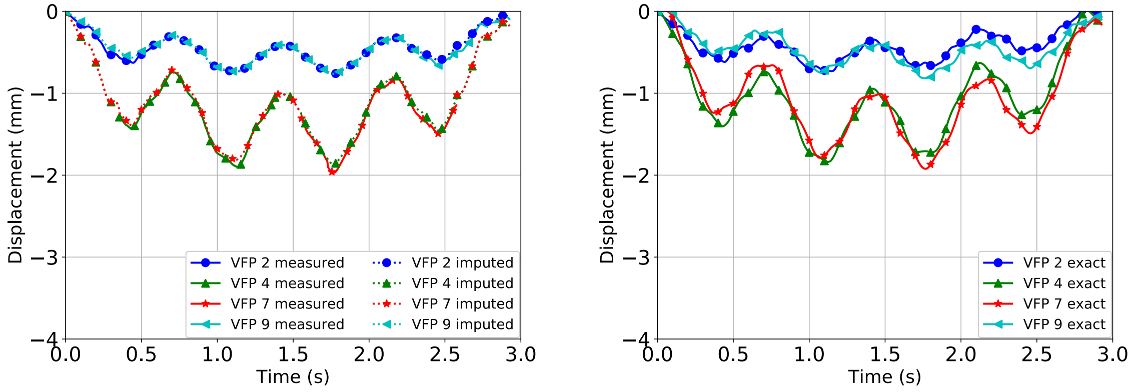
(a) 4 MMPs

(b) 6 MMPs

(c) 8 MMPs

Figure 5.11 - Integrated displacement for 4, 6 and 8 MMPs

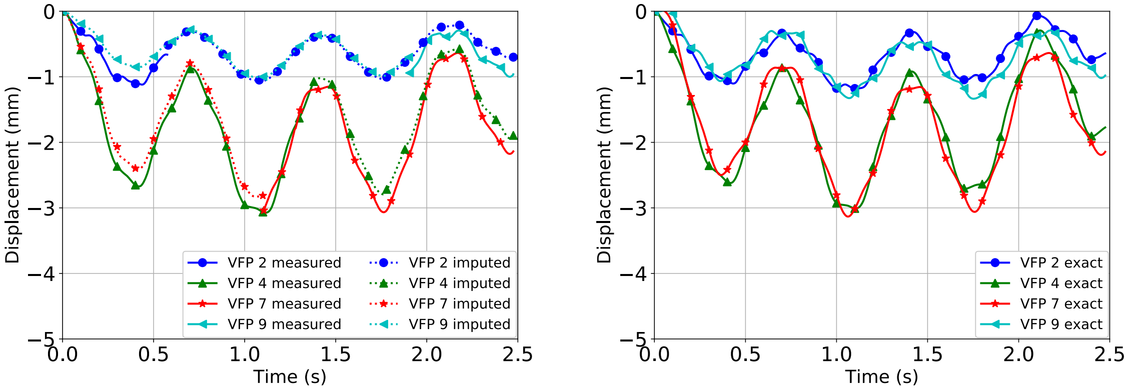
Similar to Figure 5.7 in the previous section, Figure 5.12, Figure 5.13, and Figure 5.14 show comparisons between the matrix **Z** after completing matrix **D** and ground truth displacements at VFPs for different number of MMPs. We can see in Figure 5.12(a), Figure 5.13(a), and Figure 5.14(a), the measured displacements are closer to the ground truth than the ones for 2 MMPs while using more MMPs. Comparing (a) and (b) plots in Figure 5.12, Figure 5.13, and Figure 5.14, the proposed algorithm works well to estimate the missing values in matrix **D.** For the three cases discussed in this section, the dimensions of matrices **D** are 10×587, 10×497, and 10×407 with 2166, 1472, and 911 missing values, respectively. The corresponding missing rates for these three cases are 55.40%, 44.47%, and 33.61%. Figure 5.15 presents the errors of the matrix completion relative to the amplitude of the displacement at each VFP. We can see the error level is around 20%, and more MMPs lead to lower errors.



(a) Matrix Z obtained by completing D

(b) Ground truth

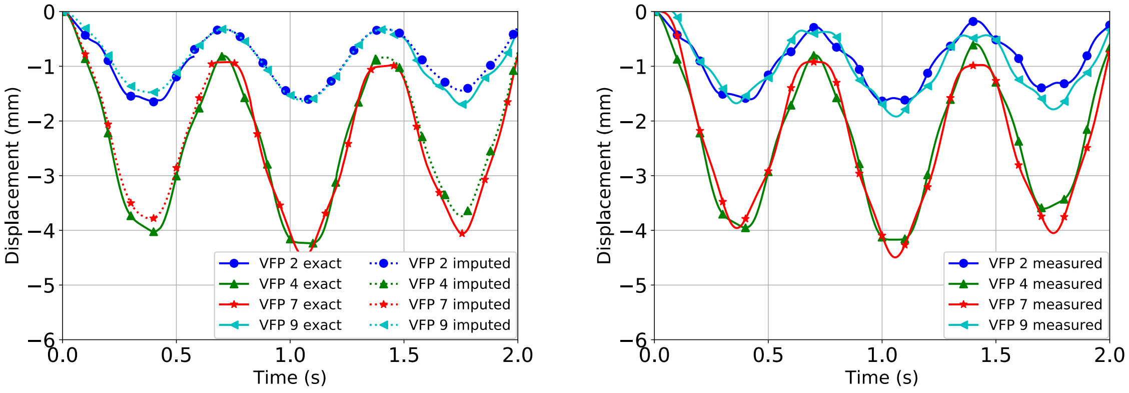
Figure 5.12 - Comparison between reconstructed displacements for 4 VFPs and ground truth (4 MMPs)



(a) Matrix Z obtained by completing D

(b) Ground truth

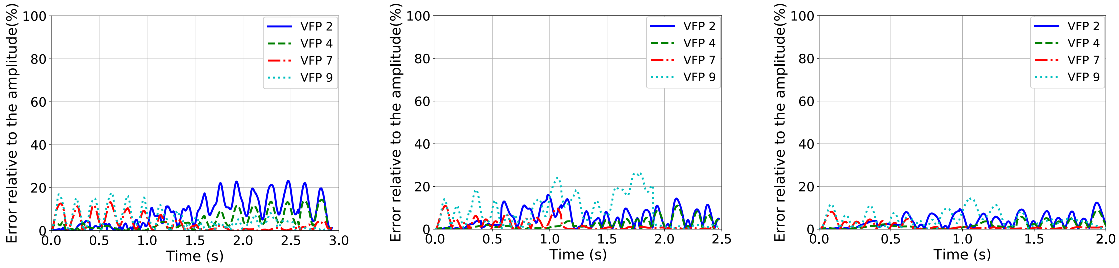
Figure 5.13 - Comparison between reconstructed displacements for 4 VFPs and ground truth (6 MMPs)



(a) Matrix Z obtained by completing D

(b) Ground truth

Figure 5.14 - Comparison between reconstructed displacements for 4 VFPs and ground truth (8 MMPs)



(a) 4 MMPs

(b) 6 MMPs

(c) 8 MMPs

Figure 5.15 - Errors relative to the amplitude of each VFP

The first three mode shapes identified in these three cases are presented in Figure 5.16, and the MAC values for these modes are summarized in Table 5.1. Generally, all the modes are identified with a high accuracy in these three cases. The MAC values for second and thirds modes are all a little lower than the ones obtained by 2 MMPs, but the influence is not very significant. The possible reason for this phenomenon is that the mapping matrix $\mathbf{N_s}$ for 2 MMPs has lower dimensions and is more robust to the instability of pseudoinverse. The MAC values for the second mode decrease as more MMPs are used, but this is not the case for the third mode. Further investigation should be conducted to find more detailed relationships between the number of MMPs and MAC values.

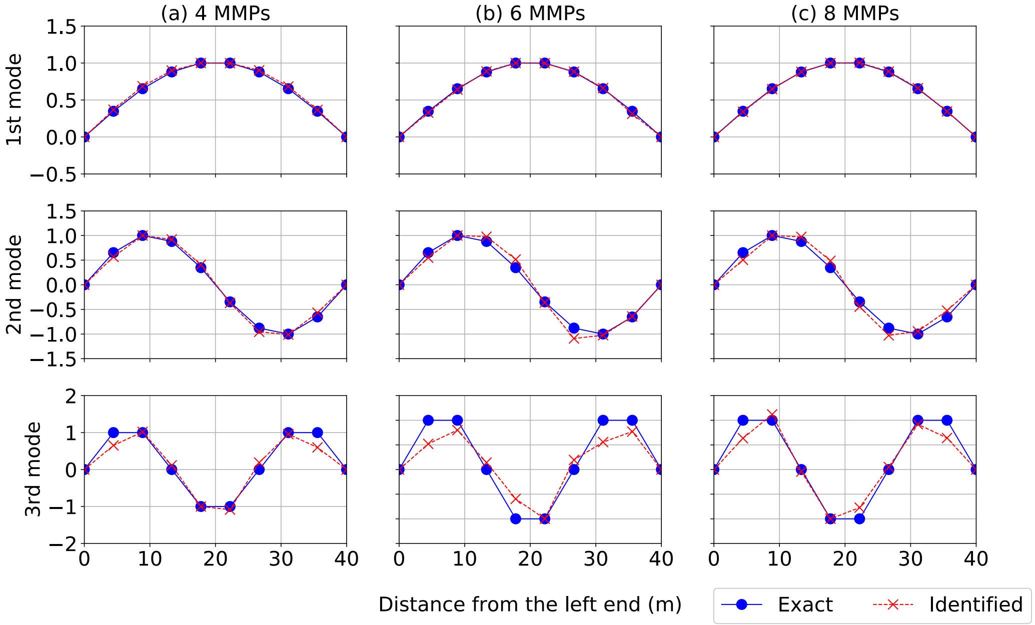


Figure 5.16 - The first 3 mode shapes identified from the proposed method

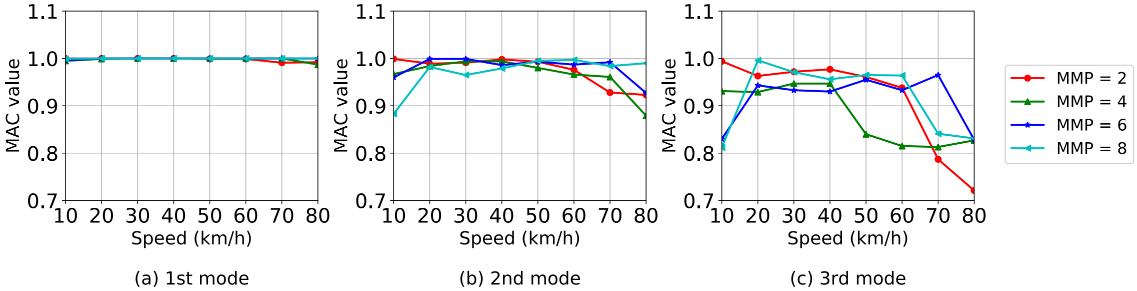
Table 5.1 - MAC values of the first 3 modes for 4, 6 and 8 MMPs

No. of MMPs	1st mode	2nd mode	3rd mode
4	1.000	0.994	0.947
6	1.000	0.986	0.930
8	1.000	0.979	0.956

5.4.2 Influence of Car Speed

The influence of the vehicle speed is discussed in this section. Figure 5.17 presents the MAC values for the first three modes at different speeds using 4, 6 and 8 MMPs. It is shown that the car speed has a very limited influence on the first mode since the MAC values for this mode are all around one for car speeds ranging from 10 to 80 km/h. The car speed has more influence on the second mode, and the most influence on the third mode. This makes sense since higher modes are less excited and harder to identify. From Figure 5.17(b), it is inferred that the MAC values of the second mode are around one, when the speed lies in the range of 20 to 60 km/h. Thus, it appears that moving either too slow or too fast can adversely affect the identification accuracy. From Figure 5.17(c), the MAC values of the third mode are higher for the speed range of 20 to 60 km/h for 2, 6 and 8 MMPs, but drop quickly starting at 50 km/h for the case of 4 MMPs. It should be also noted that the relationships of MAC values among 2, 4, 6 and 8 MMPs are different at different speeds. This also proves that not a sole factor is affecting the identification accuracy.

This section shows that unlike other mode shape identification methods as described in [58] where car speed has to stay at low levels, the proposed method can identify the first three modes very accurately even at nominal traffic speeds, which is a very important characteristic for real-life applications. The reason is that the matrix completion algorithm used in this chapter is non-parametric, and does not explicitly utilize the frequency domain information. Therefore, the accuracy of the system identification is not restricted by the Nyquist–Shannon sampling theorem [119].



(a) 1st mode (b) 2nd mode (c) 3rd mode

Figure 5.17 – MAC values for the first three modes at different speeds

5.4.3 Influence of Car Weight

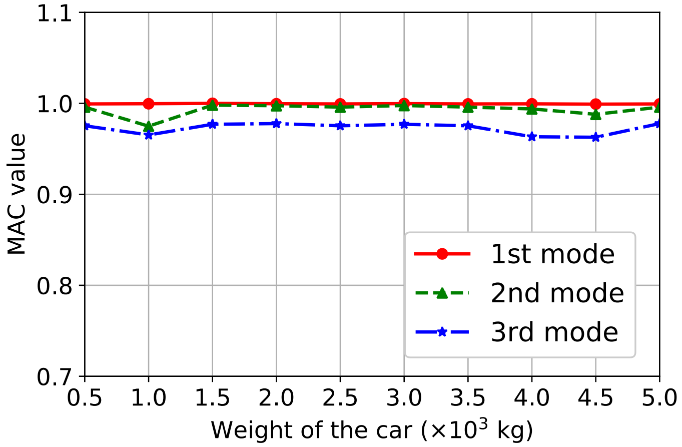


Figure 5.18 - MAC values for the first three modes for different car weights

Generally, the weight of a car varies from 1,000 kg to 3,000 kg [120]. This section discusses the influence of car weight on the mode shape identification. In this study, the car weights are simulated by assigning loads to MMPs. Considering the case of 2 MMPs, i.e., only one car, the weight of the car is evenly distributed to 2 MMPs where each MMP holds half weight of the car. Figure 5.18 shows the MAC values of the first 3 modes compared with ground truth by changing the car weight from 500 kg to 5,000 kg. It is seen that there is no significant decrease in MAC values when the car weight changes. In fact, according to the analysis, the MAC values do not change too much as long as the car weight is less than 1/10 of the bridge weight. This conclusion is consistent with Yang et al. [44]

5.4.4 Linear Interpolation Versus High-order Lagrange Interpolation

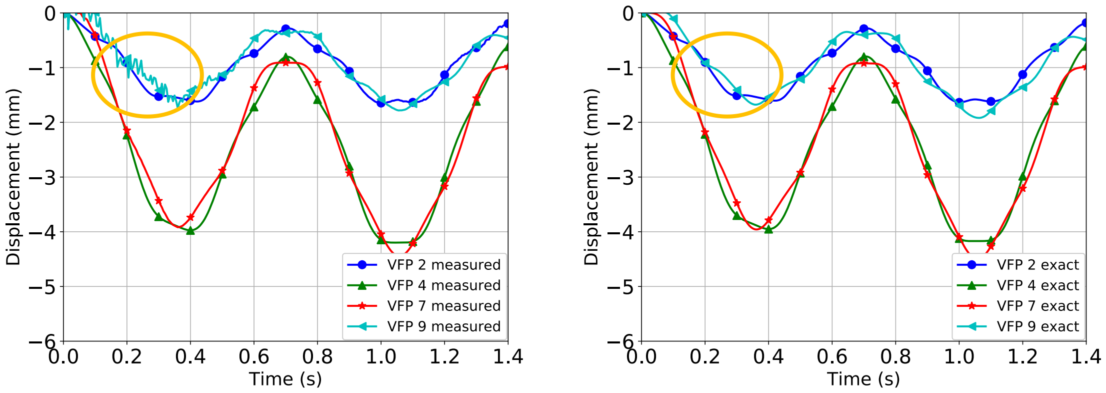
The mapping step used in this chapter was inspired by Oshima et al. [59] where they used high-order Lagrange interpolation to mapping displacements at MMPs to VFPs. Several issues are identified regarding applying high order Lagrange interpolation for the mapping which are listed below:

1) The number of MMPs has to be equal to the number of VFPs in order to calculate the inverse of their mapping matrices.
2) According to the observations, the inverse of their $\mathbf{N}(t)$ becomes unstable when the MMPs are far from the given VFP.

3) The original study uses all the VFPs for the Lagrange interpolations which results in overfitting of the data and makes the mapping sensitive to the noise and the locations of the MMPs. It was also acknowledged in [59] that the accuracy is lower when more MMPs are used.

To overcome the aforementioned issues, this chapter proposes to use linear interpolation to replace high-order Lagrange interpolation, and limit the valid data to a range that is close to the MMPs. Also, Moore–Penrose inverse (i.e., pseudoinverse) is used instead of the standard inverse for the mapping. These improvements will make the inversion more stable while creating a sparse matrix. Then, a matrix completion algorithm follows to fill the sparse matrix.

In order to make a direct comparison between the proposed mapping method and the one used in [59], we set the number of VFPs to 10, including two VFPs at boundaries, and the number of MMPs to 8 as well even though our proposed method can work with fewer MMPs. It should be noted that two VFPs at the boundaries are utilized by our proposed mapping method but not the one in Oshima et al. [59]. The other parameters are the same as in section 5.3.



(a) Matrix D

(b) Ground truth

Figure 5.19 – Comparison between Lagrange mapping proposed in Oshima et al.[59] and ground truth

Figure 5.19 presents the comparison between the Lagrange mapping proposed in [59] and the ground truth for four VFPs. As seen in Figure 5.14, the proposed mapping can reconstruct the displacements at VFPs more accurately with missing values. However, the

Lagrange mapping used in Figure 5.19 does not have missing values but have many inaccurate data points. For instance, as shown in the circled areas in Figure 5.19, the instability of the inverse results in some high frequency components which significantly reduces the performance of the system identification at higher modes.

The first three mode shapes identified from matrix **D** that is mapped by high order Lagrange interpolation are presented in Figure 5.20. The MAC values for these three modes are 1.000, 0.998 and 0.676. It is seen that the third mode is not identified correctly, which confirms that the instability of the mapping lead to inaccuracy in higher mode identification.

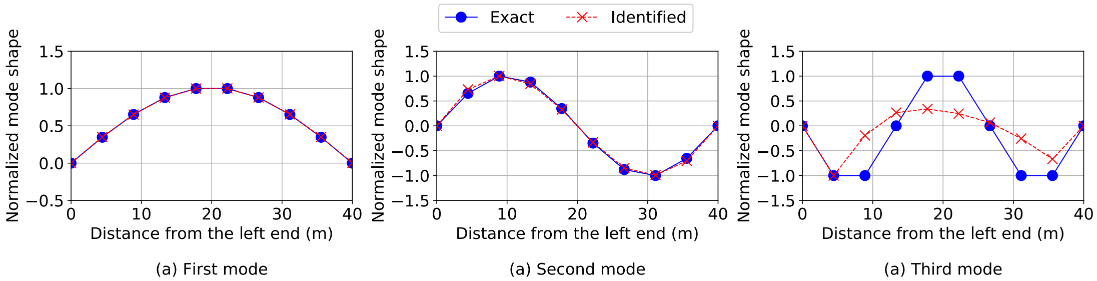
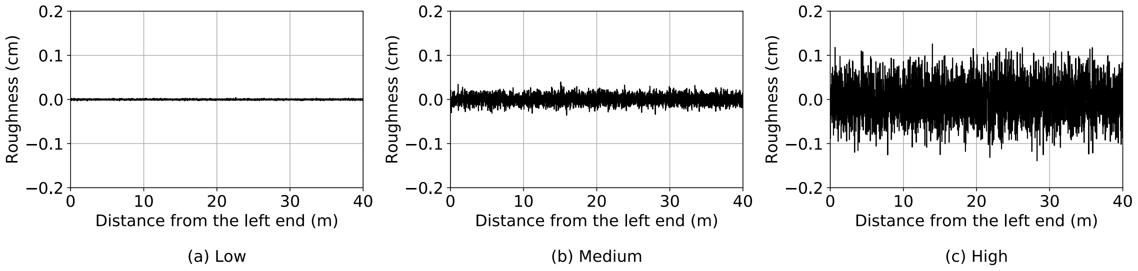


Figure 5.20 - The first three normalized mode shapes identified from matrix **D** obtained by Lagrange mapping

5.4.5 Influence of Road Roughness

It is well accepted that one of the main challenges in indirect mode shape identification is the road roughness [58], which can dominate the recorded vibrations. In this method, road roughness is added to the simulation to examine the robustness of the proposed method. Following the power spectral density (PSD) based procedure given in Yang and Lee [121], three different road profiles are generated using $G_d(n_0)$ of 0.01×10^{-6} m^3, 1×10^{-6} m^3, and 16×10^{-6} m^3, representing low, medium and high road roughness, respectively. [121] The generated road profiles are presented in Figure 5.21.



(a) Low

(b) Medium

(c) High

Figure 5.21 - Generated road roughness

The MAC values for these three road profiles are presented in Figure 5.22. It is demonstrated that for the low road roughness case, the MAC values for the first three modes are almost the same as the case without any roughness. As the road roughness increases, the MAC values decrease. The identification accuracy of the lower modes is more robust to the road roughness compared to higher modes. The MAC value for the first mode at high road roughness case is 0.876, while they are 0.750 and 0.492 for the second and third modes. It should be noted that the results in this section are obtained by directly applying the proposed method on rough road profiles. Preprocessing procedures like subtracting from two wheels [59, 63] or isolating the bridge vibration using blind source separation [65] could help reducing the effect of road roughness, but they are not studied in this chapter.

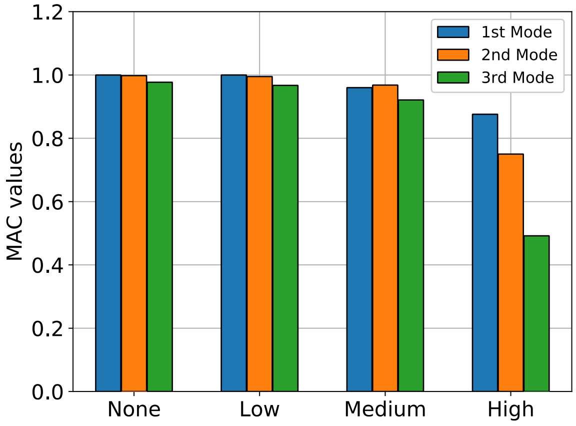
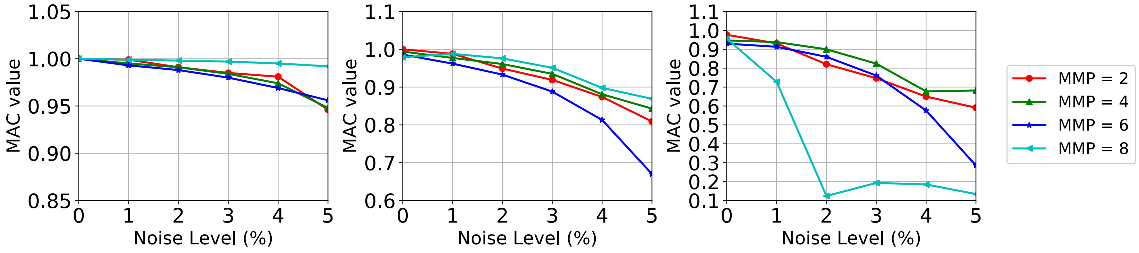


Figure 5.22 - MAC values for different road profiles

5.4.6 Influence of Noise

Except road roughness, the measurement errors could also play an important role in the performance of the proposed method. To investigate the influence of the measurement errors, artificial noise is added to all the acceleration data collected from the MMPs before calculating displacement data using integration. The noise is generated from a Gaussian distribution with the mean of 0 and the standard deviation as a percentage of the root mean square of the data. The noise levels from 0% to 5% are added in this study. The Signal-and-Ratios (SNR) for different levels are 40, 34, 31, 28 and 26 dB. Figure 5.23 presents the MAC values of different modes at different numbers of MMPs and noise levels. As expected, all MAC values decrease as the noise level increase. Among these three modes, the first one is least affected by the noise, and the third one is the most. Considering different numbers of MMPs, the first and second modes identified using 8 MMPs are the most robust to the noise, but the MAC values of the third mode for 8 MMPs drop significantly as the noise level increases.



(a) 1st mode

(b) 2nd mode

(c) 3rd mode

Figure 5.23 - MAC values of the proposed method at different noise levels

5.5 Conclusions

The chapter presents a novel method for mode shape identification using data recorded on moving vehicles by converting it to a matrix completion problem. Linear interpolation is applied to map the data collected from vehicles to fixed VFPs on the bridge. Soft imputing algorithm is introduced to fill the missing values in the matrix due to the mapping. Numerical analyses are conducted to verify the method. The proposed method can use as few as one single vehicle (two mobile sensors) to identify the first three modes with high accuracy. The following main conclusions are obtained from our studies:

1) The method works at traffic speeds ranging from 10 to 60 km/h. These speeds are more practical, as compared to many earlier methods which only allows very low speeds.
2) Through mobile scanning using the vehicle, the method can identify more modes with higher resolution, while using a much lower number of sensors than the fixed sensor scenario.
3) The method does not explicitly use information from the frequency domain, and thus it is more robust to noise and measurement errors.

CHAPTER 6: PAVEMENT CRACK DETECTION USING IN-VEHICLE CAMERAS

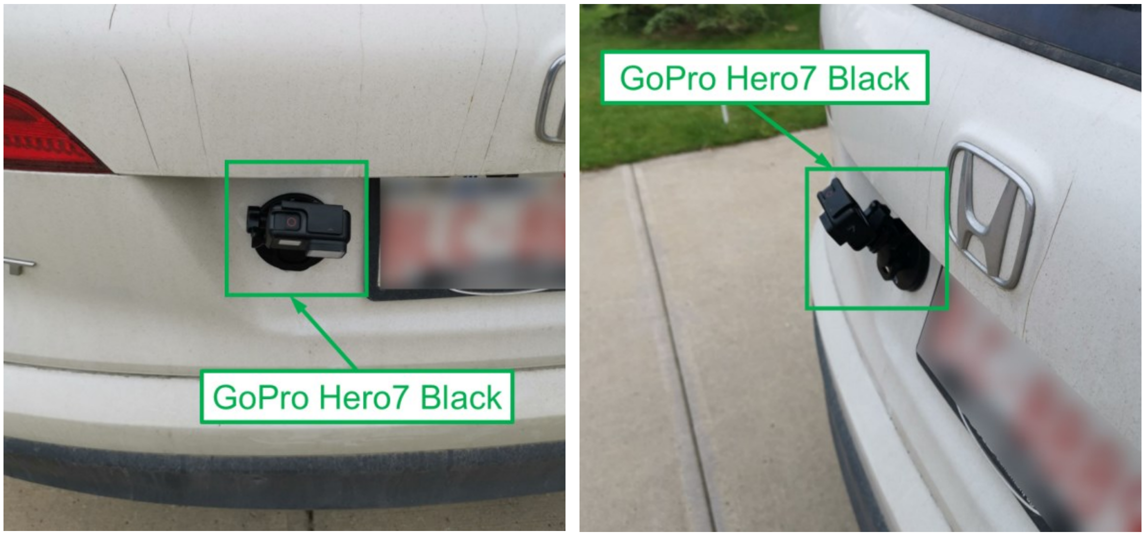
6.1 Overview

In this chapter, under the crowdsensing framework, a cost-effective pavement crack detection solution by mounting commercial-grade sport camera at the back of a moving vehicle is investigated. Data are collected from road tests to create a dataset consisting of 600 images with different cracks. A novel algorithm based on deep neural network called ConnCrack is proposed to detect the cracks at pixel level. The contribution of this chapter mainly includes: 1) The feasibility of using commercial grade sport camera mounted at the rear of a car operating at traffic speed is verified, and a new challenging pixel-level annotated dataset is introduced to consider the real life situation [122]; 2) a novel method combining conditional Wasserstein generative adversarial network and connectivity maps is developed for pixel level crack detection.

Compared with previous studies, the novelties of this chapter can be summarized as below. First, a sport camera is installed on the rear of a vehicle to mimic the behavior of a backup camera. As will be discussed in section 6.3, the rear-mount configuration could provide clearer images and higher resolution. Second, an algorithm with a conditional Wasserstein generative adversarial network (cWGAN) and connectivity maps are introduced to improve the accuracy of crack detection. Unlike the encoder-decoder architecture [22, 30, 85], the parameter updates of cWGAN come from not only samples but also the backpropagation of the discriminator, which makes this method more robust in detecting cracks. The connectivity maps are introduced to improve the accuracy by considering the connectivity of pixels in cracks.

6.2 Experimental Setup and Data Collection

In this study, field experiments were conducted with a GoPro Hero 7 Black mounted beside the license plate on the rear of a Honda Pilot 2017 (see Figure 6.1) for data collection. The cost for GoPro Hero 7 is 330 USD and the mounting device is 40 USD at the time of experiment implementation, which is significantly cheaper than specialized vehicles. It should be noted that this experiment has an initial goal to mimic the behavior of a backup camera in a vehicle. Current vehicles usually do not allow access to their backup camera systems easily. Therefore, the camera was placed at the same level and facing the same direction as the backup camera in this car to mimic its behavior. Data were collected based on this setting with the expectation that the conclusions drawn in this study could be useful in the future when the access to backup camera images becomes more practical.



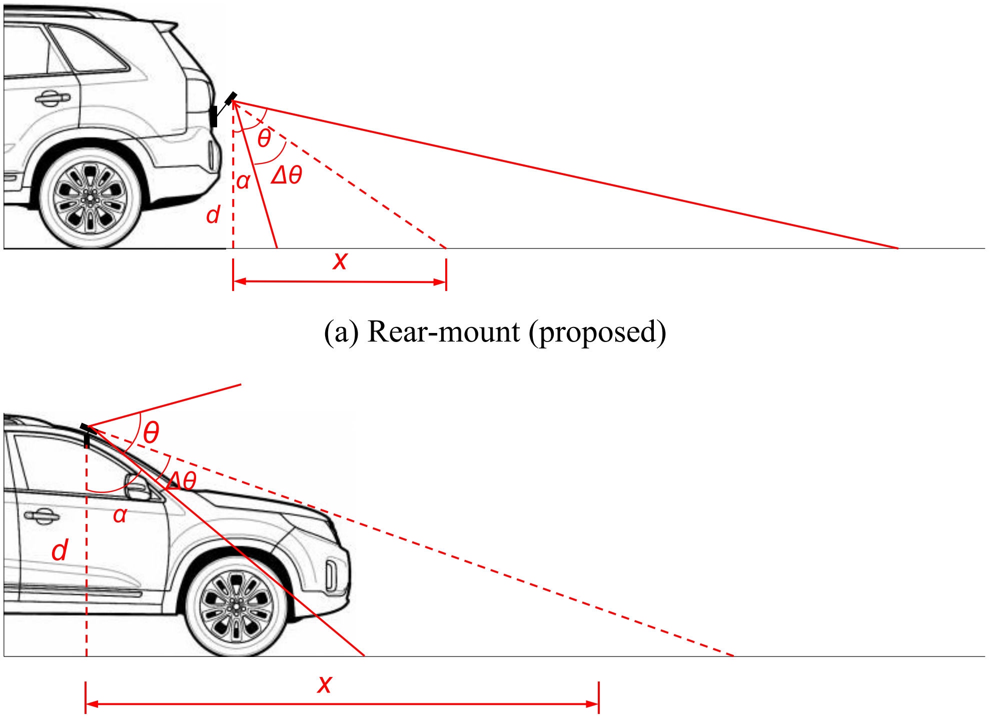
(a) front view (b) perspective view

Figure 6.1 - GoPro Hero 7 Black mounted at the rear of Honda Pilot 2017

In several previous studies [30, 123], the camera was mounted behind the windshield in the front of the car. As presented in Figure 6.2, two configurations are illustrated and compared. In rear-mount configuration, the angle of camera is set to 45° to balance the spatial resolution and scanned area. In front-mount configuration, the camera is facing forward like in previous studies [30, 123]. In these two configurations, the spatial resolution defined as number of pixels in unit length can be calculated as in Eq. (6.1). The spatial resolution represents how much detail can be captured by the camera.

$$\rho = 1/\left[d\tan(\alpha+\Delta\theta+\theta/m) - d\tan(\alpha+\Delta\theta)\right] \tag{6.1}$$

where d is the distance from the center of camera lens to the ground, $\Delta\theta$ is the angle from the bottom line of field of view (FOV), θ is the FOV, m is the total number of pixels in vertical direction and α is the angle between bottom line of FOV and vertical line.



(a) Rear-mount (proposed)

(b) Front-mount

Figure 6.2 - Comparison of two configurations (modified from [124])

Table 6.1 - Comparison between different mounting strategies

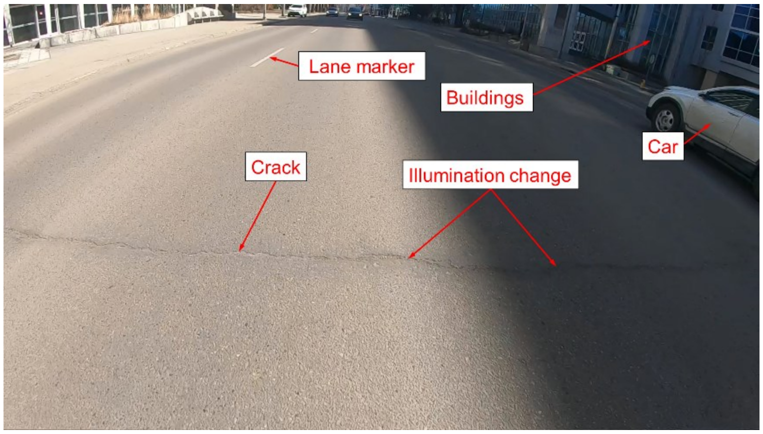
Percentage of FOV, Δθ/θ	Spatial Resolution (pixel/cm)	
	Rear-mount	Front-mount
0%	8.62	1.93 (blocked)
25%	6.99	0.53
50%	4.45	0.00
75%	1.91	N/A

100%	0.28	N/A

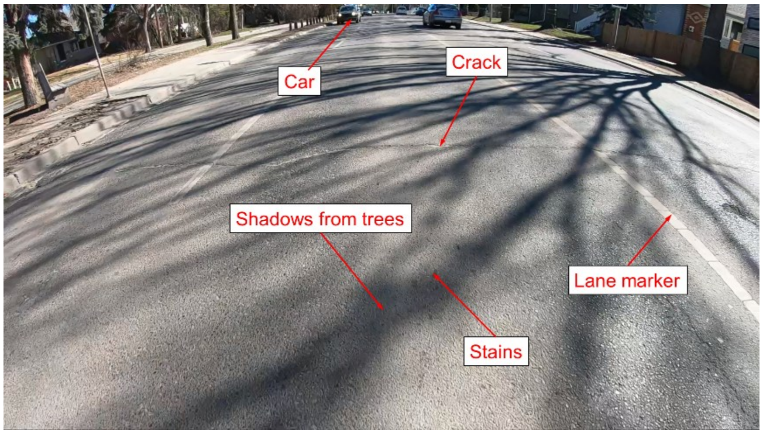
In this study, the GoPro Hero 7 black has a FOV of 69.5º. The image has a resolution of 1920×1080 pixels. Therefore, α for rear-mount configuration is 45°-69.5°/2=10.25° and for front-mount configuration is 90°-69.5°/2=55.25°. The vertical distance to the ground is 1.5 m for front-mount configuration and 1 m for rear-mount configuration. According to the above information, the parameters in Table 6.1 are calculated. In Table 6.1, the percentage of FOV is corresponding to percentage of image regarding the image bottom in vertical direction. For instance, $\Delta\theta/\theta$ of 50% means the centerline of the image in vertical direction. It is seen from the table that the spatial resolution decreases dramatically as the percentage of FOV increase, which is expected because the pavement is farther from the camera. Comparing these two configurations, we can see the front-mount configuration has significantly less spatial resolution than rear-mount configuration overall. This is because the front-mount camera is farther from the ground. Also, the 0% to 25% region is most likely to be blocked by the hood. Therefore, it is seen that rear-mount configuration to better utilize the GoPro camera.

To summarize, there are three main reason that we use a rear-mount configuration: 1) The windshield could reflect the light inside of the car and reduce the quality of the image in front-mount configuration. 2) The front camera is farther from the ground, a large part of its FOV is blocked by the hood of the car. Therefore, the front-mount configuration sacrifices too much spatial resolution corresponding to our analysis above. 3) Our eventual goal is to directly use backup camera in vehicles for crack detection while driving. In this case, no external devices need to be installed in this case.

The data were collected while the vehicle was driving at traffic speeds (40 kph – 80 kph), and 240 fps frame rate and 1/3840 sec shutter speed was used for the camera. In total, about 3-hour videos were taken from different roads in Edmonton, Canada at different times over two months by our research group. Images were extracted every 6 frames. After discarding those without cracks, we created a dataset called EdmCrack600 which includes 600 images with full annotation at pixel level. The data aim to cover various factors one could encounter on the roads like different weather conditions, different illumination conditions, existence shadows from other objects, texture difference among difference pavement surfaces, etc., so no specific restrictions are applied during the collection process. The dataset, EdmCrack600, will be made public to benefit the community [122]. Some sample images are shown in Figure 6.3. It is seen that the collected dataset is more difficult than most of the publicly available ones.



(a) Sample image 1



(b) Sample image 2

Figure 6.3 - Sample images from EdmCrack600 dataset

In the author's opinion, one of the biggest restrictions that holds back the development of novel algorithms is the lack of high quality and challenging datasets with complete annotations. In most of studies, the researchers either tested their methods on their own datasets [77, 78, 80-82], and the publicly available datasets specifically designed to evaluate crack detection algorithms are limited. Furthermore, most of the datasets have been simplified comparing to the ones that could be encountered in real life. For example, some datasets control the light conditions [125], some manually exclude any disturbance and focus only on pavement surfaces using static images [70, 71, 126], and some were created for other algorithms and simply do not have enough images for deep learning algorithms [19, 70].

A comparison of this dataset and other publicly available dataset is given in Table 6.2. It is seen that only GAPs [125] and JapanRoad [123] datasets consist of more images than our dataset. However, those two datasets are not pixel-level annotated. The cracks in their datasets are annotated by bounding boxes. In author's opinion, the bounding box is a not good way to annotate crack because of the irregular shape of cracks. Too many details will be lost if a rectangular bounding box is used to cover the cracks.

To the best of the authors' knowledge, our dataset, EdmCrack600, is the largest crack dataset so far which is annotated at pixel level. It is also a very challenging one because of all the factors that are taken into consideration during the data collection process. The challenges include: 1) change of weather conditions; 2) significant environmental effects and noise: shadows, occlusion, stains, texture difference, low contrast because of overexposure; 3) blurring effect due to moving of the car and the poor lighting condition.

Table 6.2 - Comparison among different datasets

Dataset	No. Images	Resolution	Device	Colored	Environmental effect*	Non-pavement region**	Pixel level annotation	Traffic speed	Extracted from video
CFD [71]	118	480×320	iPhone 5	yes	yes	no	yes	no	no
Aigle-RN [70]	38	991×462 311×462	professional camera	no	no	no	yes	yes	no
Crack500 [126]	500	2,000×1,500	LG-H345	yes	no	no	yes	no	no
GAPs [127]	1969	1920×1080	professional camera	no	no	no	no	yes	yes
Cracktree200 [69]	206	800×600	unknown	yes	yes	no	yes	no	no
GaMM [19]	42	768×512 1920×480	professional camera	no	yes	no	yes	yes	yes
CrackIT [128]	84	1536×2048	optical device	yes	no	no	yes	unknown	no
JapanRoad [75, 123]	9,053	600×600	LG-5X	yes	yes	yes	no	yes	no

EdmCrack600 (current study)	600	1920×1080	GoPro 7	yes	yes	yes	yes	yes	yes

*Environmental effect includes shadows, occlusions, low contrast, noise, etc.

**non-pavement region means the region of image that does not belong to pavement, such as cars, houses, sky.

6.3 Methodology

6.3.1 Overall Procedure

A novel deep learning-based algorithm called ConnCrack is proposed in this chapter. The overall procedure of ConnCrack is described in Figure 6.4. In this process, the input image is first divided into patches. Then, the patches are passed to a deep neural network termed as generator. In the generator, color image patches are taken as input, and a DenseNet121 with deconvolution layers for multiple-level feature fusion is applied. Unlike other deep learning-based crack detection methods, the generator outputs 8 connectivity maps instead of a binary probability mask. The reason and the advantages of this innovation will be explained in following sections. Later on, the predicted connectivity maps are converted to a binary mask. A post-processing technique including a depth first search (DFS) algorithm to find connected components and to threshold out connected components with a small number of pixels is applied to the output the generator [23, 24]. The reason for this post-processing is because the cracks are usually connected components with a large number of pixels, but noise has much fewer connected pixels. At last, the crack identification result of the whole image will be integrated using the results from the patches.

The training of this method is developed on the basis of a conditional Wasserstein generative adversarial network (cWGAN), and connectivity maps are used to resolve the scattered output due to deconvolution layers (see Figure 6.5). The cWGAN consists of two separate neural networks, i.e. generator and discriminator. In this context, the generator outputs connectivity maps for the identification of cracks, while the discriminator checks if the connectivity maps and the original patch are a “real” pair (ground truth) or a “fake” pair (predicted). Two networks are trained alternately to reach a Nash equilibrium after convergence [129].

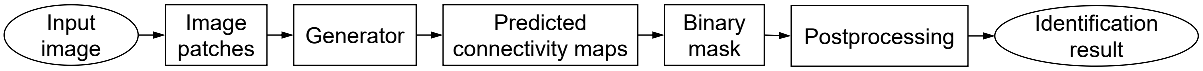


Figure 6.4 - Overview of the identification process

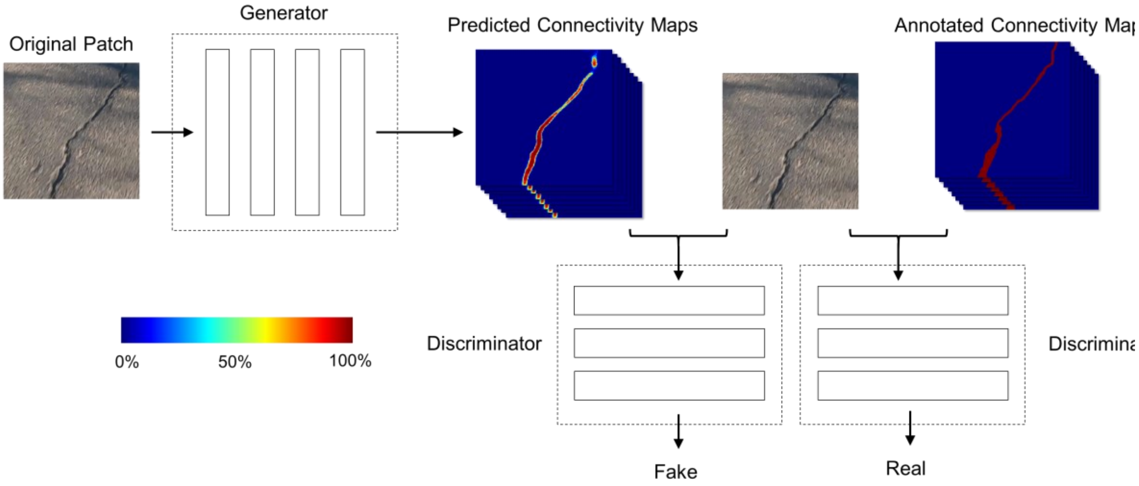


Figure 6.5 - Overview of the training process

6.3.2 Connectivity Maps

In this chapter, deconvolution layers are used for upsampling and pixel level identification similar to some other studies [30, 82] for computational efficiency. However, it is realized that deconvolution layers are likely to generate scattered output (see Figure 6.6(b)), i.e. the crack segments are not strictly connected. This is due to the mechanism of deconvolution layers where the predicted label of a pixel is solely dependent on the pixel values of a local region in original patch but is not explicitly related to the predicted labels of its neighboring pixels. Some studies suggested morphological operations, i.e., dilation and erosion, to resolve this issue [71]. However, as shown in Figure 6.6(c) and (d), the performance is highly dependent on the selection of the size of morphological operations. If the size is too small, the gaps are not fully filled. If the size is too large, unnecessary parts will be considered as cracks.

This issue comes from the definition of cross entropy loss function currently used in many deep neural networks for crack detection [22, 30]. Taking Figure 6.7 as an example, the crack pixels are labelled as 1 and the non-crack pixels are labelled as 0 in the ground truth. If the neural network mistakenly predicts one pixel within crack as 0, it is not different than predicting a non-crack as 1 in terms of loss function. However, in reality, an isolated wrong prediction is easier to fix than scattered prediction in crack segments.

To resolve this issue, we transform the crack detection into a connectivity problem inspired by [130]. Starting from the ground truth binary mask, each pixel should have 8 neighboring pixels. We generate 8 connectivity maps to reflect the relationship between a pixel and its 8 neighbors. As presented in Figure 6.8, a regular ground truth binary crack mask is converted to 8 connectivity maps. For instance, one element in A2 connectivity map is 1 only if the corresponding element in ground truth binary mask is 1 and its left neighbor is 1 as well. During the training process, the ground truth connectivity maps are compared with predicted connectivity maps as one source to update the weights of the deep neural networks. The loss function based on the connectivity maps which is termed as *Lcontent* could be written as Eq. (6.2) below.

$$\begin{aligned} L_{\text{content}}(G) &= E_{x,y}\left[-y\log G(x)-(1-y)\log(1-G(x))\right] \\ &= \sum_{k=1}^{8}\sum_{i,j\in\text{image}}\begin{bmatrix}-y_{A_k}(i,j)\log\hat{y}_{A_k}(i,j) \\ -(1-y_{A_k}(i,j))\log(1-\hat{y}_{A_k}(i,j))\end{bmatrix} \end{aligned} \tag{6.2}$$

where *G* represents the generator. It takes *x* as input and generates *G*(*x*). The true label (ground truth connectivity maps) of input *x* is termed as *y.* Also, at pixel level, $y_{A_k}(i,j)$ is the true label of a pixel at *i* and *j* in the connectivity map *Ak*. And $\hat{y}_{A_k}(i,j)$ is the predicted label for the corresponding pixel.

With the help of connectivity maps, more weights will be given to the pixels within crack segments and less weights are given to isolated pixels. In this way, the predictions are forced to be connected to each other. As can be seen in Figure 6.9, the performance of deep neural network trained with regular binary mask and our proposed connectivity maps are compared. The results based on connectivity maps are more robust and less scattered because the connectivity maps force the predictions to be connected.

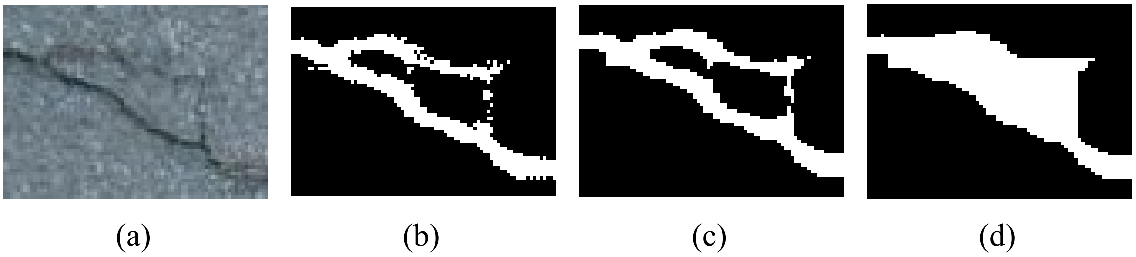
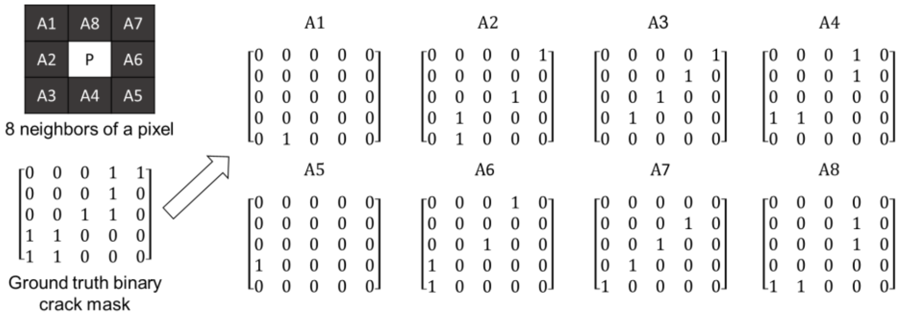


Figure 6.6 - Issues with deconvolution layer output (a) original patch; (b) raw output; (c) after 3×3 morphological operations; (d) after 15×15 morphological operations

$$
\begin{array}{ccc}
\text{Ground turth} & & \text{Predicted Labels} \\
\begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} & \Leftrightarrow & \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{array}
$$

Figure 6.7 - An example of crack detection



A1	A8	A7
A2	P	A6
A3	A4	A5

8 neighbors of a pixel

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Ground truth binary crack mask

$$
A1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}
\quad
A2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}
\quad
A3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\quad
A4 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}
$$

$$
A5 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\quad
A6 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}
\quad
A7 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}
\quad
A8 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix}
$$

Figure 6.8 - Connectivity maps for crack annotation

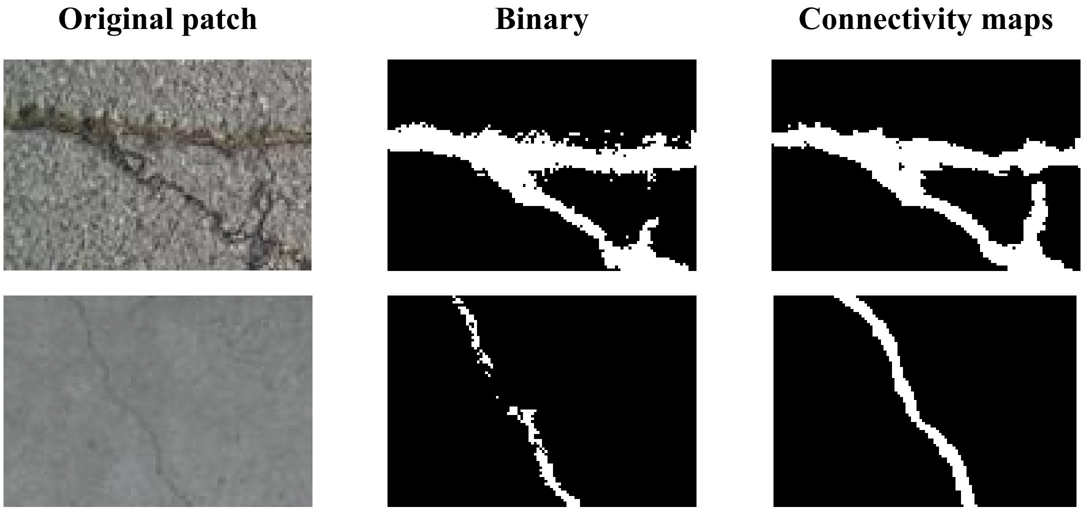


Figure 6.9 - Comparison between binary mask and connectivity maps

6.3.3 Conditional Wasserstein Generative Adversarial Network (cWGAN)

Generative adversarial network (GAN) was introduced by Goodfellow et al. [129] in 2014. It has been widely used for various computer vision tasks due to its high performance in replicating the real-world rich content [131, 132]. In general, a GAN consists of two neural networks, a generator and a discriminator. The generator is used to generate the output which is passed to the discriminator to check whether the output is a generated one ("fake") or the ground truth ("real"). A conditional Generative adversarial network (cGAN) [133] is a type of GAN that uses both the input and output of the generator as the input of the discriminator. cGAN can be used to check whether the input of the discriminator is a generated pair ("fake") or annotated pair ("real").

In some previous studies, it was observed by the researchers [134, 135] that the training of GAN is difficult. They showed that the traditional GAN that is based on Kullback–Leibler divergence has gradient vanishing issue during training. The gradient of the generator would be close to zero if the performance of the generator is not good which is likely the case at early stages of the training. To resolve this problem, Arjovsky et al. [134] replaced Kullback–Leibler divergence with Wasserstein distance. This type of GAN is called Wasserstein GAN (WGAN). The major difference between GAN and WGAN is the design of loss function, which will be discussed in following sections.

This study combines the cGAN [133] and WGAN [134] for crack detection to achieve better performance and better training stability. Connectivity maps, as described in the last section, are produced by the generator and are used together with the original image patch as the input to the discriminator.

6.3.3.1 Generator

In cWGAN, the generator is the deep neural network for crack detection. In the ConnCrack, as shown in Figure 6.10, a DenseNet121 [136] is used as feature extractor and 3 deconvolution layers are applied for multi-level feature fusion to generate target connectivity maps.

The DenseNet121 consists of a standalone convolutional layer, a max pooling layer, 4 dense blocks and 3 transition blocks. The convolutional layer was first proposed by LeCun [137], which is now widely used for computer vision problems. Similar to filters in traditional image processing techniques, a convolutional layer is applied to the input in a sliding window form. Unlike a fully connected layer, the sparsely connected neurons in a convolutional layer can lead to better efficiency and performance. Max pooling layer replaces the value of the input feature at a certain location with its neighboring features. It can reduce the size of features and make the features invariant to small translations.

One characteristic of DenseNet121 that distinguishes it from other deep neural networks is the application of the dense block. A dense block consists of a number of convolutional layers which are densely connected with each other in a feed-forward fashion. A 1×1 convolutional layer and a 3×3 convolutional layer form a basic component in a dense block. Each dense block has multiple such components, and each component is directly connected with all following basic components within this block using skip connections except the mainstream chain-like connections. In DenseNet121, the dense blocks 1, 2, 3 and 4 (see Figure 6.10) have 6, 12, 24 and 16 basic components, respectively.

The dense block does not change the height and width of the features. To follow an encoder-decoder schema for pixel level crack identification, transition blocks are applied to reduce the size of features. A transition block composes of a 1×1 convolutional layer and a 2×2 average pooling layer with a stride of 2. The reduction of size is achieved by the average pooling layer in the transition block.

Deconvolution layers are applied to fuse features from multiple levels so that the predicted connectivity maps have the same height and width of the original patch. Unlike traditional upsampling techniques, such as bilinear and bicubic interpolations which have predefined parameters, the parameters for upsampling in deconvolution layers are determined during the training process. The deconvolution layers were first time used for upsampling in semantic segmentation by Long et al. [138]

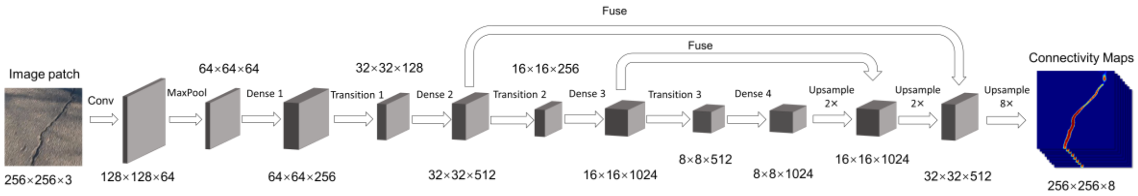


Figure 6.10 - Details of the generator

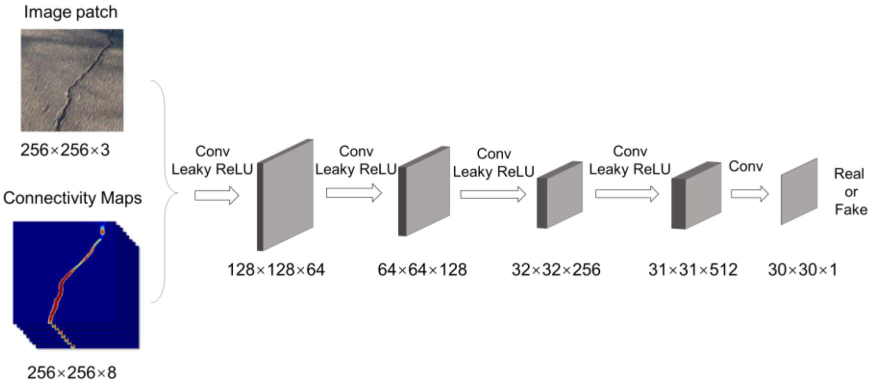


Figure 6.11 - Details of the discriminator

6.3.3.2 Discriminator

In the ConnCrack, the architecture of the discriminator is presented in Figure 6.11. It is 5-layer fully convolutional neural network. The original image patch and the corresponding connectivity maps are concatenated and passed through the discriminator. For the ground truth, the discriminator is expected to output labels as “real”. In contrast, it is expected to output “fake” when the predicted connectivity maps are used as input.

Similar to Pix2Pix [131] but different from traditional conditional generative adversarial networks (cGAN) [129], the proposed method uses a Markovian discriminator, where the output is not a single label but 30×30 labels representing “real” or “fake”. Each element of the 30×30 tensor corresponds to a small 70×70 patch, and it shows whether this patch is “real” or “fake”. These small patches are overlapped with each other. According to [131], the Markovian discriminator is better at capturing the high frequency part (details) of the image.

6.3.3.3 Loss Function

The loss function used in the proposed method combines cWGAN loss and content loss. The loss function is given in Eq. (6.3). Initially, the content loss is larger, and therefore the cWGAN is simplified to an encoder-decoder network. As the training continues, the content loss becomes smaller, and the effect of cWGAN loss becomes more significant. This can help the training of the model at early stages.

$$\begin{aligned} &L_{\mathrm{c}WGAN}(G,D)=E_{x,y}\left[D(x,y)\right]-E_{x}\left[D(x,G(x))\right] \\ &G^{*}=\arg\min_{G}\max_{D}(\lambda L_{cWGAN}(G,D)+L_{\mathrm{content}}(G)) \end{aligned} \tag{6.3}$$

where x is the input patch, y is the ground truth connectivity maps, G is the generator, D is the discriminator and λ is the parameter adjusting the weights of $L_{\mathrm{cWGAN}}(G, D)$ and $L_{\mathrm{content}}(G)$. $E_{x,\,y}$ represents the mean over multiple x, y pairs for training, and E_x represents the mean over multiple input x for training.

Unlike traditional cGANs, the log functions are removed from $L_{\mathrm{cWGAN}}(G, D)$ to achieve a Wasserstein distance following the suggestion from [134]. During the training process, the weights of the discriminator is clipped to a range [-C, C] to fulfill the requirement Lipschitz constraint [134] where C is a constant. Also, similar to [131], we add a content loss directly comparing with the output of the generator G with the ground truth. This could help the training process of the generator according to [139].

In the practical implementation, the discriminator D and generator G are trained alternatively. The generator G is trained to generate predicted connectivity maps (“fakes”) that cannot be distinguished from ground truth (“reals”) by discriminator D. In contrast, the discriminator D is trained to be better at distinguishing the “reals” from “fakes”. After the training is completed, the generator G will be used for crack detection, and the discriminator D can be discarded.

6.3.4 Evaluation

Three metrics are used for the evaluation of the proposed method, i.e., precision, recall and F1 score. The formulae to calculate these metrics are given in Eq. (6.4).

$$\begin{aligned} \text{precision} &= \frac{TP}{TP+FP} \\ \text{recall} &= \frac{TP}{TP+FN} \\ \text{F1 score} &= \frac{2\times\text{precision}\times\text{recall}}{\text{precision}+\text{recall}} \end{aligned} \tag{6.4}$$

In above equations, *TP* is true positive, *FP* is false positive, and *FN* is false negative. Following the definition given in [71], the *TP* is defined as the number of crack pixels that are within 5-pixel distance of a ground truth crack pixel. *FP* is the number of crack pixels that are beyond 5-pixel distance of a ground truth crack pixel. FN is the number of ground truth crack pixels which are incorrectly identified as non-crack pixels.

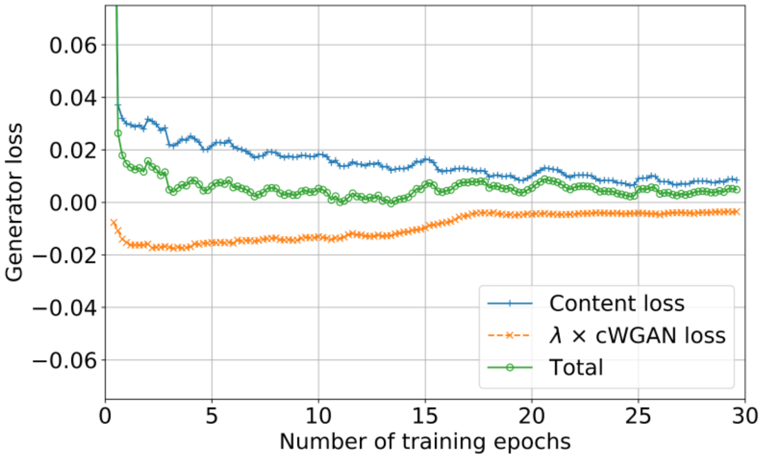
6.4 Analysis and Results

6.4.1 Pretraining on ImageNet and CFD datasets

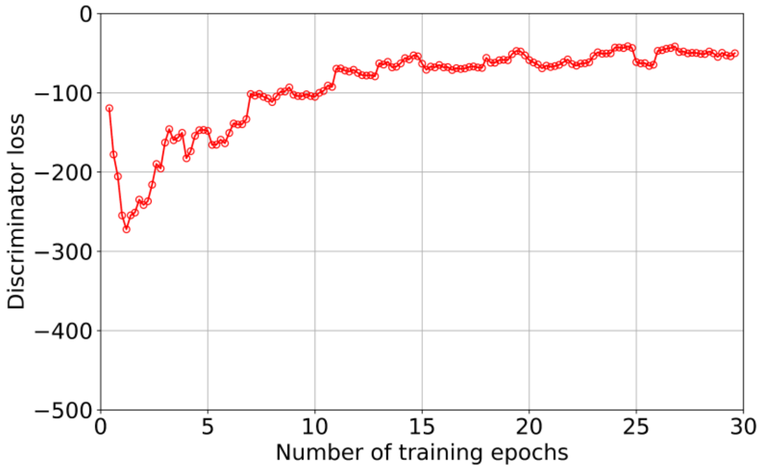
From a number of previous studies, it is well accepted that pretraining on irrelevant large datasets in advance before tackling the task can help improve the performance of the deep learning-based algorithms [140]. This strategy is called transfer learning. In this chapter, the proposed generator is first pretrained on a large object detection dataset called ImageNet [9]. It should be noted that the ImageNet dataset does not have a category related to pavement cracks.

Then, the whole proposed method is again pretrained and tested on a small crack dataset called CFD which was introduced by Shi et al. [71]. This dataset consists of 118 pavement images with resolution of 480×320 pixels. The images are taken by iPhone 5 with focus of 4 mm and aperture of *f/2.4*. In this chapter, the dataset is split into 60%/40% for training and testing. More details of the dataset can be found in [71].

For the training and testing, the images are split into 128 × 128 patches and are then integrated to the original size after being processed by the ConnCrack. Both the learning rate and λ are set to 1×10^{-6} during the training. The batch size of 16 is used. The training losses of generator and discriminator are presented in Figure 6.12. For better visualization, a 5-element moving average is taken on all the curves. As can be seen in Figure 6.12(a), the generator loss has two components, one comes from the cWGAN and the other comes from the content loss described in Eq. (6.2). It is seen that the content loss continuously decreases as the training proceeds. The cWGAN loss for generator first decreases and then increases since the discriminator has learned to distinguish the "fakes" from "reals". Looking at Figure 6.12(b), the loss for discriminator is low at the beginning but increases afterwards. This is because initially the generator is not well trained, and the discriminator can easily distinguish the generated output from the ground truth. However, as the training proceeds, the generator can output predictions that are more difficult to distinguish. In this context, the loss for discriminator starts to increase.



(a) Generator loss



(b) Discriminator loss

Figure 6.12 - Losses of the proposed method

Some sample images along with the ground truth and prediction are presented in Figure 6.13. It is seen that ConnCrack can identify the cracks with high accuracy. Table 6.3 compares the results from the proposed method with other methods. In this table, the results from canny detector, CrackTree, FFA and CrackForest were reported by [71], where canny detector [141] was a regular edge detection algorithm, CrackTree [69] utilized minimum spanning trees, FFA [142] used features calculated along every free-form paths, and CrackForest applied random structured forests for crack detection. MFCD [143] applied unsupervised feature fusion at multiple scales to detect cracks. The remaining methods were developed based on deep learning where ResNet152-FCN [30] and VGG19-FCN [82] were encoder-decoder neural networks utilized ResNet152 and VGG19 as their backbone networks. In CrackNet-V [21], an architecture that did not use pooling layers was proposed to reserve the size of input and detect the cracks. The results from all other methods were reported in their papers except ResNet152-FCN and VGG19-FCN which are implemented by ourselves with the same learning rate as ConnCrack and pretraining on ImageNet. We can see that the proposed method outperforms other methods on CFD dataset in terms of precision and F1 score with large margin.

Table 6.3 - Comparison of performance for different methods on the CFD dataset

Method	**Precision**	**Recall**	**F1 Score**
Canny	12.23%	22.15%	15.76%
CrackTree	73.22%	76.45%	70.80%
FFA	78.56%	68.43%	73.15%
CrackForest	82.28%	89.44%	85.71%
MFCD	89.90%	89.47%	88.04%
ResNet152-FCN	87.83%	88.19%	88.01%
VGG19-FCN	92.80%	85.49%	88.53%
CrackNet-V	92.58%	86.03%	89.18%
ConnCrack	96.79%	87.75%	91.96%

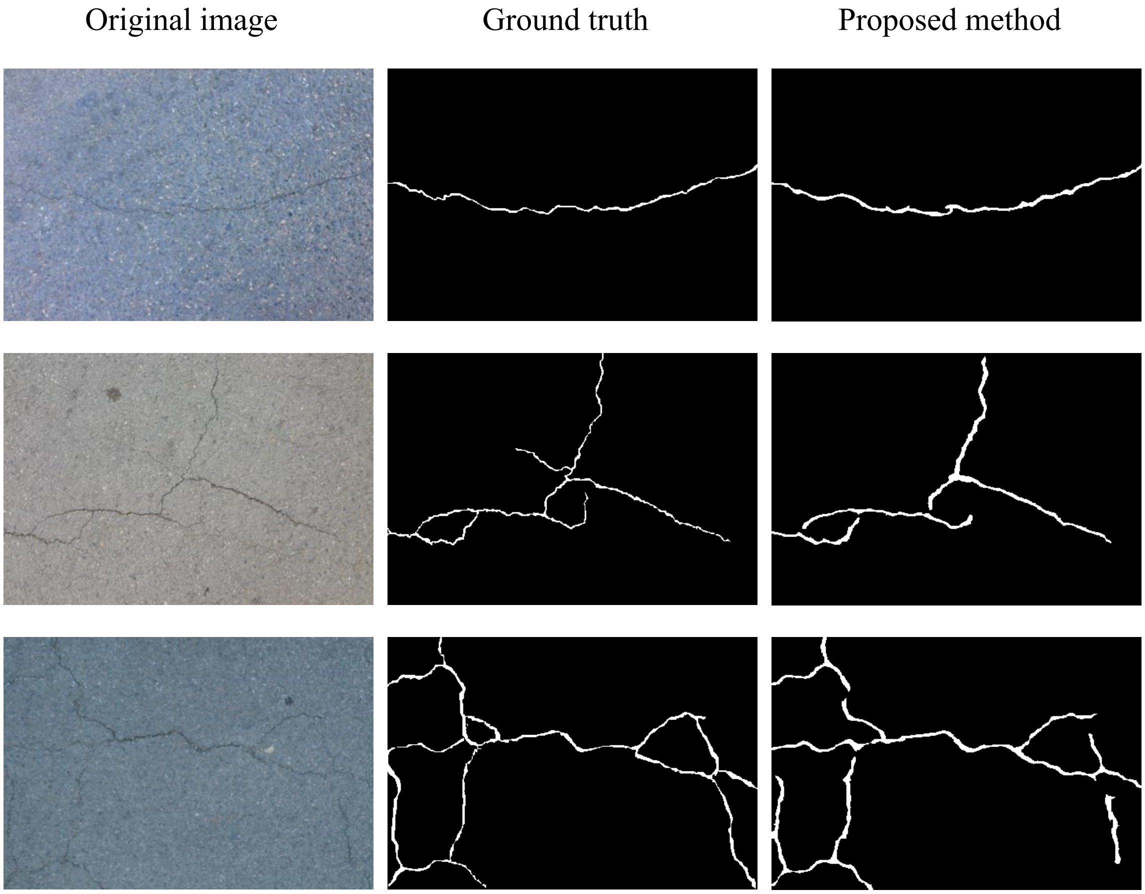


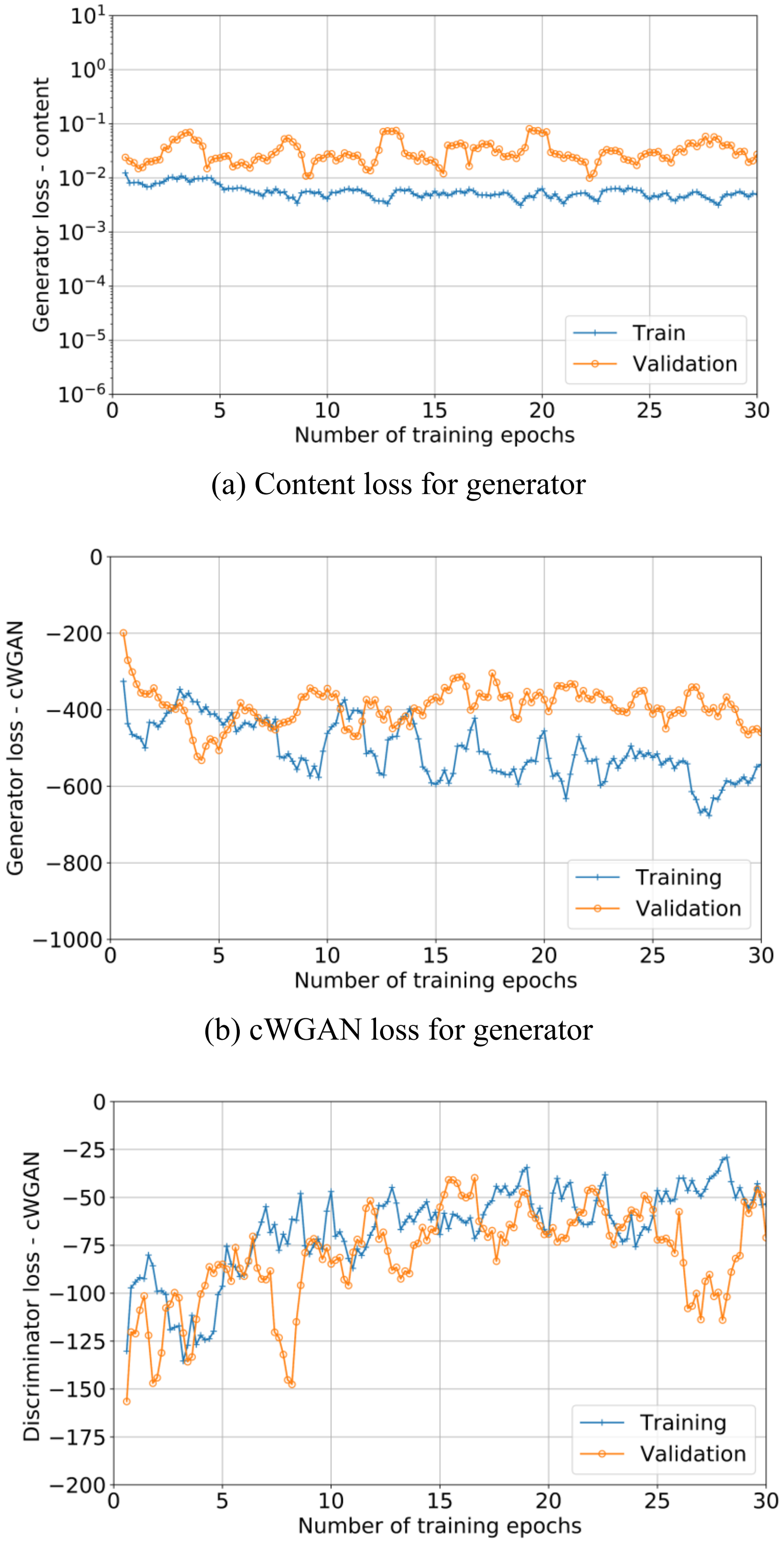
Figure 6.13 - Sample results for the CFD dataset

6.4.2 Performance on EdmCrack600 dataset

After pretraining on ImageNet and CFD datasets, the proposed method, ConnCrack, is further trained and tested on EdmCrack600 dataset. The images in EdmCrack600 are shuffled and split into 420/60/120 for training, validating and testing purposes. Similar to the pretraining, the images are first split into 256×256 patches, and are then integrated. The learning rate for the training is 1×10^{-5}, and λ is set to 5×10^{-6}. The batch size of 16 is used. The losses for training and validation sets are presented in Figure 6.14. In the figures, as the training proceeds, we can see the content loss for generator barely reduces, but the cWGAN loss decreases. This demonstrates the superior training performance of the proposed method than traditional encoder-decoder networks because there is an additional source for weight updating. The discriminator loss increases as the training goes on because the predictions output by the generator become more difficult to distinguish.

The performance of the proposed method in terms of precision, recall and F1 score is presented in Table 6.4. The Sobel and Canny detectors are standard edge detection techniques [141]. CrackIT was proposed by Oliveira and Correia [128, 144] using a series of image processing techniques. ResNet152-FCN [30] and VGG19-FCN [82] created encoder-decoder networks as suggested by [138] with ResNet152 and VGG19 as backbone networks, respectively. U-Net was introduced by [85] for crack detection. All seven methods were tested on a desktop with Intel 8700k CPU, 32GB memory and Nvidia Titan V GPU with 5120 CUDA cores where Canny, Sobel and CrackIT methods were run on CPU and the other 4 deep learning based method were run on GPU. It should be noted U-Net was not pretrained on ImageNet and CFD. We can see in the table the proposed method outperforms other methods including other deep learning-based methods with large margin.

Some sample results from the proposed method and existing methods are presented in Figure 6.15. We can see that rule-based methods cannot tackle with such complex situations where the cracks are mixed with illumination changes, shadows of trees, etc. The deep learning-based methods perform significantly better. In these methods, the illumination change and the texture of the pavement surfaces are not identified as cracks. However, ResNet152-FCN, VGG19-FCN and U-Net which utilize binary crack mask generates scattered output as described in section 4.2. Also, the noise appears at different locations in the results from those three methods. The proposed method overcomes the abovementioned issues using connectivity maps and DFS based thresholding, which results in more than 5% improvement in terms of F1 score. The computational efficiency is calculated based on the total time including deep neural network processing, post processing and outputting the results. Regarding computational efficiency, the proposed method is slightly slower than VGG19-FCN but faster than ResNet152-FCN and U-Net.



(a) Content loss for generator

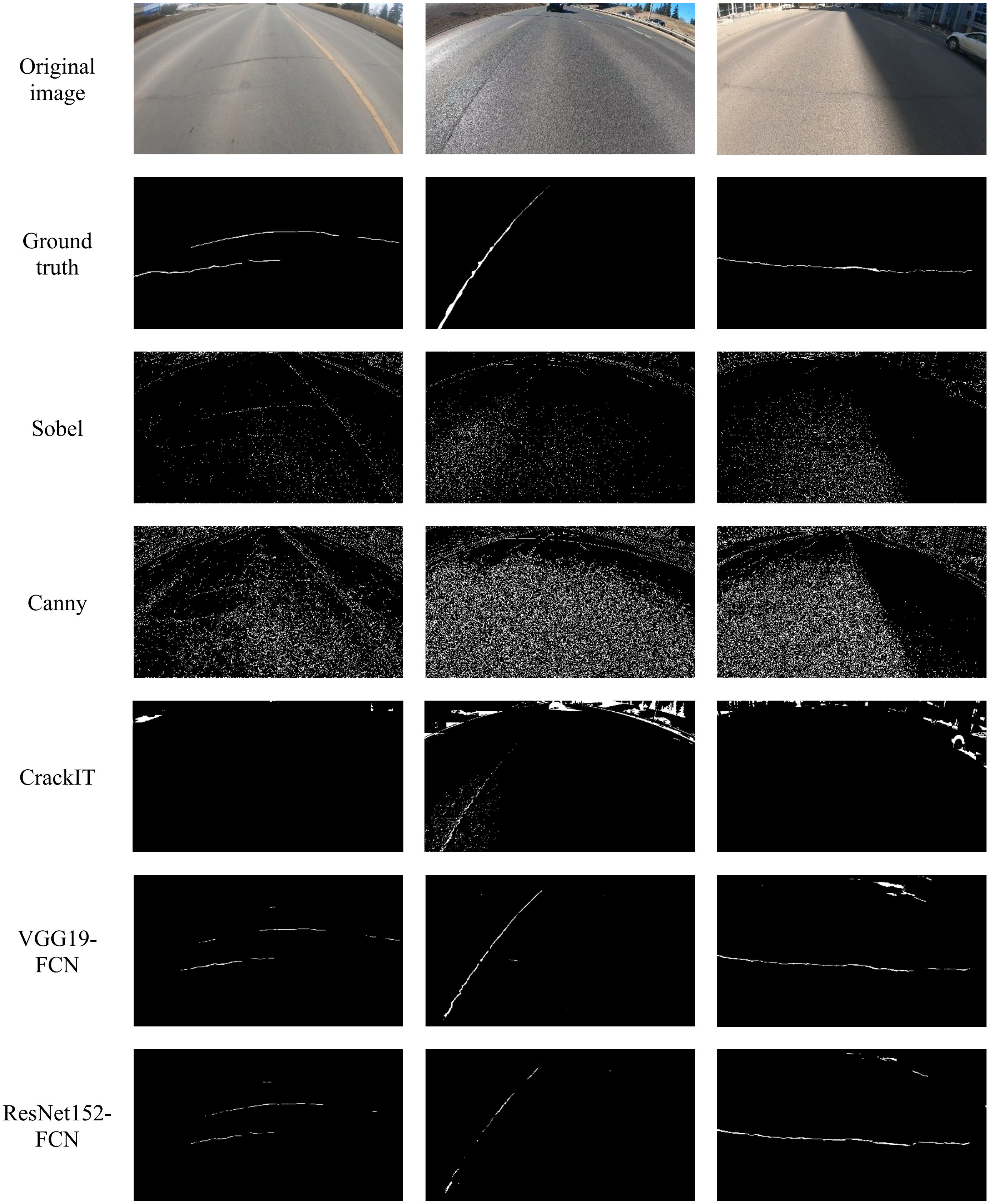
(b) cWGAN loss for generator

(c) cWGAN loss for discriminator

Figure 6.14 - Losses for EdmCrack600 dataset

Table 6.4 - Comparison of performance for different methods on the EdmCrack600 dataset

Method	**Precision**	**Recall**	**F1 Score**	**Efficiency (sec/image)**
Canny	1.69%	34.17%	3.14%	0.12
Sobel	3.00%	15.24%	4.66%	0.04
CrackIT	12.33%	7.14%	4.75%	6.71
ResNet152-FCN	78.98%	56.51%	62.78%	1.94
VGG19-FCN	80.22%	59.93%	65.18%	1.33
U-Net	76.33%	70.88%	71.52%	2.58
ConnCrack	80.88%	76.64%	76.98%	1.56


Original image
Ground truth
Sobel
Canny
CrackIT
VGG19-FCN
ResNet152-FCN

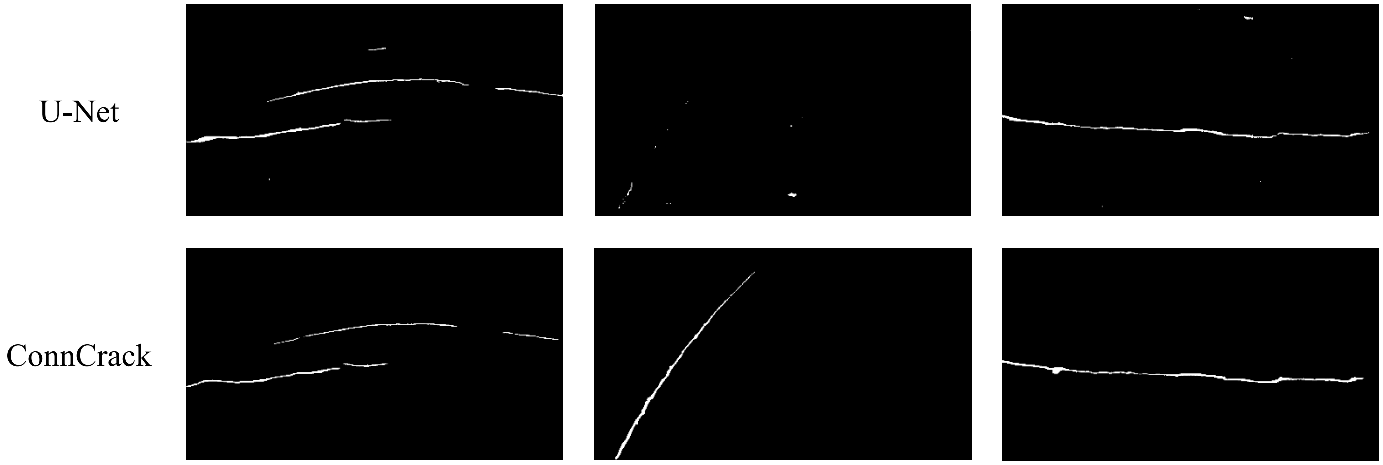


Figure 6.15 - Sample images and corresponding results for EdmCrack600 dataset

Although ConnCrack has achieved superior performance than other state-of-the-art methods, it still has difficulty in identifying the cracks correctly in some images. Figure 6.16 shows two examples of wrongly identified images. In the left plots, the crack at the bottom was not identified by the ConnCrack. Looking at the original image, it is seen that the bottom crack is relatively blurry than other parts. This could be the reason that the proposed method cannot identify it properly. In the right plots, there are a lot of shadows from trees on the road surface. Although ConnCrack can identify the long and thick crack correctly, it also misidentifies some of the shadows as cracks. One possible solution to these issues is to collect more data with such critical cases.

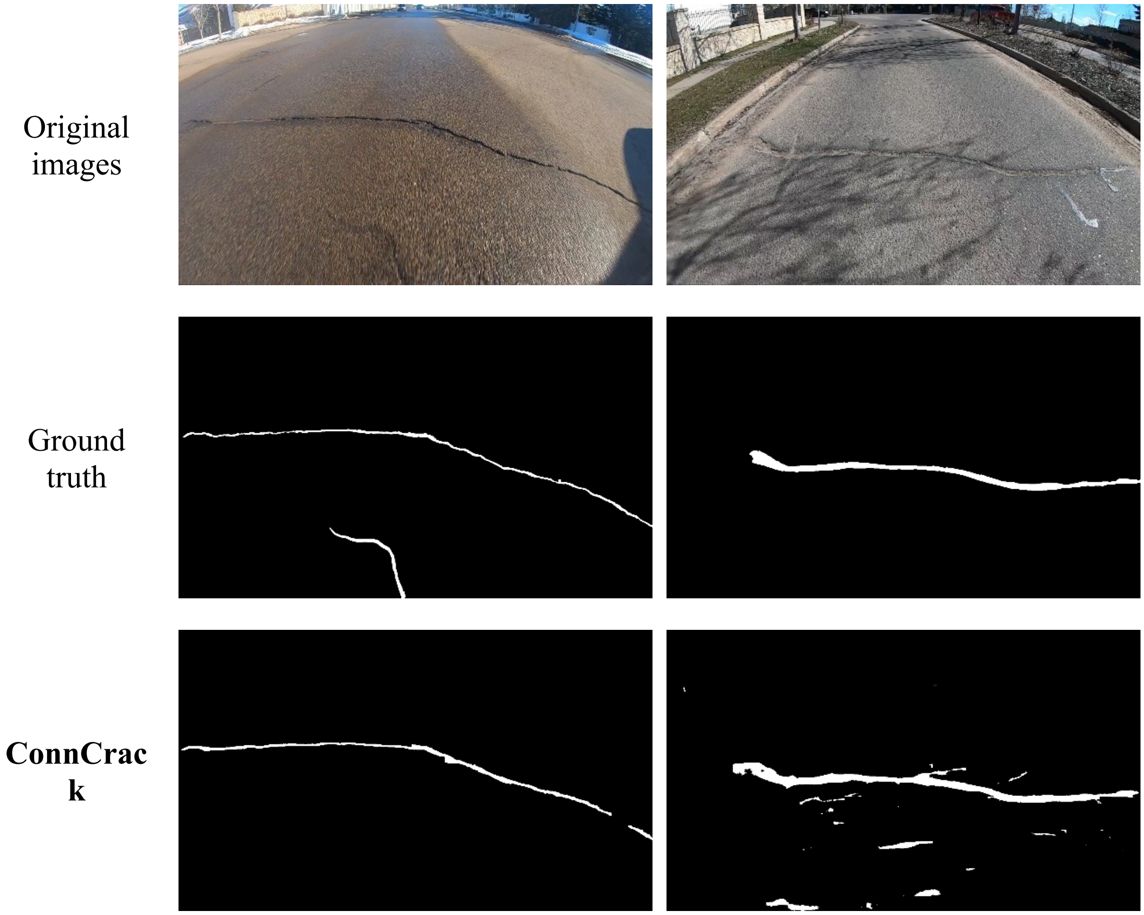
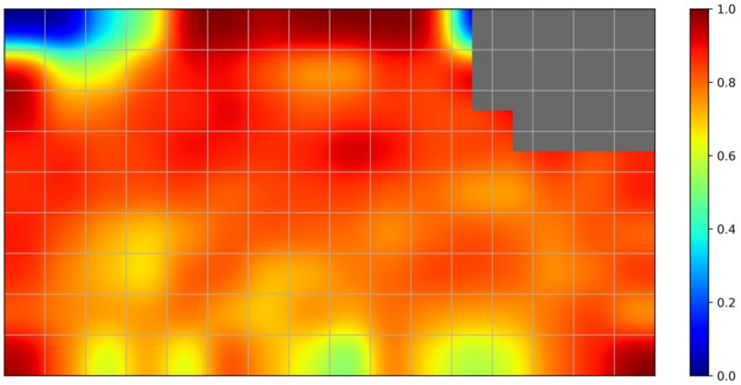
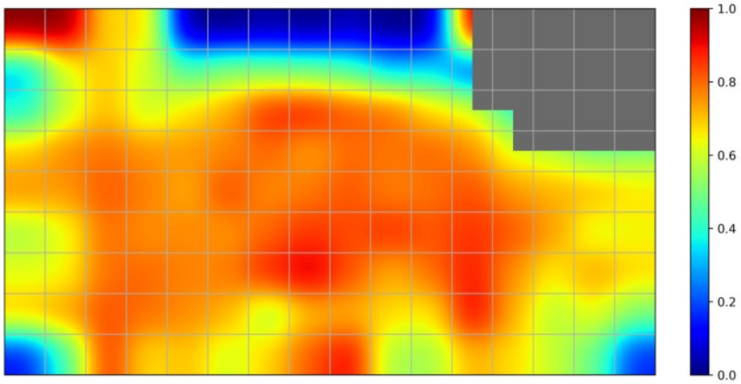


Figure 6.16 - Two wrongly identified images

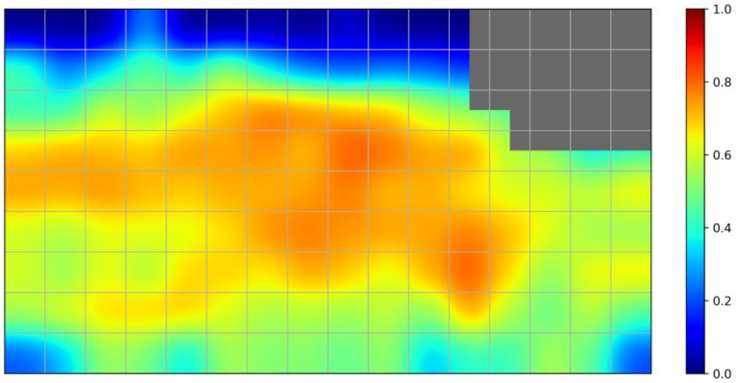
In the dataset, the images are taken in perspective view. The parts that are farther from the center of the image have lower spatial resolutions (as explained in section 3). In this study, the perspective is not taken into consideration during the training and testing process, but it is meaningful to know how the perspective view affects the performance of the proposed method. In Figure 6.17, all 120 images with 1920×1080 pixels in the test set are split into 16×9 grids. The precision, recall and F1 score are calculated for each small region separately for all 120 test images. The heat maps are generated for all three metrics where red means 100% and blue stands for 0%. The gray color represents no existence of cracks in that area. Looking at the Figure 6.16(a), there is no significant difference in different regions in terms of precision except the top left corner. This means the precision is not very sensitive to the spatial resolution of the image. However, Figure 6.17(b) shows that the recall is more sensitive to the location. The parts that are closer to the edges and corners have lower recall, which means the number of false negative pixels is higher in these regions. This shows that the proposed method is unlikely to predict the pixels that are too far from the centerline as cracks. This is because of the distortion and low resolution at the edges of images. As a combination of precision and recall, the F1 score has similar pattern as recall (see Figure 6.17(c)).



(a) Precision for different regions



(b) Recall for different regions



(c) F1 score for different regions

Figure 6.17 - Performance of different regions of EdmCrack600 dataset

6.5 Conclusions

In this chapter, a cost-effective pavement crack detection solution using commercial grade in-vehicle camera and deep neural networks is introduced. A deep learning-based algorithm called ConnCrack is proposed combining cWGAN and connectivity maps. The proposed method is first pretrained on ImageNet [9] and CFD dataset [71], and then trained and tested on EdmCrack600 dataset collected through our introduced road inspection solution. The following conclusions are drawn from this study:

1) Commercial grade in-vehicle cameras are feasible for road crack inspection.
2) The introduction of connectivity maps and cWGAN can improve the performance for crack detection.
3) The proposed method, ConnCrack, can outperform other existing methods on both publicly available dataset and our collected data.

CHAPTER 7: A PLATFORM TO MANAGE CROWDSENSING DATA FOR CIVIL INFRASTRUCTURE MONITORING

7.1 Overview

The crowdsensing data needs to be properly managed and analyzed to be useful for managing the civil infrastructure. This chapter introduces a software platform to manage the crowdsensing data. As shown in Figure 7.1, the crowdsensing process using moving vehicles can be summarized in layered architecture, which includes sensing layer, communication layer, data layer and application layer. The platform introduced in this chapter will mainly focus on the developments on data and application layers.

In this platform, the vibration data used in Chapters 3, 4 and 5 are retrieved from the remote databased through a MATLAB program. It should be noted that the Android app used for data acquisition has capabilities for pre-processing, and therefore only a few features need to be transmitted. The image data used in Chapter 6 are synchronized with the mobile devices through Google Drive Application Programming Interface (API) in Python. Algorithms proposed in previous chapters are integrated in this platform to analyze the data. The results can be stored on a web-based interactive system with visualization function developed in Python.

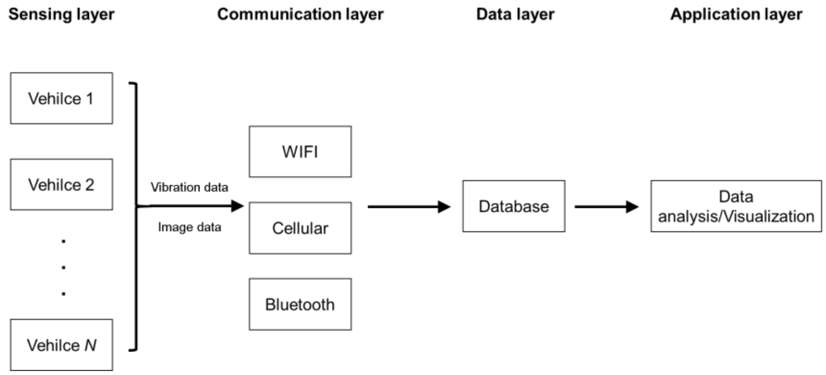
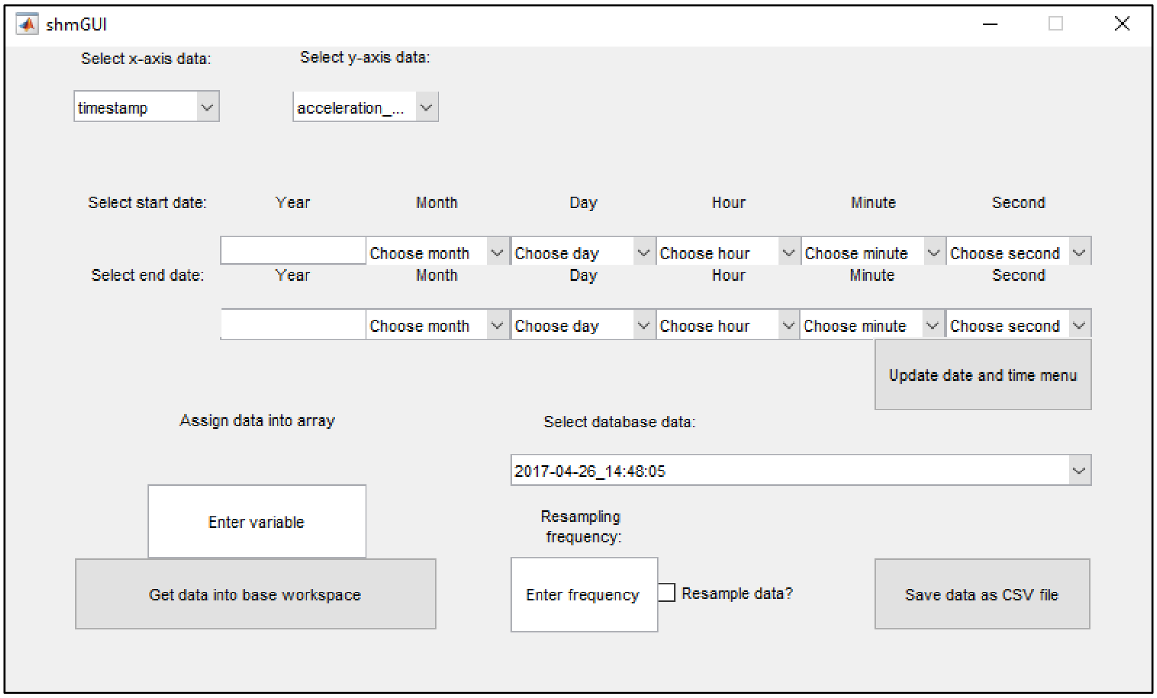


Figure 7.1 - Layer architecture of the crowdsensing based transportation infrastructure monitoring system

7.2 Data Acquisition System for Vibration Data in In-vehicle Smartphones

A data acquisition system is developed to retrieve data from the remote database built on the Amazon Web Services (AWS). The system is developed based on MATLAB. The Graphical User Interface (GUI) is presented in Figure 7.2. The program has the ability to extract tri-axial acceleration data within selected date range. It can also resample the data to the target frequency. It should be noted that the program can convert the acceleration in local coordinate system to earth's coordinate system using magnetic field data in the smartphone. The magnetic field data can be accessed through the Android API described here [145].



Select x-axis data: timestamp

Select y-axis data: acceleration_...

Select database data: 2017-04-26_14:48:05

Figure 7.2 – GUI of the data acquisiton system for smartphone data

7.3 Data Acquisition System for Image Data in In-vehicle Cameras

In this platform, the image data taken from an in-vehicle camera is transmitted to the local database through Google Drive API. The database to store the image data is developed using SQLite.

7.4 Web-based System for Analysis and Visualization

In previous sections, programs are developed to retrieve data from the mobile devices in moving vehicles. This section introduces a web-based system to manage and visualize the results to support decision-making. This system is developed in Python based on Django and Leaflet. The results are stored in SQLite database.

7.4.1 Visualization of Results from Cameras

With the high success rate of road crack detection using the method proposed in Chapter 6, further analysis is conducted by synthesizing the camera data with GPS signals collected by the GoPro. To quantitatively reflect the road condition, a simple road deterioration index (RDI) is designed as shown in Eq. (7.1).

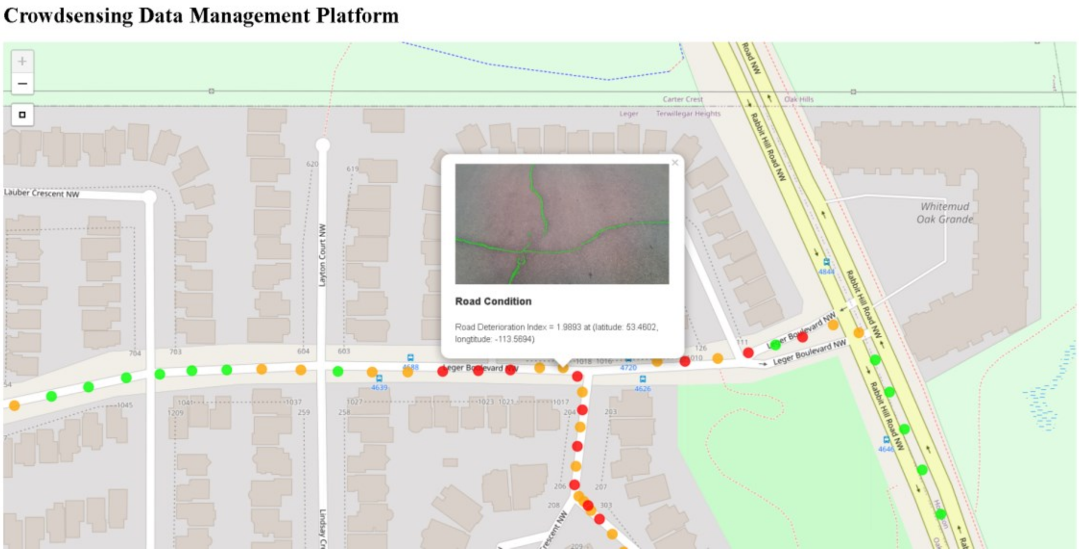
$$\mathrm{RDI} = \frac{N_{crack}}{N_{total}} \times 1000 \tag{7.1}$$

where N_{crack} is the number of pixels that are identified as crack in an image and N_{total} is the total number of pixels in the image, i.e., resolution. A higher RDI represents a worse road condition in terms of cracks.

In the web-based system, an interactive map similar to the Google Map traffic is generated according to the RDI for a series of images taken at 0.5 sec intervals. In Figure 7.3, it is seen that each dot represents a data point. The images with a RDI smaller than 0.5 are colored as green, the images with a crack index between 0.5 and 2 are yellow, and the others with a crack index higher than 2 are indicated in red. In this way, the road condition can be easily assessed and locations with unfavorable road conditions can easily be identified.

When a dot is clicked, the rendered image with cracks highlighted related to that location and the detailed information of the RDI will be presented. The information integrated in this system can be used to support decision making process. For example, from this map, the governments can know which area they should pay more attention.

An administration webpage is also developed as shown in Figure 7.4 which has access to the database directly. Information about the road datapoints can be added and modified here.

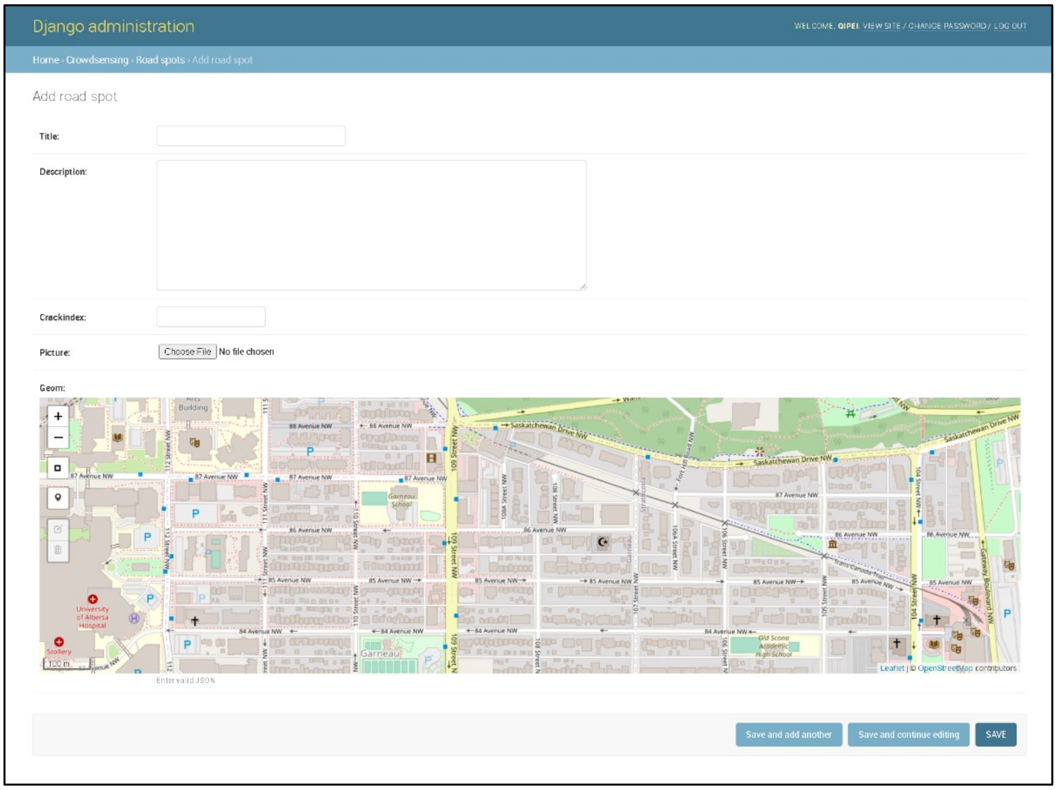


Crowdsensing Data Management Platform

Road Condition

Road Deterioration Index = 1.9893 at (latitude: 53.4602, longtitude: -113.5694)

Figure 7.3 - Screen shot of the system for road condition monitoring



Django administration
WELCOME, QIPEI. VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Crowdsensing › Road spots › Add road spot

Add road spot

Title:

Description:

Crackindex:

Picture: Choose File No file chosen

Geom:

Saskatchewan Drive NW
University of Alberta Hospital
Garneau School
Old Scona Academic High School
Leaflet | © OpenStreetMap contributors

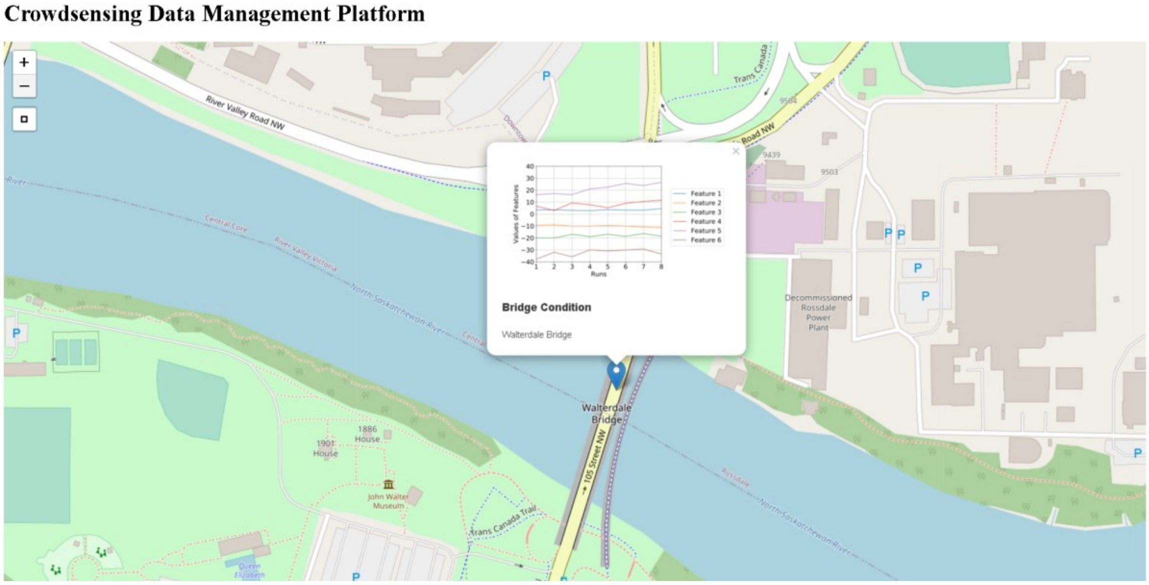
Enter valid JSON

Save and add another
Save and continue editing
SAVE

Figure 7.4 - Adminstration webpage to add and change road data points

7.4.2 Visualization of Results from Smartphone Accelerometers

Similar to the results from the roads, the features calculated using the Mel-frequency cepstral analysis proposed in Chapters 3 and 4 are presented in this system to assess the health condition of the bridge. In Figure 7.5, we can see a sample screenshot from such system. The features are plotted against different runs. The anomaly can be observed if the features are significantly shifted.



Crowdsensing Data Management Platform

Bridge Condition

Walterdale Bridge

Figure 7.5 – Screen shot of the system for bridge condition monitoring

Also, as shown in Figure 7.6, there is a webpage that can be used to manually update and maintain the existing data in the system. Program can be written to automatically upload the results to the system for decision making purpose.

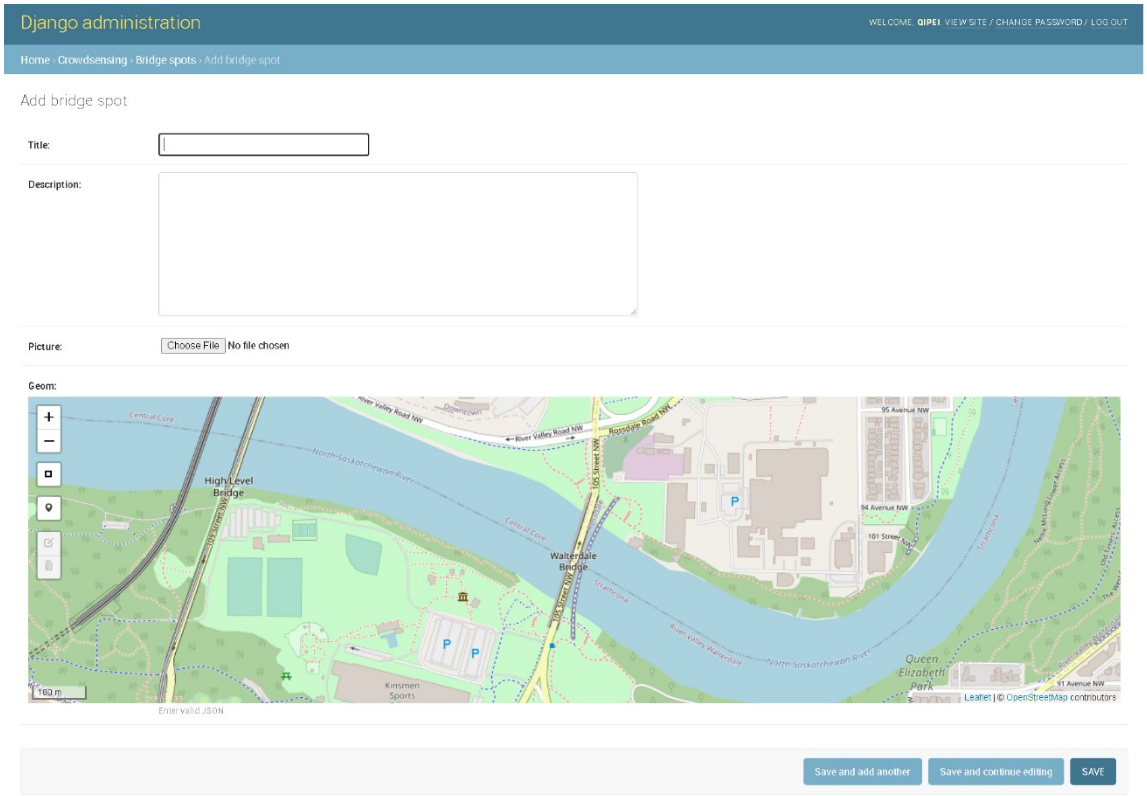


Figure 7.6 - Adminstration webpage to add and change bridge data points

7.5 Conclusions

This chapter presents a novel system to manage and analyze the crowdsensing data collected from two sensors investigated in this thesis. Efforts are made to automate the whole management process. This system bridges the gaps between the theoretical algorithms to the real decision-making process, which would be useful for decision makers.

CHAPTER 8: CONCLUSIONS AND RECOMMONDATIONS FOR FUTURE RESEARCH

8.1 Summary and Conclusions

This thesis proposes a crowdsensing-based framework for monitoring of transportation infrastructure using moving vehicles. Two commercial-grade sensors, i.e. vibration sensor and in-vehicle camera, are studied under this framework. They are used for bridge and road condition assessment, respectively.

In this thesis, methodologies are developed for identifying damage and mode shapes of bridges using vibration sensors in moving vehicles. In chapters 3 and 4, a methodology that consists of feature extraction and distribution comparison is proposed to identify damage in bridges. In these two chapters, MFCCs, which have been widely used in speech recognition, are introduced to drive-by bridge damage detection. PCA and KL divergence are used for distribution comparison in these two chapters, respectively. Finite element analysis and laboratory experiments using professional sensors and smartphones are conducted to verify this method. It is shown that the sensors and smartphones can successfully identify the existence of damage from data collected from a large number of varied moving vehicles, which shows the potential in crowdsensing based bridge condition assessment. In chapter 5, a theoretical study is conducted on mode shape identification of bridges using sensors in moving vehicles. In this study, the mode shape identification problem is first transformed into a matrix completion problem using a process called mapping. Then, soft-imputing algorithm is used to fill the sparse matrix. At last, SVD is applied to the estimated matrix for mode shape identification. Numerical analyses have shown that the method can find the mode shapes of the bridges using only two mobile sensors. Also, this method is robust with noise and road roughness.

In chapter 6, an in-vehicle camera is used to develop a cost-effective and scalable solution for road crack detection. Mounting GoPro on the rear of a vehicle, a deep learning-based algorithm is proposed for automatic crack detection at pixel level. The method can output the exact location of cracks within the image instead of drawing bounding boxes around them. Introducing densely connected layers and a novel loss function called connectivity maps into the algorithm, the system can scan the roads quickly with high accuracy. Chapter 7 presents a platform to manage and analyze the crowdsensing data from the smart devices.

In summary, it is demonstrated in this thesis that the crowdsensing-based monitoring framework may offer a promising high-level pre-screening tool which has the potential to monitor various components of the transportation infrastructure system with reduced costs and increased efficiency.

8.2 Recommendations for Future Research

While the thesis shows that the proposed framework has the potential to monitor a large number of civil infrastructures with reduced costs, there are still challenges that have to be addressed by further studies before they can be used in real life applications. The recommendations for future research include:

1) In current study of bridge condition assessment, simplified finite element models and laboratory experiments are conducted. Some important factors such as environmental effects are ignored. In future studies, more realistic models that consider environmental effects, such as wind loads or temperature effects, should be considered.
2) In the idealized models for vehicle-bridge interaction, operational effects, such as the influence of traffic, human factors and road roughness, during the data collection are ignored. In future studies, these factors should be taken into consideration.
3) Damage localization should be studied based on the mode shape information identified from moving vehicles. Experiments should be conducted to verify the mode shape identification method proposed in Chapter 5. Future research could be conducted to use multiple vehicles at different locations and at different times for damage localization.
4) In terms of the research that uses camera for road health monitoring, current study only focuses on pavement crack detection. In future studies, algorithms should be

developed to automatically identify other defects such as potholes, depressions and rutting. Also, since there are very limited pavement data with annotation, investigation should be taken on how to use less labelling data to better identify the defects.

5) Since the framework developed in this thesis is based on the data collection from general public, privacy issues should be studied.

REFERENCES

[1] H. Chourabi, T. Nam, S. Walker, J.R. Gil-Garcia, S. Mellouli, K. Nahon, T.A. Pardo, H.J. Scholl, Understanding Smart Cities: An Integrative Framework, in: 2012 45th Hawaii International Conference on System Sciences, 2012, pp. 2289-2297.

[2] E.H. Gomes, M.A. Dantas, D.D.D. Macedo, C.R.D. Rolt, J. Dias, L. Foschini, An infrastructure model for smart cities based on big data, International Journal of Grid and Utility Computing, 9 (2018) 322-332.

[3] M. Angelidou, A. Psaltoglou, N. Komninos, C. Kakderi, P. Tsarchopoulos, A. Panori, Enhancing sustainable urban development through smart city applications, Journal of Science and Technology Policy Management, 9 (2018) 146-169.

[4] American Soceity of Civil Engineers, 2017 Infrastructure Report Card, http://www.infrastructurereportcard.org/wp-content/uploads/2017/01/Bridges-Final.pdf, 2017

[5] Canadian Infrastructure Report Card, Monitoring the State of Canada's Core Public Infrastructure, in, 2019.

[6] M.R. Azim, M. Gül, Damage Detection of Steel-Truss Railway Bridges Using Operational Vibration Data, Journal of Structural Engineering, 146 (2020) 04020008.

[7] H. Ham, T.J. Kim, D. Boyce, Assessment of economic impacts from unexpected events with an interregional commodity flow and multimodal transportation network model, Transportation Research Part A: Policy and Practice, 39 (2005) 849-860.

[8] A. Mondoro, D.M. Frangopol, M. Soliman, Optimal risk-based management of coastal bridges vulnerable to hurricanes, Journal of Infrastructure Systems, 23 (2016) 04016046.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, Ieee, 2009, pp. 248-255.

[10] M.R. Azim, M. Gül, Damage detection of steel girder railway bridges utilizing operational vibration response, Structural Control and Health Monitoring, 26 (2019) e2447.

[11] T. Nagayama, S.-H. Sim, Y. Miyamori, B. Spencer Jr, Issues in structural health monitoring employing smart sensors, Smart Structures and Systems, 3 (2007) 299-320.

[12] A.H. Alavi, H. Hasni, N. Lajnef, K. Chatti, Continuous health monitoring of pavement systems using smart sensing technology, Construction and Building Materials, 114 (2016) 719-736.

[13] A. Zhang, K.C. Wang, B. Li, E. Yang, X. Dai, Y. Peng, Y. Fei, Y. Liu, J.Q. Li, C. Chen, Automated pixel-level pavement crack detection on 3D asphalt surfaces using a deep-learning network, Computer-Aided Civil and Infrastructure Engineering, 32 (2017) 805-819.

[14] Y.O. Ouma, M. Hahn, Wavelet-morphology based detection of incipient linear cracks in asphalt pavements from RGB camera imagery and classification using circular Radon transform, Advanced Engineering Informatics, 30 (2016) 481-499.

[15] J.C. Pais, S.I. Amorim, M.J. Minhoto, Impact of traffic overload on road pavement performance, Journal of transportation Engineering, 139 (2013) 873-879.

[16] N. Thom, Principles of pavement engineering, Thomas Telford London, 2008.

[17] T.S. Nguyen, M. Avila, S. Begot, Automatic detection and classification of defect on road pavement using anisotropy measure, in: 2009 17th European Signal Processing Conference, IEEE, 2009, pp. 617-621.

[18] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, H. Balakrishnan, The pothole patrol: using a mobile sensor network for road surface monitoring, in: Proceedings of the 6th international conference on Mobile systems, applications, and services, ACM, 2008, pp. 29-39.

[19] S. Chambon, J.-M. Moliard, Automatic road pavement assessment with image processing: review and comparison, International Journal of Geophysics, 2011 (2011).

[20] A. Mohan, S. Poobal, Crack detection using image processing: A critical review and analysis, Alexandria Engineering Journal, 57 (2018) 787-798.

[21] Y. Fei, K.C. Wang, A. Zhang, C. Chen, J.Q. Li, Y. Liu, G. Yang, B. Li, Pixel-Level Cracking Detection on 3D Asphalt Pavement Images Through Deep-Learning-Based CrackNet-V, IEEE Transactions on Intelligent Transportation Systems, (2019).

[22] C.V. Dung, L.D. Anh, Autonomous concrete crack detection using deep fully convolutional neural network, Automation in Construction, 99 (2019) 52-58.

[23] Q. Mei, M. Gül, Multi-level feature fusion in densely connected deep-learning architecture and depth-first search for crack segmentation on images collected with smartphones, Structural Health Monitoring, (2020) 1475921719896813.

[24] Q. Mei, M. Gül, M.R. Azim, Densely connected deep neural network considering connectivity of pixels for automatic crack detection, Automation in Construction, 110 (2020) 103018.

[25] X. Yu, E. Salari, Pavement pothole detection and severity measurement using laser imaging, in: 2011 IEEE INTERNATIONAL CONFERENCE ON ELECTRO/INFORMATION TECHNOLOGY, 2011, pp. 1-5.

[26] J. Laurent, J.F. Hébert, D. Lefebvre, Y. Savard, Using 3D laser profiling sensors for the automated measurement of road surface conditions, in: 7th RILEM international conference on cracking in pavements, Springer, 2012, pp. 157-167.

[27] W. Ouyang, B. Xu, Pavement cracking measurements using 3D laser-scan images, Measurement Science and Technology, 24 (2013) 105204.

[28] S.C. Radopoulou, I. Brilakis, Improving road asset condition monitoring, Transportation Research Procedia, 14 (2016) 3004-3012.

[29] W. Vavrik, L. Evans, S. Sargand, J. Stefanski, PCR evaluation: considering transition from manual to semi-automated pavement distress collection and analysis, (2013).

[30] S. Bang, S. Park, H. Kim, H. Kim, Encoder–decoder network for pixel-level road crack detection in black-box images, Computer-Aided Civil and Infrastructure Engineering, (2019).

[31] Q. Mei, M. Gül, A crowdsourcing-based methodology using smartphones for bridge health monitoring, Structural Health Monitoring, (2018) 1475921718815457.

[32] T.J. Matarazzo, P. Santi, S.N. Pakzad, K. Carter, C. Ratti, B. Moaveni, C. Osgood, N. Jacob, Crowdsensing Framework for Monitoring Bridge Vibrations Using Moving Smartphones, Proceedings of the IEEE, 106 (2018) 577-593.

[33] Prerforce, How Autonomous Vehicle Standards Ensure Safety, https://www.perforce.com/blog/qac/how-autonomous-vehicle-technology-driving-coding-standards, 2017 (December 23, 2019)

[34] J. Howe, The rise of crowdsourcing, Wired magazine, 14 (2006) 1-4.

[35] D.C. Brabham, Crowdsourcing as a model for problem solving: An introduction and cases, Convergence, 14 (2008) 75-90.

[36] A. Taeihagh, Crowdsourcing, sharing economies and development, Journal of Developing Societies, 33 (2017) 191-222.

[37] M. Alipour, D.K. Harris, A big data analytics strategy for scalable urban infrastructure condition assessment using semi-supervised multi-transform self-training, Journal of Civil Structural Health Monitoring, (2020) 1-20.

[38] D.K. Harris, M. Alipour, S.T. Acton, L.R. Messeri, A. Vaccari, L.E. Barnes, The Citizen Engineer: Urban Infrastructure Monitoring via Crowd-Sourced Data Analytics, in: Structures Congress 2017, 2017, pp. 495-510.

[39] E. Ozer, M.Q. Feng, D. Feng, Citizen sensors for SHM: Towards a crowdsourcing platform, Sensors, 15 (2015) 14591-14614.

[40] Wikipedia, Crowdsourcing, https://en.wikipedia.org/wiki/Crowdsourcing, 2020 (August 4, 2020)

[41] E. Ozer, D. Feng, M.Q. Feng, Hybrid motion sensing and experimental modal analysis using collocated smartphone camera and accelerometers, Measurement Science and Technology, 28 (2017) 105903.

[42] H. Zeng, H. Park, B.L. Smith, E. Parkany, Feasibility assessment of a smartphone-based application to estimate road roughness, KSCE Journal of Civil Engineering, 22 (2018) 3120-3129.

[43] H. Maeda, Y. Sekimoto, T. Seto, T. Kashiyama, H. Omata, Road damage detection and classification using deep neural networks with smartphone images, Computer-Aided Civil and Infrastructure Engineering, 33 (2018) 1127-1141.

[44] Y.B. Yang, C.W. Lin, J.D. Yau, Extracting bridge frequencies from the dynamic response of a passing vehicle, Journal of Sound and Vibration, 272 (2004) 471-493.

[45] H. Hattori, X. He, F.N. Catbas, H. Furuta, M. Kawatani, A bridge damage detection approach using vehicle-bridge interaction analysis and Neural Network technique, in: Bridge Maintenance, Safety, Management, Resilience and Sustainability, CRC Press, 2012, pp. 376-383.

[46] J. Kim, J.P. Lynch, Experimental analysis of vehicle–bridge interaction using a wireless monitoring system and a two-stage system identification technique, Mechanical Systems and Signal Processing, 28 (2012) 3-19.

[47] Y. Zhang, S.T. Lie, Z. Xiang, Damage detection method based on operating deflection shape curvature extracted from dynamic response of a passing vehicle, Mechanical Systems and Signal Processing, 35 (2013) 238-254.

[48] Z. Li, Damage identification of bridges from signals measured with a moving vehicle, in: HKU Theses Online (HKUTO), The University of Hong Kong (Pokfulam, Hong Kong), 2014.

[49] J. Keenahan, E.J. OBrien, P.J. McGetrick, A. Gonzalez, The use of a dynamic truck–trailer drive-by system to monitor bridge damping, Structural Health Monitoring, 13 (2014) 143-157.

[50] F. Cerda, S. Chen, J. Bielak, J.H. Garrett, P. Rizzo, J. Kovacevic, Indirect structural health monitoring of a simplified laboratory-scale bridge model, Smart Structures and Systems, 13 (2014) 849-868.

[51] X. He, M. Kawatani, T. Hayashikawa, C.-W. Kim, F.N. Catbas, H. Furuta, A structural damage detection approach using train-bridge interaction analysis and soft computing methods, Smart Structures and Systems, 13 (2014) 869-890.

[52] E.J. OBrien, A. Malekjafarian, A. González, Application of empirical mode decomposition to drive-by bridge damage detection, European Journal of Mechanics-A/Solids, 61 (2017) 151-163.

[53] C.-W. Kim, K.-C. Chang, P.J. McGetrick, S. Inoue, S. Hasegawa, Utilizing moving vehicles as sensors for bridge condition screening—a laboratory verification, Sensors and Materials, 29 (2017) 153-163.

[54] D. Hester, A. González, A discussion on the merits and limitations of using drive-by monitoring to detect localised damage in a bridge, Mechanical Systems and Signal Processing, 90 (2017) 234-253.

[55] J. Bu, S. Law, X. Zhu, Innovative bridge condition assessment from dynamic response of a passing vehicle, Journal of Engineering Mechanics, 132 (2006) 1372-1379.

[56] P.J. McGetrick, C.-W. Kim, A. González, E.J. Brien, Experimental validation of a drive-by stiffness identification method for bridge monitoring, Structural Health Monitoring, 14 (2015) 317-331.

[57] J.D. Sitton, D. Rajan, B.A. Story, Bridge frequency estimation strategies using smartphones, Journal of Civil Structural Health Monitoring, (2020) 1-14.

[58] A. Malekjafarian, P.J. McGetrick, E.J. OBrien, A review of indirect bridge monitoring using passing vehicles, Shock and Vibration, 2015 (2015).

[59] Y. Oshima, K. Yamamoto, K. Sugiura, Damage assessment of a bridge based on mode shapes estimated by responses of passing vehicles, Smart Structures and Systems, 13 (2014) 731-753.

[60] A. Malekjafarian, E.J. OBrien, Identification of bridge mode shapes using short time frequency domain decomposition of the responses measured in a passing vehicle, Engineering Structures, 81 (2014) 386-397.

[61] E.J. OBrien, A. Malekjafarian, A mode shape-based damage detection approach using laser measurement from a vehicle crossing a simply supported bridge, Structural Control and Health Monitoring, 23 (2016) 1273-1286.

[62] T.J. Matarazzo, S.N. Pakzad, Structural identification for mobile sensing with missing observations, Journal of Engineering Mechanics, 142 (2016) 04016021.

[63] A. Malekjafarian, E.J. OBrien, On the use of a passing vehicle for the estimation of bridge mode shapes, Journal of Sound and Vibration, 397 (2017) 77-91.

[64] Y. Yang, Y. Li, K. Chang, Constructing the mode shapes of a bridge from a passing vehicle: a theoretical study, Smart Structures and Systems, 13 (2014) 797-819.

[65] S.S. Eshkevari, S. Pakzad, Bridge Structural Identification Using Moving Vehicle Acceleration Measurements, in: Dynamics of Civil Structures, Volume 2, Springer, 2019, pp. 251-261.

[66] S.S. Eshkevari, S.N. Pakzad, Signal reconstruction from mobile sensors network using matrix completion approach, in: Topics in Modal Analysis & Testing, Volume 8, Springer, 2020, pp. 61-75.

[67] S.S. ESHKEVARI, M. TAKÁC, S.N. PAKZAD, S.S. ESHKEVARI, High resolution bridge mode shape identification via matrix completion approach, Structural Health Monitoring 2019, (2019).

[68] M. Gavilán, D. Balcones, O. Marcos, D.F. Llorca, M.A. Sotelo, I. Parra, M. Ocaña, P. Aliseda, P. Yarza, A. Amírola, Adaptive road crack detection system by pavement classification, Sensors, 11 (2011) 9628-9657.

[69] Q. Zou, Y. Cao, Q. Li, Q. Mao, S. Wang, CrackTree: Automatic crack detection from pavement images, Pattern Recognition Letters, 33 (2012) 227-238.

[70] R. Amhaz, S. Chambon, J. Idier, V. Baltazart, Automatic crack detection on two-dimensional pavement images: An algorithm based on minimal path selection, IEEE Transactions on Intelligent Transportation Systems, 17 (2016) 2718-2729.

[71] 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 (2016) 3434-3445.

[72] Y. Hu, C.-x. Zhao, H.-n. Wang, Automatic pavement crack detection using texture and shape descriptors, IETE Technical Review, 27 (2010) 398-405.

[73] S. Mathavan, M. Rahman, K. Kamal, Use of a self-organizing map for crack detection in highly textured pavement images, Journal of Infrastructure Systems, 21 (2014) 04014052.

[74] J. Redmon, A. Farhadi, Yolov3: An incremental improvement, arXiv preprint arXiv:1804.02767, (2018).

[75] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2961-2969.

[76] L. Zhang, F. Yang, Y.D. Zhang, Y.J. Zhu, Road crack detection using deep convolutional neural network, in: Image Processing (ICIP), 2016 IEEE International Conference on, IEEE, 2016, pp. 3708-3712.

[77] Y.J. Cha, W. Choi, O. Büyüköztürk, Deep learning-based crack damage detection using convolutional neural networks, Computer-Aided Civil and Infrastructure Engineering, 32 (2017) 361-378.

[78] K. Gopalakrishnan, S.K. Khaitan, A. Choudhary, A. Agrawal, Deep Convolutional Neural Networks with transfer learning for computer vision-based data-driven pavement distress detection, Construction and Building Materials, 157 (2017) 322-330.

[79] M. Alipour, D.K. Harris, Increasing the robustness of material-specific deep learning models for crack detection across different materials, Engineering Structures, 206 (2020) 110157.

[80] H. Nhat-Duc, Q.-L. Nguyen, V.-D. Tran, Automatic recognition of asphalt pavement cracks using metaheuristic optimized edge detection algorithms and convolution neural network, Automation in Construction, 94 (2018) 203-213.

[81] F. Ni, J. Zhang, Z. Chen, Pixel-level crack delineation in images with convolutional feature fusion, Structural Control and Health Monitoring, (2019) e2286.

[82] X. Yang, H. Li, Y. Yu, X. Luo, T. Huang, X. Yang, Automatic pixel-level crack detection and measurement using fully convolutional network, Computer-Aided Civil and Infrastructure Engineering, 33 (2018) 1090-1109.

[83] M. Alipour, D.K. Harris, G.R. Miller, Robust pixel-level crack detection using deep fully convolutional neural networks, Journal of Computing in Civil Engineering, 33 (2019) 04019040.

[84] X. Zhang, D. Rajan, B. Story, Concrete crack detection using context-aware deep semantic segmentation network, Computer-Aided Civil and Infrastructure Engineering, (2019).

[85] 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.

[86] D.G. Childers, D.P. Skinner, R.C. Kemerait, The cepstrum: A guide to processing, Proceedings of the IEEE, 65 (1977) 1428-1443.

[87] F. Zheng, G. Zhang, Z. Song, Comparison of different implementations of MFCC, Journal of Computer Science and Technology, 16 (2001) 582-589.

[88] W. Han, C.-F. Chan, C.-S. Choy, K.-P. Pun, An efficient MFCC extraction method in speech recognition, in: Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on, IEEE, 2006, pp. 4 pp.

[89] N. Dave, Feature extraction methods LPC, PLP and MFCC in speech recognition, International journal for advance research in engineering and technology, 1 (2013) 1-4.

[90] N. Bochud, A.M. Gomez, G. Rus, J.L. Carmona, A.M. Peinado, Robust parametrization for non-destructive evaluation of composites using ultrasonic signals, in: Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on, IEEE, 2011, pp. 1789-1792.

[91] U. Dackermann, W.A. Smith, R.B. Randall, Damage identification based on response-only measurements using cepstrum analysis and artificial neural networks, Structural Health Monitoring, 13 (2014) 430-444.

[92] G. Zhang, R.S. Harichandran, P. Ramuhalli, Application of noise cancelling and damage detection algorithms in NDE of concrete bridge decks using impact signals, Journal of Nondestructive Evaluation, 30 (2011) 259-272.

[93] L. Balsamo, R. Betti, H. Beigi, A structural health monitoring strategy using cepstral features, Journal of Sound and Vibration, 333 (2014) 4526-4542.

[94] D. O'shaughnessy, Speech communication: human and machine, Universities press, 1987.

[95] T. Litman, Smart Transportation Investments II: Reevaluating The Role Of Public Transit For Improving Urban Transportation, (2006).

[96] A. Zanella, N. Bui, A. Castellani, L. Vangelista, M. Zorzi, Internet of things for smart cities, IEEE Internet of Things journal, 1 (2014) 22-32.

[97] T. Matarazzo, M. Vazifeh, S. Pakzad, P. Santi, C. Ratti, Smartphone data streams for bridge health monitoring, Procedia Engineering, 199 (2017) 966-971.

[98] A.-M. Yan, G. Kerschen, P. De Boe, J.-C. Golinval, Structural damage diagnosis under varying environmental conditions—part I: a linear analysis, Mechanical Systems and Signal Processing, 19 (2005) 847-864.

[99] F. Magalhães, A. Cunha, E. Caetano, Vibration based structural health monitoring of an arch bridge: from automated OMA to damage detection, Mechanical Systems and Signal Processing, 28 (2012) 212-228.

[100] D. Garcia, R. Palazzetti, I. Trendafilova, C. Fiorini, A. Zucchelli, Vibration-based delamination diagnosis and modelling for composite laminate plates, Composite Structures, 130 (2015) 155-162.

[101] C. Zang, M. Imregun, Structural damage detection using artificial neural networks and measured FRF data reduced via principal component projection, Journal of Sound and Vibration, 242 (2001) 813-827.

[102] D. Tibaduiza, L. Mujica, J. Rodellar, Damage classification in structural health monitoring using principal component analysis and self-organizing maps, Structural Control and Health Monitoring, 20 (2013) 1303-1316.

[103] S. Park, J.-J. Lee, C.-B. Yun, D.J. Inman, Electro-mechanical impedance-based wireless structural health monitoring using PCA-data compression and k-means clustering algorithms, Journal of Intelligent Material Systems and Structures, 19 (2008) 509-520.

[104] I.T. Jolliffe, Principal Component Analysis and Factor Analysis, in: Principal component analysis, Springer, 1986, pp. 115-128.

[105] V. Pakrashi, A. O'Connor, B. Basu, A bridge-vehicle interaction based experimental investigation of damage evolution, Structural Health Monitoring, 9 (2010) 285-296.

[106] Y.-B. Yang, J.-D. Yau, Vehicle-Bridge Interaction Element for Dynamic Analysis, Journal of Structural Engineering, 123 (1997) 1512-1518.

[107] Y.B. Yang, C.W. Lin, Vehicle–bridge interaction dynamics and potential applications, Journal of Sound and Vibration, 284 (2005) 205-226.

[108] P.J. McGetrick, A. Gonzlez, E.J. OBrien, Theoretical investigation of the use of a moving vehicle to identify bridge dynamic parameters, Insight-Non-Destructive Testing and Condition Monitoring, 51 (2009) 433-438.

[109] A. González, E.J. Obrien, P.J. McGetrick, Identification of damping in a bridge using a moving instrumented vehicle, Journal of Sound and Vibration, 331 (2012) 4115-4131.

[110] Y. Zhang, L. Wang, Z. Xiang, Damage detection by mode shape squares extracted from a passing vehicle, Journal of Sound and Vibration, 331 (2012) 291-307.

[111] L. Muda, M. Begam, I. Elamvazuthi, Voice recognition algorithms using mel frequency cepstral coefficient (MFCC) and dynamic time warping (DTW) techniques, arXiv preprint arXiv:1003.4083, (2010).

[112] K.S.R. Murty, B. Yegnanarayana, Combining evidence from residual phase and MFCC features for speaker recognition, IEEE signal processing letters, 13 (2006) 52-55.

[113] S. Kullback, R.A. Leibler, On information and sufficiency, The annals of mathematical statistics, 22 (1951) 79-86.

[114] D.S. Berry, D.M. Belmont, Distribution of vehicle speeds and travel times, in: Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, The Regents of the University of California, 1951.

[115] A. Dhamaniya, S. Chandra, Speed characteristics of mixed traffic flow on urban arterials, International Journal of Civil, Architectural Science and Engineering, 7 (2013) 330-335.

[116] J. Duchi, Derivations for linear algebra and optimization, Berkeley, California, (2007).

[117] Y. Jin, H.-S. Toh, W.-S. Soh, W.-C. Wong, A robust dead-reckoning pedestrian tracking system with low cost sensors, in: Pervasive Computing and Communications (PerCom), 2011 IEEE International Conference on, IEEE, 2011, pp. 222-230.

[118] R. Mazumder, T. Hastie, R. Tibshirani, Spectral regularization algorithms for learning large incomplete matrices, Journal of machine learning research, 11 (2010) 2287-2322.

[119] A.V. Oppenheim, Discrete-time signal processing, Pearson Education India, 1999.

[120] Wikipedia, Gross vehicle weight rating, https://en.wikipedia.org/wiki/Gross_vehicle_weight_rating, 2020

[121] J.P. Yang, W.-C. Lee, Damping effect of a passing vehicle for indirectly measuring bridge frequencies by EMD technique, International Journal of Structural Stability and Dynamics, 18 (2018) 1850008.

[122] Q. Mei;, M. Gül;, EdmCrack600, https://github.com/mqp2259/EdmCrack600, 2019 (July 10)

[123] H. Maeda, Y. Sekimoto, T. Seto, T. Kashiyama, H. Omata, Road damage detection using deep neural networks with images captured through a smartphone, arXiv preprint arXiv:1801.09454, (2018).

[124] D. Saranga, Kia Sorento (2013), https://www.the-blueprints.com/blueprints/cars/kia/56201/view/kia_sorento_2013/, 2019 (July 10)

[125] M. Eisenbach, R. Stricker, D. Seichter, K. Amende, K. Debes, M. Sesselmann, D. Ebersbach, U. Stoeckert, H.-M. Gross, How to get pavement distress detection ready for deep learning? A systematic approach, in: 2017 international joint conference on neural networks (IJCNN), IEEE, 2017, pp. 2039-2047.

[126] F. Yang, L. Zhang, S. Yu, D. Prokhorov, X. Mei, H. Ling, Feature Pyramid and Hierarchical Boosting Network for Pavement Crack Detection, arXiv preprint arXiv:1901.06340, (2019).

[127] M. Eisenbach, R. Stricker, D. Seichter, K. Amende, K. Debes, M. Sesselmann, D. Ebersbach, U. Stoeckert, H. Gross, How to get pavement distress detection ready for deep learning? A systematic approach, in: 2017 International Joint Conference on Neural Networks (IJCNN), 2017, pp. 2039-2047.

[128] H. Oliveira, P.L. Correia, CrackIT — An image processing toolbox for crack detection and characterization, in: 2014 IEEE International Conference on Image Processing (ICIP), 2014, pp. 798-802.

[129] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Advances in neural information processing systems, 2014, pp. 2672-2680.

[130] M. Kampffmeyer, N. Dong, X. Liang, Y. Zhang, E.P. Xing, ConnNet: A Long-Range Relation-Aware Pixel-Connectivity Network for Salient Segmentation, IEEE Transactions on Image Processing, 28 (2019) 2518-2529.

[131] P. Isola, J.-Y. Zhu, T. Zhou, A.A. Efros, Image-to-image translation with conditional adversarial networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1125-1134.

[132] J.-Y. Zhu, T. Park, P. Isola, A.A. Efros, Unpaired image-to-image translation using cycle-consistent adversarial networks, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2223-2232.

[133] M. Mirza, S. Osindero, Conditional generative adversarial nets, arXiv preprint arXiv:1411.1784, (2014).

[134] M. Arjovsky, S. Chintala, L. Bottou, Wasserstein gan, arXiv preprint arXiv:1701.07875, (2017).

[135] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, A.C. Courville, Improved training of wasserstein gans, in: Advances in neural information processing systems, 2017, pp. 5767-5777.

[136] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4700-4708.

[137] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE, 86 (1998) 2278-2324.

[138] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 3431-3440.

[139] A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, arXiv preprint arXiv:1511.06434, (2015).

[140] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Transactions on knowledge and data engineering, 22 (2010) 1345-1359.

[141] R.C. Gonzalez, P. Wintz, Digital image processing, 1987.

[142] T.S. Nguyen, S. Begot, F. Duculty, M. Avila, Free-form anisotropy: A new method for crack detection on pavement surface images, in: 2011 18th IEEE International Conference on Image Processing, IEEE, 2011, pp. 1069-1072.

[143] H. Li, D. Song, Y. Liu, B. Li, Automatic Pavement Crack Detection by Multi-Scale Image Fusion, IEEE Transactions on Intelligent Transportation Systems, (2018) 1-12.

[144] H. Oliveira, P.L. Correia, Automatic road crack detection and characterization, IEEE Transactions on Intelligent Transportation Systems, 14 (2012) 155-168.

[145] Android Developers, SensorManager, https://developer.android.com/reference/android/hardware/SensorManager, 2020 (August 11, 2020)