THESIS

# A STREAMLINED BRIDGE INSPECTION FRAMEWORK UTILIZING UNMANNED AERIAL VEHICLES (UAVS)

Submitted by

Brandon J. Perry

Department of Civil and Environmental Engineering

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Summer 2019

Master's Committee:

Advisor: Yanlin Guo

Rebecca Atadero
John W. van de Lindt
Ross Beveridge

ABSTRACT

A STREAMLINED BRIDGE INSPECTION FRAMEWORK UTILIZING UNMANNED AERIAL VEHICLES (UAVS)

The lack of quantitative measures and location information for instances of damage results in human-based bridge inspections that are variable and subjective in nature. With bridge owners and managers tasked with making major maintenance/repair decisions with inadequate funding and resources, it is appealing to develop a transparent bridge inspection and evaluation system that integrates field inspection and documentation of damage with quantitative measures and geo-referenced locations in a holistic process. A new, streamlined bridge inspection framework based on unmanned aerial vehicles (UAVs) is proposed to improve the efficiency, cost-effectiveness, and objectivity of these inspections while enhancing the safety of inspectors. Since the current bridge inspection practices use a component based structural rating system, the new UAV-based bridge inspection system should also follow a component-wise damage evaluation system to enable the seamless adoption of this new technology into practice. To provide bridge managers/owners with the streamlined decision-making support, this new system uniquely integrates UAV-based field inspection, automated damage/defect identification, and establishment of an element-wise As-Built Building Information Model (AB-BIM) for the damage documentation in a holistic manner. In this framework, a UAV platform carrying visual sensors first collects data for identifying defects (i.e. cracks, spalling and scaling of concrete). Next, an automated damage detection algorithm is developed to quickly extract quantitative damage information (i.e. type, size, amount, and location) from the data. By using UAV-enabled photogrammetry and unsupervised machine learning techniques, this system can automatically segment the bridge elements (i.e. beam, girders, deck, etc.) from a 3D point-cloud with minimal user input. In the end, the damage information is mapped to the corresponding structural components of the bridge and readily visualized in the AB-BIM. The docu-

mented element-wise damage information with quantitative measures in conjunction with the 3D visualization function in the proposed system can provide bridge managers with a transparent condition evaluation and a one-stop decision making support which can greatly ease the planning of repair/maintenance. The feasibility of this approach is demonstrated using a case study of a Colorado bridge.

ACKNOWLEDGEMENTS

I would first like to thank my advisor Dr. Yanlin Guo at Colorado State University. Dr. Guo was always available and eager to assist whenever I had a question about research, writing, or life.

I would also like to thank the my committee members and other professors, who without their passionate participation and input, this project would not be successful: Dr. Rebecca Atadero, Dr. John W. van de Lindt, and Dr. Ross Beveridge.

I would like to recognize the help from Mr. Jin Wang of the City of Fort Collins, Mrs. Marilyn Hilgenberg of the City of Loveland, Mrs. Morgan Fay of Larimer County, and the Colorado Department of Transportation on this project. A sincere appreciation also goes to Mr. Greg Black as acting as a liaison between Colorado State University and the surrounding government officials.

# DEDICATION

*I would like to dedicate this thesis to my parents who have shown me what hard work and dedication is and to my friends and family for their constant support.*

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

# Chapter 1

# Introduction

With a large number of bridges in the United States that require regular inspection to ensure proper health and maintenance, there is a need for more efficient bridge inspection techniques that provide bridge owners and/or managers with more robust data to make better decisions given the limited resources available. Using current human-based bridge inspection procedures, results are often more subjective and qualitative while being cumbersome, expensive, and time-consuming to attain. In the United States, bridges are inspected at least once every two years. Of the 600,000 bridges, 9.1% are considered "structurally deficient," meaning either an element of the bridge is rated in the "poor" condition (0 to 4 on the National Bridge Inventory rating scale) or the current load carrying capacity is significantly below current design standards, and, therefore, require more frequent inspections [1, 2]. There is a need to develop an inspection framework that is more cost-efficient in terms of time and money while affording more quantitative and consistent results to ensure that state's departments of transportation can more effectively balance demand. A solution to stream-line the bridge inspection process by leveraging Unmanned Aerial Vehicles (UAVs) is proposed. In this chapter, the need for developing a new framework is highlighted along with a literature review. Lastly, the objectives of this study and structure of the thesis are outlined.

## 1.1 Motivation of Developing an Enhanced Bridge Inspection Framework

After the collapse of the Silver Bridge in Ohio in 1967, which was the most horrific bridge failure in the United States, it was determined that a routine assessment of bridges was needed to ensure public safety and welfare; therefore, the National Bridge Inspection Standards (NBIS) was created and later amended [3]. The NBIS lays out the minimum requirements for the fre-

quency and type of inspection and the qualifications of those who conduct the inspections for bridges. The accepted techniques and procedures for bridge inspections, however, are in the Federal Highway Administration's (FHWA) Bridge Inspection Reference Manual and The American Association of State Highway Transportation Officials' (AASHTO) Manual for Bridge Evaluation [4, 5]. Both references typically incorporate visual and sounding techniques to assess the bridge's performance and health [6]. These conventional techniques can be relatively subjective, unsafe, timely, and costly due to their reliance on the human inspectors' skill, experience, and, in the case of chain-dragging or sounding techniques, hearing ability [7].

For instance, Moore et al. demonstrated the variance and subjectivity of routine bi-yearly inspections. They had multiple crews perform a routine inspection on the same structures. It was found that 68% of the ratings shows $\pm 1$ point variation and 27% of ratings have $\pm 2$ points variation; while for the rest 5% of ratings, the variance is greater than $\pm 2$ [8]. This indicates an obvious inconsistency in rating data. It is also worth noting that this inconsistent rating is even more susceptible to structurally deficient bridges, where accurate data is even more critical for safe operations and planning.

Safety concerns pose a risk to inspectors on site. To gain access to the underside of large-scale bridges during in-depth inspections, roping rigs (Figure 1.1a), scaffolding, or "Snooper" Trucks (Figure 1.1b) are often required. These systems give great accessibility to the underside of the bridge and bring the components "within-arm's-reach" to the inspector, which is currently required for certain types of inspections by FHWA and AASHTO; however, extensive training and safety procedures are required. This not only puts the inspectors in a dangerous position due to the inherent risks of the rigs, but also to the proximity to vehicular traffic, especially with distracted and unsafe drivers.

Moreover, a significant amount of time is required for large-scale and in-depth inspections, typically requiring two-weeks or more for set-up, inspection, and tear-down of the equipment. Additional time is required for the final report generation in-office. With this long turn-around time, fewer bridges can be inspected by an inspector per season. During this time of on-site

**(a)** Roping Rig [9]

**(b)** "Snooper" Tuck [10]

**Figure 1.1:** Current Techniques for Accessing the Underside of a Bridge During Inspections using Roping Rigs (a) and a "Snooper" Truck (b).

inspection, highway lanes, railways, and/or boat routes may be continually altered or closed, negatively impacting traffic flow.

Lastly, the monetary impact of these inspections can be significant. The most obvious is the cost of renting/buying/operating these systems and the wages of the inspectors and crews. Additionally, the economic impact of the road closures and route alterations have an associated cost. Furthermore, there is a deadweight loss of not having the best data to make the most informed and logical maintenance and/or repair decisions, which would promote longevity of the current infrastructure.

These shortcomings of the current practice highlight the need to develop a more consistent, efficient, and cost-effective decision-making support system while maintaining a high level of safety for inspectors. Such a system would provide bridge owners and decision makers with quantitative data to plan accordingly with the limited resources available. Recently, UAVs based remote sensing technology has emerged as a promising tool to aid in the decision-making process.

## 1.2 Literature Review of Current UAV-based Inspections

With recent technology developments, UAVs have greatly improved in performance and become more user-friendly and affordable. Multi-rotor UAVs can maintain a stable position without much movement or vibration even in windy and non-preferred environmental conditions [11]. Recent advances in visual navigation systems have allowed UAVs to hover in place without the need of a strong Global Positioning System (GPS) signal by using optical, infrared, and/or ultrasonic sensors [12, 13]. The price and size of a complete system have reduced considerably, making the technology more accessible for government and research institutions. Moreover, the camera technology has continuously evolved, with improved image quality and resolution.

Several studies have been developed incorporating UAVs to inspect different types of infrastructure. Li et al. were able to develop a complete system to automatically identify visual defects in photo-voltaic panels including snail trails and dust shading [14]. Chiu et al. used a UAV to identify potential hazards in a large floating membrane cover at a water treatment plant by finding the elevation changes of the cover with a digital elevation model created with photogrammetry [15]. Much success has been achieved in the inspection of overhead transmission lines and the technology is currently being implemented [16]. More specifically in bridge inspection, Hernandez et al. developed a customized UAV system with tethered data relay and power to photograph bridges during an inspection [11]. Gillins et al. also deployed a UAV to inspect bridges; however, these two studies only used the obtained photographs as an aid to the inspectors, providing "eyes-in-the-sky" without locating, quantifying, or measuring damage to assess bridge condition [11, 17]. Furthermore, Duque et al. used a UAV system to photograph two wooden bridges and manually identify the defects in the structure [18]. This study greatly aided the inspection process by providing quantified damage assessment. It did not, however, offer a complete inspection system with the integrated data collection and analysis functions, nor did it provide the ability to easily access and navigate the damage information to facilitate decision-making. Similarly, the Minnesota Department of Transportation has com-

pleted a three-phase project on utilizing UAVs to assist inspectors on a wide variety of bridges in Minnesota. They created photogrammetric models with the use of Pix4D® to log and organize the data collected from the UAVs. Wells et al. also demonstrated the potential for cost savings of about 60% using a UAV system as opposed to traditional techniques [19–22]. The photogrammetric model created greatly aided inspectors and supplied a wealth of data that would otherwise not be possible. Despite the benefits of safety, cost-effectiveness, and valuable inspection data, these studies did not provide inspectors with an efficient data processing system that merges UAV-based inspection data into current practice.

Currently, flight around a structure and the collection of data (a UAV acting as "eyes-in-the-sky") has become trivial providing that current rules and regulations are followed. However, once the data is received and downloaded, a framework to process the hundreds of images and data and attain useful information becomes more of a challenge. To provide streamlined decision-making support, a holistic inspection system that integrates UAV-based field inspection with element-wise damage identification, documentation, and visualization incorporated in an As-Built Bridge Information Model (AB-BIM) is needed.

## 1.3 Proposed UAV-Based Bridge Inspection Framework

In response to the shortcomings of traditional bridge inspection practice and the limitation of recent UAV-based bridge inspection due to the lack of adequate data processing tools, a unique bridge inspection framework, which encompasses the entire inspection process from field survey to final AB-BIM creation is proposed. This new framework has two unique features that are expected to promote fast future technology transfer into current practice: 1) to ensure seamless merging of the new technology with current inspection practice, the data analysis module of UAV-based inspection system will be constructed to provide element-wise damage assessment that is consistent with current inspection practices outlined by AASHTO and FHWA; and 2) to enable the integration of the new inspection system with CDOT's, and other state DOT's, current Geographic Information System (GIS)-based data management system, a geo-

referenced AB-BIM model will be developed. This will allow convenient and comprehensive documentation of condition assessment results with geo-referenced and element-wise damage information. This system is safer and more efficient, cost-effective, and consistent than the current techniques. Inspectors can fly the UAV system at a safe distance from traffic. Roping rigs, scaffolding, and "snooper trucks" are not required to gain access to the underside of the bridge. On-site time could be drastically reduced to a day or less, saving on time, traffic interruptions, and equipment costs. The use of data analysis algorithms enables quantification of damage, and thus can provide objective results that are reproducible and consistent throughout each inspection.

The framework of the proposed system is illustrated in Figure 1.2. A UAV platform carrying visual sensors will hover around the structure collecting data through flight missions. Post-flight, the data will pass through a series of data processing tools. With the collected photos, a 3D point-cloud and photo-realistic model will be generated using structure-from-motion. This 3-D model will serve as the base of the AB-BIM. From this data, each bridge element (e.g. beam, girders, deck, etc.) will be automatically segmented to create an element-wise model using machine learning techniques. Next, the images will pass through automated damage/defect detection algorithms and identify surface and/or subsurface defects (e.g. cracks, spalling and scaling of concrete). This will provide quantitative information on damage location, type and severity. The extracted damage information will then be mapped to each segmented element of the bridge in the AB-BIM. The AB-BIM can be uploaded to DOT's servers and accessed from the cloud to the field. The resulting AB-BIM with element-wise damage information and 3-D visualization capability will make problem areas more obvious and provide quantitative measures to allow bridge managers/owners to efficiently rate existing structures while keeping inspectors safe from traffic and on the ground.

In the subsequent sections, each component of the proposed system will be illustrated through a case study of a Colorado bridge. First, how the UAV system collected the data in the field will be introduced in Chapter 2. This chapter explores the two test structures, i.e. the

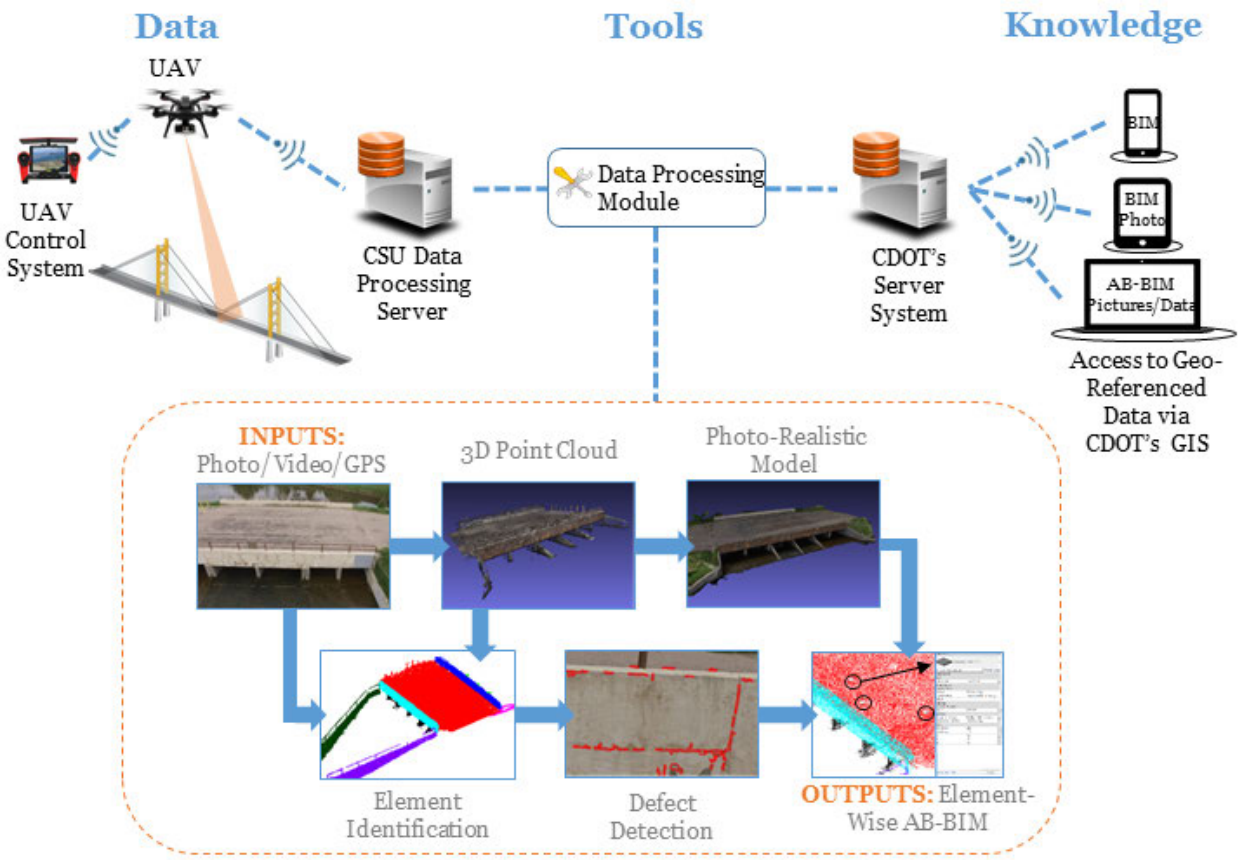



**Figure 1.2:** Proposed Automated Bridge Inspection System

"Palmer Bridge" and "Oxbow Bridge," the UAV systems used, and the flight of the UAVs including experience with Federal Aviation Administration regulations and lessons learned. Next, the methodology of each module of the data processing tools and corresponding results will be elaborated. Specifically, the modules discussed are 3D point-cloud and photo-realistic model generation using structure-from-motion in Chapter 3, defect detection and mapping onto the point-cloud in Chapter 4, element segmentation in Chapter 5, and AB-BIM establishment in Chapter 6. A comparison between the implementation of the proposed framework to current techniques is discussed in Chapter 7. Lastly, the concluding remarks and future study are explored in Chapter 8.

# Chapter 2

# Field Test

In this chapter, the technicalities and logistics of flying a UAV system around a structure will be discussed in depth. Although the technology of UAV system has advance and flight has become significantly easier and cheaper, there is still planning and practice involved before safely flying a UAV around a structure. First, the two test sites that were flown, the "Palmer Bridge" and the "Oxbow Bridge," will be examined. Next, the UAV systems used for these two bridges will be compared and analyzed. The rules and regulations, organized by the Federal Aviation Administration (FAA) will be reviewed to ensure compliance. Logistics of flight planning and preparation will be discussed. Lastly, future work of autonomous flight paths is studied.

## 2.1 Test Bridges

The structures surveyed were the "Palmer Bridge" in Southern Fort Collins, Colorado and the "Oxbox Bridge" in Loveland, Colorado. Both bridges were unique and provided different challenges within each and both were used to demonstrate the UAV's performance in the field. The "Palmer Bridge" served as a case study to demonstrate the proposed framework for UAV-enabled bridge inspections. First, the "Palmer Bridge," then the "Oxbow Bridge" will be discussed.

### 2.1.1 Palmer Bridge

Located on Palmer Drive in Fort Collins, Colorado, the "Palmer Bridge' is shown in Figure 2.1. The bridge is a spillway to a dammed-up pond with a continuous flow of water under the structure. It is composed of six concrete box girders with a 5-*inch* asphalt overlay. The top bridge surface is approximately 23-*feet* × 39.5-*feet*. This structure has a clearance of about 3-*feet* under the bridge. The last completed report was from June 5, 2013and gave the Palmer Bridge had an overall sufficiency rating of 98 and, therefore, considered "Not Deficient." How-

ever, it was reported that, “Cells 3 and 4 have S1 to S3 scaling in various locations. Pier nose 3 has a spall with exposed reinforcing. A few hairline transverse cracks in each span.” All other elements were considered to be “State 1.” The latest attempted inspection was on June 4, 2015. The inspection was not completed because it was considered too dangerous and risky to enter the culverts due to “fast water and slick bottom.”

**Figure 2.1:** North Side of Palmer Drive Test Bridge

In 2011, it was recommended to install adequate approach rails to meet AASHTO/Colorado Department of Transportation’s (CDOT) requirements. The City of Fort Collins decided the the upgrades needed were took costly, the average daily traffic flow was too low, and the inspection inside the culverts was too difficult to justify the required improvements; therefore, with support from the local neighborhood, the structure was closed to vehicular traffic and now serves only as a pedestrian bridge. Because of these reasons, this bridge provided an excellent case study for the proposed system for there being minimal, but typical damage and little to no traffic control needed during the flight.

Flight under the bridge was not possible. As emphasized in Figure 2.1, the clearance under the bridge is about 3-*feet*. The Matrice 600 Pro, discussed in Section 2.2.3, was the only UAV tested at this site and given its wide wingspan and created thrust, was deemed unsafe to fly

under the structure; therefore, images of the sides and top were taken which provided sufficient information to demonstrate the proposed framework.

Lastly, to ensure a safe flight, the location of the bridge had to be checked to verify no additional authorizations were required. It was determined that the bridge laid just outside the airspace of a regional airport. Since it was not within the protected air space of the regional airport, no FAA/airport authorization was required. However, as discussed in Section 2.3, a licensed Part 107 UAV Pilot and, as recommended, a Visual Observer, had to be present during the flight.

### 2.1.2 Oxbow Bridge

The next structure was the "Oxbow Bridge" shown in Figure 2.2. This structure was used to test the two other UAV platforms, the DJI Mavic 2 Zoom and DJI Phantom 4 Pro, discussed in Section 2.2.1 and Section 2.2.2, respectively. It is located on the western edge of the city limits in Loveland, Colorado in a new natural area called "Oxbow Natural Park." During the inspection, the bridge was a private structure over a creek to a landowner's property. Recently, the bridge and some of the surrounding lands were donated to the City of Loveland to create a bicyclist and running path around the city and parks of Loveland. Many updates were scheduled in the near future to meet safety requirements and ensure the longevity of the bridge. The Colorado State University (CSU) UAV survey team was able to collect data of the bridge before these updates began.

The bridge provided a good test site to fly under a structure. Considering the bridge was private and, therefore, had an extremely low average daily traffic, there was still some damage of the structure mainly due to the age and weather that could be identified with the proposed damage detection algorithm, discussed in Chapter 4. There was also a large clearance under the bridge; therefore, the UAVs were able to safely fly under the structure to collect upward-angled imagery. However, there were over grown trees and shrubbery around the structure which had

**Figure 2.2:** Overview of the Oxbow Bridge

to be identified and planned for accordingly. Lastly, since the structure was privately owned during the time of the flight, there was no traffic control required to ensure safe operation.

Both structures provided real-world implementation of the proposed system and demonstrated the potential of future UAV-enabled bridge inspections. Both flights were conducted safely and were considered successful. The images and data collected from the "Palmer Bridge" will be used and analyzed in subsequent sections to demonstrate the potential of UAV-enabled bridge inspections. The flights at the "Oxbow Bridge" were used to demonstrate the systems used and the feasibility of flying under a structure as discussed in Section 2.3.

## 2.2 UAV Platforms Tested

Current UAV technology has advanced significantly making applications in aerial photography, surveying, and infrastructure inspection possible, manageable, and more cost-effective. The advancements in computational processing power, optical image technologies, and GPS precision has lead to sudden increase on the UAV market place in both consumer and profes-

sional level systems making UAVs more affordable and more advanced. Leveraging all these advancements, UAVs can provide significant spatial resolution of objects from above ground providing additional perspectives and more data about areas of interest otherwise impossible or impractical to attain. Additionally, from concerns of airspace safety from governmental, private, and public entities, many consumer level UAVs are equipped with additional optical, infrared, and ultrasonic sensors to detect and avoid obstacles and collisions. These additional sensors also assist in navigation of the UAV in GPS-deprived environments which makes it feasible to fly under a bridge or in a more confined area. All of these advancements have helped to make UAV bridge inspections possible by ensuring safety, ease of piloting, and enhanced data collection while reducing costs of the system.

Three UAV platforms were tested in this study and are compared in Table 2.1. These UAV platforms were purchased for general applications in multiple research areas at CSU by the CSU Drone Center. Although not specifically acquired for the purpose of bridge inspection, all three platforms showed effectiveness in the current research applications for structure inspection and were the systems used in multiple research studies [23–30]. The systems were compared on a number of metrics including camera, size, sensor payload, and price and provides an idea of the trade-off and benefit of each system.

One should note that when comparing optical sensors, it is important to consider not only the number of pixels (megapixels), but also the camera's sensor size. This is because the resolution of real world objects captured by a camera is determined by the camera's Ground Sampling Density (GSD), which is controlled by the number of pixels, the focal length of the camera, and sensor size. The GSD is the relationship between the pixel in the image and spatial size of the object photographed in real world, for example, how many centimeters of the object/ground is represented in each pixel width in the image. It is defined as

$$\text{GSD} = \frac{S_W \cdot H}{F_R \cdot W} \tag{2.1}$$

where $S_W$ is the sensor width, $H$ is the height of the camera above-ground-level (AGL), $F_R$ is the focal length, and $W$ is the pixel width of the image. Table 2.1 compares the GSD of the platforms used in this study assuming a 5-*meter* camera height, $H$. A smaller GSD means a higher real world resolution, therefore, leading to better image performance on photogrammetric model generation and damage detection, discussed in Chapter 3 and Chapter 4, respectively.

A DJI Mavic 2 Zoom, DJI Phantom 4 Pro, and DJI Matrice 600 Pro were used in this study. These three UAVs were tested in different situations and locations. One platform did not necessarily perform better overall when compared to the others, but offered specific advantages and drawbacks which are discussed in the next subsections. For future implementation, the chosen UAV system(s) should be selected given the requirements of the job, bridge type, location, and other considerations. Each UAV is pictured in Figure 2.3 and the specifications are itemized in Table 2.1.

**(a)** DJI Mavic 2 Zoom - Wingspan: 1.05-*feet* [31]

**(b)** DJI Phantom 4 Pro - Wingspan: 1.15-*feet* [32]

**(c)** DJI Matrice 600 Pro - Wingspan: 5.47-*feet* [33]

**Figure 2.3:** UAV Platforms Tested

### 2.2.1 DJI Mavic 2 Zoom

The DJI Mavic 2 Zoom is a smaller platform with easy controllability and zoom capabilities. It is pictured in Figure 2.3a with its specifications in Table 2.1. The biggest advantage of this system is the 2X optical zoom which allows more detailed images of specific areas or defects;

**Table 2.1:** Comparison of the Tested UAV Platforms

| | DJI Matrice 600 Pro with Zenmuse X3 Camera | DJI Phantom 4 Pro V2.0 | DJI Mavic 2 Zoom |
|---|---|---|---|
| Built-in Obstacle Detection | None | 5-Directions | Omni-Directional |
| Confined Space Flight Practicality | Larger System | No Upward Obstacle Sensors | Compact |
| Resolution (MP) | 12.4 | 20 | 12 |
| Sensor (inch) | 1/2.3 | 1 | 1/2.3 |
| Optical Zoom | None | None | $2X$ |
| Flight Time (min) | 32 | 30 | 27 |
| Additional Payload | 13.2-*lbs* | No | No |
| Wingspan (feet) | 5.47 | 1.15 | 1.05 |
| Weight (lbs) | 20.9 | 3.06 | 0.675 |
| Ground Sampling Density (cm/pixel) | 0.214 | 0.137 | 0.175 |
| Flight Control and Stability | Sensitive Controls | Average | Corrected by Software |
| Price (USD) | $5,898 | $1,499 | $1,249 |

however, the zoom capabilities does not result in better images for the structure-from-motion algorithm, which creates the 3D model, due to the effect of distortion. The camera has a resolution of 12-*MP*, resulting in the GSD of 0.175-*cm/pixel* in the horizontal direction hovering at 5-*meters* AGL. The entire system is small and compact (1.05-*foot* wingspan) allowing for maneuverability and control under a structure and in tighter spaces; however, due to the small size and weight, it is more susceptible to winds. DJI accounted for this susceptibility of drift automatically in its flight software and sensor array; therefore, the controls are not as sensitive as the larger, more stable Matrice 600 Pro. The Mavic 2 Zoom has an omnidirectional obstacle sensing array that avoids collisions with objects and assists in the stability and navigation of the UAV in GPS-deprived environments. It is the only tested platform with this omnidirectional sensing capability. The Mavic 2 Zoom has a dual visual sensor array on both the front and back, a dual vision sensor and an infrared sensor on the underside, a single vision sensor on the left and right, and an infrared sensor on the top side. With this sensory array built in, if the Mavic 2 Zoom is flying within 20-*meters* AGL, the on board processor recognizes key-points of the surroundings and can either maintain its course or hover in position. The images were clear due to the stabilization of the camera from the three-axle gimbal attached. The UAV was rated with a battery life of 27 minutes; however, landing of the UAV was initiated at 20% battery remaining due to safety reasons and the full 27 minute range was not experienced. The batteries are small and it is simple to exchange.

The UAV proved to be a good candidate for general bridge inspections due to its size and enhanced ability to fly in GPS-deprived environments. A sample of the collected images are shown in Figure 2.4 with Figure 2.4a of a typical image and Figure 2.4b of an image with 2X optical zoom. The UAV, however, does not allow for additional payload or sensors. As technology progresses, different or additional payloads (i.e. LiDAR or infrared) could be required. This system does not allow for this customizability.

**(a)** DJI Mavic 2 Zoom Image Sample

**(b)** DJI Mavic 2 Zoom Image Sample with 2X Zoom

**Figure 2.4:** Sample Images Captured by DJI Mavic 2 Zoom

### 2.2.2 DJI Phantom 4 Pro

Next, a DJI Phantom 4 Pro (Figure 2.3b) was tested. This is a medium sized UAV offering obstacle avoidance and larger megapixel camera with its complete specifications in Table 2.1. The 20-*MP* camera attached does not have any optical zoom features, but delivers a 0.137-*cm/pixel* resolution at 5-*meters* AGL. It also has a similar obstacle sensor array as the Mavic 2 Zoom to avoid collisions in the air and actively avoids obstacles (a dual vision senors in the forward, backward, and downward directions, infrared sensors on the left, right, and underside, and an additional ultrasonic sensor on the underside);however, it does not have any upward-facing sensors. Therefore, the obstacle sensing ability is not as robust as the Mavic 2 Zoom. Although it can remain stable in GPS-deprived environments, it cannot sense obstacles above the aircraft which is important when flying under a structure or in a confined space. The rated flight time is similar to the Mavic 2 Zoom at around 30 minutes; however, the actual fight time was also reduced due to safety concerns and began to land when the UAV reached 20% remaining battery.

Given its ability to fly in GPS-deprived condition, this system proved applicable for flying under a structure. Additionally, the camera has a smaller GSD than the Mavic 2 Zoom. However, the larger size makes maneuverability in tighter environments more difficult. Also, the aircraft

is not customizable and additional sensors and payloads cannot be added. A sample of the images collected during a survey is shown in image Figure 2.5.

**(a)** DJI Phantom 4 Pro Image Sample 1

**(b)** DJI Phantom 4 Pro Image Sample 2

**Figure 2.5:** Sample Images Captured by DJI Phantom 4 Pro

### 2.2.3 DJI Matrice 600 Pro

Lastly, a DJI Matrice 600 Pro was tested (Figure 2.3c). This is a larger system and is extremely stable in the air and under winds. Since it is a more professional system, the reaction of the aircraft from the controls is more sensitive which creates a higher learning curve to pilot when compared to the Mavic 2 Zoom or Phantom 4 Pro. The Matrice 600 Pro has the capability of attaching up to 13.2-*lbs* of additional sensors and payload either above or below the UAV. The tested camera was a gimbaled DJI Zenmuse X3 with a 12-*MP* camera. The Zenmuse X3 has a 0.214-*cm/pixel* resolution hovering at 5-*meters* AGL, the largest GSD compared to the tested platforms. This camera can be easily exchanged between other DJI cameras, gimbals, or customized systems allowing flexibility of the sensor payload. This platform included enhanced GPS reliability with real-time kinematic (RTK) positioning technology. This allows for more precise location information of the aircraft during flight. Mulakala et al. showed survey level accuracy with UAVs using RTK positioning technology with a relative horizontal accuracy of 1.2-*cm* and, when using ground control points, 0.9-*cm* [23]. This UAV does not have standard obstacle avoidance sensors; however, there are options to add these sensors to the system as

an additional payload module. For future work, this system would allow for easy implementation of other equipment such as LiDAR, infrared sensors, or higher-quality cameras. The rated battery life is 32-*minutes*, but landing was initiated sooner for safety. The system requires six batteries, which are easy to exchange, but are cumbersome and heavy.

Because of its larger size, it would be more difficult to fly closer to a structure or in a tighter area. This system was not tested under a structure to analyse the flight performance in a GPS-deprived environment. Sample images taken by this system are shown in Figure 2.6.

**(a)** DJI Matrice 600 Pro Image Sample 1

**(b)** DJI Matrice 600 Pro Image Sample 2

**Figure 2.6:** Sample Images Captured by DJI Matrice 600 Pro

## 2.3 Flight

### 2.3.1 FAA Regulations

With the recent advancements for UAV technology prompting a significant increase in the number of hobbyist and professional UAV pilots in the past couple years, regulatory agencies are getting involved to ensure the airway safety. In the United States, the FAA regulates all airspace and creates the rules and standards professional pilots must follow. In August 2016, FAA's Part 107 Rules went into effect. The regulations most pertinent to bridge and structure inspections are listed below.

- UAV must be registered with the FAA if more than 0.55-*lbs*
- UAV's take off weight cannot exceed 55-*lbs*
- UAV cannot fly above 400-*feet* AGL unless the UAV is inspecting a structure in which case, the UAV is permitted to fly 400-*feet* above the structure given that the UAV is still within a "Class E" or "Class G" airspace
- UAV must be within "visual-line-of-sight" (VLOS) from the "person-in-control" (PIC) at all times and must maintain VLOS without the need of visual aides (i.e. binoculars or camera)
- PIC must be up to date with Part 107's required rules, licensure, and training.
- UAV is not permitted to fly above people unless the people are directly involved with the flight of the UAV
- UAV is not permitted to fly above moving vehicles
- Exceptions to most rules in Part 107 can be permitted with written consent from the FAA; however, operating over people and moving vehicles is rarely granted

The above list is not comprehensive, but gives the most pertinent rules for inspection flights. Additionally, the FAA recommends, but does not currently require, to have at least two people flying the UAV at one time. The first being the PIC in direct control of the aircraft and the second being a "visual observer" (VO). The VO is responsible for constantly maintaining eye sight with the aircraft and constantly verifying the airspace is clear and safe for the PIC. The PIC and VO should be in close proximity without communication aids (i.e. walkie-talkie or cell phone). Even-though a VO is present, the PIC should still routinely scan the airspace and maintain VLOS with the UAV. The PIC also takes full responsibility of the actions during a flight. CSU requires a PIC and at least one VO to be present during all flights; therefore, the CSU survey team met those requirements.

These regulations are not standard across the world and can vary by country, region, state, or city. Hobbyist are not required to follow Part 107; however, the rules are recommended to ensure a safe airspace. Nonetheless, in the United States, any non-hobbyist or compensated pilot is required to follow FAA's Part 107 and register the UAV in FAA's nation database. Just over a year after Part 107 adoption, one-million UAVs were registered with the FAA showing the shear number and commonplace of UAVs in the United Stated [34].

### 2.3.2 Flying Logistics

Four different flights were planned for the system. Two flights were manually controlled while the other two were autonomously flown. The autonomous flights are discussed in Section 2.3.3. In the manually controlled flights, two flying techniques were tested: flying parallel to the traffic flow and flying perpendicular to the traffic flow. Separately, both techniques produced similar photogrammetric models with equivalent quality. In the end, the image sets were combined and produced a better model due to the multiple angles captured. Although previous research recommended 50% to 60% overlap in image collection, the image set obtained from the manual flight missions had an overlap of about 95% due to the less precise control on image taking rate [15, 35]. In future studies, the image overlap rate could be reduced based on Chiu et al. and Shahnaz et al. recommendations to save on flight time, data storage, and computational expense. A subset of the images collected at the "Palmer Bridge" site is shown in Figure 2.7.

One major concern for bridge inspections is a loss of GPS signal under the structure. However, due to the advancements in UAV sensory mentioned in Section 2.2, the UAVs that were tested under the structure performed well in the GPS-deprived environment. On the Oxbow Bridge, in the open, clear space, 12 satellites were locked and positioned. Under the bridge the number of locked satellites reduced to around eight. The minimum number of satellites connections for a safe and controlled flight according to DJI is 10-12. However, because of the sensor array on the underside of the Mavic 2 Zoom and Phantom 4 Pro, the UAV was able to maintain its course and remain stable in the air under the bridge. One note is that because

**Figure 2.7:** Subset of Images Collected by the UAV at the Palmer Bridge

there was flowing water under the bridge, the UAV locked onto the key-points and descriptors of the water flow and slightly drifted in the direction of the water. This drift was maintainable once realized and control of the UAV was never lost or impeded. This caused flight of the underside of the bridge to take slightly longer than the upper side. The Matrice 600 Pro was not tested under a structure, so the effects of the GPS signal was not tested on a system without the visual guidance and obstacle avoidance sensory array.

The second concern is the safety of piloting UAV within a confined space. The test fights have shown that the obstacle avoidance sensory array installed on the UAV systems assisted to avoid collisions under the bridge effectively. For instance, the sensors on the Mavic 2 Zoom and Phantom 4 Pro, under factory settings, prevented the UAV from colliding and approaching too closely to bridge during flight. As a test to the sensors, the Mavic 2 Zoom was piloted at typical flying speed toward an object. The UAV was able to sense and avoid the object and hover at a safe distance even when the PIC manipulated the controls at full throttle toward the object. This obstacle avoidance technology ensured a safe and successful survey of the entire structure.

The third concern in particular for bridge inspection is the performance of upward facing photography under a bridge. The Mavic 2 Zoom and Phantom 4 Pro are not equipped with

upward facing cameras on the UAV; however they are able to point upwards at a 30°, and 25° angle, respectively. A sample image from the underside of each system is shown in Figure 2.8 Although this was not the best for the upward imagery, it did provide enough upwards angle to capture the underside and create a 3D point cloud model. The Phantom when pointed at the 25° angle did capture the propellers in the imagery. This did not seem to affect the 3D model or texture creation once stitched. This is shown in Figure 2.8c.

**(a)** Mavic 2 Zoom Sample Image of the Underside of Oxbow Bridge

**(b)** Phantom 4 Pro Sample Image of the Underside of Oxbow Bridge

**(c)** The Underside of the Oxbow Bridge with the Propellers in the Frame

**Figure 2.8:** A Sample of Images from the Underside of the Oxbow Bridge with the Mavic 2 Zoom (a), Phantom 4 Pro (b), and the Phantom 4 Pro with the Propellers Visible

Another consideration for bridge inspection is flying in close proximity to traffic. Although the UAV would fly within the right-of-ways and/or beside traffic (i.e. not directly over moving traffic) given its unusual and atypical nature, it may pose a risk for a distracted driver. This

could prove especially dangerous with UAV piloting crews on the ground adjacent to traffic. It is recommended to implement traffic control during flights. This traffic control will not only slow the traffic, but will make the driver cognizant of the crews on the ground.

When flying a UAV for any type of inspection project, it is important to not only follow the regulations (i.e. Part 107), but to also consider other safety concerns (i.e. traffic control). Additionally, UAVs are not completely accepted by the general public. Although it is currently legal to fly in any authorized airspace, many residents or civilian will wonder what the UAV is photographing and how the data will be used and shared. It is also important to be clear and transparent about the process to ease these worries and have an additional person present during flights to communicate to the community members so that the PIC and VO are not disturbed.

### 2.3.3 Future Automation of Flight

The feasibility of autonomous flights was tested for future applications. During the flight of the "Palmer Bridge" and "Oxbow Bridge," two consumer-level applications for autonomous flight control (PrecisionFlight by PrecisionHawk® and Flight by SkyCatch®) were used to fly the UAV autonomously [36, 37]. Using a smart phone or tablet, points around the bridge were uploaded on site and a flight plan was created within the applications. The UAV was able to automatically take off, fly around the bridge, capture images, and land without any human interaction during the flight. These two flights lasted 3-5 minutes but did not provide the best images and angles of the bridge for later photogrammetric modeling.

The reason for the limited images collection for photogrammetric modeling is due to the fix UAV elevation during the two autonomous flights. When creating a flight plan in the applications, a fixed elevation for the entire flight needs to be set; consequently, the UAV is not able to change vertical elevation during flight. When surveying land or creating digital elevation models, a fix elevation is extremely useful to evenly capture the entire area; however, for inspecting bridges or other structures, a change of elevation is needed. There are some applications (i.e. Pix4D Capture by Pix4D) which allows for a 3D path, but it only has the capability to fly in a

circular motion [22]. This is useful for the facade of a building structure, but would not provide the best results for a bridge due to the geometry difference. To improve the image quality and attain the best angles to create a robust photogrammetric model during autonomous flight, a detailed flight plan with elevation markers needs to be created and uploaded to the UAV. This would allow more precise control points with elevations; therefore, permitting better flight control and data quality. Additionally, the manually flown flight logs can be downloaded and saved providing future flights to replicate the exact same flight path. Future work discussed in Section 8.1 could use these flights plans to assist in defect propagation and more robust health monitoring.

# Chapter 3

# Creation of 3D Point-Cloud and Development of Photo-Realistic Model

An import feature of the proposed framework is the 3D visualization of damage information. The visualization serves as a way for inspectors to easily identify the damage location and retrieve quantified damage information to make more informed decisions about repair and maintenance. As discussed further in Chapter 6, an inspector is able to click on a damaged section of the bridge and get detailed, quantified damage information about that particular defect. This aids not only in localized storage of all the bridge's information, but provides a one-stop decision-making support system. To establish a base model onto which the damage information will be referenced, a point-cloud and photo-realistic model is created using structure-from-motion (SfM). The rising trend is to use 3D computer aided drawing (CAD) as a BIM during construction. These CAD models could serve as the BIM for the 3D visualization; however, with 20% of the bridges in the United States over 50 years old, these CAD models do not exist for the majority of already constructed bridges. Therefore, a point-cloud created with SfM is used as the general base for the AB-BIM and helps to visualize the geometric information and surface condition of the structure.

## 3.1 Comparison of LiDAR and Photogrammetry

To create a point-cloud or 3D points, "light detection and ranging," (LiDAR) or "photogrammetry" are often implemented to create points with associated colors and normals and generate a 3D representation of an object. Both technologies have been well researched in both structural survey and best practices and are at a mature state producing similar 3D outputs [15, 38–40]. LiDAR uses pulses of light or laser beams to measure the distance from the light source to the object. In some LiDAR systems, working in conjunction with optical sensors, RGB

values can be associated with the measured points to create real-world colorized point-cloud. In contrast, photogrammetry uses a collection of 2D images taken from various angles and locations around an object to create 3D points, the details of which will be discussed further in this chapter.

LiDAR and photogrammetry are rather different, but both offer unique advantages and drawbacks for creating a 3D point-cloud to be used as a base for the AB-BIM. A comparison of the two technologies are discussed in this paragraph and summarized in Table 3.1. Since LiDAR readings travel at the speed of light and take micro-seconds to relay (2.0-$\mu s$ at 300-*meters* AGL), the systems can take a large number of points in a short amount of time; whereas, with photogrammetry, the number of points is dependent on the number of images and GSD of the camera. Because photogrammetry relies on matching images to create the 3D points, there is a significant computation expense compared to LiDAR, which can produce points in real-time. To create these points, the only equipment required for photogrammetry is an optical sensor, typically already attached to the UAV; GPS is optional and helps to georeference the point-cloud to real-world coordinates. LiDAR systems, however, can be significantly more expensive than the UAV systems to which they are attached, mentioned in Section 2.2. An enhanced GPS system and a LiDAR sensor is needed which adds weight to the UAV and decreases battery life. LiDAR offers an advantage of penetration through coverage such as foliage or trees to the ground surface and has the ability to better detect thin, linear features such as wires and fences. These two advantages of LiDAR, however, are not necessarily preferable with inspections of bridges as these structures are large and the surroundings are cleared. To create the 3D base for the AB-BIM, photogrammetry using SfM was chosen, as it provides a cost-effective system with a longer battery life and no additional equipment required.

## 3.2 Point-Cloud Generation

Once the UAV flight to survey the bridge was completed, the the images collected by the UAV are processed using photogrammetry to create the 3D point-cloud. Photogrammetry is a

**Table 3.1:** Comparison of LiDAR and Photogrammetry

| | LiDAR | Photogrammetry |
|---|---|---|
| Data Aquisition | Pulses of Light to Measure Distances | Measurements from Large Library of Images |
| Real-Time Data Processing | Yes | Requires Significant Computational Expense |
| Additional Payload | Yes: Enhanced GPS and LiDAR Sensor | No: Optical Sensor (GPS optional) |
| Foliage Penetration | Yes | No |
| Thin, Linear Feature Detection | Yes | Not Reliably |
| Cost | $50,000+ | $1,000+ |

process of taking a 2D library of photographs and creating a 3D point-cloud using SfM. In the field study of the "Palmer Bridge," 449 collected images from the four flight missions were used to create the 3D point cloud. As mentioned in Section 2.3.2, these images have a high overlap rate, between 90% - 95%. A certain overlap rate is typically needed to ensure that there are enough matching key-points between the images to reconstruct the 3D model. As previously discussed in Chapter 2.3.2, Chiu et al. and Shahnaz et al. recommend 50% and 60% overlap rate, respectively; therefore, the overlap rate in this study was sufficient [15, 35]. To accomplish the 3D point creation, Meshroom with AliceVision, an open-source photogrammetry code that is based on SfM was used [41, 42]. SfM finds key points from every image, in this case through Scale-Invariant Feature Transform (SIFT) and stores the identified key points and their associated descriptors [43]. Each descriptor is compared through the image set using approximate nearest neighbor with L2 Distance and matched. To find incorrectly matched key-points, the a-contrario random sampling and consensus (ACRANSAC) algorithm was implemented which identifies inliers and outliers in a data set. With the found matches, and knowing the focal

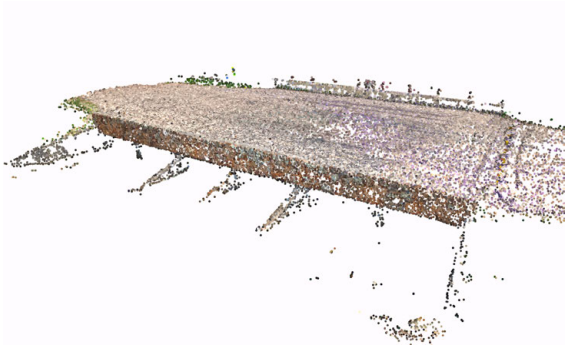
length and optical center of the camera, the pose of the camera in each image is calculated Equation (3.1).

$$\left[\begin{array}{c|c} R_C & C \\ \hline 0 & 1 \end{array}\right] = \underbrace{\left[\begin{array}{c|c} I & C \\ \hline 0 & 1 \end{array}\right]}_{\substack{\text{Translation Matrix in} \\ \text{World Coordinates}}} \times \underbrace{\left[\begin{array}{c|c} R_C & 0 \\ \hline 0 & 1 \end{array}\right]}_{\substack{\text{Rotation Matrix in} \\ \text{World Coordinates}}}$$

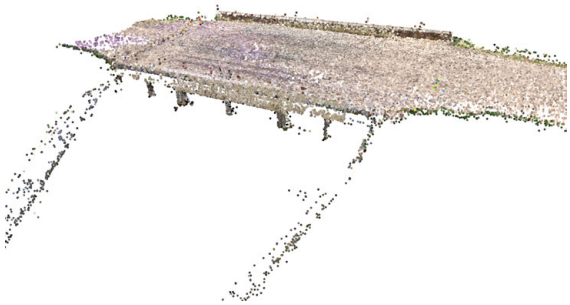
$$= \left[\begin{array}{ccc|c} R_{c_{1,1}} & R_{c_{1,2}} & R_{c_{1,3}} & x \\ R_{c_{2,1}} & R_{c_{2,2}} & R_{c_{2,3}} & y \\ R_{c_{3,1}} & R_{c_{3,2}} & r_{c_{3,3}} & z \\ \hline 0 & 0 & 0 & 1 \end{array}\right]_{4\times 4} \tag{3.1}$$

The camera pose is a dot product of a translation vector, $C$, and a rotation matrix, $R_C$ in world coordinates and describes the position and orientation of the camera in camera coordinates. For future decomposition, as discussed in Chapter 4, the matrix is typically made square by adding a row of $0,0,0,1$ to the bottom [44]. This matrix is found for each image in the data set and is unique for each image. With every camera pose and the matched key-points known for each image, 3D points can be created [42, 45, 46]. The created 3D point cloud, which is the output from Meshroom, is shown in Figure 3.1. Figure 3.1a and Figure 3.1b show a render of the north and south sides of the bridge, respectively. Figure 3.1c is an overview of the entire bridge. The dark grey cubes represent each camera pose from the images collected.

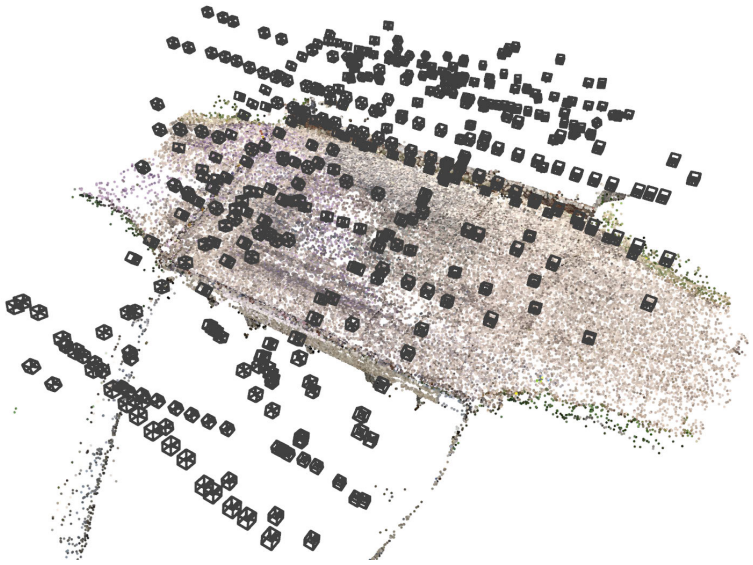
About 75,000 points were created from the SfM algorithm to represent the bridge. These points not only included the bridge, but also parts of the background and surroundings that were not of importance. To eliminate the points irrelevant to the bridge, an automated algorithm was developed to identify and delete the extraneous points. First, points of the bridge were relatively dense compared to the outlier points; therefore, the points that did not have a specified density, **j**, were deleted. Next, the average value of $[\mathbf{x},\mathbf{y},\mathbf{z}]$ was found to estimate the center point of the bridge. Based on the threshold **k**, the points farther than **k** away were



**(a)** Point Cloud Render of North Side



**(b)** Point Cloud Rendering of South Side



**(c)** Overview of Point-Cloud Showing Camera Pose

**Figure 3.1:** Point Cloud Rendering

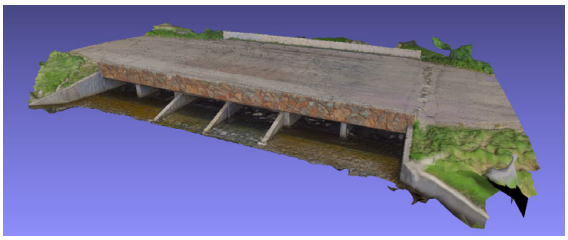
deleted. The model enhanced using the proposed algorithm is shown in Figure 3.1c with over 73,000 points.

With a 12-core Intel® i7 Processor with 32-*GB* of RAM, the feature extraction and matching took approximately 10-*minutes* and the creation of the dense point-cloud model from the 449 images took approximately 4.30-*hours* of running time. Because the number of operations to compare each key-point descriptor to each image increase superlinear, and not linearly, to the number of input images, the processing requirements is not linearly related to the number of input images. Having a smaller overlap ratio as recommended (i.e. 50 to 60%), and, therefore, fewer images, is expected to considerably reduce this processing time. If 250 images were used as input, instead of all 449, a 50% efficiency increase is expected. The efficiency of the processes of this chapter is shown in Table 3.2.

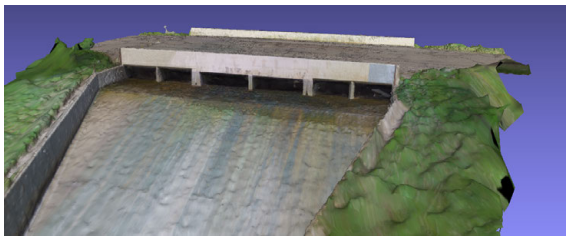
## 3.3 Development of Photo-Realistic Model

After the point-cloud is obtained, the photo-realistic model can be developed. This model will be used to help visualize the structure in full detail. Using the 3D points generated from the SfM algorithm introduced in Section 3.2, faces were calculated to make a "water-tight" model without holes or gaps, producing a more realistic appearance. Meshroom with AliceVision was, again, implemented to develop this model using the Screened Poisson Surface Reconstruction (SPSR) algorithm [41, 42, 47, 48]. The SPSR algorithm creates a triangular mesh around the points. Next, the images are stitched together to produce a texture that is overlaid on the mesh. This texture image is shown in Figure 4.3 and is later used as a possible technique to map the location of the defects in world coordinates, discussed in Section 4.2.1. The final photo-realistic model is shown in Figure 3.2

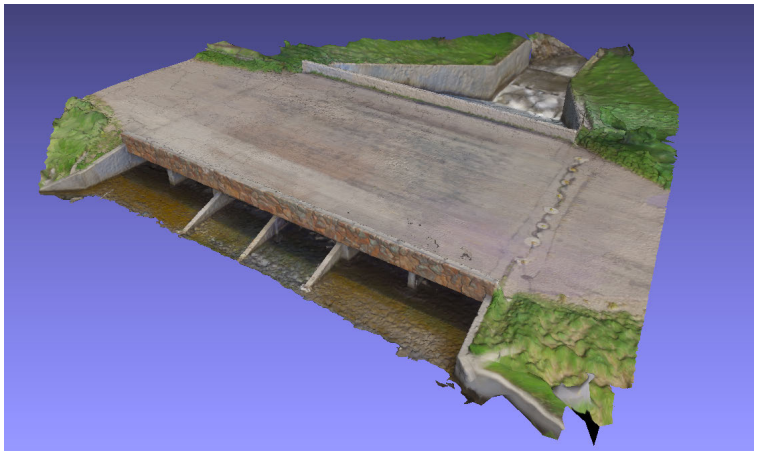
The computational time required to generate this mesh was 27-*minutes* and the texture and final photo-realistic model was approximately 26-*minutes*. The mesh creation, however, is a function of the number of points in the 3D model and not of the number of images. The texture



**(a)** Photo - Realistic Model Render Facing South



**(b)** Photo - Realistic Model Render Facing North



**(c)** Photo - Realistic Model Render From Above

**Figure 3.2:** Photo - Realistic Model Rendering

generation is reliant on the number of input images. The efficiencies of the 3D point-cloud and photo-realistic model generation is summarized in Table 3.2.

**Table 3.2:** Computational Efficiencies of the 3D Point-Cloud and Photo-Realistic Model Generation

| | Efficiency | | Time | |
|---|---|---|---|---|
| | Order | Type | (*minutes*) | (*hours*) |
| Feature Extraction and Image Matching | $\mathcal{O}(n \log n)$ | Superlinear | 10 | 0.167 |
| Dense Point-Cloud Model | $\mathcal{O}(n \log n)$ | Superlinear | 258 | 4.30 |
| Mesh | $\mathcal{O}(n)$ | Linear | 27 | 0.450 |
| Photo-Realistic Model | $\mathcal{O}(n \log n)$ | Superlinear | 26 | 0.433 |
| | | **Total:** | **321** | **5.35** |

# Chapter 4

# Damage Detection

To create a robust, stream-lined bridge inspection framework, damage of the structure needs to be automatically identified and quantified; this information is critical in the decision-making process for bridge owners and managers as it provides insight on the structure's health and maintenance needs. In current human-based bridge inspection practice, the damage rating is based on the inspector's skill, experience, or, in the case of chain-dragging or sounding techniques, hearing ability which often results in subjective and inconsistent data [8]. Additionally, since these ratings mostly come from qualitative categorical designations without quantitative measurements, the severity of the damage not explicitly delineated on the rating scale of a component. Consequently, it is difficult to track the progression of cracks or defects over time. Defects that are changing or growing over time are often of more importance and may require additional maintenance actions. In this context, this chapter proposes a defect detection algorithm capable of identifying and quantifying cracks and defects in concrete structures. Furthermore, these detected defects are mapped onto the 3D point-cloud to attain location and size information in real-world coordinates. This will provide bridge owners and/or managers with quantified information to better assess the structural condition and more precisely track the progression of defects over time.

Detecting and measuring cracks and defects in concrete can be challenging due to the low luminance of the defect and small widths of the cracks. In the literature, there are several studies on defect detection for concrete surfaces [49–51]. For example, Jahanshahi et al. and Sankarasrinivasan et al. mainly focused on exterior/interior facades that typically have longer cracks [49, 50]. Talab et al. focused on the formation of cracks in a controlled environment during laboratory stress tests [51]. Because the methods proposed in these studies were tailored to their specific applications (i.e. exterior concrete facades and laboratory concrete beam

tests), they worked well in their respective environments, but did not produce satisfactory results when used with the images of concrete bridges from a UAV.

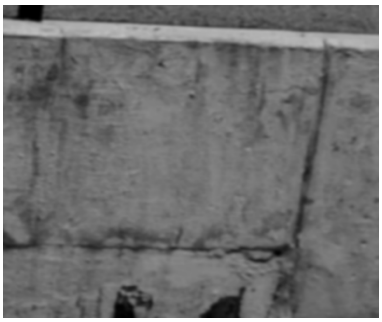
Recently, Convolutional Neural Nets (CNNs) have made tremendous progress in image classification and recognition [52–55], thus have become another alternative for damage/defect identification. CNNs are able to robustly detect and identify features in a given image; however, they require significant amount of training data to be successful. Dorafshan et al. and Hoang et al. studied implementing CNNs in combination with a variety of edge detectors to identify cracks in concrete [56, 57]. Both studies were successful with a classification accuracy of 86% and 92%, respectively. However, both used large amounts of labeled training data. For example, Dorafshan et al. used 3,420 images (319 images with cracks and 3,101 images without defects) and Hoang et al. used 400 images, 320 of which were labeled and used for training. Using CNNs on the images from the UAV flight is not possible without first labeling a large set of images in a training and validation set. Gathering this training data would negate the purpose of creating an automatic defect detection algorithm. Therefore, a new defect detection algorithm optimized for the defects typically found in concrete bridges which does not require a significant amount of training data is proposed.

## 4.1 Damage Detection Algorithm

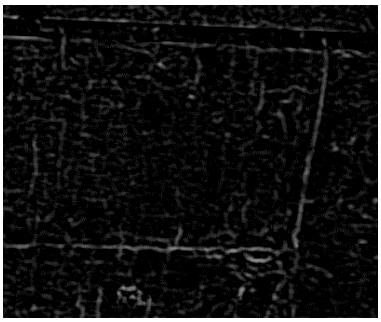
The proposed damage detection algorithm uses a Black Hat Transform and Canny Edge Detector to identify defects in an image. It was developed in Python and implements functions in the open-source OpenCV library [58]. In the following, the steps of this algorithm are introduced with the outputs of each step of the algorithm shown using two examples: 1) the south-side of the "Palmer Bridge's" six celled side (Figure 4.1); and 2) the eastern top road deck (Figure 4.2) of the "Palmer Bridge," (Figure 2.1). The algorithm and its parameters remained the same between Figure 4.1 and Figure 4.2.

In the first step, pre-processing of the image was performed which allowed the defects to become more obvious and enabled future steps to produce better results by reducing noise and

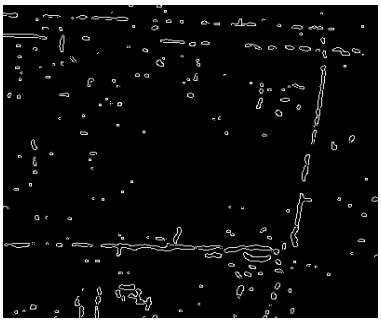
**(a)** Original Image



**(b)** Pre-Processed Image



**(c)** Black Hat Transform

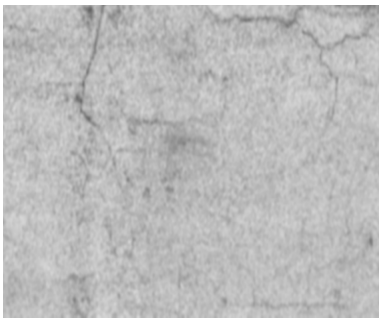


**(d)** Canny Edge Detection

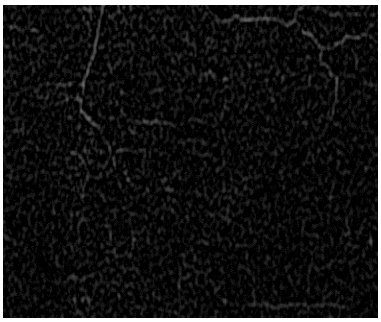
**(e)** Highlighted Cracked Section

**Figure 4.1:** Damage Detect Algorithm Output

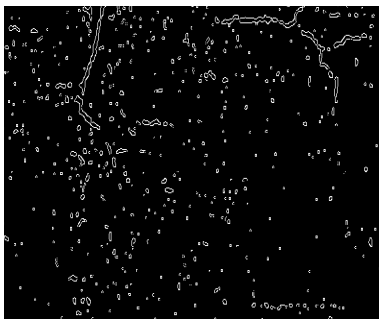
**(a)** Original Image



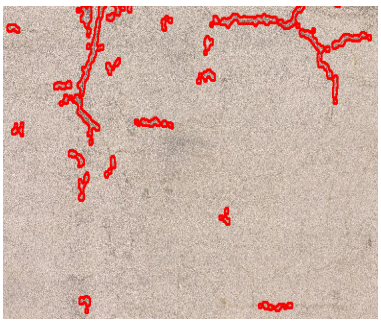
**(b)** Pre-Processed Image



**(c)** Black Hat Transform



**(d)** Canny Edge Detection



**(e)** Highlighted Cracked Section

**Figure 4.2:** Damage Detect Algorithm Output

emphasising the colors. To begin with, the image was converted into grey-scale. This reduced the computation time required for the entire algorithm by only utilizing one channel of pixels. After grey-scaling, minor color correction was done first; the contrast was increased while the brightness was reduced. When adjusting contrast, oftentimes, the noise in the image is amplified; therefore, OpenCV's Non-Local Means Denoising algorithm was used next. This not only helped with the amplified noise from the contrast adjustment, but also lessened noise organically in the image. The main idea behind denoising is to replace the pixel color with an average of the colors of similar pixels. The idea behind "Non-Local Means" denoising is assuming that the most similar pixels do not necessarily lie in close proximity to the pixel in question and could be located anywhere in the image [59]. Therefore, a "research window" is used to search for the most similar pixels. The pixel-wise implementation of Non-Local Means Denoising is shown in Equation (4.1).

$$\hat{u}(p) = \frac{1}{C(p)} \sum_{q \in B(p,r)} u(q) \cdot w(p,q) \tag{4.1}$$

*Where:*

$$C(p) = \sum_{q \in B(p,r)} w(p,q)$$

where $\hat{u}(p)$ is the new pixel color, $u(p)$ is the original pixel color in question, $B(p,r)$ is the "research window," $w(p,q)$ is the weight factor, and $C(p)$ is the normalization constanst composed of the sum of all the weight factors in the "research window." To finish the pre-processing step, a Gaussian Blur was placed over the image to also help in reducing the noise in the image. This is accomplished by convolving the image with a Gaussian kernel. It was found through trial and error that the $4 \times 4$ kernel in Equation (4.2) worked best. The result of the pre-processing for each sample is shown in Figure 4.1b and Figure 4.2b. Random noise was reduced while preserving the details of the defects.

$$K = \begin{bmatrix} 0.0145 & 0.0467 & 0.0467 & 0.0154 \\ 0.0467 & 0.1413 & 0.1413 & 0.0467 \\ 0.0467 & 0.1413 & 0.1413 & 0.0467 \\ 0.0154 & 0.0467 & 0.0467 & 0.0154 \end{bmatrix} \tag{4.2}$$

Next, a Black Hat Transform was performed on the image using a structural element shown in Equation (4.3) as recommend by Sankarasrinivasan et al. [50]. The kernels are convolved over the image. If all the pixel colors under the kernel are the same, the center pixel receives a 1; otherwise, the center pixel receives a 0. The difference between this new image and the input image create the Black Hat Transform. The process is completed with both $K_1$ and $K_2$ kernels shown in Equation (4.3). The two images are added and normalized to create Figure 4.1c and Figure 4.2c for the sample images. This produces a grey-scale image where the white portions are regions of greater contrast compared to adjacent pixels. This transform highlights the areas of interest where there is potential for a defect [60]. After the Black Hat Transform, the image was thresholded to create a simple black and white image.

$$K_1 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T, \qquad K_2 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \tag{4.3}$$

Lastly, a Canny Edge Detector was used to outline the white regions produced from the Black Hat Transform on the image [61]. This is a popular and robust edge detection algorithm. In this process, first, the x and y Sobel Kernels, shown in Equation (4.4), are convolved across an image to estimate the first derivative, $G_x$ and $G_y$ Equation (4.5). The $*$ symbol denotes the convolution operation.

$$K_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \qquad K_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \tag{4.4}$$

$$G_x = K_x * Image \qquad G_y = K_y * Image \tag{4.5}$$

Next, the Edge Gradient, $G$, from Equation (4.6), and the direction, $\theta$, from Equation (4.7) can be found given the two results from the previous convolutions, $G_x$ and $G_y$.

$$Edge\ Gradient:\ G = \sqrt{G_x^2 + G_y^2} \tag{4.6}$$

$$Angle:\ \theta = \tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{4.7}$$

Non-maximum suppression is performed to remove any unwanted pixels that are not constituted as edges to find the "largest" edges in the image. Every pixel is checked to see whether it is a local maximum in its neighborhood in the direction of the gradient, $\theta$. Lastly, the pixels are thresholded between minimum and maximum values. The results from the Canny Edge detection of the Black Hat Transformed images are shown in Figure 4.1d and Figure 4.2d. This is the outline of all the possible defects in the image.

There is still a considerable amount of noise associated in the Canny Edge Detection output; therefore, only the largest outlines are kept. A minimum area inside the outlines is set to a threshold, $\xi$. The outlines with a smaller area than $\xi$ were discarded. This eliminated most of the noisy outlines while still preserving the defects. This gives the final output image shown for the concrete side in Figure 4.1e and the asphalt road shown in Figure 4.2e for the "Palmer Bridge." From these outlines, the area, length, and location information can be extracted in the process discussed in Section 4.2.

The proposed defect detection algorithm can detect and highlight the cracks photographed by the UAV without the need of large training data. Location, area, and length of the crack can

also be extracted and used as a quantitative metrics for rating each element. This would provide more consistent and quantitative data for bridge owners and decision makers.

## 4.2 Localization of Defects

With the defects detected and highlighted, the damage information needs to be mapped onto the 3D point-cloud. Once the defects are mapped into real-world coordinates, the size and location information of the defect can be extracted to provide localized damage information and create a more robust AB-BIM for bridge owners and/or managers. Two techniques were tested to map the detected defects, found using the proposed algorithm discussed in Section 4.1, from the 2D images onto the 3D point-cloud. In the first technique, the created texture from the photo-realistic model development was passed through the defect detection algorithm. The second technique was to use the generated camera pose matrix from the point-cloud creation process. Therefore, the techniques will be discussed and analyzed in Section 4.2.1 and Section 4.2.2, respectively.

### 4.2.1 Defect Mapping Based on Texture Extraction

The first technique to attempt to map the defects detected in a 2D image into 3D space was to use the texture image created in the process of building the photo-realistic model, as discussed in Section 3.3. This texture is a set of images, used to originally create the 3D point-cloud, stitched together and essentially overlaid on the mesh of the model (shown in Figure 3.2). The texture image can have a fairly high pixel resolution of 268-*MP* or more. A sample of one of the two textures used to create Figure 3.2 is shown in Figure 4.3.

As opposed to a subset of the image collected by the UAV passing through the damage detection algorithm to highlight the defects, only the high resolution texture image was used and the defects on the texture image were highlighted. Using the same parameters and technique as discussed in Section 4.1, the detected defects were highlighted in red. The texture with highlighted defects is shown in Figure 4.4

**Figure 4.3:** Original Texture Used to Create Figure 3.2

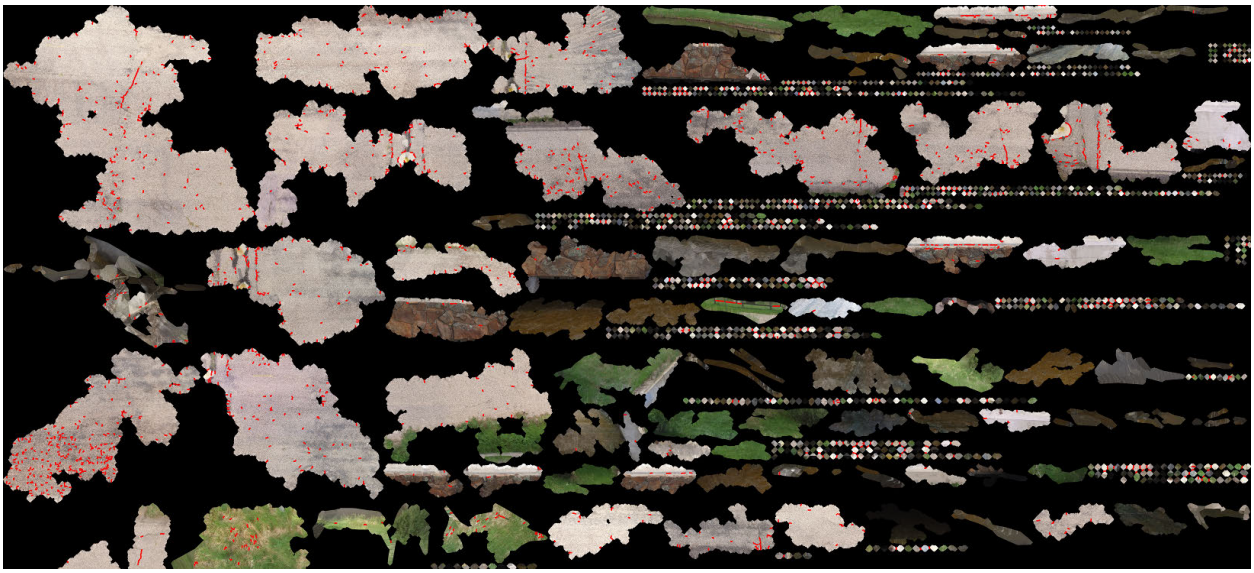


**Figure 4.4:** Detected Defects in the Texture of the Photo-Realistic Model

Once the defects were detected in the two textures, they were used to overlay on the generated mesh with the defects highlighted. After this, the defects are mapped into 3D real-world coordinates and the location and size of the defects are known. The result is shown in Figure 4.5

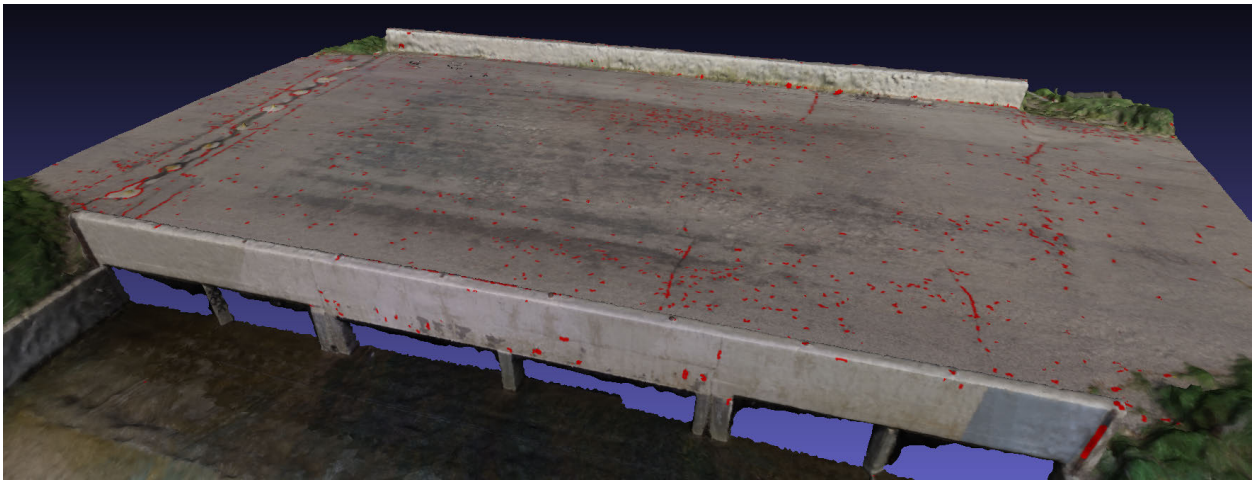
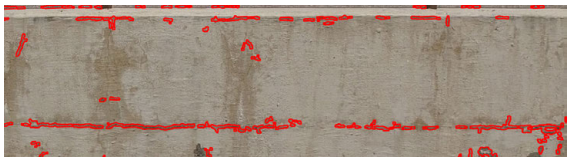


**Figure 4.5:** The Photo-Realistic Model with the Texture Passed through the Damage Detection Algorithm Discussed in Section 4.1

There were some problems with this technique. Even though the texture has a substantially higher resolution than the original image (268-*MP* in the texture as opposed to 12.4-*MP* in the original images), the texture was created by stitching the original images together. Inherently, there was some loss of quality in the images and either there was significantly more noise in image or there was not enough information left in the stitched images. One example of this is in Figure 4.6. Figure 4.6a is a close-up on the side of the photo-realistic model with the texture passed through the defect detection algorithm. Figure 4.6b is the same area with the original image taken by the UAV passed through the defect detection algorithm. There is significantly more detail and better damage identification in Figure 4.6b when compared to Figure 4.6a.

The comparison in Figure 4.6 indicates a better solution is needed to map the detected defects from the 2D images into 3D real-world coordinates. Section 4.2.2 proposes a different and more successful technique to accomplish this.

**(a)** Detected Damage of the Side of the Bridge Using the Texture Technique of Defect Mapping



**(b)** Detected Damage of the Side of the Bridge Using the Original Image in the Defect Detection Algorithm

**Figure 4.6:** Comparison of Defect Detection Using the Texture Mapping Technique and the Original Image

### 4.2.2 Defect Mapping Based on the Unique Camera Poses

To take full advantage of the higher resolution images from the UAV, the next technique uses the original image, the image passed through the damage detection algorithm, the 3D point-cloud, and the camera pose of the image to map the 2D image onto the 3D point-cloud. The Camera Pose is a matrix that relates the 2D image in pixel space to the 3D environment in camera coordinates. Each camera pose was recovered for each image during the construction of the point-cloud. Knowing the camera pose, the proposed technique relates each image back to the 3D point-cloud. The process to accomplish this projection begins with the 3D points in real-world coordinates. The points are, first, indexed and then transformed into 2D points in pixel space. With both the newly transformed points and the images in 2D pixel space, the defects can be readily mapped and the points the represent the corresponding damage are highlighted. Since the points were indexed before the transformation, they can be linked to original, non-transformed corresponding 3D point in real-world coordinates. Finally, the defect is mapped onto the 3D point-cloud.

Before discussing the specific algorithm used to automatically relate the defects in 2D pixel space to the 3D real-world coordinates, the Camera Matrix, $P$, must first be defined. The Camera Matrix relates the 2D pixels in pixel space to the real-world coordinates. It can be decomposed of two matrices, the *Intrinsic*, $K$, and *Extrinsic*, $E$, Matrices Equation (4.8).

$$[P]_{3\times4} = [K]_{3\times4} \times [E]_{4\times4} \tag{4.8}$$

The *Intrinsic Matrix*, $K$, gives information about the camera itself and the *Extrinsic Matrix* is a transformation of real-world coordinates. First, the *Intrinsic Matrix* is composed of the focal length ($f_x$ & $f_y$), the optical center ($x_0$ & $y_0$), and the shear factor ($s$). These three parameters are a function of the geometry of the camera that captured the image and, therefore, remain constant in the data set. The general Intrinsic matrix is shown in Equation (4.9). To make the matrix multiplication work with the *Extrinsic Matrix*, $E$, discussed next, a column-vector of 0s is added to the end. [62].

$$\begin{aligned}\text{Intrinsic Matrix: } K &= \underbrace{\begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix}}_{\substack{\text{2 D Translation} \\ \text{Matrix}}} \times \underbrace{\begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\substack{\text{2 D Scaling} \\ \text{Matrix}}} \times \underbrace{\begin{bmatrix} 1 & \frac{s}{f_x} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\substack{\text{2 D Shear} \\ \text{Matrix}}} \\ &= \left[\begin{array}{ccc|c} f_x & s & x_0 & 0 \\ 0 & f_y & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right]_{3\times 4} \end{aligned} \tag{4.9}$$

Next, the *Extrinsic Matrix*, $E$, needs to be found to completely describe the Camera Matrix. This matrix describes how the real-world coordinates are transformed relative to the camera. It is composed of a $3\times3$ Rotation Matrix, $R$, (the direction of the real-world axis in Camera Coordinates) and a $3\times1$ translation column-vector, $t$, (the position of the real-world origin in Camera Coordinates) Equation (4.10). Since it describes the transformation of the 3D coordinates from the camera's perspective, it is unique to every image. Often, this matrix is made square for future decomposition by adding a row of $0,0,0,1$ at the end of the matrix [44].

$$\text{Extrinsic Matrix: } E = \underbrace{\left[\begin{array}{c|c} I & t \\ \hline 0 & 1 \end{array}\right]}_{\substack{\text{Real-World Origin Position in} \\ \text{Camera Coordinates}}} \times \underbrace{\left[\begin{array}{c|c} R & 0 \\ \hline 0 & 1 \end{array}\right]}_{\substack{\text{Real-World Axis Directions in} \\ \text{Camera Coordinates}}}$$

$$= \left[\begin{array}{ccc|c} R_{1,1} & R_{1,2} & R_{1,3} & t_1 \\ R_{2,1} & R_{2,2} & R_{2,3} & t_2 \\ R_{3,1} & R_{3,2} & R_{3,3} & t_3 \\ \hline 0 & 0 & 0 & 1 \end{array}\right]_{4\times 4} \tag{4.10}$$

With the Camera Matrix completely defined, the algorithm used to automatically project the 2D images with detected damage onto the 3D point-cloud to locate the position of the defect can be described. For this process, it is assumed that an image has been passed through the defect detection algorithm (an example image is shown in Figure 4.8d) and the camera pose matrix has already be recovered, using the process discussed in Chapter 3. This Camera Pose Matrix, in Equation (4.11), describes the position and orientation of the camera in real-world coordinates and is composed of a column-vector of the camera position in real-world coordinates, $C$, and the rotational matrix of the camera's orientation in real-world coordinates, $R_C$. Again, a row of $0,0,0,1$ is added to make the matrix square [44]. This matrix is known and unique for each image.

$$\left[\begin{array}{c|c} R_C & C \\ \hline 0 & 1 \end{array}\right] = \underbrace{\left[\begin{array}{c|c} I & C \\ \hline 0 & 1 \end{array}\right]}_{\substack{\text{Translation Matrix in} \\ \text{Real-World Coordinates}}} \times \underbrace{\left[\begin{array}{c|c} R_C & 0 \\ \hline 0 & 1 \end{array}\right]}_{\substack{\text{Rotation Matrix in} \\ \text{Real-World Coordinates}}}$$

$$= \left[\begin{array}{ccc|c} R_{c_{1,1}} & R_{c_{1,2}} & R_{c_{1,3}} & x \\ R_{c_{2,1}} & R_{c_{2,2}} & R_{c_{2,3}} & y \\ R_{c_{3,1}} & R_{c_{3,2}} & r_{c_{3,3}} & z \\ \hline 0 & 0 & 0 & 1 \end{array}\right]_{4\times 4} \tag{4.11}$$

To find the Extrinsic Matrix given the Camera Pose Matrix, take the inverse of the Camera Pose Matrix and simplify Equation (4.12).

$$
\begin{aligned}
\left[\begin{array}{c|c} R & t \\ \hline 0 & 1 \end{array}\right] &= \left[\begin{array}{c|c} R_C & C \\ \hline 0 & 1 \end{array}\right]^{-1} \\
&= \left[\left[\begin{array}{c|c} I & C \\ \hline 0 & 1 \end{array}\right] \times \left[\begin{array}{c|c} R_C & 0 \\ \hline 0 & 1 \end{array}\right]\right]^{-1} \\
&= \left[\begin{array}{c|c} R_C & 0 \\ \hline 0 & 1 \end{array}\right]^{-1} \times \left[\begin{array}{c|c} I & C \\ \hline 0 & 1 \end{array}\right]^{-1} \\
&= \left[\begin{array}{c|c} R_C^T & 0 \\ \hline 0 & 1 \end{array}\right] \times \left[\begin{array}{c|c} I & -C \\ \hline 0 & 1 \end{array}\right] \\
&= \left[\begin{array}{c|c} R_C^T & -R_C^T \cdot C \\ \hline 0 & 1 \end{array}\right]_{4\times 4}
\end{aligned}
\tag{4.12}
$$

With Equation (4.12), the relationship between real-world coordinates and the camera coordinates is shown in Equation (4.13) and Equation (4.14). The *Extrinsic Matrix* in real-world coordinates described by the Camera Pose Matrix composed of $R_C$ and $C$ is shown in Equation (4.15).

$$R = R_C^T \tag{4.13}$$

$$t = -R \cdot C \tag{4.14}$$

$$\text{Extrinsic Matrix: } E = \underbrace{\left[\begin{array}{c|c} R_C^T & -R_C^T \cdot C \\ \hline 0 & 1 \end{array}\right]}_{\substack{\text{Extrinsic Matrix in} \\ \text{Real-World Coordinates}}} \tag{4.15}$$

Lastly, combining the *Intrinsic Matrix* with the known camera parameters and *Extrinsic Matrix* in real-world coordinates, described by the Camera Pose Matrix composed of $R_C$ and $C$, the unique *Camera Matrix*, $P$, for each image can be assembled Equation (4.16).

$$[P]_{3\times4} = \left[\begin{array}{ccc|c} f_x & s & x_0 & 0 \\ 0 & f_y & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right]_{3\times4} \times \left[\begin{array}{c|c} R_C^T & -R_C^T \cdot C \\ \hline 0 & 1 \end{array}\right]_{4\times4} \tag{4.16}$$

Now, using the composed *Camera Matrix* Equation (4.16), $P$, with all values known from the Camera's Parameters and Pose Matrix, the 3D point-cloud in real-world coordinates (obtained from the point-cloud discussed in Chapter 3) can be projected into 3D pixel space by multiplying the column-vector of a given point by the *Camera Matrix* shown in Equation (4.17). $x_1$, $x_2$, $x_3$ is one 3D point from the 3D point-cloud.

$$\underbrace{\begin{bmatrix} x_1' \\ x_2' \\ x_3' \end{bmatrix}}_{\substack{\text{3D Point in} \\ \text{Pixel Space}}} = [P]_{3\times4} \times \underbrace{\left[\begin{array}{c} x_1 \\ x_2 \\ x_3 \\ \hline 1 \end{array}\right]}_{\substack{\text{3D Point in} \\ \text{Real-World Coordinates}}} \tag{4.17}$$

The transformed 3D point, the output from Equation (4.17), is now in 3D pixel space. Next, the points are transformed again from 3D pixel space into 2D pixel space by Equation (4.18) and this gives the $x_1'$ and $x_2'$ of each point. $x_1'$ and $x_2'$ are the final results of the projection of the 3D point cloud into 2D points in pixel space.

$$\underbrace{\begin{bmatrix} x_1' \\ x_2' \end{bmatrix}}_{\substack{\text{2D Point in} \\ \text{Pixel Space}}} = \begin{bmatrix} \frac{x_1}{x_3} \\ \frac{x_2}{x_3} \end{bmatrix} \tag{4.18}$$

Using the technique previously mentioned, a 3D point cloud in real-world coordinates can be readily mapped into 2D pixel space. The projected 3D point cloud has the same dimensions as the original image (i.e. if the image is $3,000 \times 4,000$ pixels, the projected point-cloud will be $3,000 \times 4,000$ units); therefore, the points on the projected 2D point cloud will closely match with the pixels in the original image. Two example output images are shown in Figure 4.7. Figure 4.7a and Figure 4.7c are the original images captured by the UAV during flight. Using the technique from above and knowing the Camera's Pose and Parameters composed of $R_C$ and $C$, Figure 4.7b and Figure 4.7d are the projected point-cloud in 2D Pixel Space.

**(a)** Image Taken by the UAV

**(b)** Projection of the Point Cloud in 2D Pixel Space Using the Camera's Pose Matrix and Parameters

**(c)** Image Taken by the UAV

**(d)** Projection of the Point Cloud in 2D Pixel Space Using the Camera's Pose Matrix and Parameters

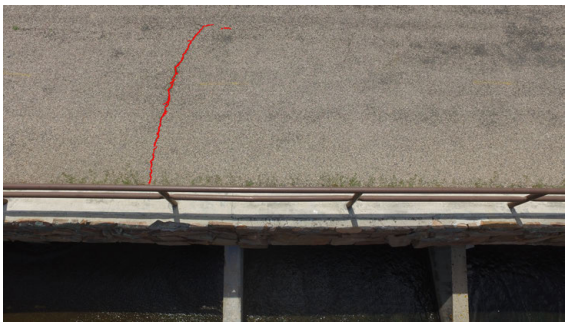
**Figure 4.7:** Sample Images of the Projection of the Point Cloud Using the Camera Pose Matrix composed of $R_C$ and $R$

A 2D pixel from an image can be readily mapped to the corresponding point and related to the point cloud in 3D real-world coordinates. The detected defect(s) from the algorithm (discussed in Section 4.1 and an example shown in Figure 4.8c) are located in this 2D pixel space with red pixels/markers. Using Euclidean distance, the closest point from the point-cloud can be matched with the red pixels/markers to identify the defected points. Before the transformation, each point is indexed. Once the defected points of the point-cloud are identified, the index number is related back to the non-transformed points in the 3D real-world coordinates. Thus, the 3D position of the defect is known. An example of the full process is shown in Figure 4.8. Figure 4.8a is the original image taken by the UAV. The 3D points are then projected into 2D pixel Space (Figure 4.8b). After the image is passed through the defect detection algorithm, discussed in Section 4.1, (Figure 4.8c), the defects are readily mapped onto the projected points shown in Figure 4.8d. Lastly, the projected points are mapped back to the 3D point-cloud and location and size are known in real-world coordinates. The final projected defects are shown in Figure 4.9. The red cube is the position and orientation of the camera at the time of the image (i.e. the Camera's Pose).

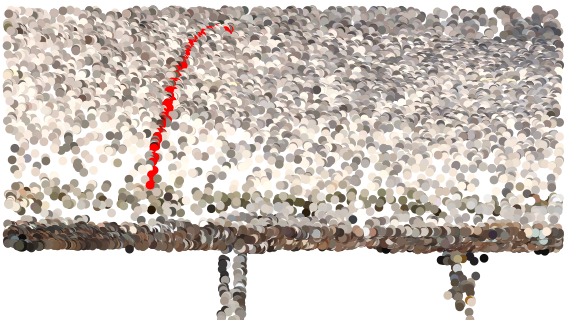
The proposed technique takes about 10-*seconds* for each image to map onto the 3D point-cloud. The most computational involved portion of the algorithm is finding the closest points from the the 2D image to the projected 2D points. This matching portion is currently done by finding a projected point with the shortest Euclidean distance to the 2D red pixel/marker in the image. It was found that the first method using the texture image mapped the defects in 3D real-world coordinates quickly and easily, but missed the majority of the defects when passed through the defect detection algorithm; however, the technique based on camera pose successfully projected the defects located in 2D pixel space into 3D real-world coordinates. Under this framework, the defects that are identified in the images, can be readily mapped onto the point cloud and located in 3D space. This allows a bridge owner or manager to locate the defects on the bridge and provide easy visualization of all the identified defects on the structure. Additionally, this localization information is used to help create the AB-BIM, which is discussed in

**(a)** Image Taken by the UAV

**(b)** Projection of the Point Cloud in 2D Pixel Space Using the Camera's Pose Matrix and Parameters



**(c)** Detected Damage from the Image Using the Algorithm Siscussed in Section 4.1



**(d)** Detected Defects Mapped onto the Points

**Figure 4.8:** Sample Images of the Projection of the Point Cloud Using the Camera Pose Matrix Composed of $R_C$ and $R$

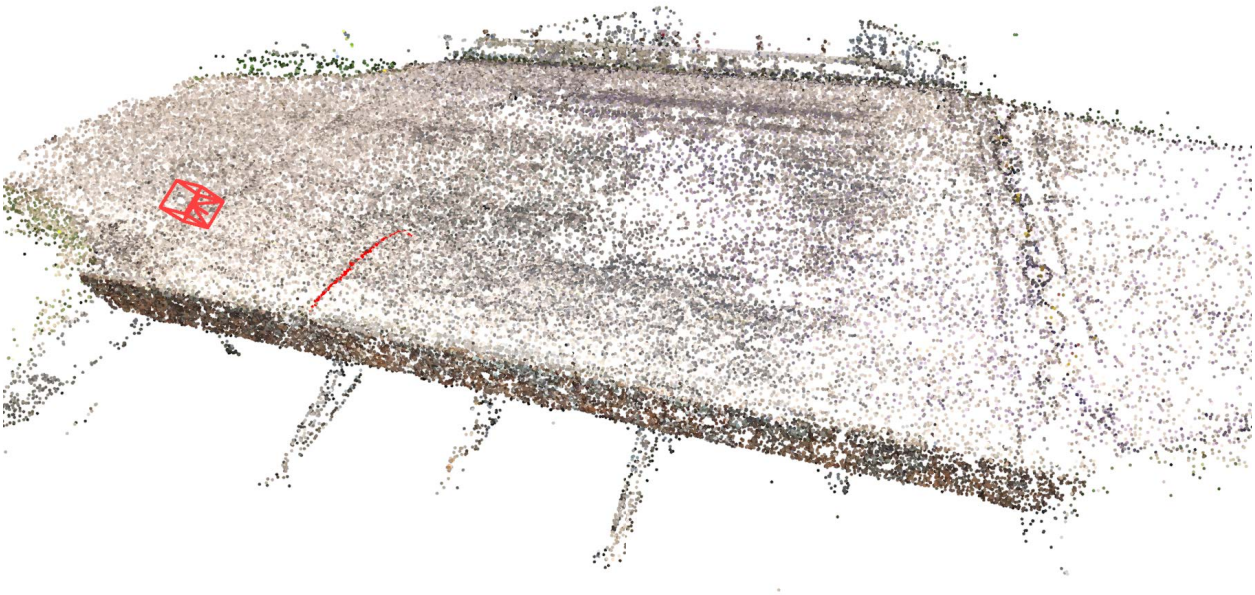


**Figure 4.9:** 3D Point Cloud with the Detected Defected Mapped in Real-World Coordinates

Chapter 6, by providing the location and size of the “damage cube” onto the model and produces quantified metrics.

# Chapter 5

# Component Identification

To ensure seamless merging of the proposed bridge inspection framework with current inspection practices, element-wise damage assessment is required that is consistent with current inspection practices detailed by FHWA and AASHTO. With the damage identified from the proposed algorithm, discussed in Chapter 4, a novel technique is proposed which automatically segments the elements of the bridge (i.e. deck, wing-wall, girders, etc.) from the created 3D point-cloud to enable mapping the damage information to each structural element. The automatically segmented bridge elements will ease and automate the quantification and mapping of damage information onto these segmented elements, simplifying and stream-lining implementation of the entire framework. Additionally, analysis programs, such as bridge management system software, use element data to more accurately assess needs and costs, and make recommendations for programs, projects, and action types; therefore, with the segmented elements, such analysis programs could still be utilized. This is a challenging step, however, due to the complexity of the point cloud information, the variability of point density, and irregular shapes of the elements. This chapter will discuss in detail the proposed process that tackles this challenge of automatic segmentation of the bridge elements from the point-cloud.

Object segmentation using the images collected by the UAV was explored. Segmentation of images using CNNs, region-based convolutional neural nets (R-CNNs), deformable part-based model (DPM), or a combination of them have proven successful in multiple object detection in images [52, 55, 63–65]. Labeled training data is required for the aforementioned techniques to be successful, rendering the automatic segmentation irrelevant. For example, Li et al. used the PASCAL VOC 2007 image data set consisting of 9,963 images, of which 4,981 images were used for training and validation. Narazaki et al. specifically explored the segmentation of bridge images into 5 bridge elements with an emphasis on reducing false-positives with a pixel-wise accuracy of 88.7%. Again, there was a large number of labeled information to train the network

(1,563 total number of images). Another approach to automatically segment the elements of the bridge from the images with minimal training data was implementing watershed and super-pixel algorithms; however, a user was required to input unique information for each image. Consequently, this was not the best solution for automatic image segmentation.

With little success from using the images to segment the elements from the AB-BIM, the next approach was to explore the advantages of using the the point-cloud itself for the segmentation. Research of the segmentation of point-cloud data from existing structures or construction sites is still fairly new; however, some studies have shown success in identifying walls, ceilings, floors and windows [66–68]. These techniques mainly use denoising of the point-cloud to make the points more planar and identify straight, flat regions. The denoising technique worked effectively to segment linear and flat features. This technique lends itself well to rooms and buildings with rectangular features like walls and windows, but the geometry of a bridge is often more complex and different. Lu et al. proposed a top-down approach to segment the bridge elements of point-clouds [69]. This study investigated the separation of piers, pier-caps, and deck using terrestrial laser scan data and found less than 5% false-positive rate in the segmentation of the piers; however, this terrestrial LiDAR data is more difficult and more expensive to attain and the proposed algorithm requires high point density in the point-cloud to identify the nuances of the data.

Given the shortcomings of the existing approaches, a new point-cloud segmentation algorithm is proposed which will simplify implementation of the proposed framework into real-world application. This segmentation approach can automatically segment the irregular shapes of a bridge from point-cloud data created with photogrammetry without requiring labeled training data. Leveraging advancements in unsupervised machine learning, the elements of a bridge are segmented with minimum user input with a high rate of “true-positive” values. With additional metadata from the point cloud, unsupervised clustering techniques are performed in three steps to separate each component of the bridge. These segmented sections are then used to build an as-built, component-wise BIM.

## 5.1 Fowlkes-Mallows Index

Before starting the automatic component identification, a metric had to be defined to quantify the accuracy of the component identification. To provide ground-truth data (GTD) for evaluating the results of this case study, the dense point cloud was manually segmented into the 17 components (bridge deck, 2 sides/guardrails, 4 abutments, 10 beams). Since the proposed algorithm only segments the point cloud, and does not label the elements, the accuracy of the segmentation (i.e. whether each point is in the correct class) becomes a challenge to implement; therefore, the Fowlkes-Mallows Index (FMI) (Equation (5.1)) was used. The output from the proposed algorithm (i.e. the segmented elements) was compared to the GTD using the FMI. This index compares the GTD with the identified components with an emphasis on True Positives. An FMI close to 1.0 indicates a precise segmentation.

$$\text{FMI} = \frac{\textit{True Positives}}{\sqrt{\big(\textit{True Positives} + \textit{False Positives}\big) \cdot \big(\textit{True Positives} + \textit{False Negatives}\big)}} \tag{5.1}$$

## 5.2 Automatic Segmentation

After the flight, a dense point-cloud was created, as discussed in Chapter 3. From this point-cloud, each structural element will be identified. Lastly, this element-segmented point-cloud will serve as the base for the AB-BIM where element-wise damage information can be documented, discussed in Chapter 6.

To begin the automatic component identification, two clustering techniques, Gaussian Mixture Model (GMM) and Agglomerative Clustering were implemented. To execute the two techniques, a Python script using the open-source Open3D and Scikit-Learn libraries was built [53, 70, 71]. The GMM assumes the clusters are generated from a Gaussian distribution and creates a probabilistic model of the clusters. This technique incorporates information about the covariance structure of the data. In this implementation, it is assumed that each cluster has its own general covariance matrix. The GMM was chosen because it is most effective for distributions and groups with different, irregular, or oblong shapes such as those found in a point

cloud of a bridge. Agglomerative Clustering, on the other hand, creates a dendrogram of the points, in this case, based on the Euclidean distances, to separate the data circumscribable in a sphere.

Next, the automatic process of component identification based on these two clustering techniques is introduced. To begin, the full point cloud is classified into three sections using a GMM, which separates out the bridge deck from the other components. After this step, the deck is completely identified and does not require additional manipulation. To automatically identify which cluster contains the deck as opposed to the other components, the surface normals of the point cloud were used. The normals are typically used for visualization to determine the reflectance of light; however, they also give information about the direction of the point by calculating the principal axis of adjacent point using covariance analysis [72]. It is assumed that the bridge deck is a relatively level, flat, and smooth surface; therefore, the normals would point in relatively the same direction and the angles between the normal vectors would be small. Calculating the average angle between every normal vector of each cluster and choosing the cluster with the smallest average angle identified the deck cluster. This deck segmentation can be automatically labeled and saved. The output is shown in Figure 5.1.

With the deck separated out from the point-cloud, the North section (red section in Figure 5.1) and South section (blue section in Figure 5.1) are each broken into three separate sections again using a GMM. This process identifies the bridge's sides/guardrail from the rest of the bridge's components. To identify which cluster contains the bridge's side/guardrail, the surface normals of the point cloud were used again. Finding the segmentation with the smallest average angle between the normals, the side/guardrail clusters were identified. Once the side was identified, the Agglomerative Clustering was implemented to separate out the rest of the components. With the deck and sides/guardrails now eliminated, the left-over components were more centralized and could be circumscribed in a sphere. Therefore, Agglomerative Clustering produced better results when compared to a GMM or other clustering technique. The final re-

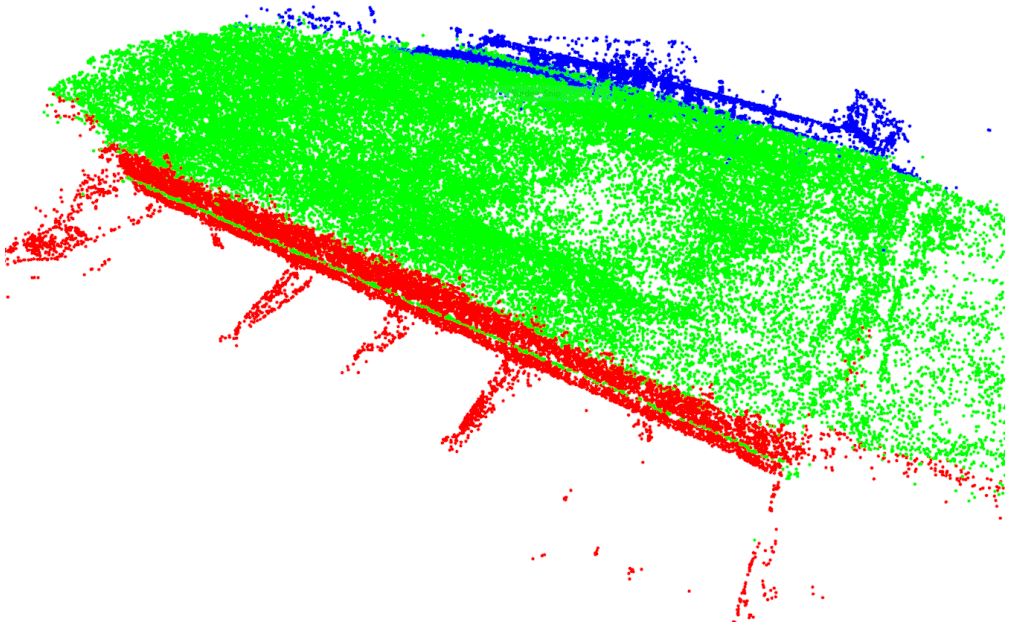


**Figure 5.1:** Initial Segmentation of Point-Cloud (Green is Segmented Deck; Red and Blue are Others)

sults from the segmentation is shown in Figure 5.2, where Red is the segmented bridge deck, grey is the north guardrail, etc.

The overall framework produced an FMI of 0.81 for the entire segmented bridge (Figure 5.2), indicating a high rate of true-positive identification. To compare with a single clustering method, such as only using GMM or Agglomerative Clustering, the FMI is significantly lower with 0.37 and 0.28, respectively. Therefore, the proposed combination of the two techniques achieves a higher FMI leading to better element segmentation.

Utilizing UAVs and leveraging current advances in photogrammetry and machine learning, a novel approach to segment a point-cloud of a bridge is proposed. Using a point-cloud created with photogrammetry, each component of the bridge (i.e. girders, guardrail, deck) is automatically segmented using unsupervised machine learning. One drawback to the proposed algorithm is the lack of automatic labeling of the segmented point-cloud. Labeling requires a user to manually provide this information. For instance, the proposed algorithm is able to separate out the bridge girders, but it cannot label the segmentations containing a bridge girder. This is the common shortfall of unsupervised machine learning. The proposed process provides a

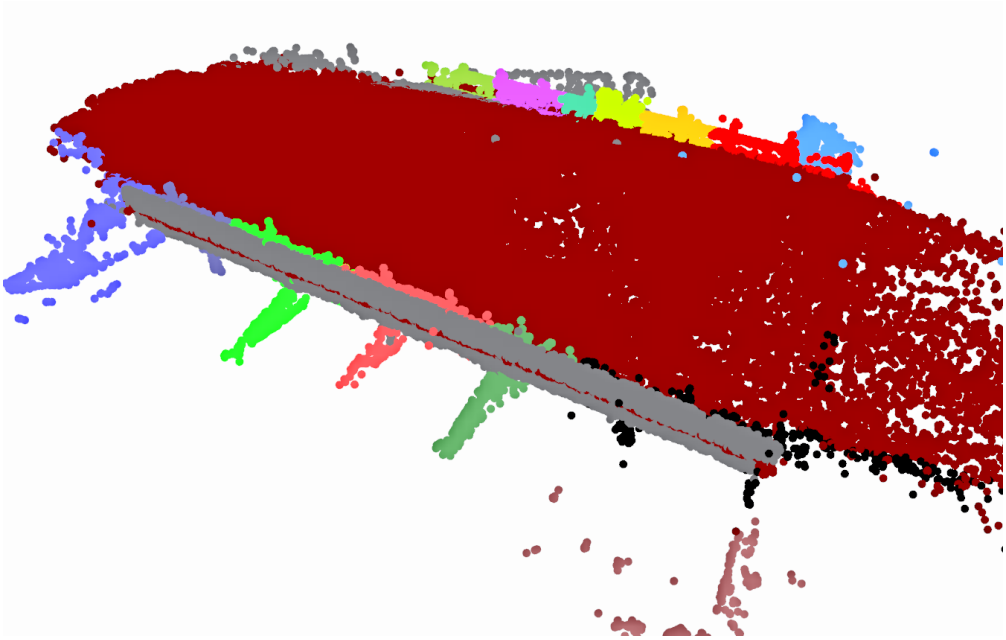


**Figure 5.2:** Final Point-Cloud Segmentation (Red is Segmented Deck; Purple is Segmented Wing Wall; etc.)

seamless transition from current practices to the implementation of a UAV-enabled bridge inspection system.

# Chapter 6

# Development of As-Built Bridge Information Model (AB-BIM)

With all the parts of the proposed framework found (including a photogrammetric point-cloud model, quantified damage information mappable onto the point-cloud, and the major elements of the bridge segmented from the points), the complete and final AB-BIM can be assembled. This AB-BIM provides a one-stop decision-making support system. The proposed geo-referenced AB-BIM enables the integration with CDOT's, and other state DOT's, current GIS-based data management system. This will allow convenient and comprehensive documentation of condition assessment results with geo-referenced and element-wise damage information and streamlined transition into practice. This chapter will discuss in detail how the proposed AB-BIM will be automatically built from the point-cloud, detected damage information, and segmented components and how all the information will be stored and visualized.

## 6.1 Automatic Information Upload and Model Establishment in Revit

To build the AB-BIM, a base software program had to be chosen. For this particular application, a robust BIM package which could handle a point-cloud data and input customizable, user-defined parameters/objects was needed. Three candidate BIM software suites were considered: LEAP Bridge, Tekla Structures, and Revit [73–75]. Based on a comparison study by McGuire et al., the Revit® software suite, designed by Autodesk, Inc., was chosen for this need, as it was able to associate any user-defined information (i.e. damage information, images, etc.) to any specific element or object, which can then be easily queried and accessed [76]. This is done by creating a Family within Revit. Additionally, the Revit® software can automatically

generate reports or schedules of these user-defined Families. For instance, with a few clicks in the user interface, all the damage information for each element (i.e. girder, deck, guardrail, etc.) can be listed and quantified in a formatted schedule. This automatic report creation ability combined with the use of Revit Families to store and locate information is convenient for a bridge owner or manager to quickly quantify and report all the damage on the bridge or on a particular element of the structure.

There is some initial set up in the Revit work environment. The 3D point-cloud is scaled to real-world coordinated and georeferenced in the Revit project, using ground-truth data from GPS points. The segmented elements (output of the process described in Chapter 5) were also uploaded and labeled in this process. This step of uploading, scaling, and geo-referencing can also be automated using the process described later in this section. Next, the concept of "damage cubes," proposed by McGuire et al., was adapted here for documentation of each detected defect [76]. A "damage cube" is a generic, user-defined Revit Family object that can have a variably-defined shape and size. The damage cube is placed on an element where damage was identified and contains any user-defined information for the specified location or defect; the proposed stored parameters are found in Table 6.1. The damage cubes are represented by the blue objects in Figure 6.1, where its size, shape and location are consistent with the actual damage area, pattern, and location in the structure. Each point cluster in Figure 6.1 of the same color is the segmented elements of the bridge (i.e. cyan is bridge deck, magenta is sides/guardrails, red is wing walls, etc.).

With the damage cube established and created as a way to store localized information about a particular defect and to easily visualize where the damage exists on a structure, the damage information and any other information about the defect is automatically uploaded into Revit and associated to a damage cube. Revit uses Dynamo to allow for automation of commands and functions. Dynamo is a node-based, visual programming language with each node representing either a variable or a function. The nodes, which can act as an input or output, are connected to create a automated program for model establishment. Using Dynamo, a script

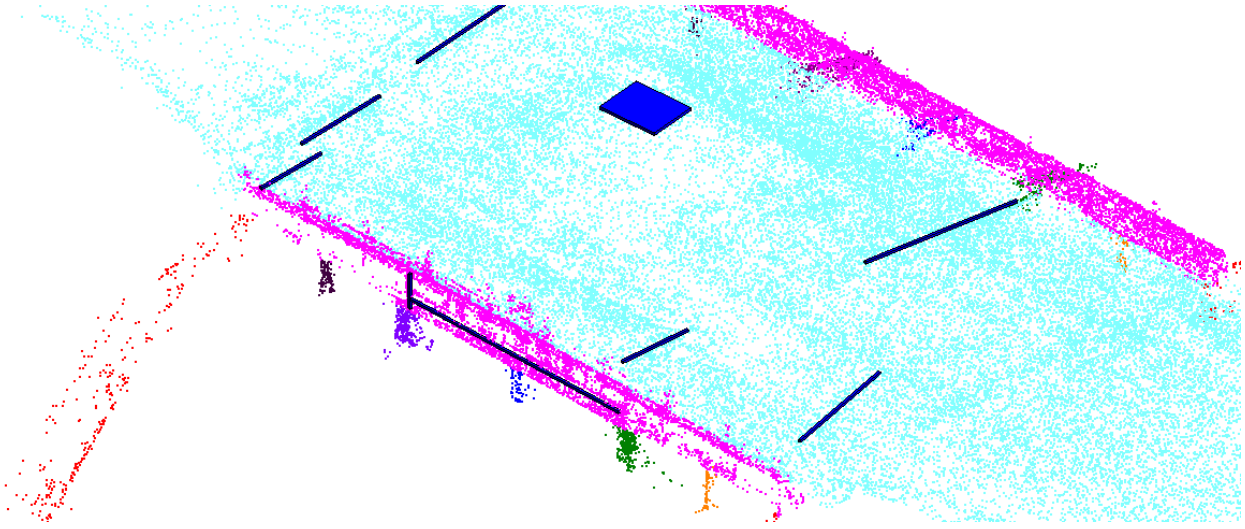


**Figure 6.1:** An Example Damage Cube (Blue Objects) Placed on the Segmented Point-Cloud (Cyan is Bridge Deck, Magenta is Sides/Guardrails, Red is Wing Walls, Etc.)

**Table 6.1:** Proposed and Example Parameters for a Damage Cube in Revit

| Proposed Damage Cube Parameters | Example Value |
|---|---|
| General Comments | "Suspected Delamination" |
| Area of Defect (*inches*$^2$) | 1728 |
| Width of Defect (*inches*) | N/A |
| Cracking | Unchecked Checkbox |
| Delamination | Checked Checkbox |
| Spalling | Unchecked Checkbox |
| Dimensions for Visualization | 5-*feet* × 4-*feet* × 2-*inches* |
| Location - Component | "Deck" |
| Location - Global | (-2.5091, 15.2083, 0.5000) |
| Original Image from UAV | "C:\0716.JPEG" |
| Damage Detected Images | "C:\0716_DD.JPEG" |
| Linked Report | "C:\FCPALM_201506.PDF" |

can be created to automatically create and place the damage cubes onto the segmented bridge elements. Assuming the damage location and size, which are found using the algorithm discussed in Chapter 4, and/or any other information that is relevant to a specific damage cube (i.e. inspector's comments, additional images, etc.) is stored in a .CSV file, Dynamo can read this information and create a custom damage cubes for each damage location. The created Dynamo visual program is shown in Figure 6.2 with each section zoomed into and discussed in Figure 6.3.

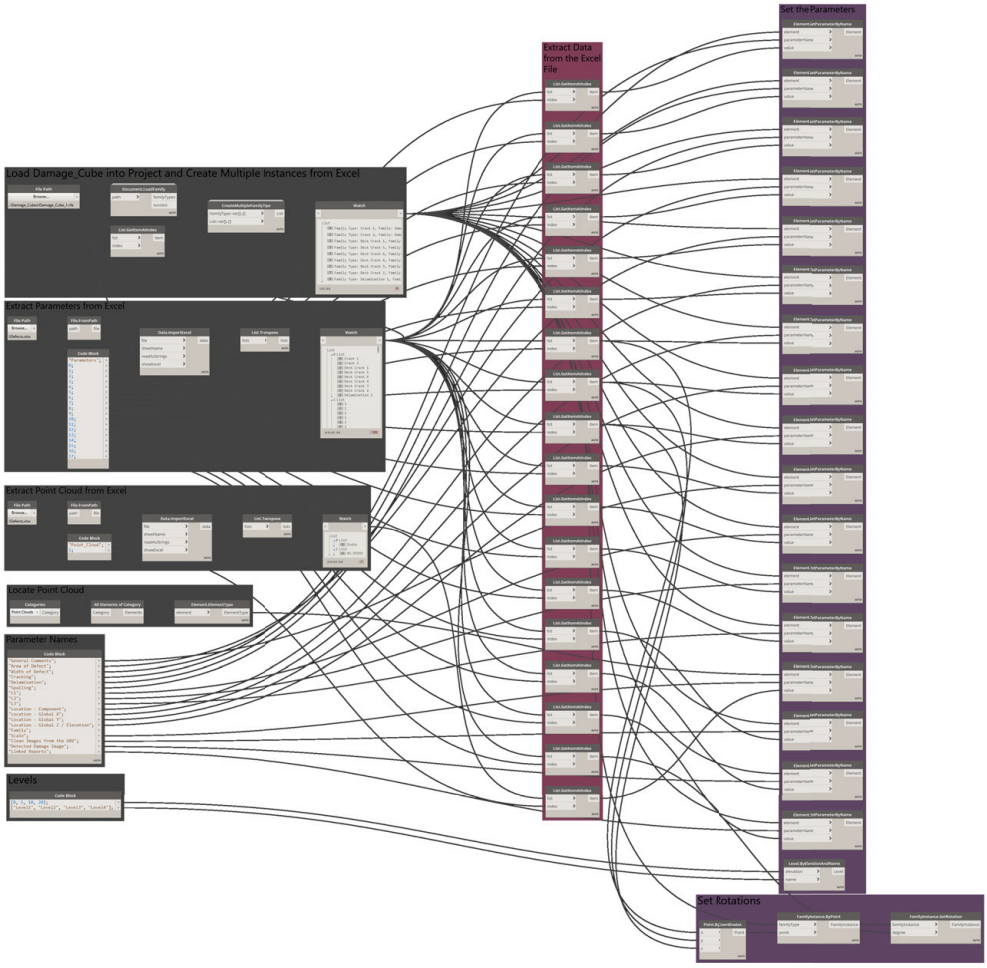
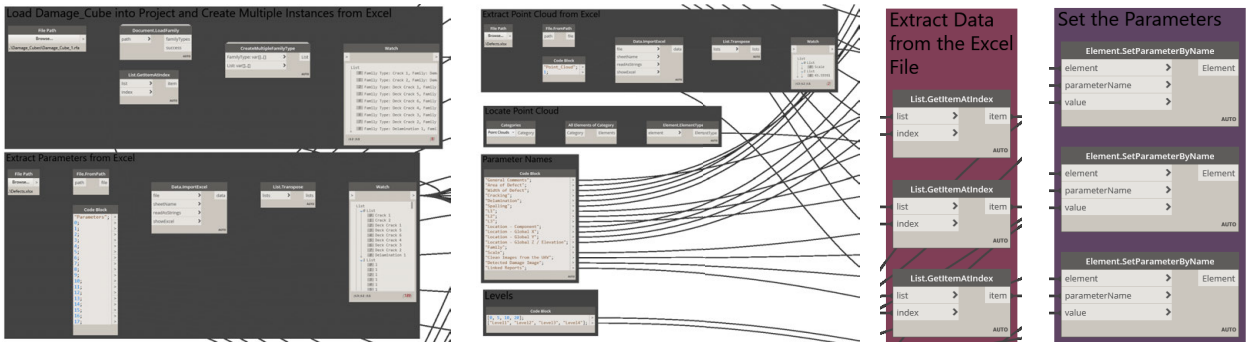



**Figure 6.2:** Full Dynamo Program to Automatically Create the Damage Cubes .CSV File

After the point-cloud is uploaded, scaled, and georeferenced in the Revit project, the damage cubes can be automatically placed on the segmented elements with the stored damage




**(a)** Nodes for Loading in "Damage Cube" Family and .CSV File with each "Damage Cube" Instance Information

**(b)** Nodes for Loading in Point Cloud and Some Constants Used

**(c)** Nodes for Extracting the Information from .CSV File

**(d)** Nodes for Setting the Parameters of the "Damage Cube" Instance

**Figure 6.3:** Node Grouping

information using Dynamo and the program described in Figure 6.2 and Figure 6.3. First, the damage cube Family and the .CSV file are loaded in the project. Dynamo creates a generic damage cube for each instance in the file. Next, Dynamo extracts all other information in the file associated with each damage cube instance and stores it in a list. With each generic damage cube instance created, the information in the extracted list from the .CSV file can be set as the damage cube's parameters. Once the parameters of the damage cube are set, the size, shape, location, and stored information of the damage cube is updated and, lastly, Dynamo draws each damage cube with the assigned information to create Figure 6.1.

Changes made in Revit to the point-cloud, damage cube, or information in the damage cube, can automatically be updated in the .CSV file for later use. Additionally, since the parameters are known for scaling and georeferenceing the point-cloud, it can readily be stored in the .CSV file which would further promote automated AB-BIM generation. Having all the information stored in a .CSV file could also condense the file size; instead of saving the entire Revit file, the information could be stored separately and the Revit file could be compiled and built when opening. More of file storage of the AB-BIM will be discussed in Section 6.2.

## 6.2 AB-BIM Usage

With the final AB-BIM assembled and usable, file size and storage becomes a concern. Table 6.2 shows the file required to run and build the AB-BIM. Under current inspection practices, only a PDF report is stored, with a size of about $1,200$-*KB*. With the proposed element-wise AB-BIM, significantly more disk space is required for storage; the majority of the disk space is the large number of extremely high resolution images. However, not all the images are required to be saved to view the model and can be kept for documentation purposes; therefore, keeping 20% or less will still provide more imagery and data than stored in the current PDF reports. Furthermore, the full resolution of the images is not required after the point-cloud generation and damage detection steps of the framework are complete. The images can be compressed after these two steps to save on storage. For example, reducing the image resolution by 50% reduces the image storage requirement by 87%. Lastly, having more imagery, data, and a robust AB-BIM that bridge owners and managers can use justifies the additional storage expense.

With the AB-BIM completely built, visualization of the overall health of the structure is more straightforward. A user can readily open the Revit project, click on a damage cube which highlights a defect on the structure to display a properties box with the damage type, (i.e. cracking, spalling, delamination), element location (i.e. deck, girder, beam, etc.), specific georeferenced location (i.e. x, y, z/elevation), area/length, remarks/comments, the images from the UAV, and any other associated information deemed necessary stored within. This visualization of the localized damage cubes with quantified, element-wise damage information and the concise information presented in the automatically generated schedules from Revit will help bridge owners and decision makers visualize the damage and make more informed decisions about planning and rehabilitation projects based on quantitative data.

In this chapter, the final AB-BIM is assembled and used as a one-stop decision-making support system. The integration with state DOT's current GIS-based data management system and quantified element-wise damage information allows for a more seamless transition from current practice to the proposed framework. This will empower bridge owners and managers

**Table 6.2:** Storage Size for AB-BIM using Revit

| | Storage Size | |
|---|---|---|
| Assembled Revit Model | 7,500-*KB* | 7.50-*MB* |
| .CSV File | 25-*KB* | 0.0250-*MB* |
| Point-Cloud Model | 7,000-*KB* | 7.00-*MB* |
| Photo-Realistic Model | 490,000-*KB* | 490-*MB* |
| Full "Palmer Bridge" Image Set (449 JPEG Images) | 2,160,000-*KB* | 2,160-*MB* |
| "Palmer Bridge" Image Subset (82 JPEG Images) | 424,000-*KB* | 424-*MB* |
| "Palmer Bridge" 50% Reduced Image Subset (82 JPEG Images) | 55,000-*KB* | 55-*MB* |
| **Assemble to View Process** | 62,025-*KB* | 62.025-*MB* |
| **Self-Contained Revit File** | 62,500-*KB* | 62.50-*MB* |
| **Current Report** | 1,200-*KB* | 1.2-*MB* |

tasked with making major maintenance decisions to better assess the overall health of a bridge and give more insight on a structure's maintenance and repair needs.

# Chapter 7

# Discussion of Results

To evaluate the efficacy of the proposed bridge inspection technique, a comparison study was conducted. Three types of bridge inspection procedures, i.e. the traditional human-based inspection, UAV-assisted inspection with manual image processing, and the fully automated inspection framework proposed in this study, were compared. The first type of approach represents the current state of practice in routine bridge inspection. The second type reflects the recent exploration of potential ways of using UAV in bridge inspection by state DOTs or private infrastructure inspection companies. In this case, a UAV would be used to fly around the structure collecting data and images, acting as "eyes-in-the-sky." The images can be used to automatically generate the 3D point-cloud and then manually processed to identify and map the damage onto the AB-BIM. The third approach takes full advantage of the new technologies with UAVs, machine learning and image computation techniques to automate the entire inspection process. The advantages and short-comings of each type of approach will be discussed in the subsequent subsections. The results are summarized in Table 7.1 at the end of the chapter.

## 7.1 Traditional Human-Based Techniques

In the first approach, an inspector and crew will travel to the job site and set up roping rigs (Figure 1.1a), "snooper trucks" (Figure 1.1b), or any additional equipment to gain access to the underside of the structure. The inspector will visually inspect all components of the structure being "within-arm's-reach" of critical components. Using a visual documentation and hammering or sounding techniques (such as chain dragging), cracks, spalling and delamination will be identified and manually documented. These techniques are more qualitative and often are related to the inspector's skill, experience, or hearing ability. The component's score tends to be more subjective and leads to less consistent data from inspector to inspector. Some images will be taken of the surrounding areas and components for later documentation. Occasionally,

markings or other notes will be recorded on the structure. This is to assist future inspectors to note the progression of existing cracks and defects over time.

Safety is a current concern of inspectors. To ensure a safe work site for the inspectors and crew, traffic control will be needed. This traffic control will impede the traffic flow and, if the structure is a major byway, negatively impact local traffic and movement of people or goods. During inspection of the underside or higher components, the inspector will be in harms way in order to gain access. However, having an inspector within this close proximity gives peace of mind to the community and bridge managers knowing that the defect was touched and viewed by a person.

Time and money costs are also relatively significant. The inspectors will take two or more weeks on site for large-scale or in-depth inspections. This not only creates cost from for each inspection for travel, wages, and traffic control, but also limits the total number of inspections one inspector can perform during the inspection season. Once all the data is collected, a report is compiled in office by the inspector, consuming more time and resources.

The access and visualization of the damage information in the future is difficult. The report, which is created and stored in the DOT's database, consists of a table with numerated information and appended images. In the future, the inspector has to use the limited damage information from the report to attempt to see the rate of change of the defects over time. With the couple of images collected for the structure, it is often difficult to visualize severity and location of the defects or to compare with a previous condition. Additionally, different crews or inspectors may perform the inspection in the future and make decisions based on these reports which are not as clear and quantitative.

## 7.2 UAV-assisted Inspection with Manual Defect Identification from Images

There are two ways to use UAV to assist bridge inspection. The first way is to use UAV to provide "eyes-in-the-sky" for inspectors. The second way is to use UAVs to collect the images

and then manually process the images and build an AB-BIM. Each of these two approaches will be discussed in this subsection.

In the first method, UAVs are used to perform pre-survey and/or enhance data collection during bridge inspection. A pre-survey will aid inspectors on site to plan for the best approach of performing the manual inspection. The UAV and its optical sensors will fly around the structure taking pictures or videos or be used as real-time surveillance of the structure. Using the collected images or videos, the inspectors on site can visualize the most important areas of interest and have the insight on where and how to perform the most efficient inspection. Additionally, the images could be included in the report and provide more data to be used in future decision making. This technique will assist the inspectors to better utilize the time on site and provide more data. However, this approach does not necessarily save time on the job site when compared to the traditional techniques. It does not take full advantage of the current technology and exploit the resources and tools available from the UAV technology; however, more data and better planning is possible using a UAV in this manner.

The second approach is to conduct the damage detection, quantification, and mapping manually from the images taken from UAV. The UAV and a crew will travel to the structure to fly around the bridge. Images will be taken all around the structure and, depending on the size of the bridge, 500 or more high resolution images will be collected. A 3D point-cloud and photo-realistic model will be created using the images as discussed in Chapter 3. An inspector will manually go through a percentage of the images. Because of the high overlap of the images, the inspector is not required to review all 500 or more images to identify the defects. Since there is a high overlap, there are multiple images of the same defect which would provide multiple perspectives to the inspector. Once the inspector finds the defect, it is marked and tagged to the 3D point-cloud and photo-realistic model. Any other relevant information can be tagged and logged on to this damaged spot with a damage cube, discussed in Chapter 6. The inspector and crew would spend significantly less time on the job site, for example, reducing time from a week or more to a day. Traffic control may still be required for certain areas. For example, the UAV,

under current Part 107 rules, cannot fly above a moving vehicle; therefore, traffic lane(s) may have to be controlled during the over head inspection. Additionally, if the structure crosses over another road, the road under the bridge may have to be controlled as well. Although traffic control will still be needed, the time that the traffic is impeded is reduced from days to hours. With this approach, substantially more time is needed for the final report and AB-BIM creation when compared to current techniques. This report creation time could increase from a day to multiple days in office. However, the final deliverable contains more damage information which helps to achieve better condition assessments and more complete data for future inspectors and decision-makers.

## 7.3 Proposed Automated Bridge Inspection Framework Implementation

The third implementation takes full advantage of the advancements in UAVs, as well as image processing and machine learning techniques. Using the framework proposed, the UAV collects images and data around the structure (Chapter 2), automatically builds a point-cloud and photo-realistic base for an AB-BIM (Chapter 3), automatically detects defects from the images and maps the defects on to the 3D point-cloud (Chapter 4). Harvesting the full potential of technology in the proposed framework, there will be minimal effort involved for the inspector and crew.

The proposed framework involves the inspector and crew traveling to the site and flying around the structure. Traffic control, as previously mentioned in Section 7.2, will still be required, but reduced from multiple days to hours. Once the 500 or more images are captured, the inspector uploads the images and automatically creates the AB-BIM based on the proposed framework. Although computing time and requirements are increase when compared to traditional techniques, the algorithms can run autonomously or be left overnight thus drastically reducing report generation time for the inspector. The inspector will also perform a quality control assessment afterwards to ensure performance of the algorithms. Once all this information

is processed and mapped, future inspectors and decision-makers can then revisit the point-cloud model and visualize the information more cohesively. The defects are located on a 3D rendering of the bridge, so the key elements can be located and accessed more easily. Additionally, later inspections are easily compared to previous inspections using this 3D visualization tool and additional photos and quantified damage information. This provides less time in the field, thus increasing the number of bridges that an inspector and crew can inspect in a season, while providing more information to be used for more robust decision-making.

**Table 7.1:** Comparison of Three Potential Implementations of the Proposed Framework

| | Traditional Human-Based Techniques | UAV-Assisted Inspection with Manual Defect Detection from Images (Second Approach) | Proposed Automated Bridge Inspection Framework Implementation |
|---|---|---|---|
| On-Site Time | two-*weeks* | one-*day* | one-*day* |
| Equipment | Roping Rigs, "Snooper" Trucks, and/or Scaffolding | UAV Platform | UAV Platform |
| In-Office Time | Minimal | Significant | Can be Autonomous |
| Deliverable | PDF Report | Element-wise AB-BIM | Element-wise AB-BIM |
| Damage Information | Manually and Qualitatively Documented | Manually Identified and Measured from Images | Automatically Detected |
| Safety | Special Equipment; Beside Traffic | Outside of Traffic | Outside from Traffic |
| Cost [19] | $58,856.64 | $20,000+ | $20,000 |

# Chapter 8

# Conclusion and Recommendations

Using the current human-based techniques outlined by AASHTO and FHWA, in-depth inspection of large-scale bridges is relatively slow, costly, and potentially dangerous while offering inconsistent and subjective results. Additionally, the extent and severity of the defects are not rigorously delineated on a rating scale. Once the final reports are created, it is often difficult to know exactly where each defect is located and to track the change of the defects over time. To overcome the short-comings in the current practice, an automated bridge inspection framework is proposed by leveraging the most recent advances in UAV technologies and machine learning techniques. Implementation of the proposed framework has been streamlined with element-wise damage information that is consistent with typical practices, and a geo-referenced AB-BIM to fit seamlessly into DOT's GIS-enabled systems. In this chapter, concluding remarks of this research are presented. Lastly, future research directions are discussed.

A complete framework for the implementation of a UAV-enabled bridge inspection is proposed. Images around the bridge are collected with a UAV. Using the collected images, a 3D point-cloud and photo-realistic model is generated using SfM and used as a base model for an AB-BIM. An automated element identification algorithm is developed to segment the 3D point-cloud into the structural elements of the bridge. In addition, a damage detection algorithm based on a Black Hat Transform and Canny Edge Detector is proposed to automatically identify defects such as cracks and spalling in the concrete bridges from images collected by the UAV. Next, an automated damage information mapping technique is developed to project the identified defects from the images onto the 3D point-cloud to find localization information and real-world measurements. Lastly, an automated modeling procedure is developed to compile all the information including both the structural elements and defects to build the final AB-BIM. Damage cubes, containing the quantified information, images, and any other relevant

information, are placed on the segmented elements of the point-cloud at the damage locations to provide simple visualization and information retrieval.

In summary, by utilizing UAVs for bridge inspection and developing automated algorithms for damage quantification and visualization, the proposed system is advantageous, yielding efficient, consistent, safe, and cost-effective data on bridge condition assessment. The proposed automated defect detection and mapping algorithms in conjunction with element identification algorithm highlight, locate, and size defects in the structure and map this information to an element-wise AB-BIM for easy visualization. This 3D visualization and damage documentation aids bridge owners and managers to make more informed decisions and more efficiently plan for future maintenance and rehabilitation projects.

## 8.1 Future Study

Future work of this project would continue to promote more robust inspection techniques by leveraging the advancements of optical sensors and additional payloads available. A module for analyzing thermal imagery to detect subsurface delamination of the bridge deck can be readily incorporated in the current framework. In addition, the integration of autonomous UAV flights with elevation changes will be explored. Autonomous flight of the UAV will promote more consistent data collection and enable more frequent and repeated inspections. Additionally, an autonomous flight would be more efficient for data collection by allowing for the optimal overlap of the images and by calculating the most efficient path of the UAV. Automatic tracking of changes of the structure over time becomes much easier with similar image collection. This information is valuable for conducting life-cycle analysis, or deterioration modeling as defects that change over time are often handled differently than defects that remain consistent. Lastly, as LiDAR systems become more affordable, they can be incorporated to allow for real-time data processing.

# Bibliography


[1] D. Reagan. *Unmanned Aerial Vehicle Measurement using Three'Dimensional Digital Image Correlation to Perform Bridge Structural Health Monitoring*. PhD thesis, University of Massachuestts Lowell, 2017.

[2] ASCE. 2017 Infrastructure report card. Technical Report 2017, 2016.

[3] A. Taddesse. *Bridge Inspection Techniques*. PhD thesis, Oklahoma State University, 2011.

[4] Thomas W. Ryan, Eric Mann, Zachary M. Chill, and Bryan T. Ott. Bridge Inspector's Reference Manual. Technical report, FHWA, Washington, D.C., 2012.

[5] AASHTO. *The Manual for Bridge Evaluation, Third Edition, 2017*. American Association of State Highway and Transportation Officials, Washington, D.C., 3rd edition, 2018.

[6] Tarek Omar and Moncef L. Nehdi. Remote sensing of concrete bridge decks using unmanned aerial vehicle infrared thermography. *Automation in Construction*, 83(April):360–371, 2017.

[7] Arun Mohan and Sumathi Poobal. Crack detection using image processing: A critical review and analysis. *Alexandria Engineering Journal*, 2017.

[8] Mark Moore, Brent Phares, Benjamin Graybeal, Dennis Rolander, and Glenn Washer. Reliability of Visual Inspection for Highway Bridges. Technical Report FHWA-RD-01-020, FHWA, Atlanta, GA, 2001.

[9] Stuart Spray. Rope Access Surveys, 2015.

[10] Branson. Inspectors Snoop Under Bridges to Check Conditions, 2014.

[11] I. Hernandez. *Overcoming the Challenges of Using Unmanned Aircraft for Bridge Inspections*. PhD thesis, University of Missouri-Kansas City, 2016.

[12] Simon Zingg, Davide Scaramuzza, Stephan Weiss, and Roland Siegwart. MAV navigation through indoor corridors using optical flow. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3361–3368, 2010.

[13] Nils Gageik, Michael Strohmeier, and Sergio Montenegro. An autonomous UAV with an optical flow sensor for positioning and navigation. *International Journal of Advanced Robotic Systems*, 10:1–9, 2013.

[14] Xiaoxia Li, Qiang Yang, Zhebo Chen, Xuejing Luo, and Wenjun Yan. Visible defects detection based on UAV-based inspection in large-scale photovoltaic systems. *IET Renewable Power Generation*, 11(10):1234–1244, 2017.

[15] W. K. Chiu, W. H. Ong, T. Kuen, and F. Courtney. Large Structures Monitoring Using Unmanned Aerial Vehicles. *Procedia Engineering*, 188(i):415–423, 2017.

[16] Chuang Deng, Shengwei Wang, Zhi Huang, Zhongfu Tan, and Junyong Liu. Unmanned aerial vehicles for power line inspection: A cooperative way in platforms and communications. *Journal of Communications*, 9(9):687–692, 2014.

[17] C; Gillins M; Simpson C Gillins, D; Parrish. Eyes in the Sky: Bridge Inspections With Unmanned Aerial Vehicles. 2018.

[18] Luis Duque, S M Asce, Junwon Seo, D Ph, M Asce, James Wacker, and M Asce. Bridge Deterioration Quantification Protocol Using UAV. 23(10):1–12, 2018.

[19] Jennifer Wells and Barritt Lovelace. Unmanned Aircraft System Bridge Inspection Demonstration Project Phase II Report No. MN/RC 2017-18. (June):1–174, 2017.

[20] Jennifer Wells, Bruce Holdhusen, and Jennifer Wells. TECHNICAL MnDOT Improves on Award-Winning Use of Drones for Bridge Inspection. (August), 2017.

[21] Jennifer Wells and Barritt Lovelace. Improving the Quality of Bridge Inspections Using Unmanned Aircraft Systems (UAS). *Minnesota Department of Transportation*, (July), 2018.

[22] Pix4D. Pix4DCapture, 2019.

[23] Jay Mulakala. Measurement Accuracy of the DJI Phantom 4 RTK & Photogrammetry. Technical report, DroneDeploy, San Francisco, CA, 2018.

[24] Junwon Seo, Luis Duque, and James P. Wacker. Field Application of UAS-Based Bridge Inspection. *Transportation Research Record*, 2018.

[25] Matthew N. Gillins, Daniel T. Gillins, and Christopher Parrish. Cost-Effective Bridge Safety Inspections Using Unmanned Aircraft Systems (UAS). *Geotechnical and Structural Engineering Congress 2016*, (August):1931–1940, 2016.

[26] Matthew N. Gillins, Daniel T. Gillins, and Christopher Parrish. Cost-Effective Bridge Safety Inspections Using Unmanned Aircraft Systems (UAS). *Geotechnical and Structural Engineering Congress 2016*, (October 2017):1931–1940, 2016.

[27] Seongdeok Bang, Hongjo Kim, and Hyoungkwan Kim. UAV-based automatic generation of high-resolution panorama at a construction site with a focus on preprocessing for image stitching. *Automation in Construction*, 84(August):70–80, 2017.

[28] William W. Greenwood, Jerome P. Lynch, and Dimitrios Zekkos. Applications of UAVs in Civil Infrastructure. *Journal of Infrastructure Systems*, 25(2):04019002, 2019.

[29] C.Brooks, R. Dobson, D. Banach, T. Oommen, K. Zhang, A. Mukherjee, T. Havens, T. Ahlborn, R. Escobar-Wolf, C. Bhat, S. Zhao, Q. Lyu, and N.Marion. Implementation of Unmanned Aerial Vehicles ( UAVs ) for Assessment of Transportation Infrastructure âĂŞ Phase II. 2018.

[30] Sattar Dorafshan, Marc Maguire, Nathan V. Hoffer, and Calvin Coopmans. Challenges in bridge inspection using small unmanned aerial systems: Results and lessons learned. *2017 International Conference on Unmanned Aircraft Systems, ICUAS 2017*, pages 1722–1730, 2017.

[31] B & H Photo and Video. Mavic, 2019.

[32] B & H Photo and Video. Phantom, 2019.

[33] B \& H Photo and Video. Matrice, 2019.

[34] Press Office. FAA Drone Registry Tops One Million, 2018.

[35] S. Shahnaz. *Gravel Road Condition Monitoring Using Unmanned Aerial Vehicle ( UAV ) Technology*. PhD thesis, South Dakota State University, 2010.

[36] PrecisionHawk. Precision Flight, 2018.

[37] Skycatch. Flights, 2018.

[38] Miller Hall, Bruce Wing, Kingston On, Miller Hall, Bruce Wing, and Kingston On. Selecting the optimal 3D remote sensing technology for the mapping, monitoring and management of steep rock slopes along transportation corridors. *TRB 2015 Annual Meeting*, pages 1–15, 2015.

[39] D. Roca, J. Armesto, S. Laguela, and L. Diaz-Vilarino. LIDAR-equipped UAV for building information modelling. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, 40(5):523–527, 2014.

[40] Adam J. Cawood, Clare E. Bond, John A. Howell, Robert W.H. Butler, and Yukitsugu Totake. LiDAR, UAV or compass-clinometer? Accuracy, coverage and the effects on structural models. *Journal of Structural Geology*, 98:67–82, 2017.

[41] Michal Jancosek and Tomas Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3121–3128, 2011.

[42] Pierre Moulon, Pascal Monasse, and Renaud Marlet. Adaptive Structure from Motion with a contrario model estimation. *Lecture Notes in Computer Science*, 7727(PART 4):1–14, 2012.

[43] D.G. Lowe. Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, pages 1150–1157, 1999.

[44] Kyle Simek and Mark Reid. Dissection the Camera Matrix, 2013.

[45] Changchang Wu. Towards linear-time incremental structure from motion. *Proceedings - 2013 International Conference on 3D Vision, 3DV 2013*, pages 127–134, 2013.

[46] Johannes L. Schoenberger and Jan-Michael Frahm. Structure-from-Motion Revisited. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[47] Michael Kazhdan, Mathew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. *Eurographics Symposium on Geometry Processing*, pages 1–10, 2006.

[48] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics*, 32(3):1–13, 2013.

[49] M. Jahanshahi. *Vision-Based Studies for Structural Health Monitoring and Condition Assessment*. PhD thesis, University of Southern California, 2011.

[50] S. Sankarasrinivasan, E. Balasubramanian, K. Karthik, U. Chandrasekar, and Rishi Gupta. Health Monitoring of Civil Structures with Integrated UAV and Image Processing System. *Procedia Computer Science*, 54:508–515, 2015.

[51] Ahmed Mahgoub Ahmed Talab, Zhangcan Huang, Fan Xi, and Liu Haiming. Detection crack in image using Otsu method and multiple filtering in image processing techniques. *Optik*, 127(3):1030–1033, 2016.

[52] Ning Zhang, Jeff Donahue, Ross Girshick, and Trevor Darrell. Part-Based R-CNNs for Fine-Grained Category Detection. *ECCV 2014*, 1:834–849, 2014.

[53] Qian-yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D : A Modern Library for 3D Data Processing. 2018.

[54] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[55] Junliang Li, Hon Cheng Wong, Sio Long Lo, and Yuchen Xin. Multiple Object Detection by a Deformable Part-Based Model and an R-CNN. *IEEE Signal Processing Letters*, 25(2):288–292, 2018.

[56] Sattar Dorafshan, Robert J. Thomas, and Marc Maguire. Comparison of deep convolutional neural networks and edge detectors for image-based crack detection in concrete. *Construction and Building Materials*, 186:1031–1045, 2018.

[57] Nhat-Duc Hoang, Quoc Lam Nguyen, and Van Duc Tran. Automatic recognition of asphalt pavement cracks using metaheuristic optimized edge detection algorithms and convolution neural network. *Automation in Construction*, 94(June):203–213, 2018.

[58] Gary Bradski. The OpenCV Library, 2000.

[59] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. Non-Local Means Denoising. *Image Processing On Line*, 1:208–212, 2011.

[60] Jean Serra. *Image Analysis and Mathematical Morphology*. Academic Press, Cambridge, MA, first edition, 1982.

[61] John Canny. A Computational Approach To Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–714, 1986.

[62] Richard Hartley and Andrew Zisserman. Multiple View Geometry in Computer Vision. In *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[63] Ning Zhang, Ryan Farrell, Forrest Iandola, and Trevor Darrell. Deformable part descriptors for fine-grained recognition and attribute prediction. *Proceedings of the IEEE International Conference on Computer Vision*, pages 729–736, 2013.

[64] Gavriil Tsechpenakis and Sotirios P. Chatzis. Deformable probability maps: Probabilistic shape and appearance-based object segmentation. *Computer Vision and Image Understanding*, 115(8):1157–1169, 2011.

[65] Y Narazaki, V Hoskere, T A Hoang, and B F Spencer Jr. Vision-based Automated Bridge Component Recognition Integrated With High-level Scene Understanding. 2017.

[66] Thomas Czerniawski and Fernanda Leite. *Advanced Computing Strategies for Engineering*, volume 10863. Springer International Publishing, 2018.

[67] HÃľlÃĺne Macher, Tania Landes, and Pierre Grussenmeyer. From Point Clouds to Building Information Models : 3D Semi-Automatic Reconstruction of Indoors of Existing Buildings. *Applied Sciences*, 7(10):1–30, 2017.

[68] Reza Maalek, Derek D Lichti, and Janaka Y Ruwanpura. Robust Segmentation of Planar and Linear Features of Terrestrial Laser Scanner Point Clouds Acquired from Construction Sites. *Sensors*, 18(8):1–30, 2018.

[69] Ruodan Lu, Ioannis Brilakis, and Campbell R. Middleton. Detection of Structural Components in Point Clouds of Existing RC Bridges. *Computer-Aided Civil and Infrastructure Engineering*, (0):1–22, 2018.

[70] Pasi Fränti, Olli Virmajoki, and Ville Hautamäki. Fast agglomerative clustering using a k-nearest neighbor graph. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1875–1881, 2006.

[71] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhfer, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and

Edouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[72] Niloy J. Mitra and An Nguyen. Estimating Surface Normals in Noisy Point Cloud Data. *SoCG'03*, pages 322–328, 2003.

[73] Bently. LEAP Bridge, 2017.

[74] Trimble. Tekla Structures, 2018.

[75] Autodesk. Revit, 2018.

[76] Brendan McGuire, Rebecca Atadero, Caroline Clevenger, and Mehmet Ozbek. Bridge Information Modeling for Inspection and Evaluation. *Journal of Bridge Engineering*, 21(4):04015076, 2016.