Determination of Elevations for Excavation Operations Using Drone Technologies

Yuhan Jiang
Marquette University

DETERMINATION OF ELEVATIONS FOR EXCAVATION OPERATIONS USING DRONE TECHNOLOGIES

by

Yuhan Jiang

A Dissertation submitted to the Faculty of the Graduate School,
Marquette University,
in Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy

Milwaukee, Wisconsin

August 2020

ABSTRACT
DETERMINATION OF ELEVATIONS FOR EXCAVATION OPERATIONS USING DRONE TECHNOLOGIES

Yuhan Jiang

Marquette University, 2020

Using deep learning technology to rapidly estimate depth information from a single image has been studied in many situations, but it is new in construction site elevation determinations, and challenges are not limited to the lack of datasets. This dissertation presents the research results of utilizing drone ortho-imaging and deep learning to estimate construction site elevations for excavation operations. It provides two flexible options of fast elevation determination including a low-high-ortho-image-pair-based method and a single-frame-ortho-image-based method. The success of this research project advanced the ortho-imaging utilization in construction surveying, strengthened CNNs (convolutional neural networks) to work with large scale images, and contributed dense image pixel matching with different scales.

This research project has three major tasks. First, the high-resolution ortho-image and elevation-map datasets were acquired using the low-high ortho-image pair-based 3D-reconstruction method. In detail, a vertical drone path is designed first to capture a 2:1 scale ortho-image pair of a construction site at two different altitudes. Then, to simultaneously match the pixel pairs and determine elevations, the developed pixel matching and virtual elevation algorithm provides the candidate pixel pairs in each virtual plane for matching, and the four-scaling patch feature descriptors are used to match them. Experimental results show that 92% of pixels in the pixel grid were strongly matched, where the accuracy of elevations was within ±5 cm.

Second, the acquired high-resolution datasets were applied to train and test the ortho-image encoder and elevation-map decoder, where the max-pooling and up-sampling layers link the ortho-image and elevation-map in the same pixel coordinate. This convolutional encoder-decoder was supplemented with an input ortho-image overlapping disassembling and output elevation-map assembling algorithm to crop the high-resolution datasets into multiple small-patch datasets for model training and testing. Experimental results indicated 128×128-pixel small-patch had the best elevation estimation performance, where 21.22% of the selected points were exactly matched with "ground truth," 31.21% points were accurately matched within ±5 cm.

Finally, vegetation was identified in high-resolution ortho-images and removed from corresponding elevation-maps using the developed CNN-based image classification model and the vegetation removing algorithm. Experimental results concluded that the developed CNN model using 32×32-pixel ortho-image and class-label small-patch datasets had 93% accuracy in identifying objects and localizing objects' edges.

ACKNOWLEDGMENTS

Yuhan Jiang

I would like to thank my advisor, Dr. Yong Bai, for guiding me and supporting me throughout every single step of writing this dissertation. He has contributed so much to my practical understanding of construction engineering and management.

I would like to thank my committee members, Dr. Saeed Karshenas and Dr. Wenhui Sheng, for their invaluable advice and words of encouragement.

I would like to thank the Civil, Construction and Environmental Engineering Department, the Graduate School and all of the Marquette University administration.

I would like to thank the MU employees who assisted me: Mr. Matthew Derosier, for maintaining the workstation system; Dr. Anna P. Scanlon, Mr. Quinn Furumo and Mr. Logan Newstrom for writing support.

Finally, but most importantly, I am thankful for my lovely family. Their belief in me has sustained me throughout my life and helped me to reach my goals.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

1.1 Background

Excavations on construction sites are multi-scale scenes for surveying and measuring – they vary from the larger area cut/fill projects with aerial-range measurements to the pit/trench excavation projects with close-range measurements (Nunnally 2004; Barazzetti et al. 2010; Nex and Remondino 2014; Spence and Kultermann 2016). Surveying plays a crucial role in determining the construction site's geometrical data – elevations and locations – which is important for measuring earth cut/ fill volume and designing the excavation plan (Nunnally 2004; Peurifoy and Garold 2014). Additionally, elevations also benefit construction professionals in optimizing earth moving path (Seo et al. 2011; Gwak et al. 2018), designing temporary hauling road (Yi and Lu 2016), estimating cost and time duration (Hola and Schabowicz 2010) and designing the site safety facilities as well (Wang, Zhang and Teizer 2015).

In the past decade, surveying on constructing site had been shifted from contact method to non-contact method. Historically, surveying operations on construction sites are accomplished by contact method – total station, GPS, measuring tape, level and theodolite (Nichols and Day 2010). Those tools help construction professionals to acquire enough site's geometrical information – distances, angles, points' positions and elevations – for measuring operations by drawing a site plan and calculating earth cut/fill quantities with the four-point method (Nunnally 2004; Peurifoy and Garold 2014; Spence and Kultermann 2016). Within the past decade, the non-contact surveying methods were developed and applied in construction surveying, which include the terrestrial laser scanning, vehicle-borne/ Unmanned Aerial Vehicle (UAV)-borne LiDAR (Du and Teng 2007; Takahashi et al. 2017; Kwon et al. 2017; Maghiar and Mesta 2018) and close-range/ aerial photogrammetry (Nassar and Jung 2012; Siebert and Teizer 2014; Sung and Kim 2016; Takahashi et al. 2017; Kwon et al. 2017; Maghiar and Mesta 2018). These non-contact surveying methods help construction professionals to acquire a point cloud – bunch of coordinated points – for creating a construction site's digital terrain model (DTM). Then construction professionals accomplish the measuring works with the DTM.

Although current surveying methods could achieve a precise result, their weaknesses are noticeable (Du and Teng 2007; Takahashi et al. 2017; Maghiar and Mesta 2018). The contact methods rely on surveyors' movement from a target point to next point in the construction site that leads to a time-consuming outdoor procedure and a high probability of interfering with other construction operations. The non-contact methods avoid the conflict issue and reduce the surveying time to some extent by scanning targets from one ground station to next station or scanning targets in a well-designed flight path that produces a huge amount of unfiltered targets, such as the vegetation and other attached objects on the construction site. However, processing the scanned data in non-contact method is not fast. A previous study stated that the duration for estimating on-site soil volume after drone photogrammetry is one processing day, under the conditions as the point cloud is generated by Agisoft PhotoScan, the geometry model is created by Autodesk ReCap with the point cloud, and the soil volume is estimated with Autodesk Civil 3D (Haur et al. 2018). In addition, the air-borne LiDAR system is not a reasonable surveying equipment for construction application until its price drops down to a low number (Guo et al. 2017). Thus, quickly and accurately determining elevations of a construction site in real-time is still a challenge for the construction industry.

A potential approach to minimize the processing time for determining the construction site elevations with image-based 3D-reconstruction method is that reducing the number of images need to be processed. Previous research tried 3D-reconstruction from single-frame image with other geometrical reference information in the past decades and confirmed that was an ill-posed problem if without any reference information (Van den Heuvel, 1998; Hassner and Basri 2006; Saxena et al. 2008). In recent years, researchers are continuously developing innovational approaches to estimate the relative-depth from a single-image, which takes the advantage of convolutional neural networks (CNNs) and deep learning (Eigen et al. 2014; Liu et al. 2015; Laina et al. 2016; Zhou et al 2017). For the construction industry, using the advanced artificial intelligence (AI) technologies to automatically determine elevations directly from an image of a construction site is an interesting research topic and meaningful challenge. Once overcome, the real-time 3D-reconstruction of a construction site become possible, then the automation degree of the excavation operations will be significantly improved (Seo et al. 2011).

1.2 Problem Statement

Recently, small sized drones (a system of quadcopter, gimbal and small sized digital camera) are increasingly regarded as the valid, cheap alternative remote imaging platform to large UAVs in civil engineering applications. Small dimensions make them easily navigable in cluttered outdoor environments and indoor environments (Takahashi et al. 2017; Siebert and Teizer 2014). In the drone application of construction site elevation determination, the main challenge is measuring vertical distances (depths) from the camera to the construction site ground surface. In Figure 1, the attached gimbal in the drone allows the camera to face any desired orientation. Specifically, when the camera's principal ray is perpendicular to the construction site ground surface plane, the captured image is the top-view of the construction site (Siebert and Teizer 2014), which is referred to as an ortho-image in this research project.

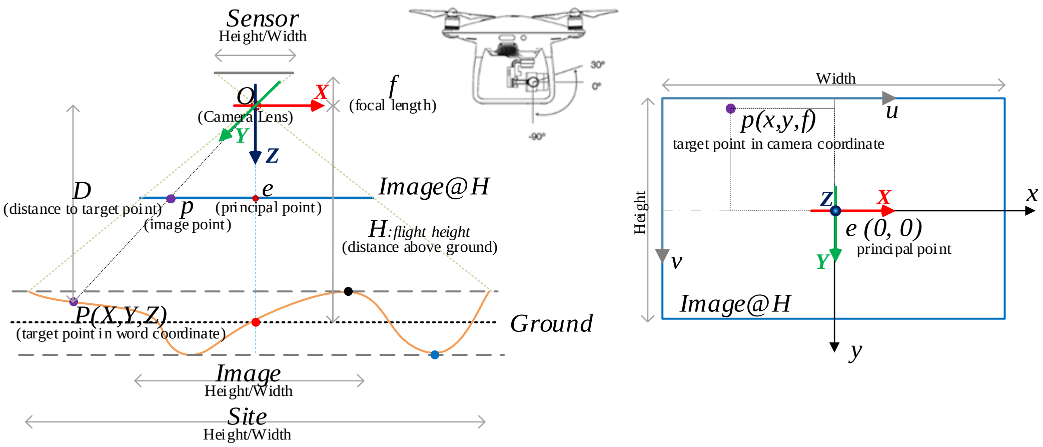


Figure 1 Drone-based ortho-image, camera model and coordinates

Using the camera model in Figure 1 to determine the distance from the camera lens to the ground is an ill-posed problem, which need at least an addition overlapping ortho-image from another position, and the spatial relationship between these two positions should be known. For example, the traditional aerial photogrammetry method needs a high overlapping ratio ortho-image series to complete image-based terrain 3D-reconstruction task (Nassar and Jung 2012; Siebert and Teizer 2014), which makes it impossible to generate and output the elevation data quickly. In contrast, the classic left-right stereo-vision method that is designed for determing depths of forward-facing objects is the fastest multiple image-based 3D-reconstruction method (Sung and Kim 2016; Sophian et al. 2017). The stereo-vision method performs a two-frame image-based 3D-reconstruction based on the triangulation model and saves all depth information in a depth-map (grayscale image) as the result. However, the stereo-vision method is limited to measure distances from objects' surface to the stereo camera system in a close-range, because its measurable depth range is limited by its small baseline (the distance between the two cameras). Furthermore, stitching stereo-vision results makes it not different from the traditional aerial photogrammetry method, and the multiple overlapping ortho-image based method had been confirmed that it is ineffectiveness with the large slope ground surface (Westoby et al. 2012; Zhao and Lin 2016). Thus, the construction industry still waits for a more rapid and simpler image-based 3D-reconstruction method for determining construction site elevations, which will help construction professionals to manage their crews and avoid excess waste during excavation operations.

Previous research results have shown the feasibility of using deep learning methods to recover the relative depth information for each pixel of an image of indoor scenes (Eigen et al 2014; Liu et al. 2015; Laina et al. 2016), outdoor scenes (Chen et al. 2016; Li and Snavely 2018) and scenes from automatic driving applications (Garg et al. 2016). In addition, convolutional neural networks (CNNs) have been verified as effective and reliable in micro-scale scenes, such as estimating the surface height map from a single image of a foam mat and mouse pad (Zhou et al. 2017). Using deep learning method to estimate the construction site elevations is equal to figure out the relationship between the elevation values' feature and ortho-images' features of construction sites, which is the reference information in the single-image 3D-reonstruction problem. To have the pixel-to-pixel relationship, the elevation values are better to save in the grayscale image, which is referred to as an elevation-map in this research project. However, the challenge is not only limited to find out an effective deep learning model from the previous research or develop a new deep learning model for this specific individual task, it also needs to create a comprehensive construction site ortho-image and elevation-map pair datasets, because there is no dataset available for training the deep learning model. Thus, an ortho-image-based 3D-reconstruction method should be developed in advance for acquiring the suitable datasets to train the deep learning model.

Additionally, the performance of image-based 3D-reconstruction method will be affected by the vegetations and other ground attached objects on the rough construction site when determing the ground elevations. This is because the light rays are reflected on the surface of vegetation instead of the "real" ground surface. In contrast, the contact surveying methods with total station, GPS, level and theodolite, can obtain the expected elevations as all selected target points are on the “real” ground surface. Thus, to improve the effective of the image-based 3D-reconstruction method in construction site elevation determination, the automatically detecting and removing the vegetation and other obstacles from the raw surveying results and determining the “real” ground elevations are needed and important for construction professionals to make the optimized decision in the excavation operations that heavily depend on the elevation information.

Previous research also shows the feasibility of deep learning methods in object detection tasks using image (Schneider et al. 2018), video (Kang et al. 2018), and image segmentation tasks (Noh et al. 2015; Badrinarayanan et al. 2017). The shortage of the current deep learning-based object detection methods is that using low-resolution images for training the deep learning-based object detector, which resizes the “ImageNet” (Deng et al. 2009) down to as small as 256×256-pixel, while the highest size is limited to 800×1000-pixel (Han et al. 2015). This low-resolution issue is caused by the limitation of computer system hardware. However, the directly exported images from a drone’s camera, such as the ortho-image captured by *DJI Phantom 4 Pro V2.0* is as large as 3648×4864-pixel, which is extremely larger than the small-resolution of 256×256-pixel. Using the small-resolution image dataset to train the object detection or image classification deep learning model can cause the loss of detail information. Reducing the image size also impacts on the image segmentation, because the number of pixels for an object will be reduced as the image size reduced.

In summary, to improve the speed and accuracy of image-based method for determing the construction site elevations for excavation operations, there is a need to develop a reliable method to collect construction site ortho-image and elevation-map pair datasets, develop an innovative way to train construction site elevation estimation deep learning model with high-resolution ortho-image and elevation-map pair datasets, and also develop a method to automatically identify and remove the vegetation dimensions on the raw elevation measures of the construction site.

1.3 Dissertation Organization

This dissertation includes eight chapters. This chapter is an introduction to the research background and problem statement. The following chapters are:

Chapter 2: Objectives, Scope, and Methodology. This chapter describes the primary objectives of this study as well as its scope and methodology.

Chapter 3: Literature Review. This chapter presents the findings from a comprehensive literature review on drone applications in construction operations and image-based 3D-reconstrution methods. This chapter also summaries the challenges and opportunities of drone and image-based method to determine elevations of a construction site.

Chapter 4: Low-high Ortho-image Pair-based Elevation Determination Algorithm Design and Testing. This chapter presents an effective, rapid and easily-implementable two-frame-image-based 3D-reconstruction method for the construction site elevation automatic determination.

Chapter 5: Ortho-Image and Elevation-Map Dataset Design and Acquisition Using Drone. This chapter details how to use the developed low-high ortho-image pair-based elevations determination method to acquire high-resolution construction site ortho-image and elevation-map pair datasets for training the deep learning-based construction site elevation estimation model.

Chapter 6: Ortho-Image and Deep Learning-Based Elevation Estimation Algorithm Design And Testing. This chapter presents a single-frame ortho-image-based 3D-reconstruction method for construction site elevation estimation, which is a convolutional encoder-decoder network model.

Chapter 7: Ortho-Image and Deep Learning-Based Vegetation Identifying and Removing Algorithm Design and Testing. This chapter presents a CNN-based image classification method to identify vegetation objects on the raw construction site using the high-resolution ortho-image and determine the "real" ground surface elevations from the raw surveying results.

Chapter 8: Conclusions and Recommendations. This chapter summarizes the procedures of the developed methods, concludes the findings of the testing experiments, and recommends potential improvements for future research on ortho-image and deep learning-based method in determining the elevation of construction sites. Contributions of this research project are outlined as well.

CHAPTER 2: OBJECTIVES, SCOPE AND METHODOLOGY

2.1 Research Objectives

To advance the construction site's elevations determination method into a non-contact, robust and rapid way by taking the advantages of drone technologies and eliminating the current construction surveying methods' shortfalls. This research project uses drone technologies, such as the gimbal-mounted camera to get a stable ortho-image, the onboard altimeter and the GPS to learn how high above the ground of a drone flies, and the camera model to calculate the geometrical data.

The primary goal of this proposed research is to advance the drone applications in construction site elevations (surface heights) determination. The general idea is to reduce the required number of images in the ortho-image based 3D-reconstruction. As the progress in success of this research project, the required ortho-image number is decreasing from multi-frame images by the traditional drone photogrammetry method to two-frame images by the developed drone-based low-high ortho-image pair-based method, and finally reducing to single-frame image by a well-trained convolutional encoder-decoder network model. This goal has been realized through achieving the specified research objectives that are described as follows:

1. To develop and test an innovational elevations determination algorithm to acquire a construction site' elevation-map with a drone-based low-high ortho-image pair – two ortho-images are captured in a high and a low position separately by a drone's camera facing down to the ground.
2. To create high-resolution construction site ortho-image and elevation-map pair datasets by the elevation determination algorithm described in the 1st objective with a drone.
3. To develop and test an elevation estimation deep learning model with the datasets created in the 2nd objective for estimating a construction site's elevations from its corresponding ortho-images captured by a drone.
4. To develop and test a high-resolution ortho-image classification method to identify and remove the vegetation obstacles from the elevation-map results in the 1st or the 3rd objectives.

2.2 Research Scope

In this research project, the proposed elevation determination methods were all based on a drone system acquired ortho-images, which means that these elevation determination methods focuses on 3D-reconstruction of a construction site's ground surface and excludes the vertical-side surfaces of all attached objects, which makes it much simpler than traditional drone photogrammetry. But the proposed methods were effective in determine elevation changes in vertical slopes, which is important to excavation operations.

This research project considered using one-frame ortho-image to cover a construction site as much as possible, which means it may not be able to cover an entire large site such as a roadway construction site. In the experiments, a drone system (*DJI Phantom 4 Pro V2.0*) flied at 10-20 m and 20-40 m over the ground, which had the measurable elevation range of [-5,5] m and [-10, 10] m and the area coverage of 8.47×8.47 m^2 and 17.6×17.6 m^2, respectively. In addition, this research project also considered the possibility of stitching ortho-images and elevation-maps results, the stitching experiments were conducted as well.

The construction site ortho-image and elevation-map pair datasets were collected from a lake beach site at Atwater Park, Shorewood, Wisconsin. The dataset acquisition happened during the year of 2019 with safe flight conditions. The construction site ortho-images were transformed from the 10-m flight height ortho-images with the high-resolution of 1568×1568-pixel. The generated elevation values were saved in the same sized 8-bit grayscale image, as elevation-map with the high-resolution of 1568×1568-pixel. Then, the construction site ortho-image and elevation-map pair datasets were built up. In addition, the high-resolution label-images were 8-bit grayscale image used for training the deep learning-based image classification (vegetation identifying) model, but they were saved in 1568-row and 1568-column spread sheet format. In this research project, these high-resolution image datasets were not resized down to training the proposed deep learning model, while a high-resolution image disassembling and model prediction image assembling methods were developed and tested.

2.3 Research Methodology

2.3.1 Literature Review

The first phase of this research is an extensive literature review. The literature survey includes state-of-the-practice in construction site surveying, image-based 3D-reconstruction, and theory of image processing and computer vision with deep learning. The reviewed literature includes journal papers, research reports, conference proceedings, theses, dissertations, and online publications.

2.3.2 Drone Photography

A *DJI Phantom 4 Pro V2.0* (a quadcopter drone equipped with automatic flight control system, GPS, altimeter, gyroscope, inertial measurement unit and other sensors) was used to hover at the desired position over the experiment site. The flight altitude data was directly read from the drone's remote controller, which has ± 0.00 set as the drone takeoff point. In addition, the 3-axis gimbal enhances the camera's stability, which was yielded to -90 ° when capturing ortho-images of experiment site.

2.3.3 Image Processing and Computer Vision with Deep Learning

A modified stereo-vision triangulation method was designed for construction site elevations determination in this research project. To automatically implement this computer vision method, the image processing of translation, rotation, resize and subpixel level image corresponding matching were conducted as well. In addition, the deep learning-based method with convolutional encoder-decoder model was used to estimate the elevation value from an ortho-image, and convolutional neural network-based image classification method was developed to identify the objects on the construction site.

The configuration of the computer system hardware and software environment is Python 3.6.8, OpenCV 3.4.2, Keras 2.3.1, TensorFlow-GPU 1.14, CUDA 10.0 and cuDNN 7.6.4.38 on a workstation system with 2×Xeon Gold 5122@3.6GHz CPUs, 96GB (8GB×12) DDR4 2666 MHz memory and 4×11GB memory GeForce RTX 2080 Ti GPUs.

2.3.4 Field Experiment

Filed experiments were conducted at a lake beach site (Atwater Park, Shorewood, WI, USA). Ortho-images were captured in the March, June and September of year 2019. Elevation data were generated from the proposed ortho-image pair-based elevation determination method. Label-images were manual drawn with an "Label-App" (programmed with Python 3.6.8 by the author) based on the corresponding ortho-image.

2.3.5 Data Analysis

Pearson correlation method was used to evaluate the relationship between the ortho-image pair matching quality and the ortho-image pair capturing quality, such as translation distance and rotation degree in the alignment of an ortho-image pair. Furthermore, descriptive statistic, histogram and contour plot were applied to evaluate the elevation differential between the deep leaning model prediction and "ground truth" of the selected grid points.

2.3.6 Summary

Conclusions were drawn based on the results of data analysis. The major findings included the effectiveness of the proposed two-frame ortho-image-based 3D-reconstruction method in determing the elevation of the experiment site and creating the ortho-image and elevation-map pair datasets for training the deep learning model; the comparison of the effectiveness of small-patch size and model training epoch in the deep learning-based elevation estimation network model and object classification network model with the high-resolution image datasets; and the effectiveness of the proposed vegetation removing method.

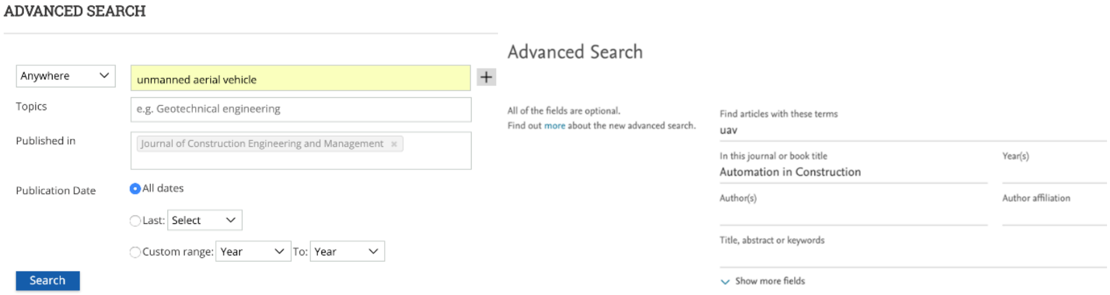
CHAPTER 3: LITERATURE REVIEW

3.1 Literature Review Procedures

The aim of this literature review is to seek an innovational approach to advance the construction site's elevations determination into a non-contact, robust and rapid way by taking advantage of drone technologies and eliminating the current methods' shortfalls. The specific objective is to investigate the potential of minimizing the processing time of drone and image-based 3D-reconstruction by reducing the number of images needed to be processed during the geometrical data determination process. To achieve that objective, two rounds of literature searches were conducted, which reversed the sequence of searches applied in previous review articles (Chan and Owusu 2017; Nasirian et al. 2019). Additionally, the theory of image processing and computer vision with deep learning were surveyed.

The first-round used the powerful "*Google Scholar*" engine to search the related terms of construction surveying such as "3D geometric measurement," "3D modeling," "3D reconstruction," "3D mapping," "3D terrain surface reconstruction," "Digital Terrain Model (DTM)," "scene depth recovery," and "image surface height recovery." This search round was not limited to the construction field, and all journal articles and conference proceedings were searched. After that, the overall statutes of 3D-reconstruction methods and technologies in different disciplines were clear. Then, comparisons of those methods were summarized in the findings section.

The second-round focused on the *Journal of Construction Engineering and Management* (COENG) and *Automation in Construction* (AUTCON), because those two journals are accepted as the top ranked publications in the construction field (Wing 1997; Nasirian et al. 2019). This search round was reserved to justify the reliability of using a drone in construction sites, and to find out which drone platform and sensor have been accepted and adopted in previous construction field research. The terms "Unmanned Aerial Vehicle," "UAV," "Drone," and "MAV" were searched based on the first-round results. In detail: in "ASCE Library" and "ScienceDirect," their "Advance Search" tools are used (see Figure 2) to search a term in "Anywhere," and keep other options blank; if the term occurs in an article, then the article returns to search result; and the review articles are excluded from the search results.



ADVANCED SEARCH

Anywhere: unmanned aerial vehicle

Topics: e.g. Geotechnical engineering

Published in: Journal of Construction Engineering and Management ×

Publication Date: ● All dates

○ Last: Select

○ Custom range: Year To: Year

Search

Advanced Search

All of the fields are optional.
Find out more about the new advanced search.

Find articles with these terms: uav

In this journal or book title: Automation in Construction

Year(s)

Author(s)

Author affiliation

Title, abstract or keywords

Show more fields

Figure 2 Search Configuration for ASCE library (left) ScienceDirect (right)

Table 1 lists the second-round search results with each screening step. The initial 15 COENG citations were manually screened and two duplicated articles were removed; the 139 AUTCON citations were exported to "RefWorks," and duplicate articles were detected and removed by the RefWorks function, then 74 unique articles remained. As the terms were matched in any part of an article, the returned citations included the articles that did not mainly discuss the drone's application, such as the selected terms that occurred in the literature review or related works part, the future research suggestion, or had the similar abbreviation, UAV and MAV. Thus, the full paper reading was conducted to filter those articles. After reading the full paper, one COENG article and 27 AUTCON articles were retained, as that research either discussed drone applications and problems or used drones to acquire the data. Additionally, the regression analysis method was adopted to predict the drone publication in COENG and AUTCON in the year of 2019 and 2020.

Table 1 Second-round Search Results

Return Citations	***Journal of Construction Engineering and Management***					***Automation in Construction***				
Initial Search Terms	Unmanned Aerial Vehicle	UAV	Drone	MAV	Total	Unmanned Aerial Vehicle	UAV	Drone	MAV	Total
	7	1	5	2	**15**	59	53	27	4	**139**
Duplicated Citations were removed	13					74				
After Full Paper was Reviewed	1					**27**				
Reference	Han et al. 2018					Aguilar et al. 2019; Bang et al. 2017; Chen et al. 2018; Ellenberg et al. 2016; Freimuth and König 2018; Hamledari et al. 2017; Han and Golparvar-Fard 2017; Inzerillo et al. 2018; Kim, D. et al. 2019; Kim, H. and Kim 2018; Kim, K. et al. 2017; Li, D. and Lu 2018; Li, F. et al. 2018; Metni and Hamel 2007; Moon et al. 2019; Morgenthal et al. 2019; Omar and Nehdi 2017; Park et al. 2019; Phung et al. 2017; Roca et al. 2013; Seo et al. 2018; Siebert and Teizer 2014; Wang et al. 2016; Yang et al. 2018; Zakeri et al. 2016; Zhang et al. 2015; Zhong et al. 2018				

3.2 Introduction to Excavation, Elevations/Point Cloud and Drone System

3.2.1 Elevations Data and Excavation Operations

Excavation is an essential construction activity to create the required planes and spaces for buildings and infrastructure facilities – such as footings, foundations, and underground utilities – to provide enough construction operating spaces for laborers and equipment (Spence and Kultermann 2016). According to the depth and area, excavations can be classified into three types: a) mass excavation, which usually removes larger amounts of earth from a huge depth and horizontal extent such as a building's basement; b) structural excavation, which removes earth in a confined area within a vertical extent and it might need a support system during excavating; c) grading, which reconfigures the construction site's landform from the irregular shape (natural/current grade) to the designed shape (finish grade).

In each type of excavation, the construction professional needs a geometry model of the construction site to accurately measure the volume of the earth to be excavated—if the current elevation is higher than the required elevation—or placed—if the current elevation is lower than the required elevation (Kraig et al. 2008; Spence and Kultermann 2016). Typically, a construction site is not an ideal level plane, which usually has an irregular topography; it can be divided into small elements in geometry, like trapezoidal bodies and cones (see Figure 3); after removing the vegetation and topsoil — then soil, rocks or mixture materials are exposed — the construction site has a rough surface (Nichols and Day 2010).

Trapezoidal Bodies
Cones

Figure 3 Excavation elements' shapes

Knowing the site's elevations is important to the construction professionals because excavation operations are complicated. Excavation operations are not limited to earthmoving activities such as grubbing, clearing, scraping, excavating, hauling, backfilling, compacting and finishing. They also include the important preparing works such as surveying, measuring, and planning before the actual earthmoving. Additionally, excavation operations require some management works, namely safety inspecting, progress monitoring and quality controlling works during excavating. Finally, documenting and recording works are needed after each excavation operations.

An excavation plan links the earthmoving, surveying and measuring, progress monitoring and quality controlling operations. The quality of the excavation planning depends on the accuracy of the site surveying and measuring, which leads to a good excavation plan and lowers the project cost by balancing the cut/fill materials in a large-area project or a roadway project. (Seo et al. 2011; Peurifoy and Garold 2014; Gwak et al. 2018).

3.2.2 Point Cloud and Site Modeling

The fundamental data used to build a 3D geometry model is a point cloud, which is a set of vertices (x_i, y_i, z_i) in a three-dimensional coordinate system (Remondino 2003; Nassar and Jung 2012; Rusu and Cousins 2011). Additionally, each point in the point cloud has its color features— either RGB (red, green, blue) or BGR (blue, green, red), depending on the programming platform used. Specifically, in the OpenCV— open source computer vision, a library of programming functions mainly aimed at real-time computer vision — the color format is BGR sequence, where each color has the value 0 ~ 255; while in the OpenGL — open graphics library, a cross-language, cross-platform application programming interface (API) for rendering 2D and 3D vector graphics — the color format "glColor3f (0.0, 0.0, 0.0)" is RGB sequence and each color has the value 0.0 ~1.0.

In contact surveying, target points are selected by surveyors to represent excavation objects' geometry features. The excavation objects are modeled in a combination of geometry elements (Figure 3) or a site plan (Figure 4), then the lengths, widths, heights, and slopes of excavation objects are measured from the site plan or the geometry model to calculate the volume of excavation (Nichols and Day 2010). For a large-area grading project, the site plan usually needs to be divided into equal-sized grids. The deviations between the current and required elevations at the four-corners represent the grid's cut / fill quantity (Peurifoy and Garold 2014; Gwak et al. 2018). While, in non-contact surveying, a target object is recorded as a point cloud, and a construction site is modeled in a Triangulated Irregular Network (TIN) model (see Figure 5) (Tsai 1993; Shewchuk 2002; Hearn et al. 2004; Sung and Kim 2016). Similar to the grid in the site plan, each triangle in the TIN has three corners with two elevations. Commercial software, such as the Autodesk Civil 3D, can semi-automatically accomplish TIN modeling works through external point cloud files (Nassar and Jung 2012).

Ideally, using point cloud to 3D model a construction site should only contain the minimum requirement number of target points, such as the corners of grids or triangles, to represent the site's geometric shape. However, that is impossible for the existing image-based feature matching technologies, without manually selecting or screening. Thus, the total station and GPS surveying still are the most accurate surveying methods to determine elevations of a construction site. But, on the other hand, they are both time-consuming and expensive. The detail of existing approaches in surveying will be discussed later.

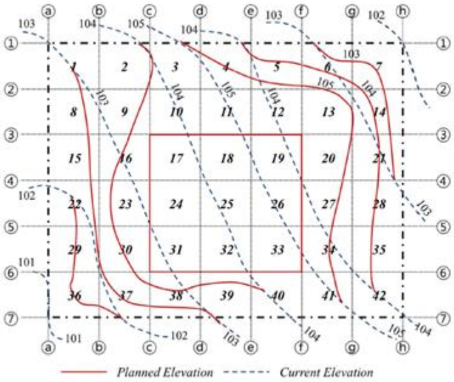


Figure 4 An example of site plan with grid lines and contour lines

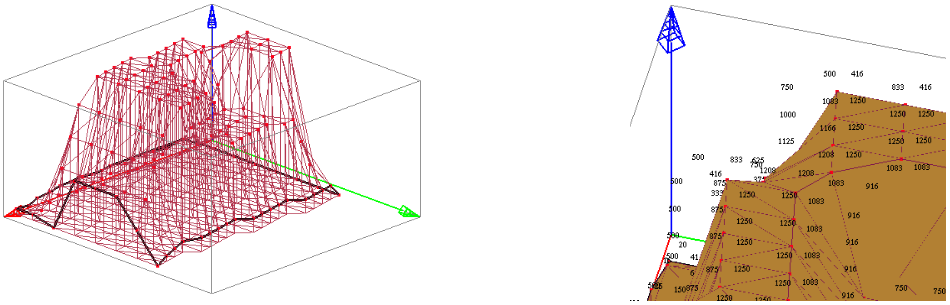


Figure 5 An example of TIN model

3.2.3 Drone Systems and Ortho-imaging

The Unmanned Aerial Vehicles (UAVs) were developed and used in military applications in the past, because their weight, size and high cost of insurance limit their commercial applications (Van Blyenburgh 1999; Siebert and Teizer 2014). With the development of precise GPS, gyroscopes and inexpensive inertial measurement units (IMUs), the performance of UAVs had been significantly improved, especially in its payload, flight endurance, stability, reliability and safety (Nex and Remondino 2014; Siebert and Teizer 2014). The micro aerial vehicles (MAVs) are increasingly regarded as the valid, cheap alternative to UAVs in civil applications, and their small dimensions make them accessible to some spatial conditions which are inaccessible with the large UAVs, such as cluttered outdoor settings and indoor settings (Bernardini et al. 2014).

The DJI Phantom series quadcopters are the most successful consumer MAV products, which are becoming the synonym of “Drone” to the public. A drone integrated with a gimbal-mounted camera (see Figure 6) is becoming the most popular remote imaging platform with diverse applications (Nex and Remondino 2014; Siebert and Teizer 2014; Takahashi et al. 2017), because the gimbal enhanced the camera’s stabilization in 3 axes (pitch, roll, yaw) and also make its rotation controllable in the pitch-axis from -90° to 30°, which is shown in Figure 6.

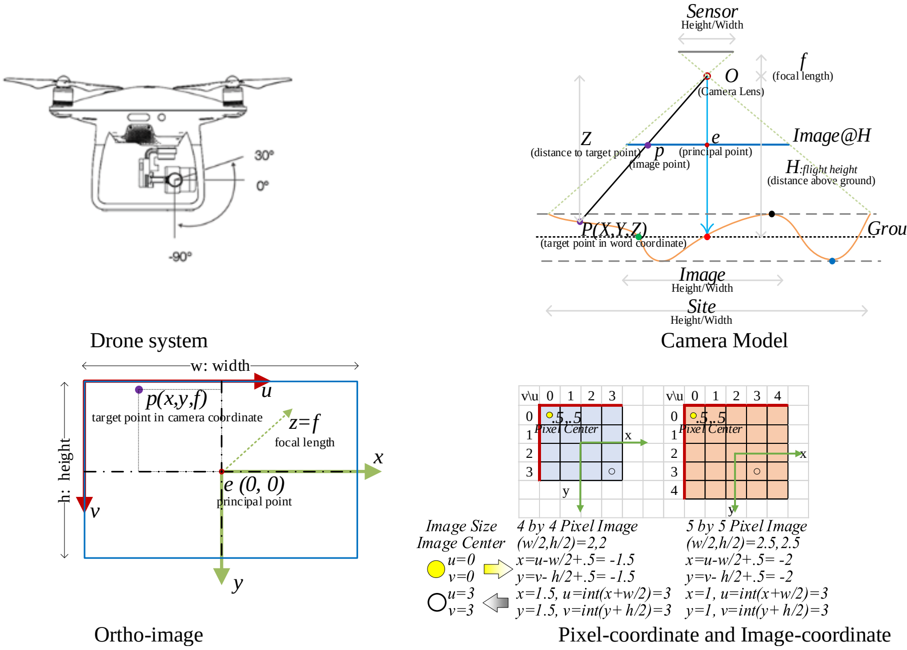


Figure 6 Drone system, drone-based ortho-image and geometry model

Specifically, when the pitch-axis of the gimbal is at -90°, the camera is just facing down to the ground, to which then the image captured is called either a plan view (Zhang et al. 2015), orthophoto (Westoby et al. 2012; Siebert and Teizer 2014), or ortho-image— the camera’s principal ray is perpendicular to the ground level plane. The ground sample distance (GSD) defined in Eq. 1 is the spatial resolution of an ortho-image taken by a drone at a specific height. The GSD has the unit $m/pixel$ or $cm/pixel$, which means each pixel of the image stands for the distance in the real-world in meters or centimeters.

$$Spatial\ Resolution\colon GSD = \min(GSD_h, GSD_w) = \min\left(\frac{Z \cdot \mathrm{h_{sensor}}}{f \cdot \mathrm{h}_{image}}, \frac{Z \cdot \mathrm{w_{sensor}}}{f \cdot \mathrm{w}_{image}}\right) \qquad \textbf{Eq. 1}$$

Where, f is focal length of the camera, with unit: mm
Z is the flight height, distance above ground, with unit: m
α is the factor to convert sensor size (mm) to image resolution size (pixel)

3.3 Drone Related Research on Construction Sites

Table 2 and Table 3 summarize all drone research articles in COENG and AUTCON. The first drone application article occurred in 2007 in AUTCON (Metni and Hamel 2007). There was a large increase in drone application articles published from 2015 to 2018 (see Figure 7). Figure 8 utilized the second-order polynomial regression to predict the number of drone application articles published based on the data from 2014 to 2018, the yearly increase will be 6 articles in COENG and AUTCON for 2019 and 2020. Comparing COENG and AUTCON, AUTCON includes more drone research than COENG. Among those 28 research articles, different types of drones (see Figure 9) have been used as data acquisition platforms, and the type of data obtained is dependent on the type of sensor installed (see Table 3):

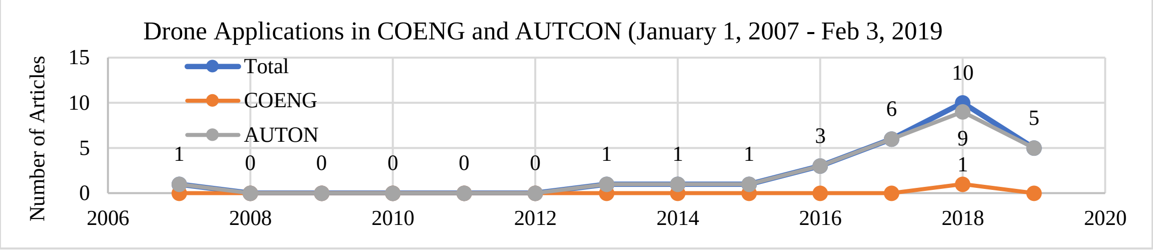


Figure 7 Number of drone applications in COENG and AUTCON

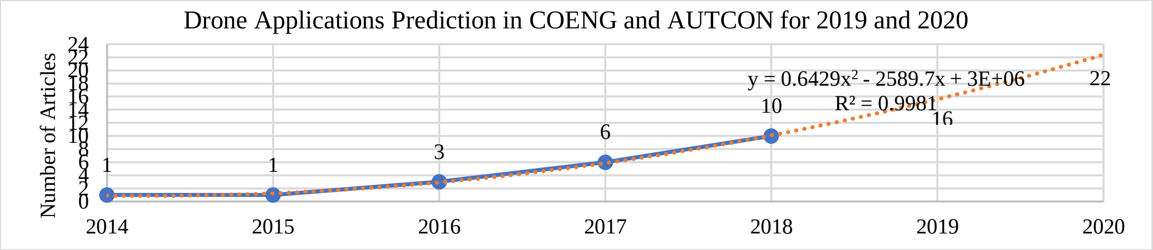


Figure 8 Prediction of drone applications in COENG and AUTCON for 2019 and 2020

Table 2 Drone Applications in COENG and AUTCON Part I

ID	***Reference***	***Year***	***Journal***	***Applications***				
				Surveying / Modeling	***Safety***	***Progress Monitoring***	***Inspection***	***Others***
1	Aguilar et al. 2019	2019	AUTCON	Building / Hybrid Point Cloud: Drone and Terrestrial Photogrammetry				
2	Bang et al. 2017	2017	AUTCON					High-resolution Construction Site Panorama Generating
3	Chen et al. 2018	2018	AUTCON	Building / Hybrid Point Cloud: Drone Photogrammetry and Laser Scanning				
4	Ellenberg et al. 2016	2016	AUTCON				Bridge Deck Delamination	
5	Freimuth and König 2018	2018	AUTCON				Visual Inspection	
6	Hamledari et al. 2017	2017	AUTCON					Indoor Under-construction Components Detecting
7	Han and Golparvar-Fard 2017	2017	AUTCON			Construction Performance Analytics		
8	Inzerillo et al. 2018	2018	AUTCON	Road Pavement / Point Cloud: Drone and Close-range Photogrammetry			Road Pavement Distress	
9	Han et al. 2018	2018	COENG			Construction Progress Monitoring		
10	Kim, D. et al. 2019	2019	AUTCON		Determine Distance Between Mobile Construction Resources			
11	Kim, H. and Kim 2018	2018	AUTCON	Concrete Mixer Truck / Point Cloud: Drone Photogrammetry				
12	Kim, K. et al. 2017	2017	AUTCON		Hazard Detection			
13	Li, D. and Lu 2018	2018	AUTCON	Construction Site / Hybrid Point Cloud: Drone Photogrammetry and Laser Scanning				
14	Li, F. et al. 2018	2018	AUTCON					Indoor Path Planning
15	Metni and Hamel 2007	2007	AUTCON				Concrete Crack	
16	Moon et al. 2019	2019	AUTCON	Construction Site / Hybrid Point Cloud: Drone Photogrammetry and Laser Scanning				
17	Morgenthal et al. 2019	2019	AUTCON				Bridge Structural Elements	
18	Omar and Nehdi 2017	2017	AUTCON				Concrete Bridge Decks	
19	Park et al. 2019	2019	AUTCON	Construction Site / Hybrid Point Cloud: Drone and UGV Photogrammetry				
20	Phung et al. 2017	2017	AUTCON				Planar Surfaces	Path Planning
21	Roca et al. 2013	2013	AUTCON				Building Facades	
22	Seo et al. 2018	2018	AUTCON				Bridge Inspection	
23	Siebert and Teizer 2014	2014	AUTCON	Construction Site / DTM				
24	Wang et al. 2016	2016	AUTCON					On Road Vehicles Detecting and Tracking
25	Yang et al. 2018	2018	AUTCON	Construction Site / DTM				Path Planning
26	Zakeri et al. 2016	2016	AUTCON				Asphalt Pavement	
27	Zhang et al. 2015	2015	AUTCON			As-built Construction Site Status		
28	Zhong et al. 2018	2018	AUTCON				Concrete Crack	
	Subtotal			9	2	3	11	6

Table 3 Drone Applications in COENG and AUTCON Part II

Articles ID	*UAVs / Drones*	*Sensors*				*Raw Data and Processed Data*						*Reference*
						3D model		*2D image*				
		Optical Camera	*RGB-D Camera*	*Thermal Camera*	*Point Cloud*	*Mesh/Texture surface*	*Others*	*RGB*	*Gray*	*Thermography*	*Depth-map*	
1	DJI Inspire 1	f =20mm, 12 MP			X							Aguilar et al. 2019
2	X	X						X				Bang et al. 2017
3	X	X			X							Chen et al. 2018
4	6-rotor	X		X						X		Ellenberg et al. 2016
5	3DR IRIS+	X						X				Freimuth and König 2018
6	quadcopter	X						X				Hamledari et al. 2017
7	X	X			X		BIM 4D					Han and Golparvar-Fard 2017
8	quadcopter	GoPro Hero 3			X	X						Inzerillo et al. 2018
9	X	X			X							Han et al. 2018
10	3DR	GoPro						X				Kim, D. et al. 2019
11	X	X			X	X						Kim, H. and Kim 2018
12	X	X						X				Kim, K. et al. 2017
13	DJI Inspire 1 Pro	X			X							Li, D. and Lu 2018
14	X	X									X	Li, F. et al. 2018
15	Helicopter	X						X				Metni and Hamel 2007
16	DJI Phantom 3	X			X							Moon et al. 2019
17	Intel Falcon 8	Sony Alpha 7R f= 35 mm			X			X				Morgenthal et al. 2019
18	DJI Inspire 1 Pro			X						X		Omar and Nehdi 2017
19	DJI Mavic	X			X							Park et al. 2019
20	X	X						X				Phung et al. 2017
21	8-rotor by Okto Xl		Kinect		X			X			X	Roca et al. 2013
22	DJI Phantom 4	X						X				Seo et al. 2018
23	Mikrokopter Qual XL	Sony NEX5N f=16 mm,16.1MP			X							Siebert and Teizer 2014
24	DJI Phantom 2	X						X				Wang et al. 2016
25	DJI Phantom 3	X			X		DTM					Yang et al. 2018
26	QUAV platform	GoPro							X			Zakeri et al. 2016
27	X	X						X				Zhang et al. 2015
28	8-rotor by Beijing TT Aviation Technology Co., Ltd.,	Canon 5D Mark III							X			Zhong et al. 2018
Count	28	26	1	2	13	2	2	12	2	2	2	28

Note: the label "X" indicates the brand and specification are not described by authors in their research articles.

1. The most popular drone brand adopted in these articles is "*DJI*", with the "*DJI Phantom*" series (Wang et al. 2016; Seo et al. 2018; Yang et al. 2018; Moon et al. 2019), "*DJI Inspire*" (Omar and Nehdi 2017; Li, D. and Lu 2018; Aguilar et al. 2019) and "*DJI Mavic*" (Park et al. 2019). Before quadcopter drones were created, remote-control helicopters were being used in bridge inspection (Metni and Hamel 2007). Although less common than the quadcopter drone, the more powerful 6-rotor (Ellenberg et al. 2016) and 8-rotor drones (Roca et al. 2013; Zhong et al. 2018; Morgenthal et al. 2019) have also been applied in construction research.
2. The most common sensor is "Optical Camera" in drone applications. Two "Thermal Camera" were applied in bridge inspection (Ellenberg et al. 2016; Omar and Nehdi 2017), and a "RGB-D Camera", the Kinect, was used in the inspection of building facades (Roca et al. 2013). The interesting thing is that none of those articles used the drone-borne LiDAR technology.
3. The most used 2D image style is RGB image, which is captured directly from the "Optical Camera" and the most used 3D model style is "Point Cloud" generated by photogrammetry or SfM method. The tendency to use "Point Cloud" to replace "RGB" images started in 2016 (see Figure 10), but for some specific applications, the processing of 2D images is more effective and faster than the 3D model, such as surface planer inspection (Phung et al. 2017).
4. The most two common applications are "Inspection" and "Surveying/Modeling" (see Figure 11). The 11 inspections listed in Table 2 were mainly based on 2D images – "RGB" and "Infrared Thermography". Surveying/Modeling by drone photogrammetry to produce a 3D point cloud is another important drone application on construction sites. The point cloud is the foundation to conduct other construction research, which are highly dependent on geometrical data, such as indoor drone path planning (Li, F. et al. 2018), and road pavement distress detection (Inzerillo et al. 2018).

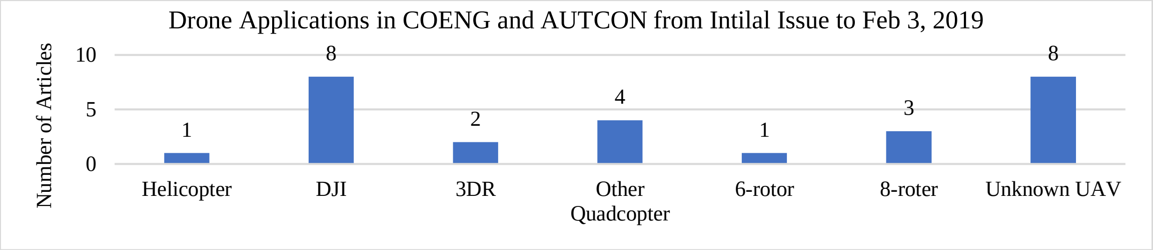


Figure 9 Drone categories of drone applications in COENG and AUTCON

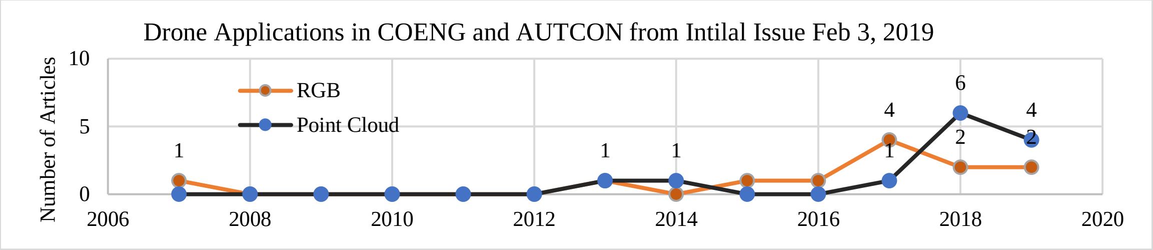


Figure 10 Data style of drone applications in COENG and AUTCON

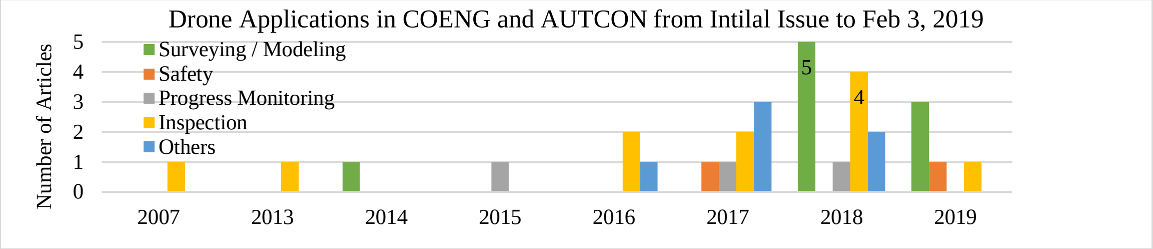


Figure 11 Applications of drone research in COENG and AUTCON

Other than those 28 research articles in COENG and AUTCON, the first-round of search also returned some drone applications which are beneficial to excavation operations. Drones started service as a safety visual inspection tool in excavation operations (Irizarry et al. 2012; Gheisari et al. 2014; Ashour et al. 2016; Gheisari and Esmaeili 2016; Kim et al. 2016). A real-time video stream of a construction site was captured by the drone and transferred to each safety responsibility official for visually detecting hazards and interacting with workers through communication speakers (Irizarry et al. 2012; Gheisari et al. 2014). Using drones to prevent excavation accidents is based on the knowledge that sharing the real-time construction site conditions to all construction participators will help participators take advantage of the real-time conditions to avoid the hazards (Toole 2002). Furthermore, with the site 3D point cloud, Wang et al. (2015) developed an algorithm to automatically extract height data from the 3D point cloud to identify and locate fall hazards at an excavated pit. Another approach to prevent excavation accidents by using guardrails to avoid workers falling into an excavation pit (Toole 2002).

Additionally, previous research also extended drone application to other fields, such as RFID materials tracking on construction sites (Hubbard 2015) and construction quality control (Wang, Sun et al. 2015). Thus, if a drone system could acquire the real-time elevations, then the efficiency of construction site safety management and quality control would be improved. Real-time quality control is important because 6~12% of cost is wasted due to reworks of defective components, which are detected late during construction (Josephson and Hammarlund 1999). The improvement of planning, real-time inspecting and feedback can ensure the quality of construction works, reduce project duration and avoid exceeding cost as well (Wang, Sun, et al. 2015).

3.4 Construction Surveying Related Research

3.4.1 Constructions Surveying Techniques

Figure 12 and Table 4 summarize the existing approaches and their general procedures in construction site surveying and modeling. Surveying (data scanning) is the starting procedure in every construction work, especially in excavation operations, which determines a construction site’s elevations and locations. The construction site surveying methods have experienced a progression from manual to automatic, from contact to non-contact, and from small to large scene size as well (Nex and Remondino 2014). Modeling is the second procedure, which processes the raw data from the surveying results and creates a geometry model, or a site plan, of the site. The comparisons of surveying techniques are compared using scanning result, measurable area and distance range, capacity, advantages and disadvantages.

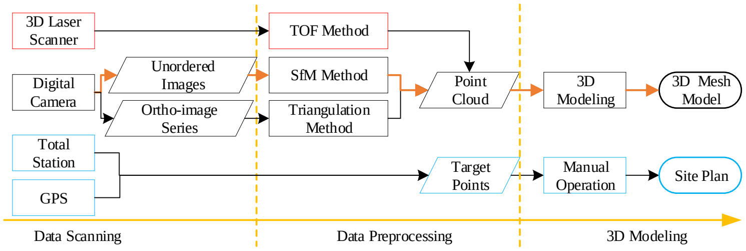


Figure 12 Approaches in construction site surveying and modeling

Table 4 Comparison of Construction Surveying Techniques

Techniques/Types			*Scanning results*	*Scanning Capacity*	*Scene Size*
Contact	Manual	Measuring tape, Level, Theodolite	Angular deviations, horizontal, vertical and slope distances between two target points	1,000 selected target points	≤100 m
	Semi-automatic	Total station			≤10 km
		GPS surveying	A bunch of selected target points with GPS coordinates		≤1000 km
Non-	Ground	Terrestrial Laser Close-range Photogrammetry	3D point cloud and Digital Terrain Model (DTM)	≥10 million raw points	≤100 m by setting up several ground stations
	Drone-based	Drone-borne LiDAR Drone Photogrammetry / Drone-SfM			≤ 5 km by a pre-planned drone path

3.4.1.1 Contact Surveying

Measuring tape, level and theodolite are manual surveying tools for small size construction sites. Their surveying results are angular deviations, horizontal, vertical and slope distances between two target points (Nichols and Day 2010). They are suitable for surveying the range of 0 ~ 100 m, and the surveying capacity is about 1,000 target points (Remondino and El-Hakim 2006; Nex and Remondino 2014).

Total station and GPS surveying devices are semi-automatic surveying equipment, which are the most popular surveying methods on construction sites. They extend the surveying sense size to 10 km and 1,000 km respectively (Remondino and El-Hakim 2006; Nex and Remondino 2014). Total station—an electronic theodolite integrated with an electronic distance measurement (EDM)—records the distance, the angle and the height between two target points. GPS surveying records many selected target points with GPS coordinates. Both have a 1,000 target points surveying capacity, which is the same as the manual surveying tools (Nex and Remondino 2014).

These manual tools and semi-automatic equipment are used in contact surveying methods, which rely on surveyors' movement on the construction site and the placing of the surveying device on the target points in a sequence. That means the target points are manually selected by surveyors, and their accuracy could be guaranteed. However, the manual tools and the total station need at least two cooperating surveyors to complete the surveying task on the construction site. This time-consuming outdoor procedure leads to a high probability of interfering with other construction operations on the construction site. In addition, they cannot provide the in-time progress data after the excavation starts.

3.4.1.2 Non-contact Surveying

Remote sensor based surveying methods are continuously being developed and tested in the construction industry, which include 3D Laser Scanning — by Terrestrial Laser (Du and Teng 2007), Drone-borne LiDAR (Tulldahl and Larsson 2014; Guo et al. 2017) , and Photogrammetry — by Close-range Photogrammetry (Arias et al. 2005; Barazzetti et al. 2010; Sung and Kim 2016) and Drone Photogrammetry, known as Drone-SfM in computer vision (Nassar and Jung 2012; Siebert and Teizer 2014; Haur et al. 2018).

These non-contact surveying methods help construction professionals to obtain a 3D point cloud and generate a construction site's DTM (Digital Terrain Model). They can scan targets in the distance range of 100 m by setting up several ground stations, and up to 5 km using a pre-planned drone path. Their surveying capacity is more than 10 million raw points, while those points are recorded without manually selecting target points and excluding the non-target points (Nex and Remondino 2014). In addition, those remote surveying procedures avoid interfering with other construction operations and also reduce the surveying time by scanning multi-points at the same time.

3.4.1.3 Terrestrial Laser Scanning

The Terrestrial Laser Scanning (or ground-based 3D laser scanning) has been adopted in construction surveying for several years. It needs to be set up on a tripod at a fixed location in front of the target object, and the time of flight (TOF) method is used to determine distances from the scanner to targets, with a high speed of 10,000 ~100,000 points per second (Du and Teng 2007). In engineering practices, multi-stations laser scanning and vehicle-based laser scanning solve the coverage limitations.

Although the drone-borne LiDAR system had been used for 3D habitat mapping in forest ecosystems (Guo et al. 2017), there are no published articles using the drone-borne LiDAR in COENG and AUTCON (see Table 3) to replace the ground-based or vehicle-based laser scanning. It is because the small-sized LiDAR device still needs a powerful UAV to carry, such as the DJI Matrice series, which is impossible for a small drone. Additionally, the investment (see Table 5) is another issue; the drone-borne LiDAR systems are quoted for $ 60, 000 to $ 280, 000, which is too expensive to be adopted in construction site surveying before its price drops down to a reasonable number.

Among the reviewed literatures, there are two interesting applications of ground-based 3D laser scanning:

1. Using ground-based 3D laser scanning as the baseline for evaluating drone photogrammetry. Takahashi et al. (2017), Maghiar et al. (2018) and Moon et al. (2019) designed experiments to compare drone photogrammetry with ground-based 3D laser scanning. The results confirmed that drone photogrammetry is precise enough for use in excavation operations.

2. Merging the point clouds from ground-based 3D laser scanning and drone photogrammetry to create an integrated point cloud (Kwon et al. 2017; Li, D. and Lu 2018; Chen et al. 2018; Moon et al. 2019). Kwon et al. (2017) tested the hybrid scanning method, which merged ground-based 3D laser scanning with drone photogrammetry to generate a 3D point cloud for an under-construction bridge. The laser scanner scanned the sides of the target in multi-stations; the drone's camera scanned the top view of the target, where is hard to be reached by the ground laser scanner. Additionally, the side views can also be supplement with ground-camera images and vehicle-mounted camera images (Barazzetti et al. 2010; Sung and Kim 2016; Inzerillo et al. 2018; Aguilar et al. 2019; Park et al. 2019)

Table 5 Comparisons of Drone-based Surveying System

System	*Price*	*Drone Platform*	*Sensors*	*Calculated Spatial Resolution*	*Manufactory Accuracy*
Matrice 200 and 210 LiDAR	$ 60,000 for education	DJI M200	Velodyne PUCK-LITE	@10m: 0.4°*3.14rad/180°*10m = 6.9cm or 2.7 inch @5m: 0.4°*3.14rad/180°*5m = 3.49cm or 2.7 inch	@50m 4.6 cm+/-
Snoopy-V-Series	$ 280,000 for education	DJI M600	Unknow Laser Sensor		@50m 3.2 cm+/-
DJI Phantom 4 Pro V2.0	$1,799 for retail	DJI Phantom 4 Pro	8.8 mm*13.2 mm COMS, Focal length =8.8 mm	@20m: GSD=0.54 cm/px @40 m: GSD=1.08 cm/px	-
DJI Inspire 2 (X4S)	$4,249 for retail	DJI Inspire 2	8.8 mm*13.2 mm COMS, Focal length =8.8 mm	@40 m: GSD=1.08 cm/px	-
DJI Inspire 2 (X5S-15mm)	$10,309 for retail	DJI Inspire 2	13 mm*17.3 mm COMS, Focal length =15 mm	@40 m: GSD=0.88 cm/px	-
DJI Inspire 2 (X5S-45mm)	$10,309 for retail	DJI Inspire 2	13 mm*17.3 mm COMS, Focal length =45 mm	@40 m: GSD=0.29 cm/px	-

Based on Table 2, it is clear that drone photogrammetry has the flexible range in object 3D-reconstruction, especially in the construction field. It is not only limited to construction site surveying, but it also has been used to perform building 3D-reconstruction (Aguilar et al. 2019; Chen et al. 2018), create 3D texture models of construction equipment (Kim, H. and Kim 2018) and pavement surface mesh models as well (Inzerillo et al. 2018). Table 5 compares the spatial resolution and price between drone photogrammetry and drone-borne LiDAR. Based on that comparison, the drone photogrammetry is the most reasonable option for excavation operations. However, determination of elevations of a construction site in real-time during the excavating still is a challenge, because drone photogrammetry needs at least one workday to transform the raw images to a coarse 3D model for measuring with commercial photogrammetry software, Agisoft PhotoScan (Haur et al. 2018). The advantages and shortfalls in drone photogrammetry will be discussed later.

3.4.2 Image-based 3D-reconstruction Methods

Table 6 categorizes the image-based 3D-reconstruction methods by the type of sensor and the method of distance measurement between the sensor and targets.

Table 6 Image-based Methods to Acquire Targets' 3D-geometrical Data

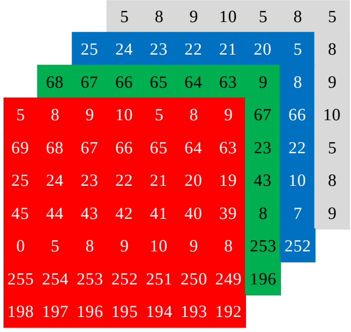
Sensors	*principles*	*Scene Scale*	*Raw Results*	*Measuring Products*	*Applications*
SAR	Synthetic-aperture radar	Large Landscape	SAR Images	DEM/DTM	Kirscht and Rinke 1998; Nico et al. 2005; Huang, Q., et al. 2017
IR Distance Sensor	TOF	Small/Indoor	RBG-D Images	Depth Map	Holz et al. 2011; Huang A. S. et al. 2017
Stereo Cameras	Triangulation	Small/Indoor	RGB Image Pairs	Distance /Depth Map	Sung and Kim 2016; Sophian et al. 2017;
Single Camera	Photogrammetry / SfM	Small /Large	RGB Images	Points Cloud / Geometry Model/ DEM	Westoby et al. 2012; Siebert and Teizer 2014

3.4.2.1 Synthetic-aperture Radar (SAR)

Synthetic-aperture radar (SAR) is a plane-mounted system for scanning large-scale landscapes. The scanning results are SAR images, and the modeling result is a DEM (digital elevation model) (Kirscht and Rinke 1998) or a DTM (Nico et al. 2005) depending on the application environment. In infrastructure construction, the ground-based SAR system (GB-SAR) has been used in dam deformation monitoring (Huang, Q., et al. 2017) and landslide monitoring (Noferini et al. 2007). However, SAR has the same issue with LiDAR, they are unable to be carried by a MAV/drone.

3.4.2.2 Infrared Radiation (IR) Distance Sensor

Infrared radiation (IR) distance sensor, also known as RGB-Depth camera, is a device that uses TOF to determine distances between the sensor and target objects. The scanning results are four-channel RGB-D images (see Figure 13). The resolution of the distance sensor is smaller than the color sensor, such as 320×240 pixels depth resolution for the old Kinect V1, and 512× 424 pixels for the latest Kinect V2. The accuracy of depth measurement gets worse when the distance increases; 1.5 ~ 3m is an accepted distance range for measurement (Litomisky 2012). In most cases, this sensor is only used indoors, such as indoor scenes 3D-reconstruction (Holz et al. 2011; Huang, A.S. 2017) and body activities capturing (Guo 2018).



			5	8	9	10	5	8	5
		25	24	23	22	21	20	5	8
	68	67	66	65	64	63	9	8	9
5	8	9	10	5	8	9	67	66	10
69	68	67	66	65	64	63	23	22	5
25	24	23	22	21	20	19	43	10	8
45	44	43	42	41	40	39	8	7	9
0	5	8	9	10	9	8	253	252	
255	254	253	252	251	250	249	196		
198	197	196	195	194	193	192			

Figure 13 Four-channel RGB-D matrix, red, green, blue pixel, and gray depth value

3.4.2.3 Stereo Camera

Stereo camera system is the most common close-range photogrammetry device for measuring distance between cameras and target objects. It is made up of two cameras with the same specifications and parameters, the only difference between those two cameras is the baseline T in spatial position. The scanning result is an image pair (see Figure 14); the $left\ image$ (x_l, y_l) and $right\ image$ (x_r, y_r) are in the same plane and perpendicular to each cameras' principal ray. The triangulation method (see Eq. 2) is used to calculate distances between the targets and cameras, because the distances in front of the cameras have a negative relationship with the $Disparity = x_l^p - x_r^p$. With this relationship, it is feasible to generate a small sense depth-map from the stereo camera image pair by traversing all common pixels of those two images (Sung and Kim 2016; Sophian et al. 2017).

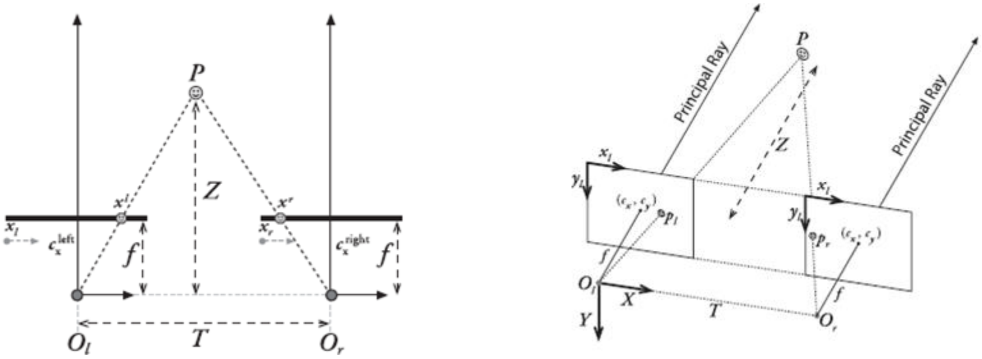


Figure 14 Stereo camera model with triangulation

$$\left.\begin{array}{l}\frac{x_l^p - c_x}{f} = \frac{T_l}{Z} \\ \frac{c_x - x_r^p}{f} = \frac{T_r}{Z}\end{array}\right\} \Rightarrow (x_l^p - c_x) + (c_x - x_r^p) = \frac{T_l + T_l}{Z} f \\ T_l + T_l = T \Bigg\} \Rightarrow \begin{array}{c} Disparity = x_l^p - x_r^p = \frac{T}{Z} f \Leftrightarrow Z = \frac{T}{x_l^p - x_r^p} f \\ (x_l^p - x_r^p \neq 0) \end{array} \qquad \textbf{Eq. 2}$$

Where, P is a target in front of cameras O_l and O_r; $p_l(x_l^p, y_l^p)$ and $p_r(x_r^p, y_r^p)$ are image points of P;
(c_x, c_y) is image point of cameras, named as principal point, its ideally be the center of image plane.
T is the distance between cameras, named as Baseline; f is the focal length of cameras;
Z is the distance between P and Cameras

3.4.2.4 Single Camera

Structure from Monition (SfM), where 3D structure can be resolved from unordered images (Ullman 1979; Westoby et al. 2012), often used in drone photogrammetry, is replacing traditional aerial photogrammetry /aerial triangulation to generate the DTM. Both methods use cameras to capture multiple overlapping images, then match the same target objects in those adjacent image pairs to generate DTM. Figure 15 shows a three-frame aerial triangulation model, where the camera moves from left to right without rotation; Figure 16 shows a two-frame SfM model, where the camera moves from C_0 to C_1 with a translation ($t = [x\ y\ z]^T$) and a rotation (R). That is more complex than the stereo camera model (see Figure 14), which only has the x-axis translation between the two cameras. So, a key task in SfM is to find out the external position and orientation parameter $[t \;\; R]$ to align the sequence camera stations' coordinates to the initial station's coordinate. Similar to the stereo camera model, the matched point pairs in those two images can be found in the epipolar line pairs, while the local feature matching methods — such as SIFT, SURF, which will be discussed later— have replaced the epipolar line method to extract keypoint pairs from image pairs in SfM. Pix4D, Autodesk ReCap Pro, Agisoft PhotoScan and DroneDeploy are commercial softwares of drone photogrammetry, and OpenSfM is an opensource SfM library written in Python.

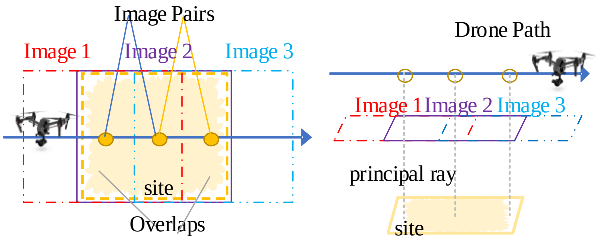


Figure 15 Overlapping ortho-image series

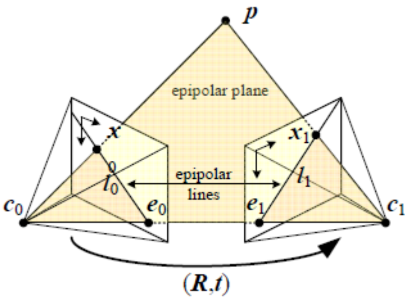


Figure 16 Two-frame structure from motion and epipolar geometry

Compared with SAR, IR and stereo camera, the single digital camera with drone photogrammetry/ drone-SfM has several unbeatable advantages for scanning excavation objects on construction sites:

1. A small-sized digital camera could be mounted to a drone with a 3-axis gimbal, so that the camera can move over a construction site without interfering with other construction operations.
2. The digital camera has a larger spatial resolution than RGB-Depth cameras, so a large single RGB image can contain more objects than an RGB-D image.
3. Fewer images are required to cover an entire construction site which helps to reduce the error caused by image matching and reduce the processing time as well (Schenk 1999; Kaehler and Bradski 2016).

3.5 Drone Ortho-imaging Related 3D-reconstruction Research

3.5.1 Requirements in Ortho-imaging

An object that needs to be 3D-reconstructed by drone photogrammetry /drone-SfM using overlapping ortho-images should have a rough surface, a relatively small slope, and should be captured in a bright environment. This is because:

1. Only objects with a rough enough surface can be recorded as complicated textured images, while objects that lack a sufficient quantity of unique detectable features — such as transparent materials (i.e. windows), reflective materials (i.e. windows, mirrors, glossy paint, water surface, snows), or the object with uniform surfaces with little variation— are unable to generate sufficient feature keypoints due to low contrast (Lowe 2004, Solem 2012,Westoby et

al. 2012). Applying SfM in 3D-reconstructing uniform texture objects, such as vehicles, requires attaching marks to target objects' surfaces (Erickson et al. 2013).

2. The accuracy of drone photogrammetry/drone-SfM using ortho-images is affected by the object's surface slope. It has a low error rate for relatively flat ground surfaces (Haur et al. 2018), while its error rate increases as the ground slope increases. Zhao and Lin (2016) concluded that the error rate has a positive relation to the slope in the range of 55° to 90°. That means it is unable to handle the steep, or near vertical topography (Westoby et al. 2012; Zhao and Lin 2016).
3. Additionally, the brightness of the environment also impacts drone photogrammetry/drone-SfM method's accuracy. Westoby et al. (2012) stated that the SfM method has a questionable accuracy in dense vegetation areas. Zhao and Lin (2016) found that the accuracy of the SfM method has a negative relationship with the hillshade value in range of 0 to 170.

Fortunately, most construction sites meet those requirements. After removing the vegetation and topsoil, the soil, rocks or mixture materials are exposed as seen in Figure 17, which is a richly-textured surface. The uniform brightness ortho-image requirement can be satisfied by using a drone to take the images on a sunny day. In general, the SfM works for vertical surfaces, such as the building 3D-reconstruction cases (Aguilar et al. 2019; Chen et al. 2018), because the camera can angle toward any object; but, it is better to limit drone to a narrow flight region to avoid interruption with excavation operations for safety reasons. Therefore, the main challenge of applying ortho-imaging based 3D-reconstruction for construction site elevation determination is that the slope of the ground surface might be changed to 90° after excavated, such as on the side wall of a pit.

Figure 17 Example ortho-image of construction site's surface

3.5.2 Ortho-Image Matching Methods

Figure 18 shows the general procedures of 3D-reconstruction with image feature keypoint-based drone photogrammetry/drone-SfM method. For 3D-reconstrction of a construction site, the overlapping ortho-image series (Figure 15) can be either captured in a sequence or extracted from a video, where the adjacent ortho-images require a minimum of 70% and 40% overlap in longitudinal and traversal coverage, respectively (Siebert and Teizer 2014).A previous experiment (Takahashi et al. 2017) tested the higher longitudinal overlapping ratios from 80% to 90%, but it did not lead to a significant enhancement in the measurement accuracy with the increasing overlapping ratio. However, the extra unnecessary images require additional time to process image matching. Haur et al. (2018) reported the required time to estimate on-site soil volume by drone photogrammetry is one workday, as the point cloud is produced using Agisoft PhotoScan, the geometry model is created using Autodesk ReCap with the point cloud, and the soil volume is estimated with Autodesk Civil 3D.

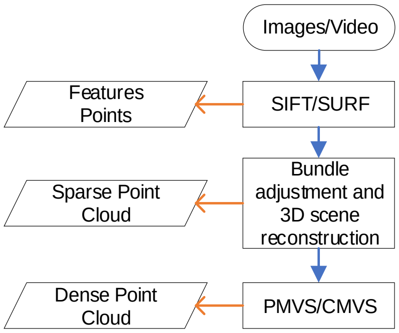


Figure 18 Keypoint-based SfM workflow

The next step is extracting and matching keypoints from adjacent ortho-image pairs, which is called local feature detection and description in computer vision (Kaehler and Bradski 2016). The most common and widely used image local features are: the Scale-invariant Feature Transform (SIFT) – a famous feature detection algorithm in computer vision to detect and describe local features in images, which was patented in Canada by the University of British Columbia and published by David Lowe (Lowe 1990; Lowe 2004) – and the Speeded Up Robust Features (SURF) – another patented local feature detector and descriptor, which is several times faster than SIFT and more robust against different image transformations than SIFT (Bay et al. 2008). Figure 19 is a SIFT example, the green dots are SIFT keypoints, and the matched keypoint pairs were linked with green lines. Although those green keypoints were matched and located correctly, the noticeable weaknesses of SIFT still exist: a) sparse keypoints are selected with rules, and the keypoints are randomly distributed in an image, b) the number of detected keypoints in an image is less than the number of pixels in the image, and c) the number of matched keypoints in the image pair is much less than the number of detected keypoints. Similarly, the result by SURF may also not be dense enough to represent a construction site's geometrical features, because most candidate points are excluded by a number of criteria, like low contrast and points on edges (Solem 2012). This is why the SfM method does not work well with steeply sloped ground, as the points on edges have been removed. After getting the sparse point clouds, the Patch-based Multi-view Stereo (PMVS) / Clustering Views for Multi-view Stereo (CMVS) (Furukawa and Ponce 2010) will be used to generate dense point clouds.

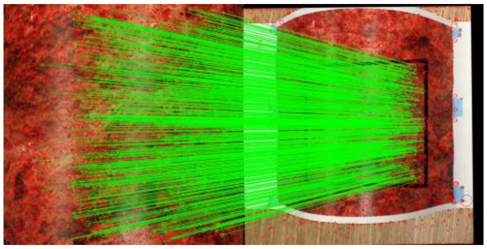


Figure 19 Example of SIFT keypoints matching

The Normalized Cross Correlation (NCC) matching method is used to determine the relations between a reference patch and a target patch (Lewis 1995), which are not limited to grayscale values or gradient values (Solem 2012). It might be a suitable approach to match the customized pixel pairs which are dense and uniformly spread in the overlaps of adjacent image pairs, since that has been verified in PMVS (Furukawa and Ponce 2010). This is because reference pixels can be manually selected in the preferred styles such as implementing a densely packed and uniformly spaced pixel grid, and the best matched corresponding target pixels will be determined from the candidate target pixels. The NCC method calculates the correlation between two equally sized image patches $I_{x,y\in w}(x,y)$ and $I'_{x,y\in w}(x',y')$ (Kaehler and Bradski 2016), where the image patch is a rectangular portion (W) which is centered around the interest pixel with size $(2N+1)\times(2N+1)$. Its formula is defined as Eq. 3, by subtracting the mean $\bar{I}$, $\bar{I}'$ and scaling with the standard deviation $\sqrt{\sum_{x,y\in W}[I(x,y)-\bar{I}]^2}$, $\sqrt{\sum_{x,y\in W}[I'(x,y)-\bar{I'}]^2}$, the NCC method becomes robust to image brightness changes (Solem 2012). However, compared to SIFT/SURF, the patch-based NCC is worse at image scaling, rotation and projection transformations, because the rectangular patch is not invariant to scale or rotation, and the patch size affects the matching results as well (Solem 2012).

$$NCC = \frac{\sum [I(x,y) - \bar{I}][I'(x',y') - \bar{I}']}{\sqrt{\sum [I(x,y) - \bar{I}]^2}\sqrt{\sum [I'(x',y') - \bar{I}']^2}} \qquad \textbf{Eq. 3}$$

Where, $I_{x,y \in W}(x,y)$: the patch for the reference image pixel (x,y) ; $I'_{x',y' \in W'}(x',y')$: the patch for the target image pixel (x’, y’);
$\bar{I} = \frac{1}{N}\sum_{x,y \in W} I(x,y)$: mean of patch $I_{x,y \in W}$; $\bar{I'} = \frac{1}{N}\sum_{x',y' \in W'} I'(x',y')$: mean of patch $I'_{x',y' \in W'}$;
W and W' are rectangular patches with size (2R+1)×(2R+1), which are centered around the reference pixel (x,y) or candidate target pixel (x’,y’).

3.5.3 Other Related Methods and Works

3.5.3.1 Multi-scale Image-based Methods

In traditional drone photogrammetry the overlapping images are captured at a constant altitude (see Figure 15). Then these images have the same constant scene scale and spatial resolution, and the objects in the overlapping parts have the same size, because these images meet at the pinhole camera model (see Figure 6). In contrast, Daftry et al. (2015) proposed a novel SfM framework of using multi-scale images — capturing in various depth (distance) from a building — to enhance the accuracy of the facade 3D-reconstruction. Additionally, Matthies et al. (1997, 2007), Li, R., et al. (2002),Xiong et al. (2005) and Meng et al. (2013) continuously applied descent images to determine the topography of landing terrain for space aircraft. Those descent images were captured in the landing path at different times and at different altitudes, so those descent images are multi-scale images of the same terrain surface. That image-based 3D-reconstruction result is suitable for choosing a safe landing area in planetary landing exploration tasks (Meng et al. 2013). Thus, capturing images at different altitudes may be an approach to enhance the image-based 3D-reconstruction of construction sites.

Considering the success of descent image-based research, Figure 20 shows a modified, faster drone photogrammetry for construction site elevation determination, which uses an ortho-image to cover the entire building construction site (like image 1) at the special altitude, $Z = f \cdot H_{site}/h_{image}$. Then another image (like image 2) is captured above this altitude which also covers this site. Meng et al. (2013) discussed that the low camera’s altitude should be half of the high camera’s altitude, so that the disparity of neighbor pixels around the image center could be detected in 0.5-pixel level. If this multi-scale image-based method can be automatically implemented using a computer system and program, then the required number of images used in the ortho-image based 3D-reconstruction method is minimized to two.

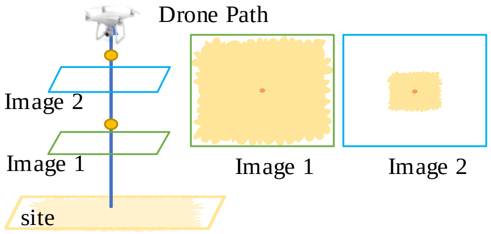


Figure 20 Modified drone path

Additionally, the multiple image-based construction site elevations determination can be improved in regard to model alignment. Currently, point cloud or the TIN mesh model generated from drone photogrammetry/drone-SfM is a scale model, which needs at least 3 ground control points (GCPs) to align the scale model to the real-world coordinate (Westoby et al. 2012). Furthermore, the modeling error should less than 50 millimeters in elevation coordinate compared with the real-world coordinates (Takahashi et al. 2017). Then the developed image-based 3D-reconstruction method can be adapted in determing the construction site elevations.

3.5.3.2 Deep learning-based Method

Recovery the 3D geometrical data from a single-image without reference information is an ill-posed problem (Van den Heuvel 1998; Hassner and Basri 2006; Saxena et al. 2008). The methods discussed in the previous section either used the equipment's physical properties or the camera model and epipolar geometry as the reference information. However, previous research has shown the feasibility of using deep learning methods to recover the relative depth information for each pixel of an image of indoor scenes (Eigen et al 2014; Liu et al. 2015; Laina et al. 2016), outdoor scenes (Chen et al. 2016; Li and Snavely 2018) and scenes from automatic driving applications (Garg et al. 2016). In addition, convolutional neural networks (CNNs) have been verified as effective and reliable in micro-scale scenes, such as estimating the surface height map from a single image of a foam mat and mouse pad (Zhou et al. 2017).

However, the challenge of training a deep learning model is acquiring a dataset. Previous research also discussed the approach for creating enough data for training the CNN model. In the automatic drive application, the CNN model is trained for determining objects' distances from a single forward-facing view of a car; the dataset, depth and front-view image pair are created by a stereo camera system that is installed on the front of the car (Garg et al. 2016). Another interesting approach to create a labeled dataset is using artificial images, such as generating different view images from a photogrammetry-based concrete mixer truck 3D-model and using these images to train a construction equipment object detector (Kim and Kim 2018). Thus, to guarantee the accuracy of CNN-based depth recovery or image surface height estimation, the multiple image-based 3D-reconstruction methods still are important for acquiring datasets.

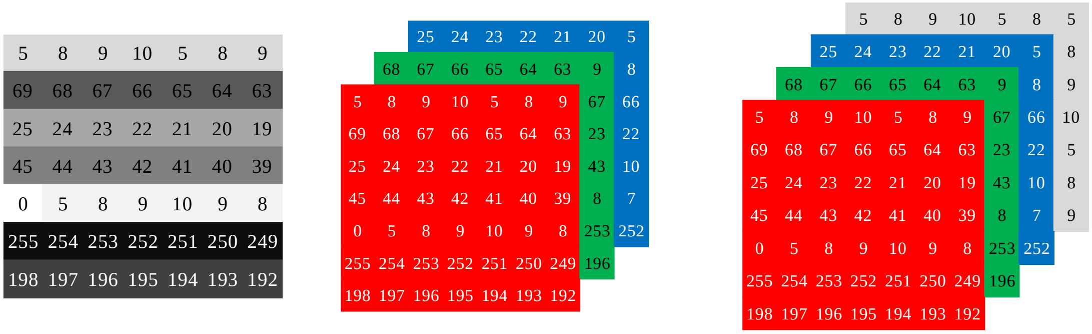
Therefore, for the construction industry, using the advanced artificial intelligence (AI) technologies, such as deep learning to automatically determine elevations directly from an image of the construction site, is an interesting research topic and meaningful challenge. Once overcome, the real-time 3D-reconstruction of a construction site becomes possible, and then the degree of automation of the excavation operations will be significantly improved (Seo et al. 2011).

3.6 Image Processing and Computer Vision with Deep Learning

3.6.1 Image Formation, Transformation and Convolution

3.6.1.1 Image Formation

Currently, a digital image is easily acquired by various cameras and other image sensors, especially by mobile phones and home digital cameras in our daily life; a digital image is used conveniently to show and share with smart phones and computers. Behind those, the most essential thing is that a digital image is formatted in matrices (see Figure 21). This approach is sampling an image at rectangular grids' centers; the color, or intensity, at each of these center points is converted into a numerical value; apart from the color / intensity, everything else is discarded when storing the image in a computer (Hearn et al. 2004; Szeliski 2010). Thus, the computer can read and write a gray digital image as one layer (channel) two-dimensional matrix of gray pixel values. Similarly, the RGB image and the RGB-D image, which have been discussed in previous sections, are able to be read and written as a three-layer (channel) two-dimensional matrix and a four-layer(channel) two-dimensional matrix separately. Therefore, an image processing problem is a kind of matrix math problem.



5	8	9	10	5	8	9
69	68	67	66	65	64	63
25	24	23	22	21	20	19
45	44	43	42	41	40	39
0	5	8	9	10	9	8
255	254	253	252	251	250	249
198	197	196	195	194	193	192

A. A Matrix of Gray Pixel Values

B. Three-layers RGB Matrix. Each Layer is a Two-dimensional Matrix of Red, Green or Blue Pixel Values

C. Four-layers RGB-D Matrix. Each Layer is a Two-dimensional Matrix of Red, Green or Blue Pixel Values, and Gray Depth Value.

Figure 21 Multi-layers image matrix formation

In this matrix formation, each grid is called a pixel, and the numerical value is called the pixel value. Pixels can be read and written by the pixel coordinate – row index and column index of the matrix. In the OpenCV-Python, the row index increases from left to right and the column index increases from top to bottom of the image plane (see Figure 14). For example, the pixel value ‘22’ (in third row, forth column) of the gray image I in Figure 21-A, can be read by value $= I[2,3]$ or written as $I[2,3] = 22$, because the row and column index start with 0 in Python3 (Solem 2012).

3.6.1.2 Image Transformation

Figure 22 shows the basic set of 2D planar transformations. With this pixel coordinate (in matrix formation), the image 2D transformations, such as rotation or translation, could be accomplished by matrix multiplications (Hearn et al. 2004; Szeliski 2010). In detail: a) 2D points, pixel coordinates in an image can be denoted using a pair of values, $x = [x\ y]^T$, or using the homogeneous coordinates as $x = [x\ y\ 1]^T$, then an image translation and rotation can be done with Eq. 4.

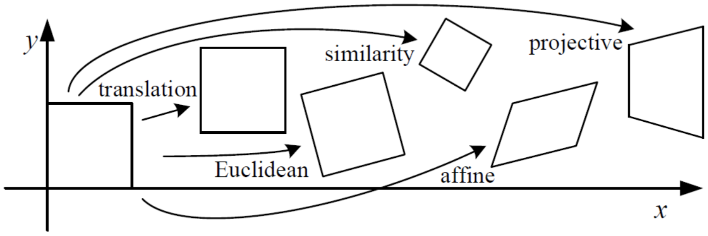


Figure 22 2D planar transformation

$$translation: \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}_{image@H'} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}_{image@H}$$

$$rotation: \begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix}_{image@H''} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}_{image@H'}$$

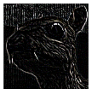
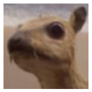
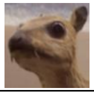
Eq. 4

Where, $[x \quad y \quad 1]^T$, $[x' \quad y' \quad 1]^T$, $[x'' \quad y'' \quad 1]^T$is the homogeneous coordinate; t_x , t_y are the distances translated; θ is the degree rotated in anticlockwise fashion.

3.6.1.3 Image Convolution

A kernel is a small convolution matrix (see Table 7), which is used for smoothing (blurring) , sharpening, edge detection, and more image processing operations. Those image processing operations are accomplished by doing a convolution between a kernel and an image (Kaehler and Bradski 2016). The general expression of convolution is Eq. 5, and explained in Figure 23.

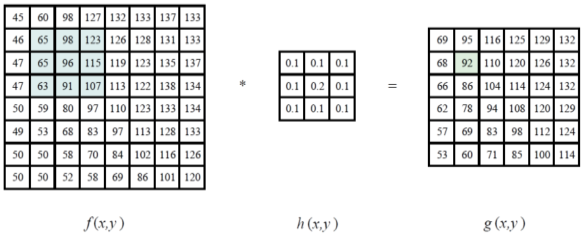
Table 7 Common Convolution Kernel

Operation	Kernel	Image result g(x,y)
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur 3 × 3 (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

$$g = f * h, \text{ or } g(x,y) = \sum_{k,l} f(i-k, j-l)h(k,l) = \sum_{k,l} f(k,l)h(i-k, j-l)$$

Eq. 5

Where, $g(x,y)$ is the filtered image;$f(x,y)$ is the original image;$h(k,l)$ is the filter kernel.



45	60	98	127	132	133	137	133
46	65	98	123	126	128	131	133
47	65	96	115	119	123	135	137
47	63	91	107	113	122	138	134
50	59	80	97	110	123	133	134
49	53	68	83	97	113	128	133
50	50	58	70	84	102	116	126
50	50	52	58	69	86	101	120

$f(x,y)$

*

0.1	0.1	0.1
0.1	0.2	0.1
0.1	0.1	0.1

$h(x,y)$

=

69	95	116	125	129	132
68	92	110	120	126	132
66	86	104	114	124	132
62	78	94	108	120	129
57	69	83	98	112	124
53	60	71	85	100	114

$g(x,y)$

Where, the original image on the left is filtered (convolved) with the filter kernel in the middle to yield the filtered image on the right; the light blue pixels indicate the source neighborhood for the light green destination pixel.

Figure 23 Neighborhood filtering (convolution)

3.6.1.4 Image Pooling

Image Pooling or down-sampling is the operation to extract image features after convolution. It is similar to image scale transformation. Figure 24 is the example of max pooling, which uses the max pixel value to stand for the feature of the 4 pixels. Another common pooling is mean pooling, which uses the mean of the 4 pixels to stand for their feature.

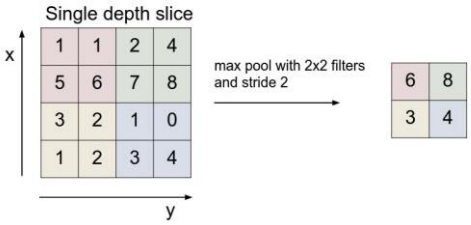


Figure 24 Max pooling with a 2x2 filter and stride = 2

3.6.2 Image Feature Detection and Matching

3.6.2.1 Pixel Feature Matching

Image features are defined base on their applications, such as edges, corner/interest points and blobs/regions of interest points. The basic feature is a pixel’s color or intensity. Assuming all pixels’ colors / intensities are unique in two images, then the pixels’ colors / intensities can be used to match the corresponding objects in those two images, as the same objects have the same pixel value. However, directly comparing the pixel value is an ineffective method for matching pixel pairs in two adjacent images captured by a drone, because the environment conditions effect the images’ brightness.

Previous sections have described that the normalized cross-correlation matching method is a suitable approach to match the image pairs. The correlation matching methods calculate the correlation between two equally sized image patches $I_{x,y\in w}(x,y)$ and $I'_{x,y\in w}(x',y')$ rather than two pixels only (Solem 2012; Kaehler and Bradski 2016).

3.6.2.2 Keypoint Matching and Homography

In image-based 3D-reconstruction, matching keypoint pairs is the most essential processing step. The SIFT and SURF detections and matching results are not dense enough to represent a construction site, because most points are dismissed based on several criteria, like low contrast and points on edges (Solem 2012). That is why drone photogrammetry/drone-SfM using ortho-images does not work well in big slope surfaces, as the points on edges have been removed. Figure 25 shows an example of template matching using the SIFT keypoint and homography method. The green lines show the same points in the images captured in different positions. The white outline in the right image indicates the edges of the template. It uses matched SIFT keypoints to calculate a 3×3 perspective transformation matrix. Then, using the matrix to transform the four corners of the template (the left image) to its corresponding four points in the right image, template detection is completed (Kaehler and Bradski 2016).

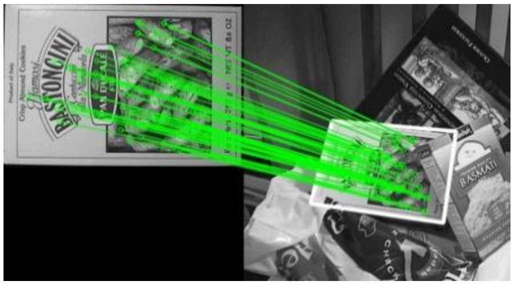


Figure 25 Template detection with keypoints matching and homography

3.6.2.3 Other Image Features

Image gradients are directional changes in intensities/colors of the image. Gradients in the x-axis and y-axis directions are computed in Eq. 6. Gradients extract feature information from images, such as edge detection (Solem 2012). It also can be used in feature and texture matching for images with different brightness or captured with different cameras. That is another approach to solve the brightness issue.

Histogram of oriented gradients (HOG) is a feature descriptor used in object detection, which is used in particular suites for human detection in images (Dalal and Triggs 2005). Memarzadeh et al. (2013) detected the construction equipment and workers from a construction site video stream by HOG plus colors features. Kim, H. and Kim, H. (2018) also developed a concrete truck detector with HOG feature and SVM (support vector machine) model.

$$\nabla f = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \qquad \textbf{Eq. 6}$$

Where, $\frac{\partial f}{\partial x}$ is the derivative with respect to x (gradient in the x direction);
$\frac{\partial f}{\partial y}$ is the derivative with respect to y (gradient in the y direction).

3.6.3 Object Detection, Image Classification and Image Segmentation

In general, object detection includes the task of object classification and object localization. The results usually are marked with different-colored rectangular boxes for identifying different objects' categories and their locations in the original image. Image classification task only needs to identify the main object in the image or the specific image region. Image segmentation is more detail than object detection and can get the result of a same sized label-image, which uses several colors to draw the different objects' categories in each pixel instead of the texture in the original image.

3.6.3.1 Limitations

Currently, detecting vegetation from a photogrammetric point cloud based on vegetation indexes and points' spatial geometrical relations (Anders et al. 2019; Cunliffe et al. 2016) has limitations because it only allows a ground point subset and non-ground (vegetation) point subset to be classified. In addition, the vegetation index methods are effective in identifying green and yellow vegetations, but ineffective with other colors such as the withered vegetations and shaded vegetations. That also results in the issue of treating other green or yellow texture objects as the vegetations.

Previous research results have shown the feasibility of deep learning methods in objects detection using image (Schneider et al. 2018), video (Kang et al. 2018), point cloud (Engelcke et al. 2017), and image segmentation (Noh et al. 2015; Badrinarayanan et al. 2017). The limitation of the current deep learning-based methods is that they use low-resolution images for training the deep learning-based object detector, which resize the ImageNet (Deng et al. 2009) down to as small as 256×256-pixel, while the high-resolution is limited to 800×1,000-pixel (Han et al. 2015). This issue is caused by the limitation of computer hardware. A directly exported image from a digital camera, such as the image captured using the

DJI Phantom 4 Pro V2.0, is as large as 3,648×4,864-pixel, which is extremely larger than the 256×256-pixel. Reducing the image size impacts the effectiveness of image segmentation because the number of pixels for representing each single object decreases as the image resize down. For example, if the image has been shrunk three times, an 8×8-pixel patch in the original image becomes a single pixel in the shrunken image.

3.6.3.2 Solutions

A potential approach to avoid resizing down the high-resolution image is that separately identifying each 8×8-pixel small-patch of the original image and assembling them to be the high-resolution result. The result of this approach is same as the result of image segmentation using the low-resolution image. In addition, the small image patch size is better for training a deep learning-based image classification model, where the image classification task only needs to identify the main object in the image. In detail, the main object is distinguished from the background; and, no matter what other objects are included in the background, this small image patch will be identified as the main object's category. Therefore, the possible procedures to segment a high-resolution image without resizing it may be to disassemble it to small-patches and recording the sequence ID, classify each small-patch with a pre-trained deep learning-based image classification model, assign the class-label to each small-patch, and assemble these small-patches to build the high-resolution results.

3.7 Literature Review Summary

This literature review mainly discussed the feasibilities, weaknesses, and research opportunities in drone technologies and image-based 3D-reconstruction methods for determing construction site elevations. The review was carried out using several steps. Firstly, this review evaluates the drone related publications in the *Journal of Construction Engineering and Management* and the *Automation in Construction*. The drone related research in the top ranked construction research publications has been rapidly increasing starting from 2015. From this comprehensive review and quantitative assessment, the following interesting points are outlined:

1. “DJI Phantom” series and “DJI Inspire” series drones are the most popular drones that have been adopted in the construction research projects.
2. An optical camera is the most reasonable sensor to acquire the RGB image for inspection applications.
3. The point clouds generated by photogrammetry / SfM methods have been benefiting drone applications in the construction research since 2016.
4. No published article mentioned the usage of the drone-borne LiDAR technology at this moment.

Secondly, this review compared the current construction surveying techniques and image-based 3D-reconstruction method by reviewing the relative literatures from “*Google Scholar*.” From this comparative review, the following points are summarized:

1. Determination of elevations for excavation operations is similar to 3D mapping and object 3D-reconstruction tasks, which have a complex procedure with data collection, data preprocessing and 3D modeling. Previous researchers applied the commercial drone photogrammetry software to generate the DTM for the construction sites, while future research is still needed to accelerate the process of image-based 3D-reconstruction method to generate the real-time as-built model.
2. 3D-reconstruction with specific equipment’s physical properties, such as LiDAR, or using the camera model and epipolar geometry could get state-of-the-art results, while the cost and time is not good enough for determing elevations on construction sites. Also, the robust algorithm for extracting the desired features and eliminating the noisy features from an image pair is still a challenge with the current image processing and computer vision methods.
3. Taking a single ortho-image over a construction site by a drone system, then using this single-image to estimate the construction site elevations could be a feasible approach with deep learning, which will reduce drone’s flying time and minimize risk of drone crash on a construction site.

Therefore, considering the success of descent image cases, acquiring an ortho-image pair by drone at a low altitude and a high altitude may be a possible approach for faster construction site elevation determination, where the low ortho-image is the overlap and the overlap parts have different scales. In Chapters 4 and 5, this goal was implemented by addressing three tasks: 1) to determine the distance from the ground surface to the drone, a modified triangulation model is required; 2) to get the most accurate dense corresponding matching results by NCC, the proposed image patch feature descriptors should be sensitive to the different scales in the ortho-image pair, and the patch size should be self-adjusting between small-patches for complex textured regions and large-patches for poorly textured regions; and 3) to rapidly and automatically compute distances, an innovative approach and algorithms need to be developed for generating matched pixel pairs in a dense pixel grid style while simultaneously determining the distances. After that, Chapter 6 discussed how to use a single ortho-image to determine geometrical data using the dataset created in Chapter 5.

CHAPTER 4: LOW-HIGH ORTHO-IMAGE PAIR-BASED ELEVATION DETERMINATION ALGORITHM DESIGN AND TESTING

4.1 Introduction

This chapter presents a modified stereo-vision triangulation method for construction site elevations determination, which uses a drone’s camera to capture a low-high ortho-image pair instead of a left-right ortho-image pair of a construction site. This low-high ortho-image pair triangulation method is designed to enlarge the baseline distance and increase the measurable depth range compared to the classic stereo-vision method. This proposed method focuses on 3D-reconstruction of the ground surface of a construction site and excludes the side surfaces of the attached objects, which makes it simpler than traditional drone photogrammetry. In detail, the low ortho-image, which covers the entirety or sometimes a large portion of the construction site, is captured at half the height of the high ortho-image (see Figure 26). Then the entire low ortho-image is contained in the overlap of the ortho-image pair. Additionally, if a construction site is larger than a single ortho-image frame, this proposed method can stitch its results from adjacent ortho-image pairs with a very narrow overlapping strip compared to the high overlapping ratio in traditional drone photogrammetry.

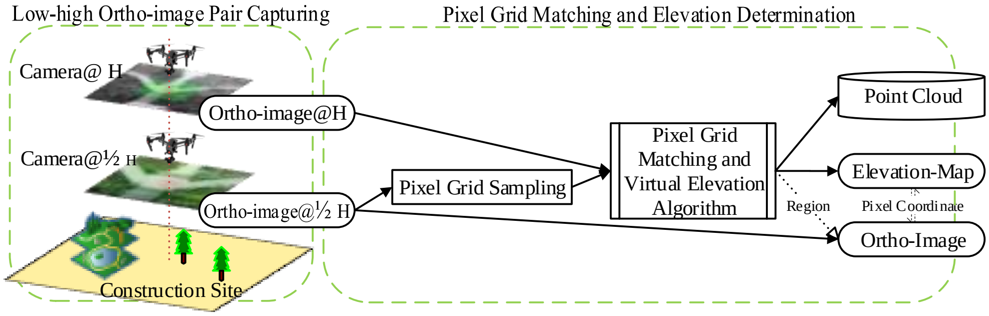


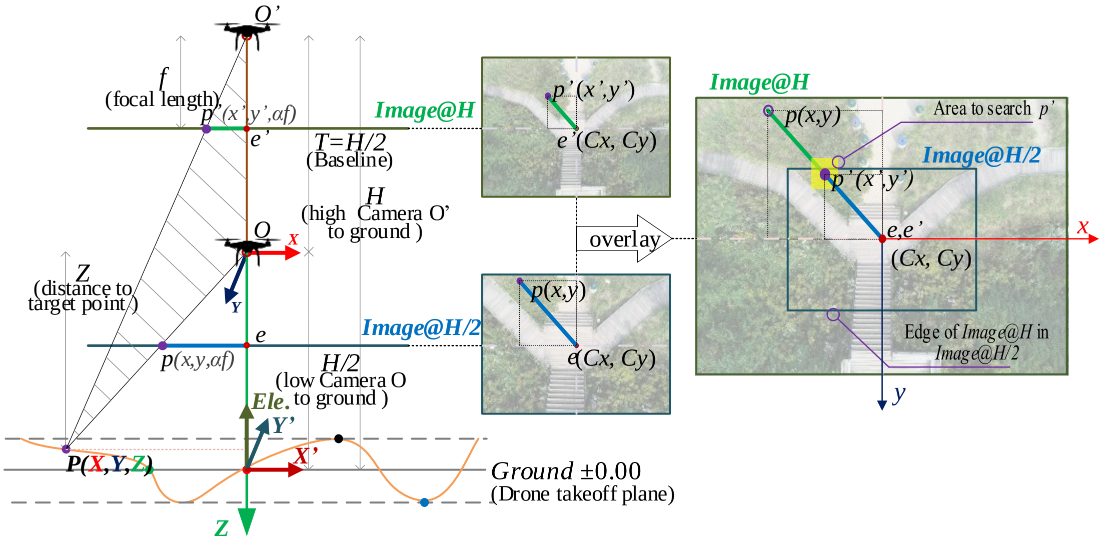
Figure 26 Workflows of low-high ortho-image pair-based method

The biggest challenge in this low-high ortho-image pair-based 3D-reconstruction method is to performing subpixel level image corresponding matching. Most of the current developed image matching methods are based on extracting and matching feature keypoints. The problem is that the extracted feature points are not evenly distributed throughout the image pair. In addition, the traditional image-based 3D-reconstrution methods, such as SfM, separate the image matching and geometrical data recovering into two different sequential processes, which wastes some computing resources. Therefore, the author developed a low-high ortho-image pair pixel matching and virtual elevation algorithm, which aims to generate matched pixel pairs in pixel grid, while simultaneously determine the elevation data based on the low-high ortho-image pair triangulation method and virtual elevation plane method. Additionally, testing was conducted on a construction site. Experimental results were evaluated and presented in this chapter to show the efficiency of the proposed method and the developed algorithms in a real construction site.

4.2 Low-high Ortho-image Pair-based 3D-reconstruction Method

4.2.1 Low-high Ortho-image Pair Triangulation Model

Using a single ortho-image (the size of a target object in an ortho-image has a negative relationship with the drone flight height) to determine geometrical data is an ill-posed problem; more reference information is needed such as additional ortho-images and camera positions (Eigen et al 2014). Th developed a low-high ortho-image pair triangulation model is shown in Figure 27, where the drone moves vertically along its camera's principal ray without any horizontal shift or rotation.



* express in Image Coordinate (x, y); p and p' are image point; e and e' are principal point

Figure 27 Low-high ortho-image pair triangulation model

The low ortho-image ($Image@H/2$) is captured at the low camera position O. This low altitude $H/2 = \alpha f \, H_{site}/h_{image}$ should be high enough to capture the entire construction site. The high ortho-image ($Image@H$) is captured at the high camera position O' with altitude H. These two ortho-images have the same principal point e with a 2:1 scaling relation, and the altitude differential between the low-high camera stations is the triangulation baseline $T = H/2$. The $World\ Coordinate(X, Y, Z)$ is set at $Camera\ O\ Coordinate$, where the *Z-axis* is aligned with the camera's principal ray downward to the ground. If image point pair $p(x, y, \alpha f)$ and $p'(x', y', \alpha f)$ are matched, then, the target point $P(X, Y, Z)$ can be calculated by Eq. 7. Especially when $x' = x/2, y' = y/2$, it has $Z = H/2$, which means the target point on $Ground \pm 0.00$. Therefore, a target point's elevation (relative to the $Ground \pm 0.00$) can be determined by $Elevation = H/2 - Z$.

4.2.2 Geometry Data Determination

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \frac{H}{2f_x}\frac{xx'}{x - x'} \\ \frac{H}{2f_y}\frac{yy'}{y - y'} \\ \frac{H}{2}\frac{x'}{x - x'}\ or\ \frac{H}{2}\frac{y'}{y - y'} \end{bmatrix} \qquad \textbf{Eq. 7}$$

Where, $P(X, Y, Z)$ (in Camera O coordinate) is a target point in the construction site;
$p(x, y, \alpha f)$ (in Camera O coordinate) is the image point of P on Image@H/2;
$p'(x', y', \alpha f)$ (in Camera O' coordinate) is the image point of P on Image@H;
f_x and f_y are focal length for image in x and y direction, ideally, it has $f_x = f_y = \alpha f$.
α is the factor to convert sensor size (mm) to image size (pixel)

Eq. 7 derivation processes are listed as follows:

In Figure 27, from *△Ope ≌ △OPZ* has,

$$\frac{X}{x} = \frac{Z}{f_x} \qquad (\text{Eq.}\ 7 - 1)$$

From △Op'e' ≌ △O'PZ has,

$$\frac{X}{x'} = \frac{Z + H/2}{f_x} \qquad (\text{Eq.}\ 7 - 2)$$

Minus (Eq. 7-1) from (Eq. 7-2) has,

$$X\left(\frac{1}{x'} - \frac{1}{x}\right) = \frac{H}{2}\frac{1}{f_x} \Rightarrow X = \frac{H}{2f_x}\frac{xx'}{x - x'} \qquad (\text{Eq.}\ 7 - 3 - 1)$$

Similarly,

$$Y = \frac{H}{2f_y}\frac{yy'}{y - y'} \qquad (\text{Eq.}\ 7 - 3 - 2)$$

From (Eq. 7-1) has,

$$Z = \frac{X}{x}f_x \qquad (\text{Eq.}\ 7 - 4 - 1)$$

Similarly,

$$Z = \frac{Y}{y}f_y \qquad (\text{Eq.}\ 7 - 4 - 2)$$

Take (Eq. 7-3-1) into (Eq. 7-4-1) has,

$$Z = \frac{H}{2}\frac{x'}{x - x'} \qquad (\text{Eq.}\ 7 - 5 - 1)$$

Similarly, take (Eq. 7-3-2) into (Eq. 7-4-2) has,

$$Z = \frac{H}{2}\frac{y'}{y-y'} \qquad \text{(Eq. 7 − 5 − 2)}$$

Thus, combine $[X, Y, Z]^T$ get the Eq. 7.

4.2.3 Method Discussion

The low-high ortho-image pair is easily acquired in a very short time without interfering with other construction operations. This is because the drones' small dimensions and their equipped automatic flight control system and sensors make them easily navigable in cluttered outdoor environments and allow them to hover at desired positions. The drone flight altitude data can be easily read directly from the remote controller, which has ± 0.00 set as the drone takeoff point. The *3-axis* gimbal helps the camera lens stably face the ground to capture ortho-images.

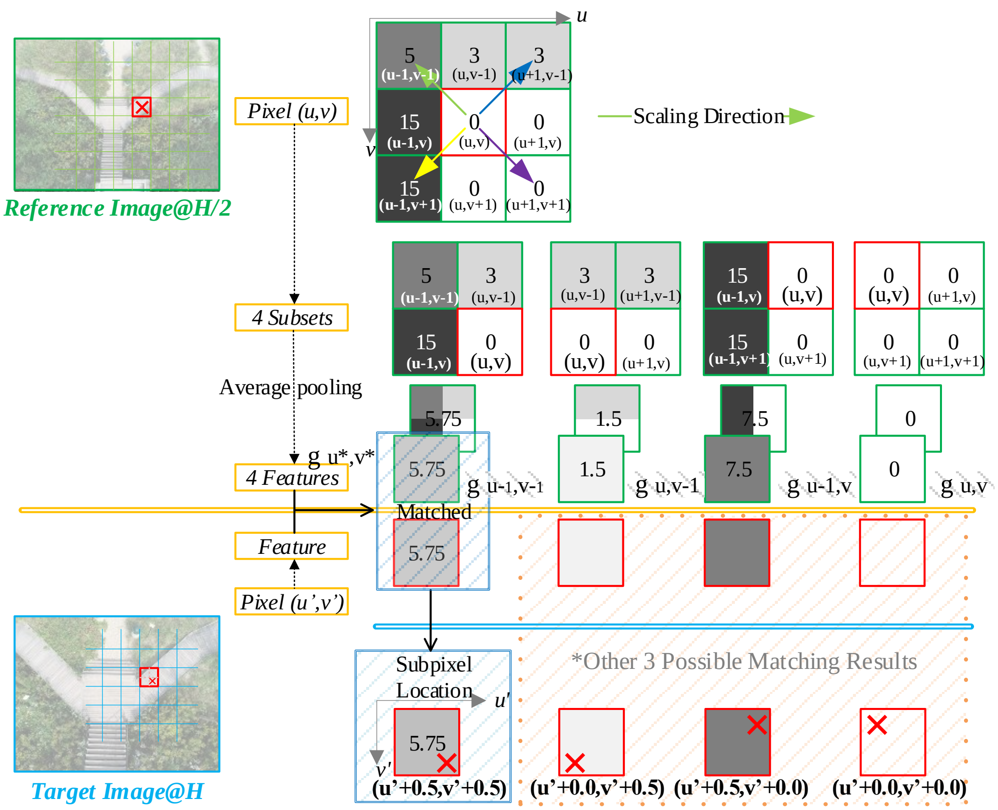
Since the impacts of wind and GPS signal interference cannot be eliminated, a slightly horizontal shift and rotation may occur during the drone's movement from the low position to the high position, which make the high ortho-image's principal point slightly different from that of the low ortho-image. Thus, it is necessary to align the low-high ortho-image pair to the same center with a slight image rotation and translation.

4.3 Pixel Grid Matching and Elevation Determination Algorithm Design

4.3.1 Low-high Ortho-image Pair Patch Feature Descriptors

4.3.1.1 Pixel-to-subpixel Matching and Locating

The reference image ($Image@H/2$) and target image ($Image@H$) have a 2:1 scale, necessitating the creation of separate patch feature descriptors for each. Figure 28 indicates the four scaling directions for generating the four features $g_{u^*,v^*}(u, v)$ for a reference pixel $p(u, v)$. The example shows that $g_{u-1,v-1}(u, v)$ matches with target pixel 5.75, meaning that the reference pixel is the bottom-right corner of the target pixel. Thus, the reference pixel and the target subpixel $p'(u' + 0.5, v' + 0.5)$ are matched. Eq. 8 states the other three matching conditions.



* express in Pixel Coordinate (u, v)

Figure 28 Pixel-to-subpixel matching and locating

$$If\ Target\ Pixel\ p'(u',v')\ matchs\ with\ scaling\ created\ Pixel\ g_{u^*,v^*}(u,v)\begin{cases} g_{u-1,v-1}(u,v)\\ g_{u,v-1}(u,v)\\ g_{u-1,v}(u,v)\\ g_{u,v}(u,v)\end{cases},$$

$$then\ return\ Target\ Subpixel\begin{cases} p'(u'+0.5,v'+0.5)\\ p'(u'+0.0,v'+0.5)\\ p'(u'+0.5,v'+0.0)\\ p'(u'+0.0,v'+0.0)\end{cases},Target\ Point\ p'(x',y')\begin{cases}(u'-w/2+0.75,v'-h/2+0.75)\\(u'-w/2+0.25,v'-h/2+0.75)\\(u'-w/2+0.75,v'-h/2+0.25)\\(u'-w/2+0.25,v'-h/2+0.25)\end{cases}, \qquad \textbf{Eq. 8}$$

$$and\ Reference\ point\ p(x,y)=(u-w/2+.5,v-h/2+.5)$$

Eq. 8 derivation processes are listed as follows:

The $Pixel\ Coordinate$ and $Image\ Coordinate$ can be converted by (Eq. 8-1) and (Eq. 8-2). The $Pixel\ Coordinate$ is a 2D-coordinate with the origin on the upper-left pixel. The $Image\ Coordinate$ is a 3D-coordinate with the origin on the image center $(w/2, h/2)$, and a fixed z-axis value (f, focal length).

$$Convert\ Pixel\ Coordinate\ (u,v) to\ Image\ Coordinate (x,y),\quad \begin{cases} x=u-\dfrac{w}{2}+0.5\\ y=v-\dfrac{h}{2}+0.5\end{cases} \qquad (\text{Eq.}\,8-1)$$

$$Convert\ Image\ Coordinate(x, y) to\ Pixel\ Coordinate\ (u, v), \quad \begin{cases} u = int\left(x + \frac{w}{2}\right) \\ v = int\left(y + \frac{h}{2}\right) \end{cases} \qquad (\text{Eq}. 8 - 2)$$

Thus, from (Eq. 8-1), has the reference point $p(x, y)$,

$$Reference\ point\ p(x, y) = \left(u - \frac{w}{2} + 0.5, v - \frac{h}{2} + 0.5\right) \qquad (\text{Eq}. 8 - 3)$$

In (Eq. 8-1) $Pixel\ Coordinate$ adjusts 0.5 to get its $Image\ Coordinate$, similarly, the 0.5 subpixel needs adjusts 0.25 to get its $Image\ Coordinate$. Thus, from Eq. 8, the target pixel $p'(u', v')$ can be converted to target point $p'(x', y')$,

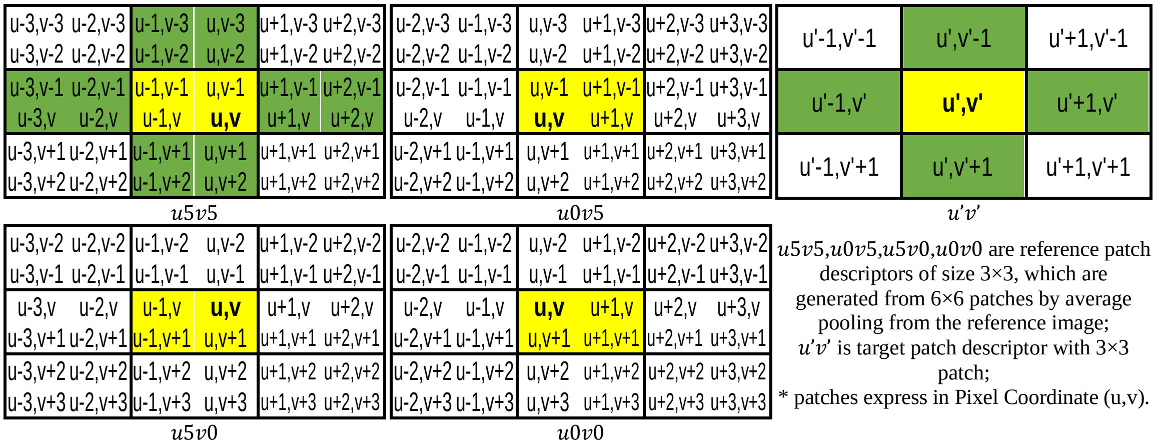
$$Target\ point\ p'(x', y') = \begin{cases} \left(u' - \frac{w}{2} + 0.75, v' - \frac{h}{2} + 0.75\right) \\ \left(u' - \frac{w}{2} + 0.25, v' - \frac{h}{2} + 0.75\right) \\ \left(u' - \frac{w}{2} + 0.75, v' - \frac{h}{2} + 0.25\right) \\ \left(u' - \frac{w}{2} + 0.25, v' - \frac{h}{2} + 0.25\right) \end{cases} \qquad (\text{Eq}. 8 - 4)$$

4.3.1.2 Patch Feature Descriptors Matching in Low-high Ortho-image Pair

In this research project, the single pixel feature descriptor is extended to a patch feature descriptor using target patch $u'v' = I'_{u\prime,v\prime \in (2R+1)\times(2R+1)}(u', v')$ to represent the target pixel/point in the target image. The patch size R is self-adapting and depends on the previous matching result. R will be increased during the matching process until the minimum threshold is satisfied. Similarly, $g_{u^*,v^*}(u, v)$ is extended to patches $u5v5,$ $u0v5, u5v0, u0v0,$ which are reference patches of size $(2R + 1) \times (2R + 1)$ used to represent the reference pixel/point in its four scaling directions. Each reference patch is generated from a patch $I_{u,v \in [2\times(2R+1)]\times [2\times(2R+1)]}(u, v)$ in the reference image with the average pooling operation.

As reference patch descriptors have the same size as the target patch descriptor, the NCC method can be used to match them. In detail, using the NCC method a) calculate the four NCC values between reference patch descriptors $u5v5, u0v5, u5v0, u0v0$ and target patch descriptor $u'v'$; b) choose the largest NCC value as the matched scaling direction; and c) calculate the subpixel location for target pixel/point by Eq. 8.

Figure 29 shows an example of a target patch descriptor and four-scaling reference patch descriptors with R=1, in which the 3×3 reference patches are scaled from 6×6 pixels patch $u5v5$. Thus, with the predefined image scaling, the patch-based NCC method is effective for the low-high ortho-image pair matching.



u-3,v-3	u-2,v-3	u-1,v-3	u,v-3	u+1,v-3	u+2,v-3
u-3,v-2	u-2,v-2	u-1,v-2	u,v-2	u+1,v-2	u+2,v-2
u-3,v-1	u-2,v-1	u-1,v-1	u,v-1	u+1,v-1	u+2,v-1
u-3,v	u-2,v	u-1,v	**u,v**	u+1,v	u+2,v
u-3,v+1	u-2,v+1	u-1,v+1	u,v+1	u+1,v+1	u+2,v+1
u-3,v+2	u-2,v+2	u-1,v+2	u,v+2	u+1,v+2	u+2,v+2

$u5v5$

u-2,v-3	u-1,v-3	u,v-3	u+1,v-3	u+2,v-3	u+3,v-3
u-2,v-2	u-1,v-2	u,v-2	u+1,v-2	u+2,v-2	u+3,v-2
u-2,v-1	u-1,v-1	u,v-1	u+1,v-1	u+2,v-1	u+3,v-1
u-2,v	u-1,v	**u,v**	u+1,v	u+2,v	u+3,v
u-2,v+1	u-1,v+1	u,v+1	u+1,v+1	u+2,v+1	u+3,v+1
u-2,v+2	u-1,v+2	u,v+2	u+1,v+2	u+2,v+2	u+3,v+2

$u0v5$

u'-1,v'-1	u',v'-1	u'+1,v'-1
u'-1,v'	**u',v'**	u'+1,v'
u'-1,v'+1	u',v'+1	u'+1,v'+1

$u'v'$

u-3,v-2	u-2,v-2	u-1,v-2	u,v-2	u+1,v-2	u+2,v-2
u-3,v-1	u-2,v-1	u-1,v-1	u,v-1	u+1,v-1	u+2,v-1
u-3,v	u-2,v	u-1,v	**u,v**	u+1,v	u+2,v
u-3,v+1	u-2,v+1	u-1,v+1	u,v+1	u+1,v+1	u+2,v+1
u-3,v+2	u-2,v+2	u-1,v+2	u,v+2	u+1,v+2	u+2,v+2
u-3,v+3	u-2,v+3	u-1,v+3	u,v+3	u+1,v+3	u+2,v+3

$u5v0$

u-2,v-2	u-1,v-2	u,v-2	u+1,v-2	u+2,v-2	u+3,v-2
u-2,v-1	u-1,v-1	u,v-1	u+1,v-1	u+2,v-1	u+3,v-1
u-2,v	u-1,v	**u,v**	u+1,v	u+2,v	u+3,v
u-2,v+1	u-1,v+1	u,v+1	u+1,v+1	u+2,v+1	u+3,v+1
u-2,v+2	u-1,v+2	u,v+2	u+1,v+2	u+2,v+2	u+3,v+2
u-2,v+3	u-1,v+3	u,v+3	u+1,v+3	u+2,v+3	u+3,v+3

$u0v0$

$u5v5, u0v5, u5v0, u0v0$ are reference patch descriptors of size 3×3, which are generated from 6×6 patches by average pooling from the reference image;
$u'v'$ is target patch descriptor with 3×3 patch;
* patches express in Pixel Coordinate (u,v).

Figure 29 Example of patch feature descriptors

4.3.2 Low-high Ortho-image Pair Pixel Matching and Virtual Elevation Algorithm

4.3.2.1 Virtual Depth-Elevation Model

The virtual depth-elevation plane model is shown in Figure 30, which avoids using a brute-force algorithm to match all pixels in the target image for determing the corresponding target pixel for a reference pixel. In detail, a construction site is divided into several discrete virtual planes and set the drone takeoff plane as the origin plane ($Depth$=0). The *Depth-axis* has positive values below the origin plane, while the *Elevation-axis* has positive values above the origin plane. The origin plane has distance *H/2* to the drone's low altitude position, so the real-word point P on a virtual plane $Depth$ has distance $Z = H/2 + Depth$ to the drone.

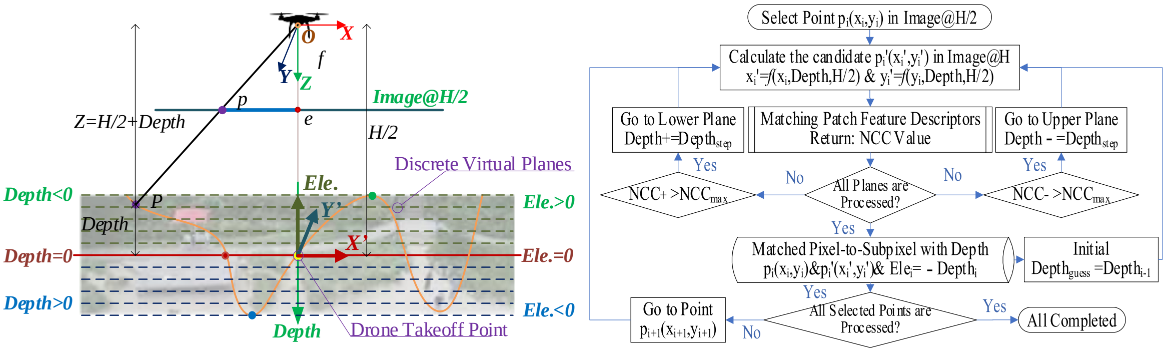

O
X
f
Y
Z
p
e
Image@H/2
Z=H/2+Depth
H/2
Discrete Virtual Planes
Ele.
Y'
Depth<0
P
Ele.>0
Depth
X'
Depth=0
Ele.=0
Depth>0
Ele.<0
Depth
Drone Takeoff Point
Select Point $p_i(x_i,y_i)$ in Image@H/2
Calculate the candidate $p_i'(x_i',y_i')$ in Image@H
$x_i'=f(x_i,Depth,H/2)$ & $y_i'=f(y_i,Depth,H/2)$
Go to Lower Plane
$Depth+=Depth_{step}$
Matching Patch Feature Descriptors
Return: NCC Value
Go to Upper Plane
$Depth - =Depth_{step}$
Yes
No
All Planes are Processed?
No
Yes
$NCC_+ >NCC_{max}$
$NCC_- >NCC_{max}$
Yes
Matched Pixel-to-Subpixel with Depth
$p_i(x_i,y_i)$&$p_i'(x_i',y_i')$& $Ele_i= - Depth_i$
Initial
$Depth_{guess} =Depth_{i-1}$
Go to Point
$p_{i+1}(x_{i+1},y_{i+1})$
Yes
All Selected Points are Processed?
Yes
All Completed
No

Figure 30 Virtual depth-elevation model and pixel matching and elevation determination flowchart

4.3.2.2 Geometry Data Determination

Taking $Z = H/2 + Depth$ expression into Eq. 7 will results in $x' = f(x, Depth, H/2)$ and $y' = f(y, Depth, H/2)$. Thus, for a given reference point $p(x, y)$ and $H/2$ (is fixed), each virtual plane can generate a candidate target point $p_i'(x_i', y_i') = f(x, y, Depth_i, H/2)$. If the reference point matches with the candidate target point $p_i'(x_i', y_i')$ on virtual plane $Depth_i$, then, the real-world point P is located on that virtual plane with the specific $Ele_i = -Depth_i$. The $Elevation\ Coordinate\ (X', Y', Ele.)$ can be determined by Eq. 9.

$$\begin{bmatrix} X' \\ Y' \\ Ele. \end{bmatrix} = \begin{bmatrix} x \cdot GSD \\ -y \cdot GSD \\ -Depth \end{bmatrix}, \text{where } Ele. \in \left(-\frac{H}{2}, \frac{H}{2}\right) \qquad \textbf{Eq. 9}$$

Eq. 9 derivation states as follows:

In Figure 30, Z is the distance from drone to point P, $H/2$ is the distance from drone to its takeoff plane, $Depth$ is the distance from the point P to the drone takeoff plane. It has,

$$Z = \frac{H}{2} + Depth \qquad (\text{Eq.} 9 - 1 - 1)$$

$$Elevation = -\ Depth \qquad (\text{Eq.} 9 - 1 - 2)$$

Take (Eq. 9-1-1) into (Eq. 7-5-1) and (Eq. 7-5-2) has,

$$\frac{H}{2}\frac{x'}{x{-}x'} = \frac{H}{2} + Depth \Rightarrow x' = x\frac{1 + Depth\frac{2}{H}}{2 + Depth\frac{2}{H}} \qquad (\text{Eq.} 9 - 2 - 1)$$

$$\frac{H}{2}\frac{y'}{y{-}y'} = \frac{H}{2} + Depth \Rightarrow y' = y\frac{1 + Depth\frac{2}{H}}{2 + Depth\frac{2}{H}} \qquad (\text{Eq.} 9 - 2 - 2)$$

Thus, for each $Depth$, the target point $p'(x', y')$ has the relationship with reference point $p(x, y)$, $Depth$, $H/2$,

$$p(x', y') = f\left(x, y, Depth, \frac{H}{2}\right) \qquad (\text{Eq.} 9 - 3)$$

Assume candidate target point $p'(x', y')$ at virtual plane $Depth$ matches with the reference point $p(x, y)$, then the real-world point P falls on that virtual plane $Depth$.

In Figure 27, it has,

$$X' = X = x \cdot GSD \qquad (\text{Eq.} 9 - 4 - 1)$$

$$Y' = -Y = -y \cdot GSD \qquad (\text{Eq.} 9 - 4 - 2)$$

As the x, y and $Depth$ can be determined by (Eq. 9-3), thus combine (Eq. 9-1-2), (Eq. 9-4-1) and (Eq. 9-4-2) get the Eq. 9.

4.3.2.3 Pixel Matching and Virtual Elevation Algorithm

The proposed matching procedure is stated in the flowchart in Figure 30, which shows that a given reference point/pixel can match a target point/pixel and return a virtual plane value/elevation value simultaneously. For matching a series of point/pixel pairs, the $Depth_{guess}$ set to the previous point/pixel's virtual plane value. A while-loop starts at $Depth_{guess}$, and goes to the adjacent virtual planes by plus and minus the $Depth_{Step}$, until the best or most acceptable matching result is returned. The pseudocode of the low-high ortho-image pair pixel matching and virtual elevation algorithm is presented in Figure 31.

```
Assume $\boldsymbol{NCC\_MATCH\_SCALING\_LABEL}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{u'}, \boldsymbol{v'}, \boldsymbol{R})$ returns the biggest NCC value and its Scaling Direction $Label_{Scaling}$ from the match results of reference $p(u, v)$ and target $p(u', v')$ and their feature descriptors $u5v5, u0v5, u5v0, u0v0$ and $u'v'$ in size $(2R + 1) \times (2R + 1)$;
Assume $\boldsymbol{IM2PX}(\boldsymbol{x}, \boldsymbol{y})$ returns the Pixel Coordinate $p(u, v)$ from the Image Coordinate $p(x, y)$;
Assume $\boldsymbol{PX2IM}(\boldsymbol{u}, \boldsymbol{v})$ returns the Image Coordinate $p(x, y)$ from the Pixel Coordinate $p(u, v)$;
Assume $\boldsymbol{SUBPX2IM}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{Label_{Scaling}})$ returns the Image Coordinate from Pixel Coordinate $p(u, v)$ and $Label_{Scaling}$ .

Input:   $Img_{H/2}, Img'_{H}, p(u, v), Depth_{guess}, Depth_{step}, R, H/2$
Output: $p(x, y),\ p'(x', y'),\ Depth,\ NCC$

$\boldsymbol{IMAGE\_PAIR\_MATCHING\_VIRTUAL\_ELEVATION}(\boldsymbol{Img_{H/2}}, \boldsymbol{Img'_{H}}, \boldsymbol{p(u, v)}, \boldsymbol{Depth_{guess}}, \boldsymbol{Depth_{step}}, \boldsymbol{R}, \boldsymbol{H/2})$
Initial $Depth = Depth_{guess}$; $NCC_{max} = 0$; $R_{adj.ratio+} = 1$; $R_{adj.ratio-} = 1$; $Depth_{current-} = Depth_{guess}$; $Depth_{current+} = Depth_{guess} + Depth_{Step}$
$p(x, y) = \boldsymbol{PX2IM}(u, v)$
while $Depth_{current-} > -H/2$ or $Depth_{current+} < H/2$
    if $Depth_{current-} > -H/2$
        $p'(x', y') = f(x, y, Depth_{current-}, H/2)$;  $p(u', v') = \boldsymbol{IM2PX}(x', y')$
        $NCC_{current-}, Label_{scaling-} = \boldsymbol{NCC\_MATCH\_SCALING\_LABEL}(u, v, u', v', R \times R_{adj.ratio-})$
        if $NCC_{current-} > NCC_{max}$
            $NCC_{max} = NCC_{current-}$; $Depth_{matched} = Depth_{current-}$; $p'_{matched}(x', y') = \boldsymbol{SUBPX2IM}(u', v', Label_{scaling-})$
    if $Depth_{current+} < H/2$
        $p'(x', y') = f(x, y, Depth_{current+}, H/2)$;  $p(u', v') = \boldsymbol{IM2PX}(x', y')$
        $NCC_{current+}, Label_{scaling+} = \boldsymbol{NCC\_MATCH\_SCALING\_LABEL}(u, v, u', v', R \times R_{adj.ratio+})$
        if $NCC_{current+} > NCC_{max}$
            $NCC_{max} = NCC_{current+}$; $Depth_{matched} = Depth_{current+}$; $p'_{matched}(x', y') = \boldsymbol{SUBPX2IM}(u', v', Label_{scaling+})$
    if $NCC_{max} < Threshold_{low}$
        $R_{adj.ratio-} += 0.2$
        $R_{adj.ratio+} += 0.2$
    else
        $Depth_{current-} -= Depth_{Step}$
        $Depth_{current+} += Depth_{Step}$
return $p(x, y),\ p'_{matched}(x', y'),\ Depth_{matched},\ NCC_{max}$
```

Figure 31 Pseudocode of low-high ortho-image pair pixel matching and virtual elevation algorithm

4.3.3 Low-high Ortho-image Pair Pixel Grid and Elevation-map Algorithm

4.3.3.1 Pixel Grid Formation

Figure 32 explains the pixel grid formation for sampling a low-high ortho-image pair matching. In detail, pixels are selected with an interval of $Grid\ Size$, and each selected pixel in a pixel grid is designed to share its elevation data to its neighbors within a *Grid Size* × *Grid Size* patch to create the *Ele_map*. *Margin* in each image edge is used to guarantee that all selected pixels have their patch descriptors.

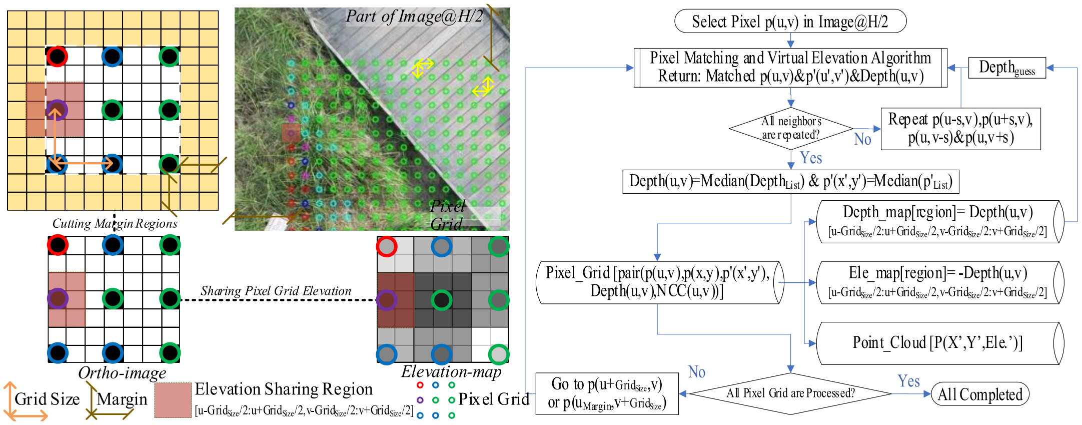

Part of Image@H/2
Cutting Margin Regions
Pixel Grid
Sharing Pixel Grid Elevation
Ortho-image
Elevation-map
Grid Size
Margin
Elevation Sharing Region
[u-Grid_Size/2:u+Grid_Size/2,v-Grid_Size/2:v+Grid_Size/2]
Pixel Grid
Select Pixel p(u,v) in Image@H/2
Pixel Matching and Virtual Elevation Algorithm
Return: Matched p(u,v)&p'(u',v')&Depth(u,v)
Depth_guess
All neighbors are repeated?
No
Repeat p(u-s,v),p(u+s,v), p(u,v-s)&p(u,v+s)
Yes
Depth(u,v)=Median(Depth_List) & p'(x',y')=Median(p'_List)
Depth_map[region]= Depth(u,v)
[u-Grid_Size/2:u+Grid_Size/2,v-Grid_Size/2:v+Grid_Size/2]
Pixel_Grid [pair(p(u,v),p(x,y),p'(x',y'), Depth(u,v),NCC(u,v))]
Ele_map[region]= -Depth(u,v)
[u-Grid_Size/2:u+Grid_Size/2,v-Grid_Size/2:v+Grid_Size/2]
Point_Cloud [P(X',Y',Ele.')]
No
Go to p(u+Grid_Size,v) or p(u_Margin,v+Grid_Size)
All Pixel Grid are Processed?
Yes
All Completed

Figure 32 Pixel grid and elevation-map formation

4.3.3.2 Pixel Grid and Elevation-map Formation Algorithm

The pseudocode of pixel grid and elevation-map formation algorithm is presented in Figure 33.

```
Assume $\boldsymbol{A}[\boldsymbol{N}]$ returns a new List $A[\,]$ with size $N$;
Assume $\boldsymbol{MEDIAN}(\boldsymbol{A}[\,])$ returns the median value of List $A[\,]$;
Assume $\boldsymbol{A}[\,].\boldsymbol{APPEND}(\boldsymbol{B})$ add element $B$ to the end of List $A[\,]$;
Assume $\boldsymbol{ELE2GRAY}(\boldsymbol{Ele.})$ returns Gray value $0\sim255$ from the Elevation $-H/2\sim H/2$;
Assume $\boldsymbol{Img}(\boldsymbol{u},\boldsymbol{v}).\boldsymbol{COPY}()$ returns a same size Grayscale Image/Array;
Assume $\boldsymbol{SQDIFF\_NORMED}(\boldsymbol{p}(\boldsymbol{u},\boldsymbol{v}),\boldsymbol{p}'[\boldsymbol{n}](\boldsymbol{u}',\boldsymbol{v}'))$ returns the best matched $\boldsymbol{p}'[i]$ for $\boldsymbol{p}$ by Normalized Sum of Squared Differences (SSD);
Assume $\boldsymbol{CUT}(\boldsymbol{Img}(\boldsymbol{u},\boldsymbol{v}),\boldsymbol{Margin})$ returns $Img(u,v)$'s central region without the Margin region.

Input:  $Img_{H/2}(u,v)$, $Img'_H(u',v')$, $Grid_{size}$, $H/2$
Output: $Pixel\_Grid[pair(p(u,v),p(x,y),p'(x',y'),Depth)]$, $Point\_Cloud[P(X',Y',Ele.)]$, $Ortho\_image(u,v)$, $Elevation\_Map(u,v)$

$\boldsymbol{PIXEL\_GRID\_ELEVATION\_MAP}(\boldsymbol{Img_{H/2}}(\boldsymbol{u},\boldsymbol{v}),\boldsymbol{Img'_H}(\boldsymbol{u}',\boldsymbol{v}'),\boldsymbol{Grid_{size}},\boldsymbol{H/2})$
Initial $v = Margin$; $Depth\_map(u,v) = Img_{H/2}(u,v).\boldsymbol{COPY}()$; $Ele\_map(u,v) = Img_{H/2}(u,v).\boldsymbol{COPY}()$
while $v \le Img_{H\_height} - Margin$
    $u = Margin$; $p_{List}(x,y)[5]$; $p_{List}'(x',y')[5]$; $Depth_{List}[5]$; $NCC_{List}[5]$
    while $u \le Img_{H\_width} - Margin$
        $Depth_{guess} = Depth\_map(\boldsymbol{SQDIFF\_NORMED}(p(u,v),[p(u-Grid_{size},v),p(u-Grid_{size},v-Grid_{size}),p(u,v-Grid_{size}),p(u+Grid_{size},v-Grid_{size})]))$
        $p_0(x,y),p_0'(x',y'),Depth_0,NCC_0 = \boldsymbol{IMAGE\_PAIR\_MATCHING\_VIRTUAL\_ELEVATION}(Img_{H/2},Img'_H,p(u,v),Depth_{guess},Depth_{step},R,H/2)$
        $p_1(x,y),p_1'(x',y'),Depth_1,NCC_1 = \boldsymbol{IMAGE\_PAIR\_MATCHING\_VIRTUAL\_ELEVATION}(Img_{H/2},Img'_H,p(u-s,v),Depth_{guess},Depth_{step},R,H/2)$
        $p_2(x,y),p_2'(x',y'),Depth_2,NCC_2 = \boldsymbol{IMAGE\_PAIR\_MATCHING\_VIRTUAL\_ELEVATION}(Img_{H/2},Img'_H,p(u+s,v),Depth_{guess},Depth_{step},R,H/2)$
        $p_{\boldsymbol{3}}(x,y),p_3'(x',y'),Depth_3,NCC_3 = \boldsymbol{IMAGE\_PAIR\_MATCHING\_VIRTUAL\_ELEVATION}(Img_{H/2},Img'_H,p(u,v-s),Depth_{guess},Depth_{step},R,H/2)$
        $p_{\boldsymbol{4}}(x,y),p_4'(x',y'),Depth_4,NCC_4 = \boldsymbol{IMAGE\_PAIR\_MATCHING\_VIRTUAL\_ELEVATION}(Img_{H/2},Img'_H,p(u,v+s),Depth_{guess},Depth_{step},R,H/2)$
        $p(x,y)$, $p'(x',y')$, $Depth, NCC = \boldsymbol{MEDIAN}(p_{List}(x,y)), \boldsymbol{MEDIAN}(p_{List}'(x',y')), \boldsymbol{MEDIAN}(Depth_{List}), \boldsymbol{MEDIAN}(NCC_{List})$
        $Pixel\_Grid[pair(p(u,v),p(x,y),p'(x',y'),Depth,NCC)].\boldsymbol{APPEND}(pair(p(u,v),p(x,y),p'(x',y'),Depth,NCC))$
        $X' = x \times GSD$; $Y' = -y \times GSD$; $Ele. = -Depth$
        $Point\_Cloud[P(X',Y',Ele.)].\boldsymbol{APPEND}(P(X',Y',Ele.))$
        $Ele\_map(u - Grid_{size}/2 : u + Grid_{size}/2, v - Grid_{size}/2 : v + Grid_{size}/2) = \boldsymbol{ELE2GRAY}(Ele.)$
        $Depth\_map(u - Grid_{size}/2 : u + Grid_{size}/2, v - Grid_{size}/2 : v + Grid_{size}/2) = Depth$
        $u \mathrel{+}= Grid_{size}$
    $v \mathrel{+}= Grid_{size}$
$Ortho\_image(u,v) = CUT(Img_{H/2}(u,v),Margin)$; $Elevation\_Map(u,v) = CUT(Ele\_Map(u,v),Margin)$
return $Pixel\_Grid[pair(p(u,v),p(x,y),p'(x',y'),Depth,NCC)], Point\_Cloud[P(X',Y',Ele.)], Ortho\_image(u,v), Elevation\_Map(u,v)$
```

Figure 33 Pseudocode of pixel grid and elevation-map formation algorithm

In line 6 to 10, the pixel matching and virtual elevation algorithm is repeated 5 times in pixel $p(u,v)$ and its neighbors to enhance matching results using their median values. The distance (s) to neighboring pixels can be adjusted from 1 to $Grid\ Size$. After traversed and matched all selected pixels, the matched results are stored in the $Pixel\_Grid$. A 3D point cloud $Point\_Cloud$ of a construction site is created as well. Furthermore, an ortho-image and elevation-map (stores elevation data as 0~255 grayscale value) pair is created by cutting off the low ortho-image and Ele_map margin separately, then pixels in the ortho-image and elevation-map are linked.

4.3.3.3 Pixel Grid and Elevation-map Enhancement Algorithm

The pseudocode of pixel grid enhancement algorithm is presented in Figure 34.

```
Assume $\mathbf{ROTATE}(\mathbf{Img},\mathbf{D})$ returns a new $Img'(u',v')$ by anticlockwise rotating $Img(u,v)$ with $D$ degrees;
Assume $\mathbf{PIXEL\_GRID\_ELEVATION\_MAP}(\mathbf{Img_{H/2}},\mathbf{Img'_H},\mathbf{Grid_{size}},\mathbf{H/2})$ returns $Pixel\_Grid\_Array[row(x,y,Depth,NCC)]$;
Assume $\mathbf{ARRAY.SORT\_1\_2}()$ sorts the Array by its 1st and 2nd Columns;
Assume $\mathbf{ARRAY.COL}[\mathbf{i}:\mathbf{j}]$returns Array's $\mathbf{i^{th}}$ to $\mathbf{j^{th}}$ Columns;
Assume $\mathbf{A}[\,].\mathbf{LowFence}()$ returns the Low Fence $Q1 - 1.5 \times (Q3 - Q1)$ of List $A[\,]$;
Assume $\mathbf{LEN}(\mathbf{A}[\,])$ returns the size of List $A[\,]$.

Input:   $Img_{H/2}(u,v), Img'_H(u',v'), Grid_{size}, H/2$
Output: $Enhanced\_Pixel\_Grid\_Array[row(x,y,Depth,C)]$, $Point\_Cloud[P(X',Y',Ele.)]$, $Ortho\_image(u,v)$, $Elevation\_Map(u,v)$

$\mathbf{PIXEL\_GRID\_ENHANCEMENT}(\mathbf{Img_{H/2}(u,v)},\mathbf{Img'_H(u',v')},\mathbf{Grid_{size}},\mathbf{H/2})$
1   Initial $Img_0 = Img_{H/2}$; $Img_{90} = \mathbf{ROTATE}(Img_0,90)$; $Img_{180} = \mathbf{ROTATE}(Img_0,180)$; $Img_{270} = \mathbf{ROTATE}(Img_0,270)$;
    $Img'_0 = Img'_H$; $Img'_{90} = \mathbf{ROTATE}(Img'_0,90)$; $Img'_{180} = \mathbf{ROTATE}(Img'_0,180)$; $Img'_{270} = \mathbf{ROTATE}(Img'_0,270)$
2   for i in [0,90,180,270]
3       $Pixel\_Grid\_Array_i[row(x,y,Depth,NCC)] = \mathbf{PIXEL\_GRID\_ELEVATION\_MAP}(Img_i, Img'_i, Grid_{size}, H/2)$
4   $R\_Pixel\_Grid_1[row(x,y,Depth,NCC)] = Pixel\_Grid\_Array_0\ [row(x,y,Depth,NCC)].\mathbf{SORT\_1\_2}()$;
    $R\_Pixel\_Grid_2[row(x,y,Depth,NCC)] = Pixel\_Grid\_Array_{90}\ [row(-y,x,Depth,NCC)].\mathbf{SORT\_1\_2}()$;
    $R\_Pixel\_Grid_3[row(x,y,Depth,NCC)] = Pixel\_Grid\_Array_{180}[row(-x,-y,Depth,NCC)].\mathbf{SORT\_1\_2}()$;
    $R\_Pixel\_Grid_4[row(x,y,Depth,NCC)] = Pixel\_Grid\_Array_{270}[row(y,-x,Depth,NCC)].\mathbf{SORT\_1\_2}()$
5   for i in [1,2,3,4]
6       $x_i[], y_i[], D_i[], W_i[] = R\_Pixel\_Grid_i[row(x,y,Depth,NCC)].\mathbf{COL}[1:4]$
7       $Q_i = \mathbf{MAX}(W_i.\mathbf{LowFence}(),0.001)$
8   $N = \mathbf{LEN}(x_1[])$; $x_{list}[N]$; $y_{list}[N]$; $Depth_{list}[N]$; $C_{list}[N]$
9   for i in [1, ..., N]
10      $x_{list}.\mathbf{APPEND}(x_1[i])$; $y_{list}.\mathbf{APPEND}(y_1[i])$; $Depth_{list}.\mathbf{APPEND}\left(\mathbf{MEDIAN}\left(\mathbf{D_{i|W_i \geq Q_i}}[\,]\right)\right)$; $C_{list}.\mathbf{APPEND}(1_{W_1 \geq Q_1} 2_{W_2 \geq Q_2} 3_{W_3 \geq Q_3} 4_{W_4 \geq Q_4})$
11      $Enhanced\_Pixel\_Grid\_Array[row(x,y,Depth,C)].\mathbf{APPEND}(row(x_{list}[i], y_{list}[i], Depth_{list}[i], C_{list}[i]))$
12      $X' = x_{list}[i] \times GSD$; $Y' = -y_{list}[i] \times GSD$; $Ele. = -Depth_{list}[i]$; $p(u,v) = \mathbf{IM2PX}(x,y)$
13      $Point\_Cloud[P(X',Y',Ele.)].\mathbf{APPEND}(P(X',Y',Ele.))$
14      $Ele\_map(u - Grid_{size}/2 : u + Grid_{size}/2, v - Grid_{size}/2 : v + Grid_{size}/2) = \mathbf{ELE2GRAY}(Ele.)$
15  $Ortho\_image(u,v) = \mathbf{CUT}(Img_{H/2}(u,v), Margin)$; $Elevation\_Map(u,v) = \mathbf{CUT}(Ele\_Map(u,v), Margin)$
16  return  $Enhanced\_Pixel\_Grid\_Array[row(x,y,Depth,C)]$, $Point\_Cloud[P(X',Y',Ele.)]$, $Ortho\_image(u,v)$, $Elevation\_Map(u,v)$
```

Figure 34 Pseudocode of pixel grid and elevation-map enhancement algorithm

In the pixel grid and elevation-map algorithm, the selected pixels are traversed row by row, each pixel uses the previous pixel's $Depth$ value as the input variable $Depth_{guess}$ for matching its own $Depth$ value.

To make this algorithm robust, a low-high ortho-image pair is proposed to rotate 90°, 180° and 270° in a counterclockwise fashion and the pixel grid and elevation-map algorithm is repeated to generate four $Pixel\_Grid_i$ results starting from each corner of the ortho-image pair. In addition, the four results are transformed back to the original coordinate. In each $Pixel\_Grid_i$, if a selected pixel has NCC value $W_i[u,v]$ larger than 0.001 and $W_i.Lower\ Fence_i$ (the lower fence of NCC values of all selected pixels in $Pixel\_Grid_i$, any NCC value less than the lower fence is considered as an outlier), it is considered as a strongly matched pixel pair, otherwise it is a weakly matched pixel pair. Combining the four $Pixel\_Grid_{i=1,2,3,4}$ matching results, each selected pixel/point has 16 matching quality conditions that are listed in Table 8. For example, the 2nd condition means a selected pixel has strongly matched results in the original, 90° rotation, and 180° rotation ortho-image pairs and weakly matched result for 270°. It is then assigned “123” for its matching quality label, and the median value of $[\,D_1, D_2, D_3]$ for its enhanced $Depth$.

Table 8 Matching Quality Mark and Enhanced Depth

Matching Quality		***Compare with*** $Q_i = MAX(W_i.Lower\ Fence, 0.001)$				***Matching Quality Label*** C_{list}	***Enhanced Depth*** $Depth_{list}$
		W_1	*W_2*	*W_3*	*W_4*	$1_{W_1 \geq Q_1} 2_{W_2 \geq Q_2} 3_{W_3 \geq Q_3} 4_{W_4 \geq Q_4}$	$MEDIAN\left(D_{i \mid W_i \geq Q_i}[\,]\right)$
Strongest	1	≥	≥	≥	≥	1234	$MEDIAN(\,D_1, D_2, D_3, D_4)$
strong	┌ 2	≥	≥	≥	<	123	$MEDIAN(\,D_1, D_2, D_3)$
	├ 3	≥	≥	<	≥	124	$MEDIAN(\,D_1, D_2, D_4)$
	├ 4	≥	<	≥	≥	134	$MEDIAN(\,D_1, D_3, D_4)$
	└ 5	<	≥	≥	≥	234	$MEDIAN(\,D_2, D_3, D_4)$
weak	┌ 6	≥	≥	<	<	12	$MEDIAN(\,D_1, D_2)$
	├ 7	≥	<	≥	<	13	$MEDIAN(\,D_1, D_3)$
	├ 8	≥	<	<	≥	14	$MEDIAN(\,D_1, D_4)$
	├ 9	<	≥	≥	<	23	$MEDIAN(\,D_2, D_3)$
	├ 10	<	≥	<	≥	24	$MEDIAN(\,D_2, D_4)$
	└ 11	<	<	≥	≥	34	$MEDIAN(\,D_3, D_4)$
weaker	┌ 12	≥	<	<	<	1	D_1
	├ 13	<	≥	<	<	2	D_2
	├ 14	<	<	≥	<	3	D_3
	└ 15	<	<	<	≥	4	D_4
Weakest	16	<	<	<	<	0	$MEDIAN(\,D_1, D_2, D_3, D_4)$

*W_i is the NCC values of all selected pixels in $Pixel\_Grid_i$; $W_i.Lower\ Fence_i$ = Q1-1.5×(Q3-Q1) , the lower fence of NCC values of the selected pixels in $Pixel\_Grid_i$, any NCC value less than the lower fence is considered as an outlier; $1_{W_1 \geq Q_1}$means the $W_1[u,v]$ is a strongly matched result in the original ortho-image pair, “1” will be assign to assemble the matching quality label $C_1[u,v]$, similarly, “2” is for 90° rotation, “3” for 180° rotation and “4” for 270° rotation of the ortho-image pair; $D_{1|W_1 \geq Q_1}[u,v]$ means pixel $p(u,v)$ is strongly matched in the original ortho-image pair, if not, $D_{1|W_1<Q_1}[u,v]$ will not be used to enhance the $Depth[u,v]$.

In Table 8, the “weakest” means four-rotation matching results of a pixel are all weakly matched. Another kind of weak matching is on the central of the ortho-image pair. When the reference point (x,y) is close to center (0,0), the target point (x',y') is close to (0,0) and becomes insensitive to $Depth$ variation (see

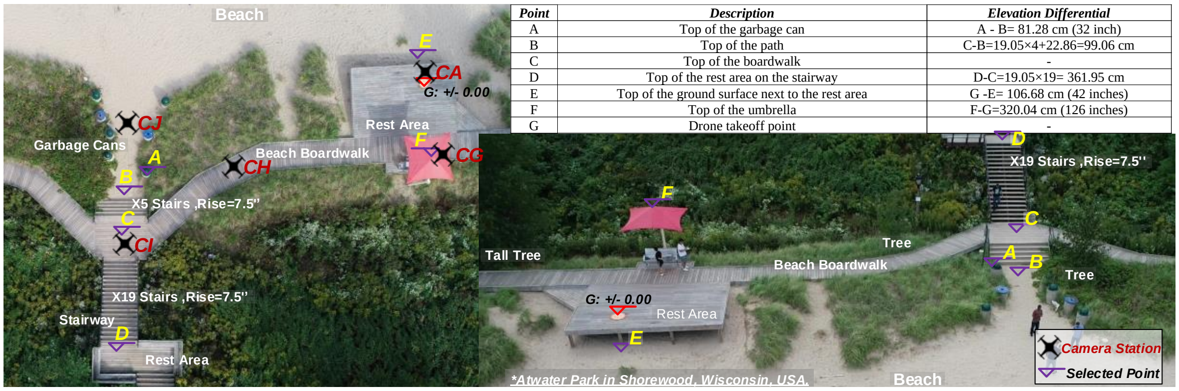
Eq. 9-2-1, Eq. 9-2-2). This leads to the pixel matching and virtual elevation algorithm to generate the same NCC value from different $Depth$ values. Fortunately, the center region is usually a flat plane for drone takeoff; its elevation can be easily confined to its neighbors' elevations. Therefore, the $Depth$ of a weakest pixel can be inherited from an adjacent pixel ($C \geq 1$) that has the closest texture feature. The updated pixel will be assigned a new matching label C=5 to participate in enhancing the remaining pixels.

4.4 Pixel Grid Matching and Elevation Determination Experiment Design

4.4.1 Experiment Dataset

4.4.1.1 Experiment Site

The developed algorithms were programmed in Python 3.6.8 and verified on the construction site shown in Figure 35. This beach site includes a stairway, a boardwalk with rest area, several garbage cans and vegetation. The elevation differentials between the selected points were measured for evaluating the developed method. The height of the bottom stair (above the ground) is 22.86 cm (9 inches) and the height of other stairs is 19.05 cm (7.5 inches).



Point	***Description***	***Elevation Differential***
A	Top of the garbage can	A - B= 81.28 cm (32 inch)
B	Top of the path	C-B=19.05×4+22.86=99.06 cm
C	Top of the boardwalk	-
D	Top of the rest area on the stairway	D-C=19.05×19= 361.95 cm
E	Top of the ground surface next to the rest area	G -E= 106.68 cm (42 inches)
F	Top of the umbrella	F-G=320.04 cm (126 inches)
G	Drone takeoff point	-

Figure 35 Elevation determination experiment site

4.4.1.2 Low-high Ortho-image Pairs

During this research, a *DJI Phantom 4 Pro V2.0* ($focal\ length$=8.8 mm, $Sensor_{height}$=8.8 mm) took off at point *G* and flew to point *C*, and captured ortho-images at five selected camera stations. At stations CA and CI, the drone captured the ortho-image series at 10m, 20m and 40 m of heights, which have flat central regions. At stations CG and CJ ortho-images at 10m and 20m of heights were captured, which have concavo-convex central regions. At station CH ortho-images at 20m and 40m of heights were captured, which are used to stitch with other ortho-images and experimental results. Thus, four 10-20 ortho-image pairs and three 20-40 ortho-image pairs were assembled. Additionally, three pre-processing steps were implemented to generate low-high ortho-image pairs shown in Figure 36: a) shrink original images (4864×3648 pixels) to half resolution; b) cut images to square shape (1824×1824 pixels); and c) align high images to low images by slight translation and rotation.

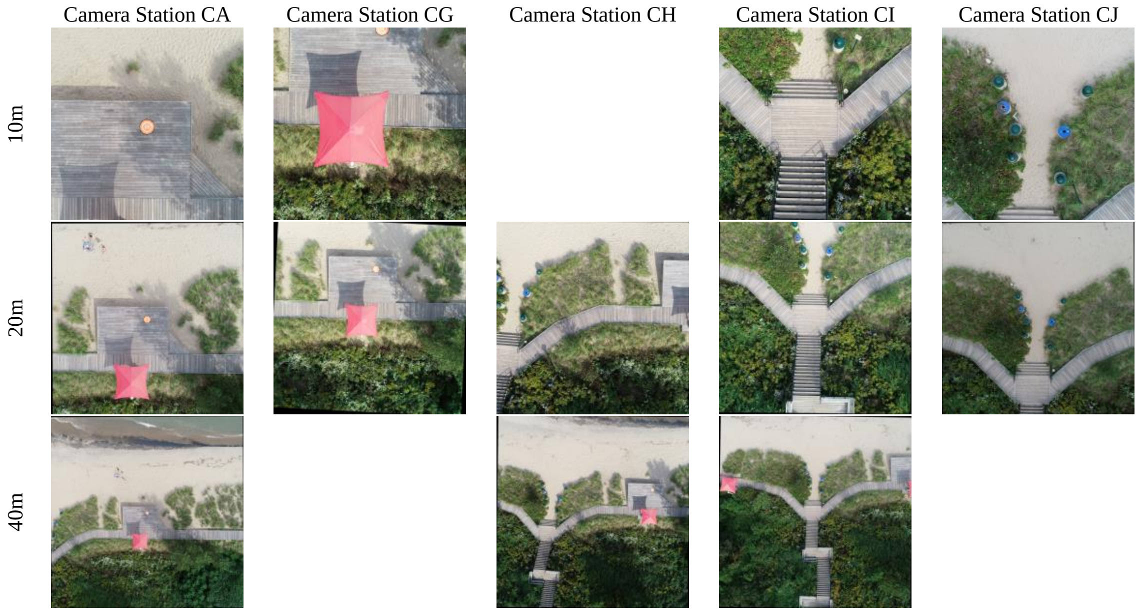


Figure 36 Ortho-image pairs

4.4.2 Pixel Grid Matching Algorithm Configuration

The experimental parameters configurations for the developed method are explained in Table 9. The experimental elevation range was first set as $[-H/4, H/4]$, then, each pixel in an elevation-map used a grayscale value [0,255] to represent its elevation data. There are 200 major virtual planes and 1000 minor virtual planes in the range of $[-H/4, H/4]$. The pixel matching and virtual elevation algorithm searches all major planes. If two adjacent major planes return the same matching value, then the algorithm adjusts the $Depth_{step}$ to $Depth_{step}/5$ and searches the five minor planes between those two major planes to find the best matching result. Additionally, the $Grid\ Size$ for 20-40 ortho-image pairs is 3/4 of the 10-20 ortho-image pairs, and they have different pixel grid numbers.

After running the developed Python program, the output includes a matched pixel grid with matching quality labels(see Figure 37), an ortho-image and elevation-map pair(see Figure 38 and Figure 39), and a point cloud(see Figure 48).

Table 9 Pixel Grid Matching Algorithm Parameters Configuration

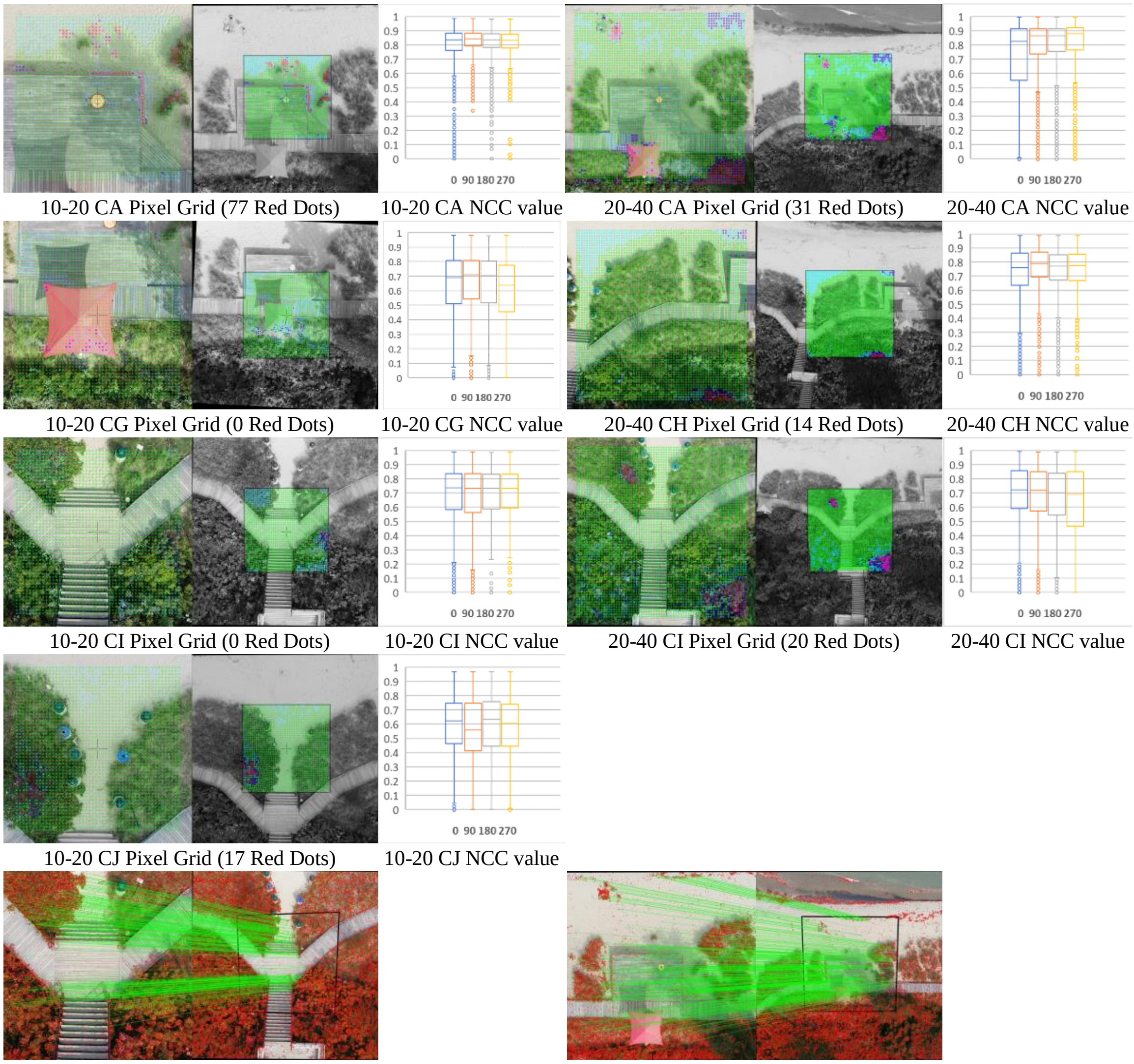
Parameters	***Value (H/2-H)***		***Comments***
	10-20	***20-40***	
Ortho-image Size	1824×1824 pixels		-
Grid Size	32 pixels	24 pixels	Pixel Grid formation, see Figure 32
Initial Patch Size	R =19 pixels $Window\ Size$ = 39×39		$R \times R_{adj.ratio*}$ is self-adapting, see Figure 31 $Window = (2R+1) \times (2R+1)$, see Figure 29
Maximum Patch Size	$R \times R_{adj.ratio*}$ = 76 pixels		$R_{adj.ratio*} \in [1,4]$, see Figure 31
Margin Size	128 pixels	96 pixels	$4 \times\ Maximum\ (Grid\ Size, R)$, see Figure 32
Expected Output Size	1568×1568 pixels	1632×1632 pixels	$Ortho \cdot image\ Size\ -\ 2\ \times Margin\ Size$
Pixel Grid Number	2500	4761	$(Expected\ Output\ Size/GridSize + 1)^2$
Ground Sample Distance	0.54 cm/pixel	1.08 cm/pixel	Eq. 1
Horizontal Space Resolution	8.47×8.47 m^2	17.6×17.6 m^2	$GSD \times\ Output\ Image\ Size$
Elevation Range	[-5 m,5 m]	[-10m, 10m]	$[-H/4, H/4]$
Virtual Plane Number	200		Virtual Plane formation, see Figure 30
Major Depth Step	0.05 m	0.1 m	$Depth_{Step} =\ H/2\ /\ 200$, see Figure 31
Minor Depth Step	0.01 m	0.02 m	$Depth_{Step}\ /5$
Elevation-map	$gray_{u,v} = 255 \times \frac{Ele._{u,v} + H/4}{H/2}$		An 8bit Grayscale Image, see Figure 32 $Ele._{u,v} = H/2 \times gray_{u,v}/255 - H/4$
Distance to Neighbor Pixels	$s =\ Grid\ Size/2$		$s \in [0, Grid\ Size]$, see Figure 33
Image Center Region	$Radius$ = 192 pixels		Pixels are confined to matching between virtual planes $Depth_{guess} - Depth_{Step}$ and $Depth_{guess} + Depth_{Step}$
Strong-matching Threshold	$Maximum\ (Lower\ Fence_i, 0.001)$		see Table 8 and Figure 34
Matched Pixel Label/Mark	Label 0 as Red Dot; Label 1,2,3,4 as Pink Dot; Label 12,13,14,23,24,34 as Blue Dot ; Label 123,124,134,234 as Cyan Dot; Label 1234 as Green Dot; See Table 8		

4.5 Pixel Grid Matching and Elevation Determination Evaluation

4.5.1 Pixel Grid Matching Results and Analysis

4.5.1.1 Pixel Grid Matching Results

The pixel grid matching results are listed in Figure 37. These experimental results show that the developed four-scaling patch feature descriptor and pixel grid matching algorithm can generate the dense pixel grids from the low-high ortho-image pairs, where strongly matched pixel pairs are evenly distributed throughout each low-high ortho-image pair, even in the poorly textured beach regions and dense vegetation regions.



10-20 CA Pixel Grid (77 Red Dots) 10-20 CA NCC value 20-40 CA Pixel Grid (31 Red Dots) 20-40 CA NCC value

10-20 CG Pixel Grid (0 Red Dots) 10-20 CG NCC value 20-40 CH Pixel Grid (14 Red Dots) 20-40 CH NCC value

10-20 CI Pixel Grid (0 Red Dots) 10-20 CI NCC value 20-40 CI Pixel Grid (20 Red Dots) 20-40 CI NCC value

10-20 CJ Pixel Grid (17 Red Dots) 10-20 CJ NCC value

10-20 CI by SIFT method 20-40 CA by SIFT method

*In boxplots, "0" is the original ortho-image pair, "90", "180" and "270" are the rotated ortho-image pairs; in the pixel grid plot, the red dots are weakest matched pixels; in SIFT matching, the red dots are unmatched keypoints.

Figure 37 Ortho-image pair matching results

4.5.1.2 Pixel Grid Matching Evaluation

The developed patch feature descriptors have a self-adapting mechanism ($R \times R_{adj.ratio*}$, line 14 see Figure 31), which uses a large $Patch\ Size$ to improve the matching results in poorly textured regions such as the area with a red umbrella. Furthermore, the shaded regions of the umbrella on the rest area, and most shaded regions of the tall tree on the rest area and the beach are well matched, which overcame the impact of environment brightness changes. However, the SIFT method only matched 432 sparse keypoints shown in Figure 37 due to low contrast.

The NCC value distributions for each rotation of each ortho-image pair are different (see Figure 37) because their different starting corners result in different $Depth_{guess}$ for the remaining selected pixels (line 5 in Figure 33). In these results, the number of outliers on the boxplots has a positive correlation with the number of weakest pixels, such as the 10-20 CA, which has the largest number of weakest pixels (77 of 2500). In addition, Table 10 shows the strongly matched ratio is $[92.52\%, 98.64\%]$, while the weakest matching ratio is only $[0.00\%, 3.08\%]$ in the experimental results. Therefore, the developed pixel grid enhancement algorithm that repeats matching from four starting corners of the squared ortho-image (line 3 in Figure 34) can enhance the pixel matching results.

Table 10 Pixel Grid Matching Quality

Matching Quality Label		***10-20CA***		***10-20CG***		***10-20CI***		***10-20CJ***		***20-40CA***		***20-40CH***		***20-40CI***	
		Count	***%***	***Count***	***%***	***Count***	***%***	***Count***	***%***	***Count***	***%***	***Count***	***%***	***Count***	***%***
Strongest/Green	1234	1960	78.4	2037	81.48	2141	85.64	2254	90.16	3855	80.97	4288	90.07	4108	86.28
Strong/Cyan	234	287	11.48	124	4.96	156	6.24	11	0.44	240	5.04	277	5.82	6	0.13
	134	13	0.52	28	1.12	147	5.88	87	3.48	93	1.95	51	1.07	57	1.2
	124	54	2.16	74	2.96	8	0.32	67	2.68	87	1.83	52	1.09	130	2.73
	123	49	1.96	186	7.44	14	0.56	6	0.24	130	2.73	14	0.29	216	4.54
Weak/Blue	34	2	0.08	3	0.12	28	1.12	1	0.04	67	1.41	1	0.02	29	0.61
	24	1	0.04	2	0.08	1	0.04	9	0.36	1	0.02	1	0.02	1	0.02
	23	1	0.04	16	0.64	-	-	4	0.16	1	0.02	1	0.02	7	0.15
	14	7	0.28	8	0.32	-	-	14	0.56	33	0.69	21	0.44	49	1.03
	13	4	0.16	2	0.08	-	-	1	0.04	10	0.21	2	0.04	10	0.21
	12	14	0.56	14	0.56	-	-	1	0.04	62	1.3	13	0.27	5	0.11
Weaker/Pink	4	7	0.28	1	0.04	-	-	5	0.2	-	-	10	0.21	111	2.33
	3	5	0.2	2	0.08	5	0.2	20	0.8	-	-	2	0.04	5	0.11
	2	-	-	2	0.08	-	-	3	0.12	-	-	3	0.06	1	0.02
	1	19	0.76	1	0.04	-	-	-	-	151	3.17	11	0.23	6	0.13
Weakest/Red	0	77	3.08	-	-	-	-	17	0.68	31	0.65	14	0.29	20	0.42
	N=	2500		2500		2500		2500		4761		4761		4761	
Strongly Matched=		2363	94.52	2449	97.96	2466	98.64	2425	97.00	4405	92.52	4682	98.34	4517	94.88
SIFT matched		216		473		432		325		569		1324		1002	

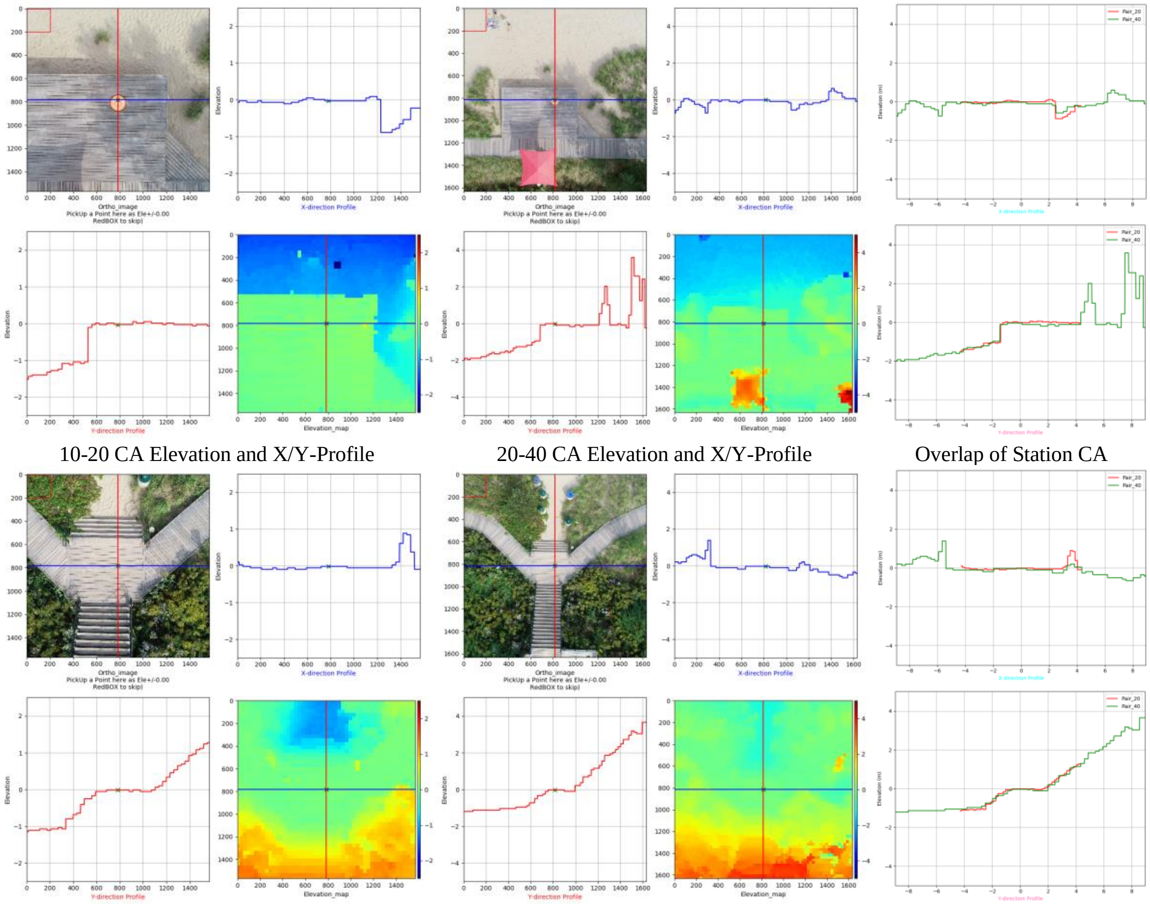
4.5.1.3 Pixel Grid Matching Discussion

The weakest matched pixel pairs, marked as red dots, primarily occurred on the regions of singular trees, because their heights are suddenly different from their surroundings in a very small region compared to the umbrella in 10-20 CG, which is strongly matched. What's more, plants have limited impact on elevation determination; further work should consider removing plants and restoring the ground surface under them. Other weakest matched pixel pairs occur on the ground next to the upper-right corner of the rest area in 10-20 CA, because the rest area and the beach have the low contrast texture caused by the shade of the nearby tall tree. However, the 10-20 CA elevation results in Figure 37 show that their elevations were determined well by the developed method because the incorrect elevations were replaced by the strongly matched neighbors' elevations.

4.5.2 Elevation Determination Results and Analysis

4.5.2.1 Elevation Determination Results

The elevation date (converted from the grayscale elevation-map) results are shown in Figure 38 and Figure 39, which were aligned to the ortho-image center as the elevation origin.



10-20 CA Elevation and X/Y-Profile

20-40 CA Elevation and X/Y-Profile

Overlap of Station CA

10-20 CI Elevation and X/Y-Profile

20-40 CI Elevation and X/Y-Profile

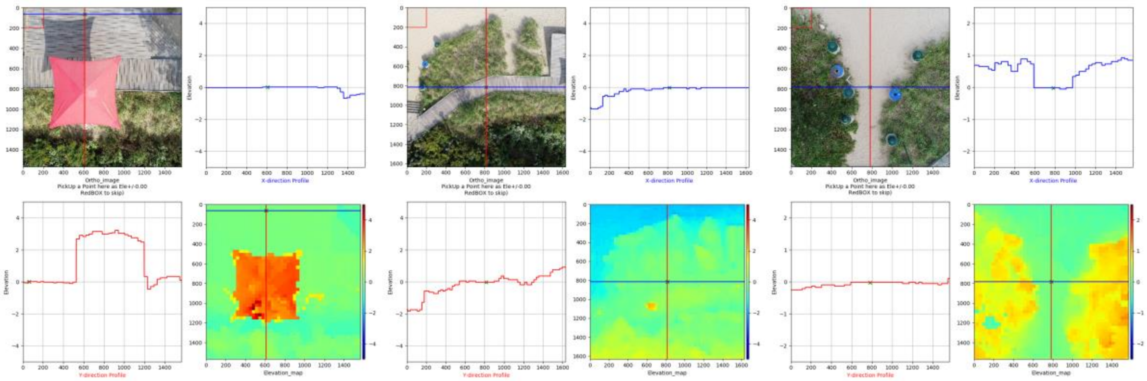
Overlap of Station CI

* ortho-image shown in RGB color; the blue line is the x-profile, unit : m; red line is the y-profile, unit : m; elevation data shown in jet colormap, unit : m; 10-20 CA, 20-40 CA, 10-20CI and 20-40 CI were aligned to image center as ± 0.00.

*red line 10-20, unit: m; green line 20-40, unit: m.

Figure 38 Elevation results 1

The experimental results show that the developed method is valid in flat central regions such as CA station and CI station shown in Figure 38, and also works perfectly in the concavo-convex central regions (see Figure 39). Furthermore, the developed method can handle steep and near vertical topography such as the vertical side of the garbage can in CJ station, the edge of the rest area in CA station, the umbrella in CA and CG stations, and the stairways in CI station. This is better than traditional drone photogrammetry using ortho-image.



10-20 CG Elevation and X/Y-Profile 20-40 CH Elevation and X/Y-Profile 10-20 CJ Elevation and X/Y-Profile

*ortho-image shown in RGB color; the blue line is the x-profile, unit : m; red line is the y-profile, unit : m; elevation data shown in jet colormap, unit : m; 10-20 CG was aligned to a point on the rest area; 20-40CH and 10-20 CJ were aligned to image center as ± 0.00.

Figure 39 Elevation results 2

4.5.2.2 Elevation Determination Evaluation

The overlapped X/Y-Profile of 10-20 pairs and 20-40 pairs at stations CI and CA are matched at most parts in Figure 38. As the 20-40 pairs' GSD and $Depth_{step}$ are twice that of the 10-20 pairs' (see Table 9), it is reasonable to have more detailed elevation variations such as edges, salient pole and concave pole in the lower altitude ortho-image pairs. Furthermore, for the common objects in different low-high ortho-image pairs, the developed method generated quite accurate elevation results.

The measured elevation differentials between the selected points are compared with the true elevation differentials in Table 11. The measurement differences are [-4.36, 4.86] cm for the 10-20 ortho-image pairs, and [-2.39, 2.76] cm for the 20-40 ortho-image pairs, which are satisfied with 5.00 cm error standard (Takahashi et al. 2017). Therefore, the developed method is robust at different camera stations with different altitudes.

Table 11 Elevation Measurement

Point	*Elevation-map*	*Elevation Coordinate (m)*	*Measured Differential (cm)*	*True Differential (cm)*	*Elevation Differential (cm)*	*Elevation Error (cm)*
A	10-20 CJ	A(0.8039)-B(0.00)	80.39	81.28	-0.89	0.89
	10-20 CI	C(0.00)-A(-0.1765)	17.65	17.78	-0.13	0.13
B	10-20 CI	C(0.00)-B(-1.0392)	103.92	99.06	4.86	4.86
	20-40 CI	C(0.00)-B(-0.9804)	98.04		-1.02	1.02
D	20-40 CI	D(3.6471)-C(0.00)	364.71	361.95	2.76	2.76
E	10-20 CA	G(0.00)-E(-1.0784)	107.84	106.68	1.16	1.16
	20-40 CA	G(0.00)-E(-1.0588)	105.88		-0.8	0.8
F	10-20 CG	F(3.1568)-G(0.00)	315.68	320.04	-4.36	4.36
	20-40 CA	F(3.1765)-G(0.00)	317.65		-2.39	2.39

4.5.2.3 Elevation Determination Discussion

The disassembled discrete virtual elevation plane result for the selected points are compared in Table 12. Based on the virtual plane model (see Figure 30) and the pixel matching and virtual elevation algorithm (see Figure 31), the matched result should fall within a three-virtual-plane-range, within the interval $[-Depth\_Step, Depth\_Step\ ]$. In this chapter, the experimental site was broken into 200 major virtual planes in the range of [-5,5] m for 10-20 ortho-image pairs and [-10,10] m for 20-40 ortho-image pairs. The designed interval between two major virtual planes are 5.00 cm and 10.00 cm for 10-20 and 20-40 ortho-image pairs respectively. 3 of 9 experimental results fell into the lower interval $[-Depth\_Step, 0]$, and 6 of 9 fell into the upper interval $[0,Depth\_Step]$; they are all matched within the expected discrete virtual planes based on the true elevation data. In other words, the developed virtual elevation algorithms are sensitive to major plane changes. Therefore, the matched pixel grids from the low-high ortho-image pairs contain the correct elevation data.

Table 12 Virtual Elevation Evaluation

Point	*Elevation-map*	*Depth Step (cm)*	*Experimental Results*		*Discrete Virtual Plane Based on Ture Elevation*					*Comparison*	
			Elevation Coordinate (m)	*Ele. /Step*	*Elevation Coordinate (m)*	*Ele. /Step*	*Lower Plane*	*Virtual Plane*	*Upper Plane*	*Fall in lower interval (<plane)*	*Fall in upper interval (>plane)*
A	10-20 CJ	5	0.8039	16.078	0.8128	16.26	15	16	17		Yes
	10-20 CI	5	-0.1765	-3.53	-0.1778	-3.56	-5	-4	-3		Yes
B	10-20 CI	5	-1.0392	-20.784	-0.9906	-19.81	-21	-20	-19	Yes	
	20-40 CI	10	-0.9804	-9.804	-0.9906	-9.91	-11	-10	-9		Yes
D	20-40 CI	10	3.6471	36.471	3.6195	36.20	35	36	37		Yes
E	10-20 CA	5	-1.0784	-21.568	-1.0668	-21.34	-23	-22	-21		Yes
	20-40 CA	10	-1.0588	-10.588	-1.0668	-10.67	-12	-11	-10		Yes
F	10-20 CG	5	3.1568	63.136	3.2004	64.01	63	64	65	Yes	
	20-40 CA	10	3.1765	31.765	3.2004	32.00	31	32	33	Yes	

CHAPTER 5: ORTHO-IMAGE AND ELEVATION-MAP DATASET DESIGN AND ACQUISITION USING DRONE

5.1 Introduction

Image and deep learning based methods have been applied to determine the relative depth information for each pixel of an image of indoor scenes (Eigen et al 2014; Liu et al. 2015; Laina et al. 2016), outdoor scenes (Chen et al. 2016; Li and Snavely 2018) and scenes from automatic driving applications (Garg et al. 2016). The main challenge of training a deep learning model is acquiring the suitable dataset. In the application of determining construction site elevation, the elevation data can be acquired by either contact or non-contact methods discussed in the literature review, but the challenge is linking the ortho-image's pixels with the elevation value in this same coordinate.

The developed low-high ortho-image pair-based elevation determination method in Chapter 4 provides the possible approach that stores the elevation value in an equal size 8-bit grayscale image, referred to as an elevation-map, which uses 0 as the elevation lower boundary and 255 as the elevation upper boundary. In Figure 40, the elevation-map is represented in viridis colormap for better visualization, and the X/Y-profiles show the elevation changes at the selected point (elevation unit: m). Acquiring elevation data for each pixel of the ortho-image is unreasonable. To save time, Chapter 4 also provides the grid pixel formation to simplify and share the same grayscale value / elevation value for a patch, such as a 32×32-pixel patch. For example, the 1st selected $pixel\ (16,16)$ shares its grayscale value / elevation value with the patch $Elevation\_map[0{:}31,0{:}31]$. Therefore, this chapter summarized the findings in using the developed method to setup the high-resolution ortho-images and elevation-maps dataset.

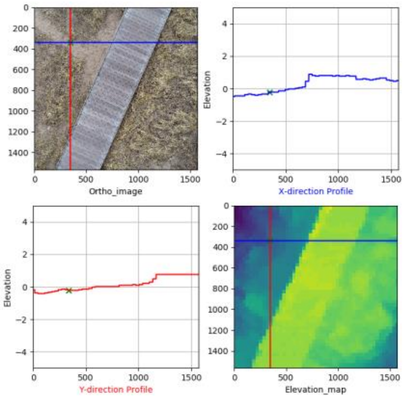


Figure 40 Ortho-image, elevation-map and X/Y-profiles (w/o alignment)

5.2 Ortho-image and Elevation-map Dataset Design

5.2.1 Ortho-Image Formation

The developed method in Chapter 4 will capture an *H-H/2* ortho-image pair at two flight altitude positions. Such can be done, for example, as the low-height ortho-images have *Image Size*=3648×4864-pixel, *GSD*=0.27 cm/pixel, *Site Size*=9.85×13.13 m^2 with *H/2*=10 m; and the high-height ortho-images *GSD*=0.54 cm/pixel, *Space Size*=19.70×26.26 m^2 with *H*=20 m.

After the processes of the developed elevation determination method in Chapter 4, the expected output ortho-image is transformed (see Table 13) from the low-height ortho-image with the *Image Size*=1568×1568-pixel, *GSD*=0.54 cm/pixel, *Site Size*=8.47×8.47 m^2, which is referred to as high-resolution 24-bit RGB ortho-image in this research project.

Table 13 Image Processing Parameters with 10 m Altitude

Processing Step	***Image Size***	***GSD***	***Site Size***
Original	3648×4864-pixel	0.27 cm/pixel	9.85 x 13.13 m^2
Cutting to square shape	3648×3648-pixel	0.27 cm/pixel	9.85 x 9.85 m^2
Scaling, 0.5	1824×1824-pixel	0.54 cm/pixel	9.85 x 9.85 m^2
Removing margin, 128 pixels	1568×1568-pixel	0.54 cm/pixel	8.47 x 8.47 m^2

5.2.2 Elevation-Map Formation

Same as the high-resolution ortho-image, the high-resolution 8-bit grayscale elevation-map also has the *Image Size*=1568×1568-pixel, *GSD*=0.54 cm/pixel, *Site Size*=8.47×8.47 m^2 with *H/2*=10 m. After the processes of the proposed elevation determination method, each pixel of the generated elevation-map has the grayscale value ranges from 0 to 255 to represent the elevation value from $[-H/4, H/4]$, which is [-5, 5] m, and the transformation equation is Eq. 10, as the *H/2* is set as 10 m.

$$Elevation@(u,v) = Elevation_{map[u,v]} \times \frac{H/2}{255} - \frac{H}{4} = Elevation_{map[u,v]} \times \frac{10m}{255} - 5m \quad \textbf{Eq. 10}$$

In addition, pixel grid formation is set as 32×32-pixel patch sharing the same grayscale value / elevation value. For example, the 1st selected $pixel\ (16{,}16)$ shares its grayscale value / elevation value with the patch $Elevation\_map[0{:}\,31{,}0{:}\,31]$. Furthermore, the elevation-map will be aligned to the ortho-image center as the elevation origin.

5.3 Ortho-image and Elevation-map Acquisition Configuration

5.3.1 Drone Flight Path Design

5.3.1.1 Drone Fight Height

The developed method in Chapter 4 has an adjustable measuring space range, which depends on the drone flight altitude and camera parameters. Raising the drone's altitude will increase the ortho-image pair's coverage, which is better for getting the overall construction site topography. On the other hand, to get detailed structures' shapes, it is better to use a lower altitude ortho-image pair, which use more pixels to represent the small objects. The author recommends using a DJI Phantom 4 Pro V2.0 , which can give an $8.47\times8.47\ \mathrm{m}^2$ coverage in 10-20 ortho-image pair and $17.6\times17.6\ \mathrm{m}^2$ coverage in 20-40 ortho-image pair

5.3.1.2 Drone Fight Path

Where the construction site is larger than a single image frame, such as roadway projects, a series of ortho-image pairs can be captured through a serpentine style path (see Figure 41).

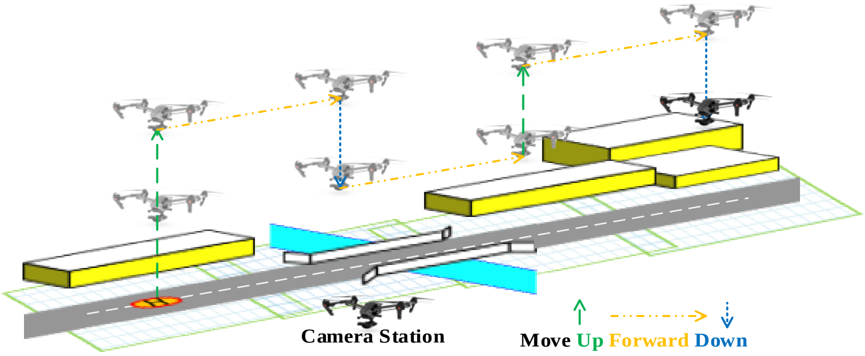


Figure 41 Serpentine style drone path for roadway construction project

In detail, the drone is planned to takeoff and reach the desired altitudes $H/2$ and H to capture the low-high ortho-image pair at the takeoff station. After the drone finishes the high ortho-image capture at altitude H, it moves forward to the next station where the distance between two stations should make the adjacent low ortho-images have enough overlap for image stitching. The drone takes high ortho-image at altitude H at the 2nd location first, then it moves downward to capture the second low ortho-image at the 2nd location after it reached to the desired altitude $H/2$. After that, the drone will continue movies forward and repeat the previous steps until it acquires enough low-high ortho-image pairs to cover the entire construction site. This designed path will guarantee that each low-high ortho-image pair has the same center. Furthermore, it is convenient to add and modify some ortho-image pairs beyond the acquisitions with the designed flight path.

5.3.1.3 Ortho-image Pair Capturing

It is important to avoid the drone’s shift and rotation as much as possible during the vertical moving and capturing of the low-high ortho-image pair at each station. In this research project, the pre-processing steps of image rotation and image translation are based on the SIFT keypoints. The high ortho-images are rotated in the range of [-2.862, 0.321] degrees, which have the minimum absolute rotation in 10-20 CA with 0.260 degrees, and the maximum absolute rotation in 10-20 CG with -2.862 degrees. The high ortho-images translated in the range of [-3.99, 13.02] pixels in x-direction and [-22.57, 13.16] pixels in y-direction, which have the minimum absolute translation in 20-40 CI with x/y-direction translation [1.83,1.30] pixels, and the maximum absolute translation in 10-20 CA with [13.02, 13.16] pixels and 10-20 CG with [-1.85, -22.57] pixels.

Table 14 shows the correlations between the absolute rotation degree and translation distance with the number of weakest pixels in Table 10. Based on the correlation results, the X-direction translation (in image width) has a significant positive correlation with the pixel matching quality. The Y-direction translation (in image height) and the rotation have no significant correlation with the matching quality. The maximum X-direction translation occurred in 10-20 CA, which has the largest number of weakest pixels. Therefore, minimizing the image width direction shift is most important in acquiring the best low-high ortho-image pair at each camera station.

Table 14 Pairwise Pearson Correlations

Sample 1	*Sample 2*	*Correlation*	*95% CI for ρ*	*P-Value*
Num. of Weakest Matching	X-translation Distance	0.764	(0.026, 0.963)	0.046
Num. of Weakest Matching	Y-translation Distance	-0.062	(-0.779, 0.725)	0.895
Num. of Weakest Matching	Absolute Rotation Degree	-0.498	(-0.910, 0.409)	0.256
Num. of Weakest Matching	Translation Distance $\sqrt{X^2+Y^2}$	0.144	(-0.683, 0.809)	0.759
Num. of Weakest Matching	Rotation × X	-0.142	(-0.808, 0.684)	0.762
Num. of Weakest Matching	Rotation × Y	-0.395	(-0.885, 0.510)	0.381
Num. of Weakest Matching	Rotation × Distance	-0.391	(-0.884, 0.513)	0.386

5.3.2 Pixel Grid Matching and Elevation Determination

5.3.2.1 Pixel Grid Configuration

The algorithm parameters configuration in Table 9 should be adapted for the elevation determination on a construction site. The proposed pixel grid formation simplifies the ortho-image pairs' matching. Reducing $Grid\ Size$ can generate more detailed results while also raising the computing time. But the additional computing cost in matching all pixels ($Grid\ Size$ =1) gives no additional benefits from the extra dense pixel grid.

To save time and avoid wasting computing resources, it is better to add an early stop function to the pixel matching and virtual elevation algorithm (Figure 31). If $NCC_{current\,+} < \alpha\, NCC_{max}$ then stop matching $Depth_{current\,+}$. If $NCC_{current-} < \alpha\, NCC_{max}$.then stop matching $Depth_{current-}$. The author recommends using α = 0.7, which balances the computing time and accuracy. The matching time is also impacted by the site shape. A relatively flat site takes less time than one with a lot of elevation changes. The tested matchings of 11449-pixel grid ($Grid\ Size$ =16) in 20-40 ortho-image pairs take slightly longer than 12 minutes, with the experimental computer configuration of Python 3.6.8, Intel® Xeon® Gold 5122 CPU@3.6 GHz. The computing time for 2500-pixel grid ($Grid\ Size$ =32) in 10-20 ortho-image pairs and 4761-pixel grid ($Grid\ Size$ =24) in 20-40 ortho-image pairs are around 2 to 5 minutes, which are the recommended pixel grid configuration.

5.3.2.2 Elevation Determination Configuration

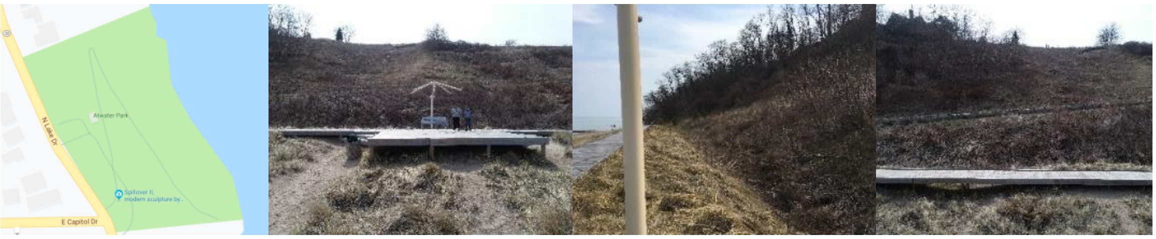
The configuration of 200 major planes and 1000 minor planes balanced the accuracy and computing time. Table 12 shows 3 of 9 experimental results fell into the lower interval $[-Depth\_Step, 0]$, and 6 of 9 fell into the upper interval $[0,Depth\_Step]$; they are all matched within the expected discrete virtual planes based on the true elevation data. The $Threshold_{low}$ (line 14 in Figure 31) was set as 0.4 in this research. The author recommended range is [0.3,0.7]. Raising it can improve the matching accuracy in poorly textured ortho-image pairs but can result in errors as well. The noise points in 10-20 CA (see Figure 38) were matched on the wrong virtual planes. Additionally, the distance s (in lines 7,8,9, and 10 in Figure 33) was set as $Grid\ Size/2$ to balance the smoothing of the elevation-map and retaining detailed of elevation changes.

Additionally, as the drone's shift and rotation are unable to be totally eliminated, the image center region, in Table 9, is an important parameter. Pixels in this region are limited to matching within the upper and lower adjacent virtual planes $[Depth_{guess} - Depth_{Step}, Depth_{guess} + Depth_{Step}]$. This setting can help avoid incorrect elevation results in the center region pixels and make their elevation results close to their surroundings. Otherwise, due to the reason discussed in section 4.3.3.3, incorrect matching will happen there. In addition, this research also applied this setting for pixels on $y\text{-}axis\,(x \le Grid\ Size)$. With the repeated matching in pixel grid and elevation-map enhancement algorithm (line 3 in Figure 34), the noise points on both the $X\text{-}axis$ and $Y\text{-}axis$ are reduced. The author recommends using a circular center region with radius=192 pixels, shown as the red regions in Figure 26.

5.4 Ortho-image and Elevation-map Dataset Setup

5.4.1 Construction Site

The high-resolution ortho-images and elevation-maps datasets are set up to train and test the proposed deep learning-based method in this research. The selected construction site is a lake beach (Atwater Park, Shorewood, WI, USA), which includes a stairway, boardwalk with rest area, several garbage cans and vegetations (see Figure 35, Figure 42,Figure 56, Figure 80). The ortho-images were captured in this site from March 2019 to September 2019. Thus, different vegetation growing situations occurred in the dataset.



Site Location Rest Area Ditch Boardwalk

Figure 42 Ortho-image and elevation-map acquiring site

5.4.2 Ortho-image and Elevation-map Dataset on Different Seasons

In Figure 43, Figure 44, Figure 45, the 1st and 2nd column ortho-images were taken in Atwater Park (Shorewood, WI, USA) during different seasons. In detail, a) Data A and B were taken on 3/24/2019, when the vegetation had not recovered yet; b) Data C and D were taken on 6/5/2019, when the vegetation was growing; c) Data AC, AO, CA were taken on September 2019, when the vegetation was fully grown; and d) Data B, D, AC and AO have the same wooden platform, which is different to Data CA. The 3rd column high-resolution 24-bit RGB ortho-image has the *Image Size*=1568×1568-pixel, *GSD*=0.54 cm/pixel, *Site Size*=8.47×8.47 m^2 with *H/2*=10 m. The 4th column high-resolution 8-bit grayscale elevation-map also has the *Image Size*=1568×1568-pixel, *GSD*=0.54 cm/pixel, *Site Size*=8.47×8.47 m^2. After the processes of the developed elevation determination method, each pixel of the generated elevation-map has the grayscale value ranges from 0 to 255 to represent the elevation value from $[-5\,m, 5\,m]$.

5.4.2.1 Spring Season

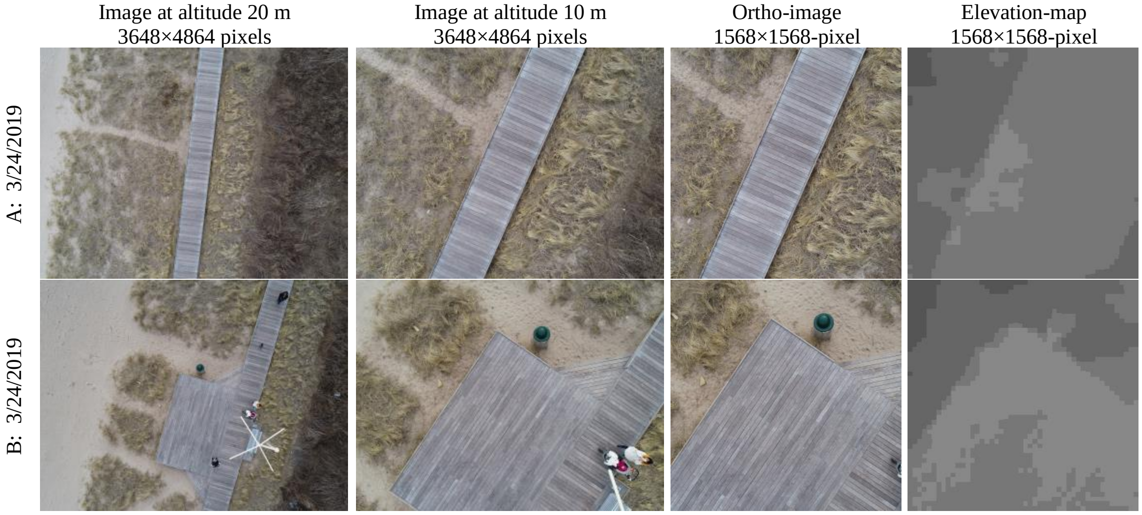


Figure 43 Spring season dataset

5.4.2.2 Summer Season

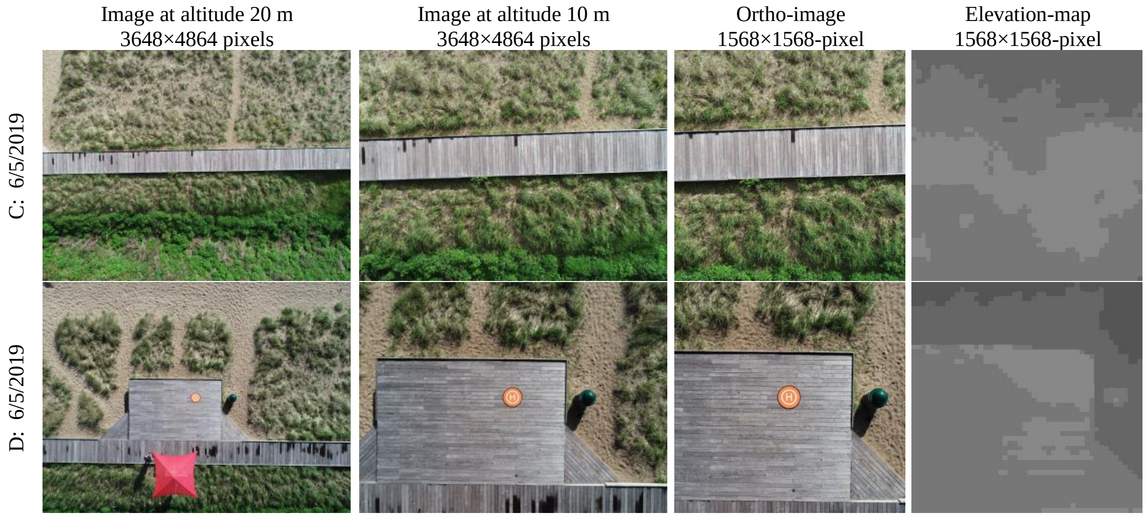


Figure 44 Summer season dataset

5.4.2.3 Fall Season

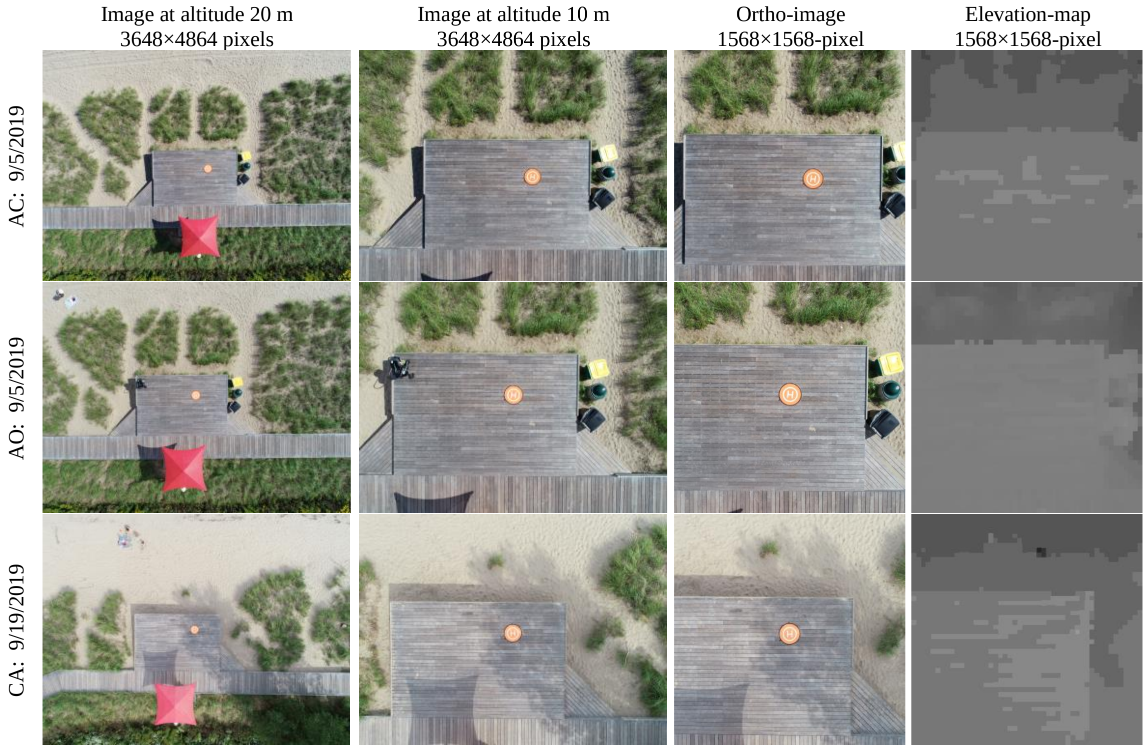


Figure 45 Fall season dataset

5.4.3 Ortho-image and Elevation-map Dataset on Detailed Objects

In Figure 46, the 1st and 2nd column ortho-images were taken in Atwater Park (Shorewood, WI, USA). In detail, data CG detail the umbrella, CI detail stairways and CJ detail garbage cans.

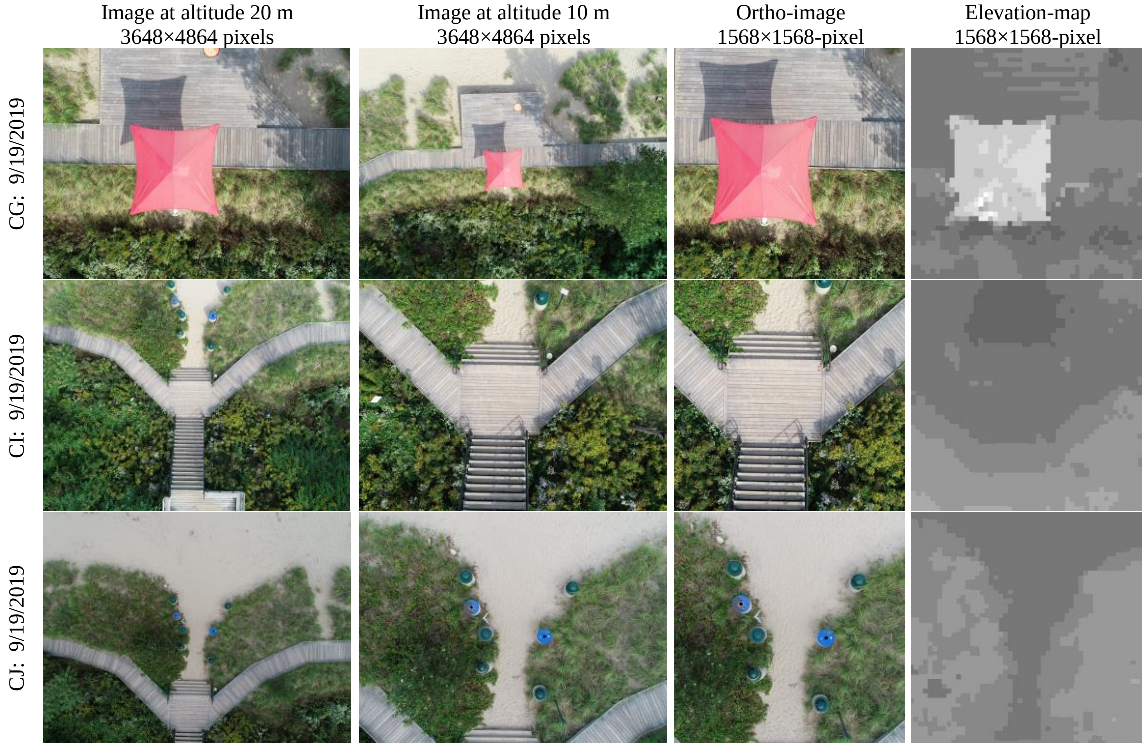


Figure 46 Detailed objects dataset

5.4.4 Ortho-image and Elevation-map Dataset on 20-40 m

In Figure 47, the 1st and 2nd column ortho-images were taken in Atwater Park (Shorewood, WI, USA). The 3rd column high-resolution 24-bit RGB ortho-image has the *Image Size*=1632×1632-pixel, *GSD*=1.08 cm/pixel, *Site Size*=17.6×17.6 m^2 with *H/2*=20 m. The 4th column high-resolution 8-bit grayscale elevation-map also has the *Image Size*=1632×1632-pixel, *GSD*=1.08 cm/pixel, *Site Size*=17.6×17.6 m^2. After the processes of the developed elevation determination method, each pixel of the generated elevation-map has the grayscale value ranges from 0 to 255 to represent the elevation value from $[-10\,m, 10\,m]$.

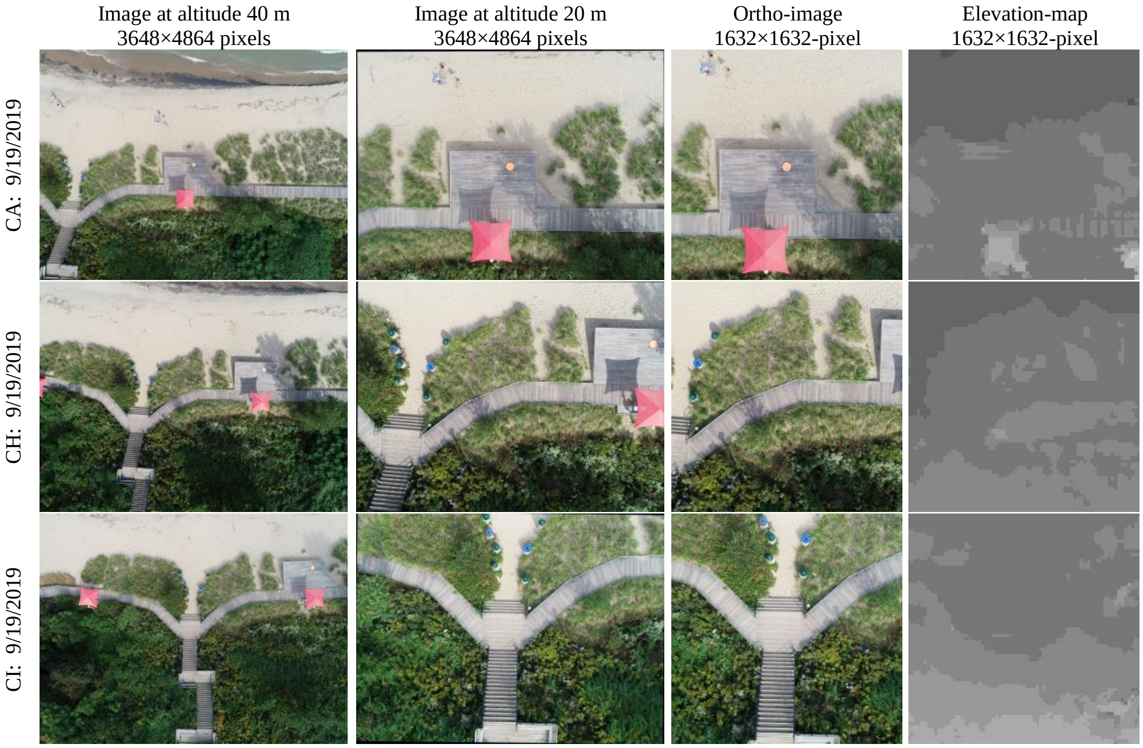
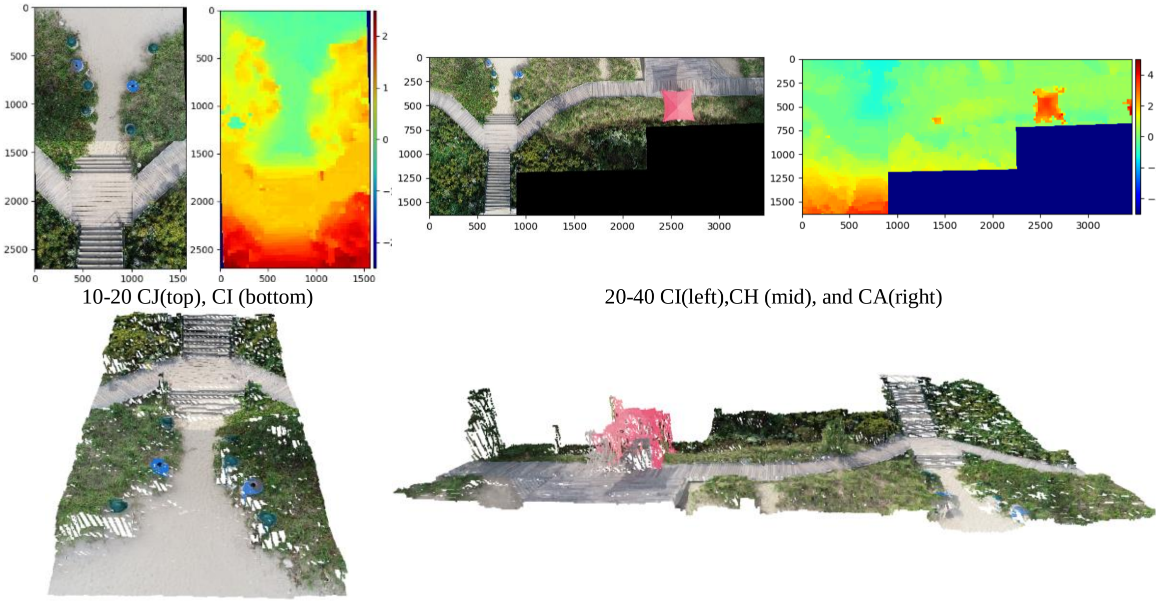


Figure 47 20-40m dataset

5.5 Ortho-image and Elevation-map Stitching and Discussion

5.5.1 Stitching Results

As elevation data are saved in the grayscale elevation-map format (see Figure 40), it is convenient to stitch adjacent elevation-maps into a larger elevation-map by simply selecting two corresponding points in their associated ortho-images as the boundary and aligning the elevation data at the selected boundary. Figure 48 shows the results of up-down stitching and left-right stitching. Although the 10-20 CJ and 10-20 CI ortho-images have different exposure values, the combined elevation-maps are smooth at their junctions. The accuracy of the developed method was not impacted by the brightness of the environments.



10-20 CJ(top), CI (bottom)

20-40 CI(left),CH (mid), and CA(right)

10-20 CJ-CI Point Cloud

20-40 CI-CH-CA Point Cloud

*elevation, unit: m; each point is corresponded to a pixel in the ortho-image (R,G,B) and elevation-map (Z).

Figure 48 Elevation-map stitching results

5.5.2 Stitching Evaluation

In Figure 48 , point clouds were converted using each pixel of the stitched elevation-maps by Eq. 9. The overall shape of the experiment site was well reconstructed. The small objects, such as the single tree on 20-40 CH, were also well reconstructed. The side points of vertical surfaces are missed because the developed method only used top-view ortho-images, and the missed side points have no impact on determining the elevations of a construction site. When the drone flew at 10 m, some small objects' side surfaces were recorded in the ortho-image, such as the garbage can in ortho-image 10 CI, because the reflected rays converged through the camera lens instead of passing parallel into the lens. Enlarging the altitude or flying the drone over these objects can eliminate this kind of effect, and the horizontal position of a point on the vertical side surfaces can be corrected by Eq. 7 if necessary. What's more, these small and easily removeable objects have limited impacts on elevation determination.

5.5.3 Stitching Discussion

There are noticeable elevation errors on the edge of the red umbrella on 20-40 CA, where the pixel pairs were weakly matched as pink and blue dots in Figure 37. This is different to the single tree, as its pixel pairs were all strongest matched as green dots (see Figure 37). The state-of-the-art SIFT method is invalid there, as no keypoint was matched in Figure 37. However, there are several approaches that can be used to fix this issue:

1. Lower down the drone flight altitude to 10-20 m for capturing more detail, such as 10-20 CG in Figure 37.
2. Decrease *Grid Size* for dense matching more pixel pairs and smoothing vertical changes, which is the same as low altitude for using more pixels to represent an object.
3. Remove weakly matched pixels, then use the PMVS method for dense reconstruction (Furukawa and Ponce 2010).
4. Fix it with additional processes, such as using a convolutional neural network first to distinguish the umbrella surface from other surfaces, then assign the correct elevation values to each of them. This will be discussed at sections 6.4.2.2 and 6.5.4.

CHAPTER 6: ORTHO-IMAGE AND DEEP LEARNING-BASED ELEVATION ESTIMATION ALGORITHM DESIGN AND TESTING

6.1 Introduction

Chapter 4 presents a two-frame-image-based 3D-reconstruction method, which can automatically generate the output as an ortho-image and elevation-map pair. In the case of automatic driving, the forward-facing view has the camera's principal ray perpendicular to the objects in front of the car. So, the images captured in front of automatic driving cars and above construction site surfaces have the common characteristic in that the objects in the same depth level / elevation level have common texture features in the forward-facing view / ortho-image. Therefore, capturing an ortho-image over a construction site by drone, then using this image to estimate the site elevations, is a feasible approach, which will reduce drone flying time and avoid hazards of drone crashes in the construction site.

In this chapter, a deep learning based-method, convolutional encoder-decoder network model, is proposed to estimate elevations from the ortho-images of a construction site, which links each pixel of the ortho-image with the same coordinate pixel of an elevation-map (see Figure 49). This chapter also evaluates the effectiveness of the single-image-frame-based 3D-reonstruction method, which requires much fewer images in estimating elevation than the developed low-high ortho-image pair method in Chapter 4. To explain how to estimate site elevations from a single-frame drone-based ortho-image, the rest of this chapter presents the dataset acquisition, model designs, training and testing, field experiments and result discussions.

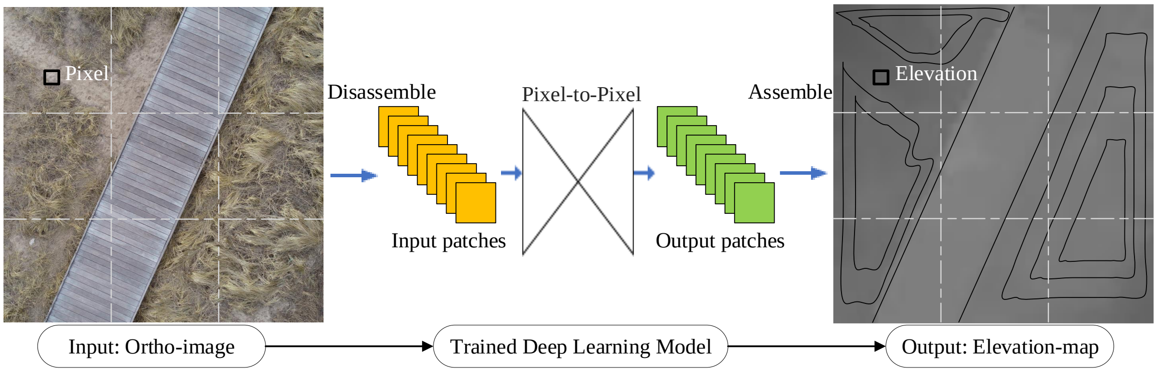


Figure 49 Workflows of the single ortho-image based method

6.2 Elevation Estimation Dataset Creation

6.2.1 Patch Size and Number

Considering the computing capacity of the workstation system, in Figure 50 the 1st to 5th columns list the possible model input and output small-patch examples of 32×32-pixel, 64×64-pixel, 128×128-pixel, 256×256-pixel, and 512×512-pixel, which are cropped from [0:31,0:31], [0:63,0:63], [0:127,0:127], [0:255,0:255], and [0:511,0:511] of the high-resolution 1536×1536-pxiel ortho-image and elevation-map (the 6th column) respectively. For the elevation-map small-patches, each larger patch contains four times more elevation values than the smaller patch. For example, the 64×64-pixel small-patch contains elevation values from $pixel\,(16,16)$, $pixel\,(16,48)$, $pixel\,(48,16)$ and $pixel\,(48,48)$, while the 32×32-pixel patch only contains the elevation value from $pixel(16,16)$. Thus, a smaller patch size is better for the deep learning model to learn the local features from the input and output dataset. On the other hand, a larger patch size is better for learning the global features from the input and output dataset.

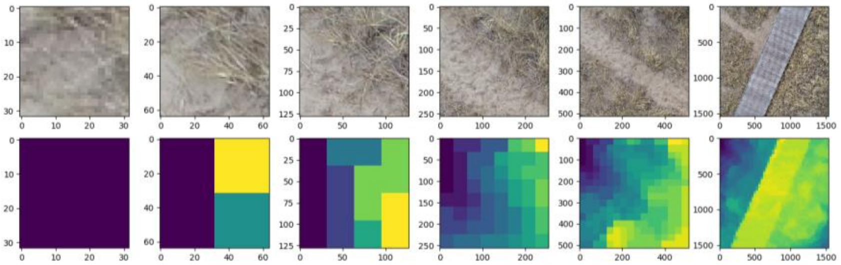


Figure 50 Example of input dataset and output dataset

In addition, when creating these overlapping 32×32-pixel, 64×64-pixel, 128×128-pixel, 256×256-pixel, and 512×512-pixel patches, the stride is set as 16, 32, 64 or 96 pixels for moving these square boxes on the ortho-image and elevation-map (larger strides are used to avoid workstation system memory shortages) and the number of patches can be determined by Eq. 11, where "⌊ ⌋" is the floor function. Moreover, in order to make the deep learning model robust in different image orientations, the ortho-image and elevation-map are planned to rotate 90, 180 and 270 degrees to increase the dataset by four times. Table 15 lists the detailed parameters in creating the small-patch datasets from a high-resolution ortho-image and elevation-map pair with size 1536×1536-pixel.

$$Num. of\ Dataset = \lfloor\frac{Image\ Height - \ Patch\ Size}{Stride} + 1\rfloor \times \lfloor\frac{Image\ Width\ \ - Patch\ Size}{Stride} + 1\rfloor \times 4 \qquad \textbf{Eq. 11}$$

Table 15 Dataset Parameters

Patch Sizes	***Strides***	***Rows***	***Columns***	***Num.***	***Num. after 4-rotation***
32x32	16	95	95	9025	36100
64x64	32	47	47	2209	8836
128x128	32	45	45	2025	8100
256x256	64	21	21	441	1764
512x512	96	11	11	121	484

6.2.2 Dataset shape

An ortho-image acquired by the drone has RGB 3-channel. Considering that the color textures are important in distinguishing different objects on the construction site, the texture information is kept rather than using a grayscale image. Therefore, using a high-resolution ortho-image can produce the model training datasets with $shape\ (36100{,}32{,}32{,}3)$, $shape\ (8836{,}64{,}64{,}3)$, $shape\ (8100{,}128{,}128{,}3)$, $shape\ (1764{,}256{,}256{,}3)$, or $shape\ (484{,}512{,}512{,}3)$, where the first number is the quantity of the small-patches.

The elevation-map generated from the low-high ortho-image pair-based method only has one channel. Disassembling a high-resolution elevation-map can produce the small-patch datasets with $shape\ (36100{,}32{,}32{,}1)$, $shape\ (8836{,}64{,}64{,}1)$, $shape\ (8100{,}128{,}128{,}1)$, $shape\ (1764{,}256{,}256{,}1)$, or $shape\ (484{,}512{,}512{,}1)$.

6.3 Elevation Estimation Algorithm Design

6.3.1 Elevation Estimation Deep Learning Model Architecture

6.3.1.1 Convolutional Encoder-decoder Architecture Design

In this research project, the proposed deep learning model is a convolutional encoder-decoder network model (see Figure 51), which has an equal number of max pooling layers and up sampling layers. This type of model is referred to as an “hourglass-like” model, which has been widely used in image segmentation, such as SegNet (Badrinarayanan et al. 2017). Another “hourglass-like” model uses deconvolution network in the decoder, such as DeconvNet (Noh et al. 2015), where each deconvolution (also known as transposed convolution) layer is the opposite operation of normal convolution (Chollet

2015). During this research project, the convolutional decoder and deconvolutional decoder were compared and their generated results do not have any significant difference. What's more, the proposed model is different from SegNet, in which the up sampling layer is the first layer in the decoder, but the proposed model uses a convolution layer first (see Figure 51).

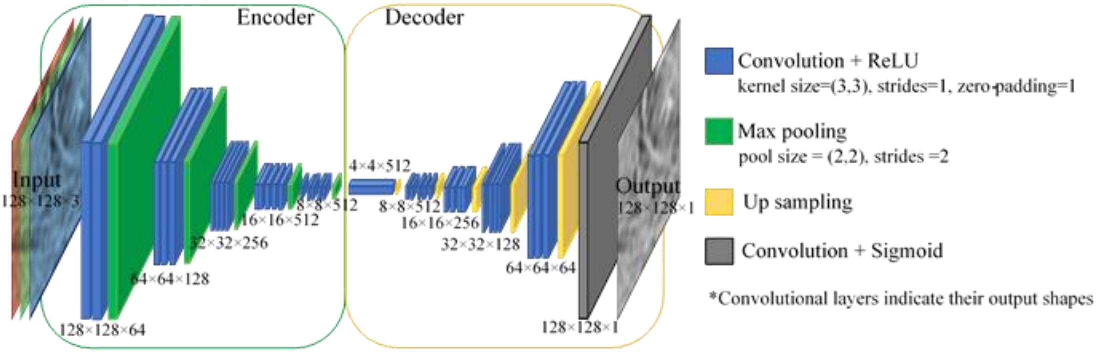
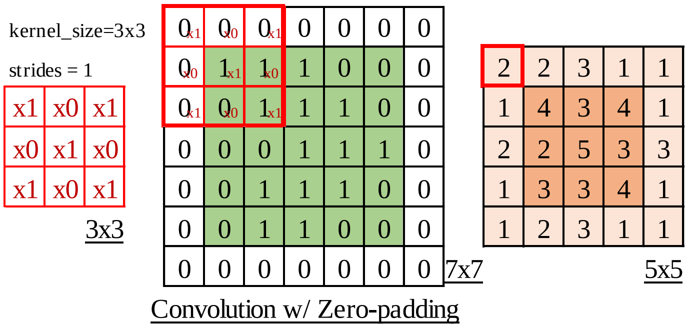


Figure 51 Proposed encoder-decoder model with 128×128-pixel patch

In the encoder block, the five convolution layers learn the model input ortho-image patch as feature-maps; each convolution layer contains a 2D convolution operation with zero-padding (see Figure 52), and the layer output has the same size as the layer input (Chollet 2015). In detail, Figure 52 shows an example of a zero-padded convolution operation. The original input is 5×5 in size, which has been padded to 7×7; the 3×3 kernel convolution has a 5×5 output, which has the same size as the original input. If the original input is not padded with zero, the convolution output is the filled 3×3 region only. Additionally, each max pooling layer next to the convolution layer is a max pooling operation (see Figure 53), which reduces the layer input (convolution layer output) to half size as the layer output. For example, Figure 53 shows an example of how max pooling (2×2 filter and strides =2)



x1	x0	x1
x0	x1	x0
x1	x0	x1

0 x1	0 x0	0 x1	0	0	0	0
0 x0	1 x1	1 x0	1	0	0	0
0 x1	0 x0	1 x1	1	1	0	0
0	0	0	1	1	1	0
0	0	1	1	1	0	0
0	0	1	1	0	0	0
0	0	0	0	0	0	0

2	2	3	1	1
1	4	3	4	1
2	2	5	3	3
1	3	3	4	1
1	2	3	1	1

Figure 52 Example of convolution operation with zero-padding

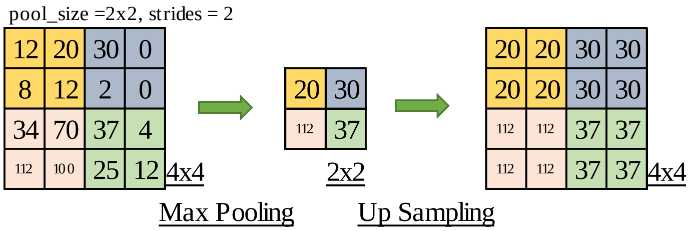


Figure 53 Examples of max pooling and up sampling operations

In the decoder block, the five convolution layers interpret the feature-maps to model an elevation-map output; each convolution layer contains a 2D convolution operation with zero-padding as well; the up sampling layers are the reverse operations of max pooling operations, which enlarge the layer input to its double size as the layer output. Figure 53 shows an example of how up sampling work in the model.

6.3.1.2 Convolutional Encoder-decoder Model Layers Setup

To make this encoder-decoder model able to interpret an ortho-image patch and predict an equivalent size elevation-map patch, the intersection part of the encoder-decoder is proposed as a 512-channel feature-map, which is generated from the “max_pooling2d_5” layer (see Table 16). For example, the encoder generates a 4×4×512 feature-map for the 128×128×3 input (see Figure 51). This intermedia feature-map is required by the model output. Based on the dataset creation, each elevation-map shares a common integer from 0 to 255 (8-bit grayscale value) in every 32×32 patch, thus a 256-channel feature-map with size 4×4 is required for the decoder to generate the 128×128×1 output. That can be explained as each channel is the probability of the integer from 0 to 255. As a 128×128 elevation-map patch contains 16 (4×4) elevation values, thus at least a 4×4×256 feature-map is required for the decoder. The proposed “conv2d_5” layer uses 512 filters (see Table 16), which doubled the required channel number. The 512-channel feature-map can be understood as each channel is the probability of the element in list [0.0, 0.5, 1.0, …, 245.5, 255.0]. What’s more, in this research project, adding the interaction feature-map to 1024-channel had no difference from the 512-channel. In addition, 5 max-pooling-layer is the maximum number for the encoder because the smallest model input 32×32-pixel patch is transformed to 1×1-pixel feature-map after 5 max pooling operations.

Table 16 Model Layers Parameters

Model Architecture for 32×32,64×64,128×128,256×256 and 512×512-pixel Patch							Output Shapes for Each Patch					
Blocks	Layers (Type and kernel size)	Strides	Padding	Activations	Filters /Channels	Parameters Number	32	64	128	256	512	Channels
								Rows/Columns				
Input	input_1 (Input Layer)	-	-	-	3	0	32	64	128	256	512	3
Encoder	conv2d_1 (Conv2D 3x3)	1	same	ReLU	64	1792	32	64	128	256	512	64
	max_pooling2d_1 (Max Pooling 2x2)	2	same	-	-	0	16	32	64	128	256	64
	conv2d_2 (Conv2D 3x3)	1	same	ReLU	128	73856	16	32	64	128	256	128
	max_pooling2d_2 (Max Pooling 2x2)	2	same	-	-	0	8	16	32	64	128	128
	conv2d_3 (Conv2D 3x3)	1	same	ReLU	256	295168	8	16	32	64	128	256
	max_pooling2d_3 (Max Pooling 2x2)	2	same	-	-	0	4	8	16	32	64	256
	conv2d_4 (Conv2D 3x3)	1	same	ReLU	512	1180160	4	8	16	32	64	512
	max_pooling2d_4 (Max Pooling 2x2)	2	same	-	-	0	2	4	8	16	32	512
	conv2d_5 (Conv2D 3x3)	1	same	ReLU	512	2359808	2	4	8	16	32	512
	max_pooling2d_5 (Max Pooling 2x2)	2	same	-	-	0	1	2	4	8	16	512
Decoder	conv2d_6 (Conv2D 3x3)	1	same	ReLU	512	2359808	1	2	4	8	16	512
	up_sampling2d_1 (Up Sampling 2x2)	1	-	-	-	0	2	4	8	16	32	512
	conv2d_7 (Conv2D 3x3)	1	same	ReLU	512	2359808	2	4	8	16	32	512
	up_sampling2d_2 (Up Sampling 2x2)	1	-	-	-	0	4	8	16	32	64	512
	conv2d_8 (Conv2D 3x3)	1	same	ReLU	256	1179904	4	8	16	32	64	256
	up_sampling2d_3 (Up Sampling 2x2)	1	-	-	-	0	8	16	32	64	128	256
	conv2d_9 (Conv2D 3x3)	1	same	ReLU	128	295040	8	16	32	64	128	128
	up_sampling2d_4 (Up Sampling 2x2)	1	-	-	-	0	16	32	64	128	256	128
	conv2d_10 (Conv2D 3x3)	1	same	ReLU	64	73792	16	32	64	128	256	64
	up_sampling2d_5 (Up Sampling 2x2)	1	-	-	-	0	32	64	128	256	512	64
Output	conv2d_11 (Conv2D 3x3)	1	same	Sigmoid	1	577	32	64	128	256	512	1

Total parameters: 10,179,713
Trainable parameters: 10,179,713
Non-trainable parameters: 0

Layer output shape
(Rows, Columns, Channels)

Furthermore, each convolutional layer also includes an activation function, which performs the non-linear transformation of the features generated from the convolution operation (Dettmers 2015). In the proposed model, the input and output datasets, 24-bit RGB ortho-image and 8-bit grayscale elevation-map pairs with value range [0,255] are normalized to the range [0,1] by dividing them by 255. Thus, the activation function should progressively change from 0 to 1 with no discontinuity for generating the output. The Rectified Linear Unit activation function (ReLU), $f(x) = max(0, x)$, is a very popular choice for use in hidden layers; it is faster than many activation functions, such as Sigmoid. The ReLU function does not always output a non-zero, so it results in less neurons being utilized and less dependence between features (Nair and Hinton 2010). In addition, the Sigmoid activation function (also known as Logistic), $f(x) = 1/(1 + exp(-x))$ is used in the output layer to generate the continuous values for the elevation-map, instead of using SoftMax function to classify the objects in SegNet (Badrinarayanan et al. 2017). The detailed model layers and each layer output shape for each patch size trial are shown in Table 16, where the type of layers are described in the Keras 2.3 style (Chollet 2015).

6.3.1.3 Convolutional Encoder-decoder Compiling Configuration

This research project uses the “Sequential model API” in Keras to set up the convolutional encoder-decoder network model. When compiling the model, it uses “rmsprop” as the optimizer, and “mean_squared_error” as the loss function(Chollet 2015); “validation_split” is set to 0.05, which means that 95% of the datasets is used for training the model and 5% of the dataset is used for validation. In this research project, the efficiency of “early stopping” compared to non-stopping has been evaluated. The “early stopping” technique stopped model training when the monitored quantity had stopped improving (Chollet 2015), such as the training loss or validation loss had not decreased for 10 epochs. This research project uses “EarlyStopping(monitor='val_loss', patience=10)”.

What’s more, this research project uses the “same” padding for max pooling layers. As the model input sizes are 32, 64, 128, 256 and 512, which can be divided by 32 (2^5), the padding setting in max pooling should have no impact on the result because in each max pooling layer, the layer input size is halved, while the layer output size is still an integer which can be divided by 2. However, the model results varied on this setting. Using “same” padding generated a better result than “valid” padding.

6.3.2 Ortho-image Disassembling and Elevation-map Assembling Algorithm

The input layer and output layer of the proposed model (see Figure 51) indicate that the trained model predicts an elevation-map patch from an input ortho-image patch. A model prediction example is shown in Figure 54, while the edged area of each prediction patch is different from the center area. This is because the zero-padding is used in convolution operations. The normal convolution operation shrinks the input image size down to the filled center region in Figure 52. In this research project, the added padding operation enlarges the image size with “0” before the convolution operation (see Figure 52). Then, the zero-padding convolution ensures that the output maintains the same size as the input. However, the added “0” produces unwanted features in the edge of the prediction patches. The side-by-side assembly of predictions in Figure 54 shows the unexpected gridlines.

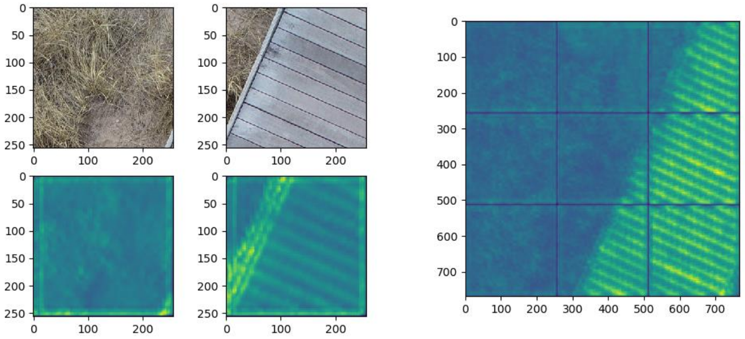


Figure 54 Prediction of side-by-side assembly

Figure 55 shows the workflow of the ortho-image disassembling and elevation-map assembling algorithm, which generates the elevation-map without unexpected “gridlines”. This algorithm needs to disassemble the ortho-image into several overlapping patches. The required number of patches is determined by Eq. 12. When assembling the elevation-map, only selected parts of each patch will be used. Compared with the side-by-side approach, the proposed overlapping algorithm replaces the patch edges with other predictions’ center regions. Additionally, the assembly of the elevation-map has the same GSD as the ortho-image. Then, the 3D geometry data can be reconstructed by Eq. 13.

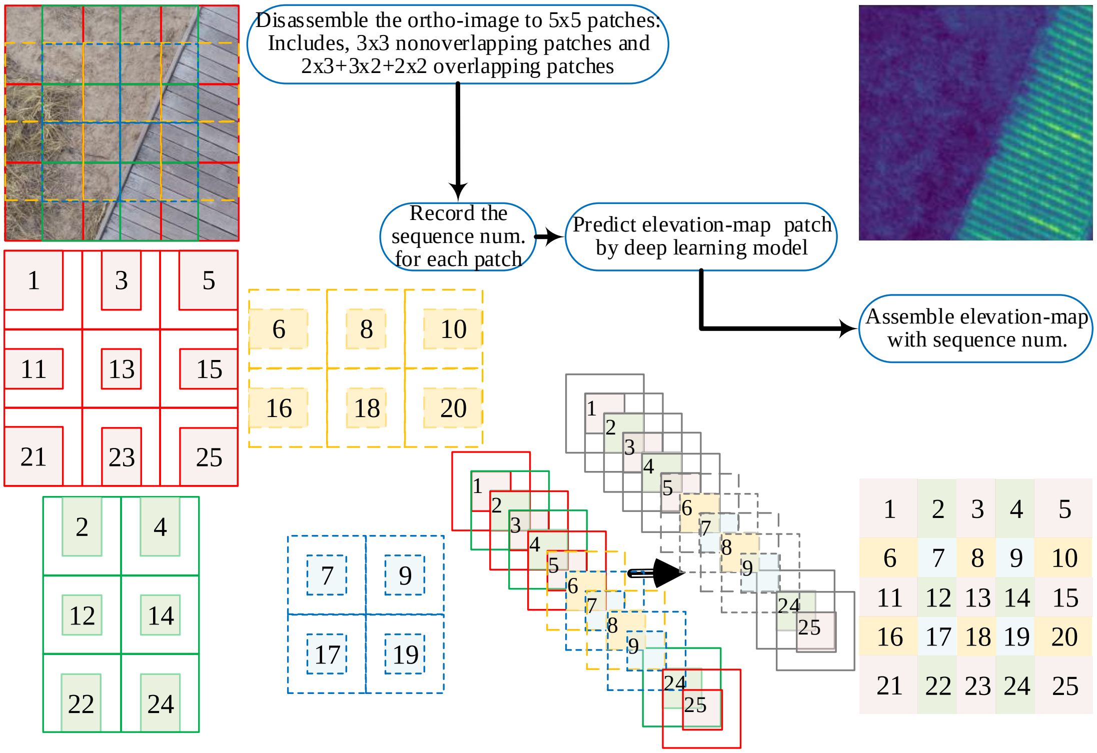

Disassemble the ortho-image to 5x5 patches: Includes, 3x3 nonoverlapping patches and 2x3+3x2+2x2 overlapping patches
Record the sequence num. for each patch
Predict elevation-map patch by deep learning model
Assemble elevation-map with sequence num.
1 3 5 11 13 15 21 23 25
6 8 10 16 18 20
2 4 12 14 22 24
7 9 17 19
1 2 3 4 5 6 7 8 9 24 25
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23 24 25

Figure 55 Workflow of the ortho-image disassembling and elevation-map assembling algorithm

$$Num.\,of\ Small\_Patches = \left(2 \times \frac{Image\ Height}{Patch\ Size} - 1\right) \times \left(2 \times \frac{Image\ Width}{Patch\ Size} - 1\right) \qquad \textbf{Eq. 12}$$

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} x \cdot GSD \\ -y \cdot GSD \\ Elevation \end{bmatrix} = \begin{bmatrix} (u - \frac{Image\ Width}{2}) \cdot GSD \\ -(v - \frac{Image\ Height}{2}) \cdot GSD \\ Elevation_{map[u,v]} \times \frac{Elevation_{Range}}{255} + Elevation_{LowerBoundary} \end{bmatrix} \qquad \textbf{Eq. 13}$$

6.4 Elevation Estimation Experiment

6.4.1 Experiment Dataset

6.4.1.1 Experiment Site

In this research project, the experiment datasets, the ortho-image and elevation-map pairs are selected from Chapter 5. In addition, the edges of the ortho-images and elevation-maps are removed to make their width (1,536-pixel) and height (1,536-pixel) which are exactly divisible by 32, 64, 128, 256 and 512. This is because the various patch size configurations will be compared. Figure 56 shows the spatial relation among these selected datasets.

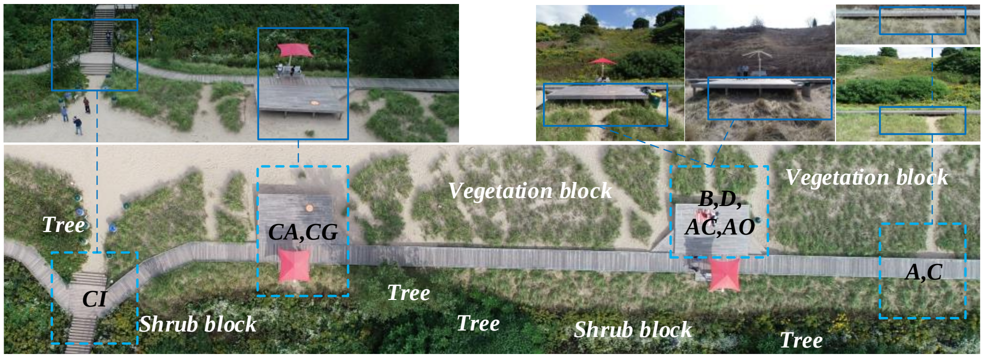


Figure 56 Elevation estimation experiment site condition

6.4.1.2 Ortho-images and Elevation-maps

Figure 57 lists the model training and validation datasets. The ortho-images were taken in Atwater Park (Shorewood, WI, USA) during different seasons. Data A and B were taken on 3/24/2019, when the vegetation had not recovered yet. Data C and D were taken on 6/5/2019, when the vegetation was growing.

Additional data AC, CA, CG and CI were taken on September 2019, when the vegetation was fully grown. Data AC is the same wooden platform as B and D; data CA is another wooden platform on this site; data CG and CI detail the umbrella and stairways. In addition, Figure 58 includes an additional ortho-image and elevation-map pair which is proposed to be used for quantitatively evaluating the trained model. Furthermore, the elevation-maps were aligned by picking a point on the wooden platform / path and setting its elevation as ±0.00.

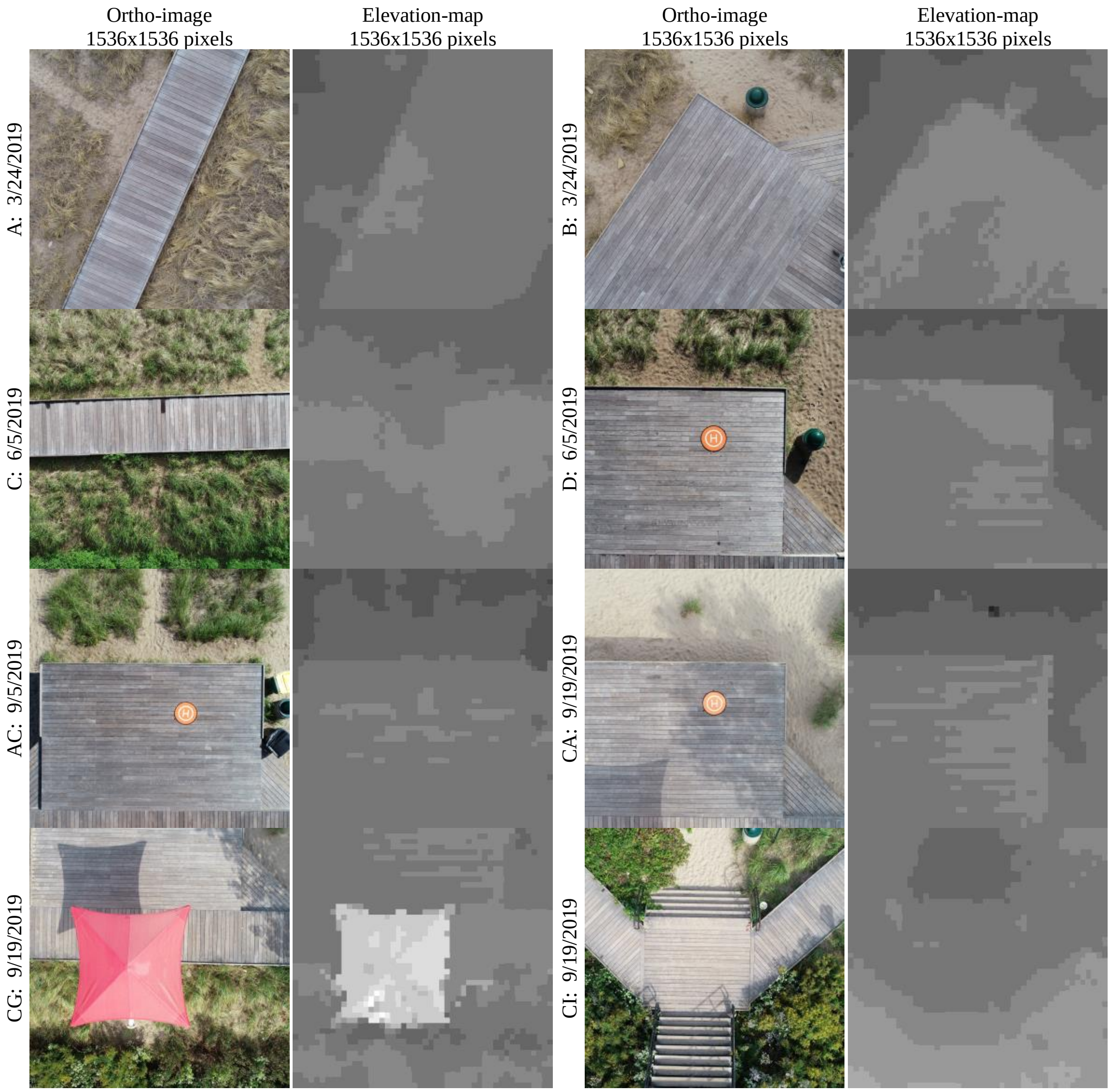


Figure 57 Model training and validation datasets

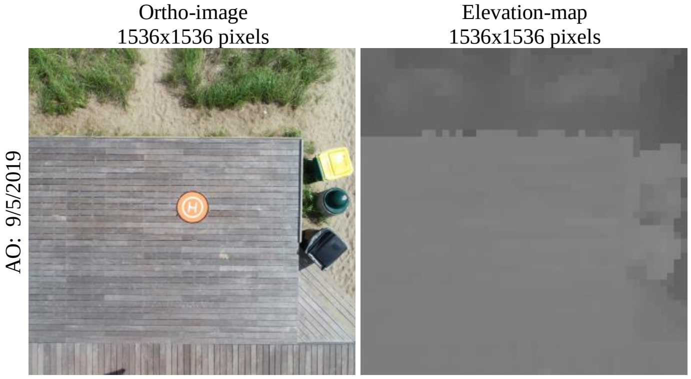


Figure 58 Model testing dataset

6.4.2 Elevation Estimation Deep Learning Model Training and Validation

6.4.2.1 Model Training Configuration

The model training parameters including batch sizes, epochs and dataset numbers are listed in Table 17. The “100 epochs” and “early stopping” were shared for the five different patch size trials. Eight ortho-image and elevation-map pairs (see Figure 57) and their 4-rotations were used to train the model. Thus, the total number of datasets is eight times the number listed in the last column of Table 15. The dataset numbers varied for the five different patch size trials, because the system memory limitation resulted in different strides being used for creating datasets. What’s more, in this research project, when training the model, the “batch size=32” was used in 512×512-pixel patch trial and “batch size=128” was used in the other trials. This is because of the single GPU’s memory limitation (11GB or 10.24GiB); an additional 3.38 GiB memory and 2.29 GiB memory are needed for each GPU with batch size 128 and 64 respectively. Fortunately, the small batch size in 512×512-pixel patch trial only results into more model training times in each epoch.

Table 17 Model Training Parameters and Results

	Patch Size Trials				*Training Epoch Trials*	
Patch Sizes	*Datasets (Validation Split = 0.05)*			*Batch Sizes*	*EarlyStopping(monitor='val_loss', patience=10), Epochs=100*	
	Total Num.	*Training*	*Validation*		*w/ Early Stop*	*w/o Early Stop*
32x32	288800	274360	14440	128	29	100
64x64	70688	67153	3535	128	27	100
128x128	64800	61560	3240	128	18	100
256x256	14112	13406	706	128	35	100
512x512	3872	3678	194	32	32	100

6.4.2.2 Training and Validation with Early Stopping

The loss results of model training for each trial and the loss results of model validation (also known as model testing) for each trial in Figure 59. The five different patch size trials were stopped at different epochs (see Table 17). The 128×128-pixel patch trial stopped at 18th epoch is the earliest trial, and the 256×256 patch stopped at 35th epoch. 256×256-pixel patch took the most epochs for the validation loss to reach stable for 10 epochs.

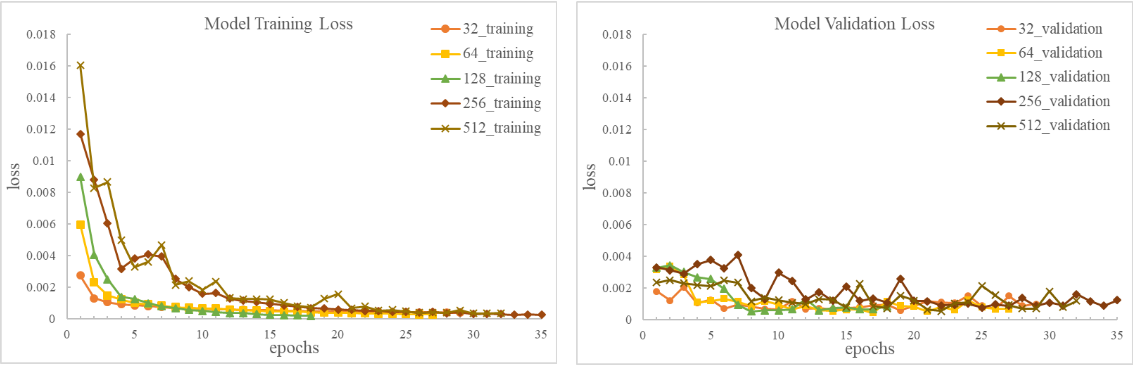


Figure 59 Loss of model training and validation (w/ early stopping)

Furthermore, the validation results of each patch size are shown in Figure 60, where the "ground truths" are the elevation-map patches used in training the model, the predictions are the model outputs generated from the trained model with the corresponding inputs. The "ground truths" and model predictions are shown in the same viridis colormap range, the more similar the color the more accurate the predictions. In visual, the model predictions are not a constant color (grayscale value) for a 32×32-pixel patch as the elevation-map patches. The developed model decodes the elevation values for each pixel of the input patch instead of a single elevation value for the whole patch. The model output results show that the trained model can distinguish different objects, such as the wooden paths that are distinguished from the ground in the 256×256 and 512×512 trials. The trained model also shows the ability to correct elevation value errors that occur in the wooden path of 256×256 and 512×512 trials. In detail, the wooden paths of 256×256 and 512×512 in the "ground truth" have incorrect elevation values, while the predictions for the wooden paths show corrected elevations.

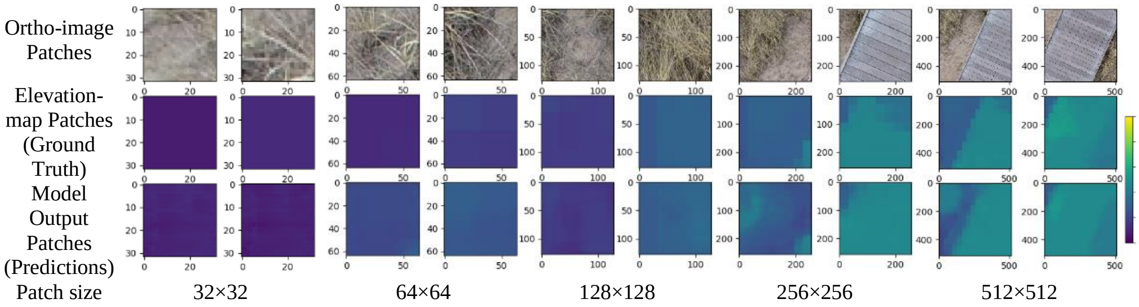
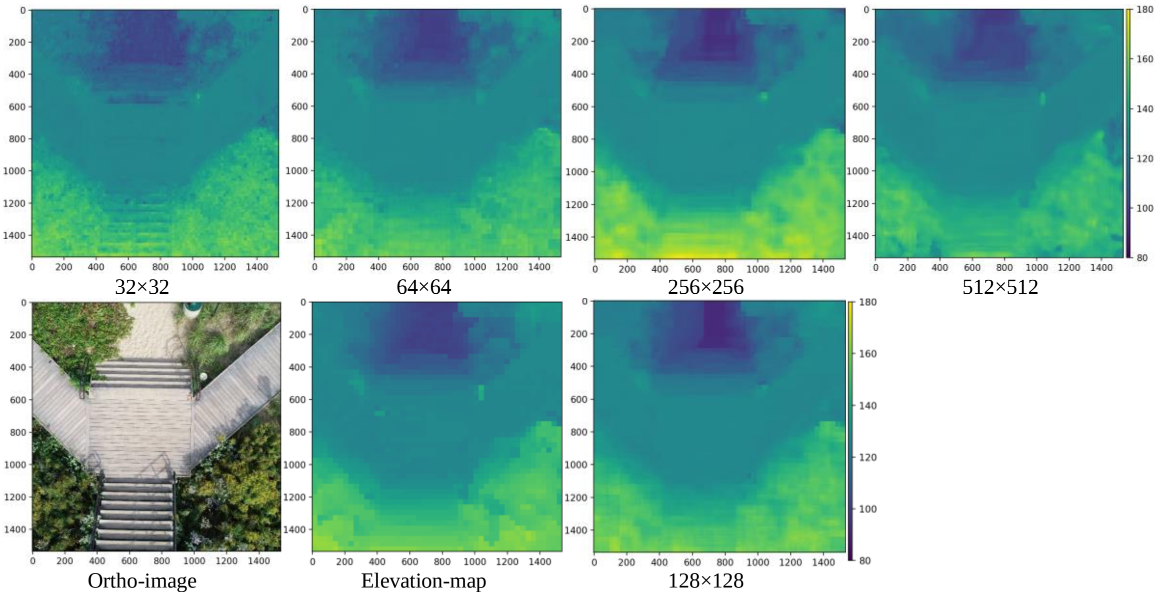


Figure 60 Data A: ground truth patches and model prediction patches (w/ early stopping)

For the five “early stopping” different patch size trials, the minimum model training loss occurred on 128×128-pixel patch trial at its 18th epoch (see Figure 59). The 128×128-pixel patch also has the smaller model validation loss, while the minimum model validation loss occurred on 64×64-pixel patch trial at its 27th epoch. The Data A predictions in Figure 60 indicates that the 128×128-pixel patch trial has better performance than other patch sizes in the “early stopping” trials, and the overlapping assembled predictions in Figure 61 confirms that the 128×128-pixel patch has the best performance in the “early stopping” trial for Data CI as well. That may be because the 128×128-pixel patch balances the local features of each 32×32 patch and contains global features to connect each single 32×32 patch as well. The detailed comparisons of the different patch sizes will be stated in the discussion section.


32×32 64×64 256×256 512×512

Ortho-image Elevation-map 128×128

Figure 61 Data CI: overlapping assembly of model predictions (w/ early stopping)

6.4.2.3 Training and Validation with 100-epcoh

Another model training was conducted without “early stopping”, the 18 to 100 epochs model training loss and validation loss of the five different patch size trials are shown in Figure 62. The 128×128-pixel patch has the minimum model training loss of 8.74E-06 at 100 epochs, which is smaller than 1.82E-04 at the “early stopping” trial. The 64×64-pixel patch and 256×256-pixel patch trials have a more stable decreasing trend and smaller values for training loss compared to the extreme size patches 32×32 and 512×512. Therefore, using the 128×128-pixel patch for the developed convolutional encoder-decoder network model has the best model training and validation performance, followed by the 64×64-pixel patch and 256×256-pixel patch.

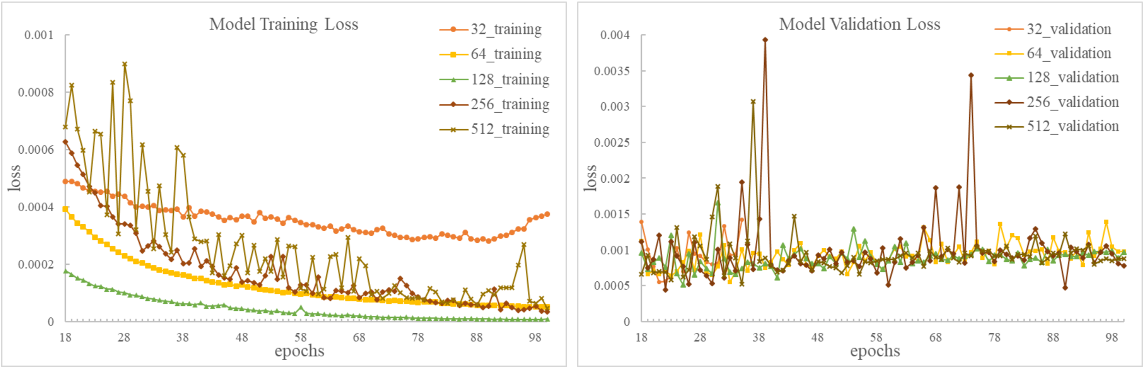
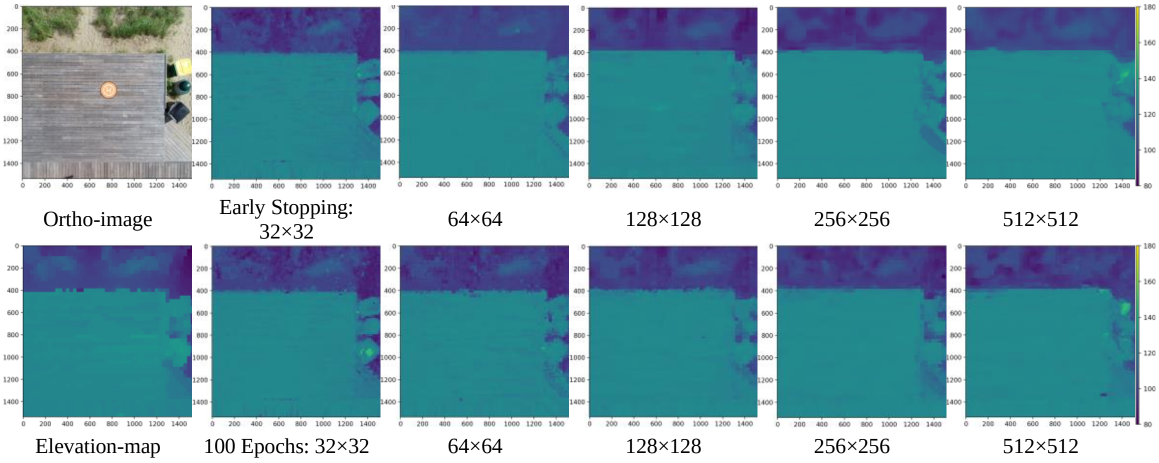


Figure 62 Loss of model training and validation (18~100 epochs)

6.4.3 Elevation Estimation Testing

The testing data AO in Figure 58 is different from the training data AC in Figure 57; they were captured on the same day but in different fight paths and sequences; the drone landed after captured the AC low-high ortho-image pair and took off again to capture AO pair; for the AC pair, the 10 m ortho-image was captured first followed by the 20 m ortho-image; but for the AO pairs, the 20m ortho-image was captured first followed by the 10 m ortho-image.

Figure 63 contains the model predictions for data AO . Visually, the 128×128-pixel patch has the best result in the “early stopping” trial and the 64×64-pixel patch is better than others, while the patches 128×128 and 256×256 are better than others in the 100 epochs trials. The 100 epochs results are more detailed than the "early stopping" ones. These 2D predictions can be easily converted to 3D point clouds by Eq. 13 with the selected 2,304 (48×48) points (strides=32 pixels in column and row directions).

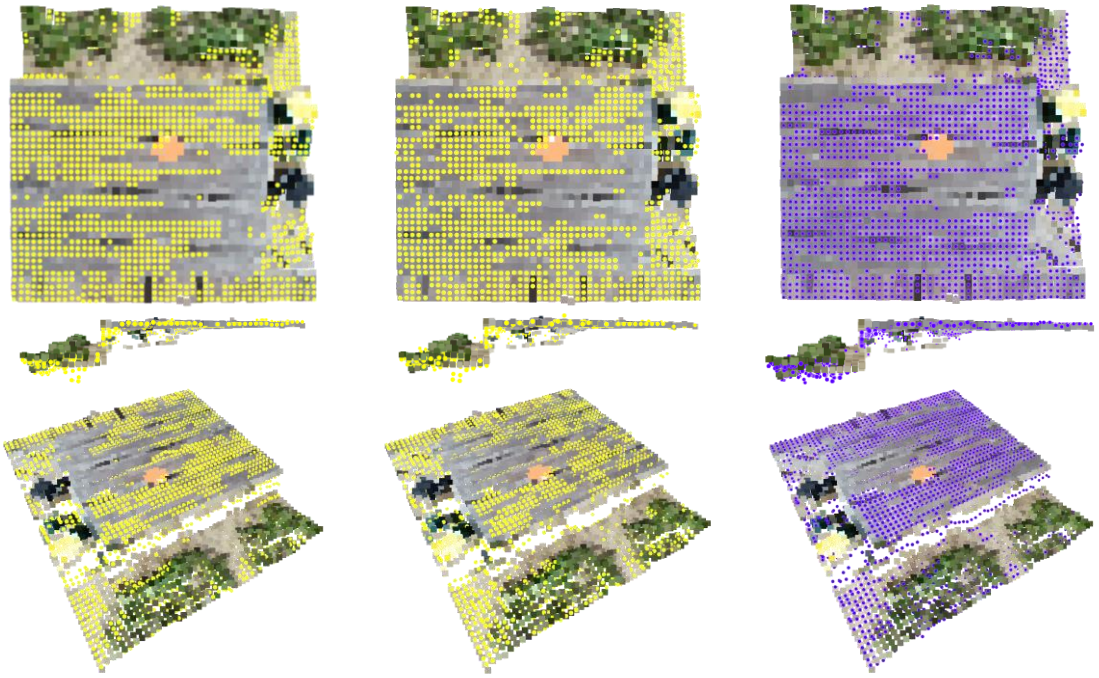


Ortho-image | Early Stopping: 32×32 | 64×64 | 128×128 | 256×256 | 512×512

Elevation-map | 100 Epochs: 32×32 | 64×64 | 128×128 | 256×256 | 512×512

Figure 63 Data AO: predictions with different patch size and different epochs

Figure 64 overlaps the 128×128 and 256×256 prediction point clouds (one pixel is one point) with the "ground truth", which is converted from the elevation-map and plotted with RGB cubes. The model predictions have the similar shape as the "ground truth" and are more accurate than the "ground truth" for the wooden platform surface and its edges.



Early Stopping: 128×128 | 100 Epochs: 128×128 | 100 Epochs: 256×256

* RGB Cubes are Elevation-map, Yellow Points are 128×128 patch results, Purple Points are 256×256 patch results

Figure 64 Data AO: point cloud comparison between predictions and ground truth

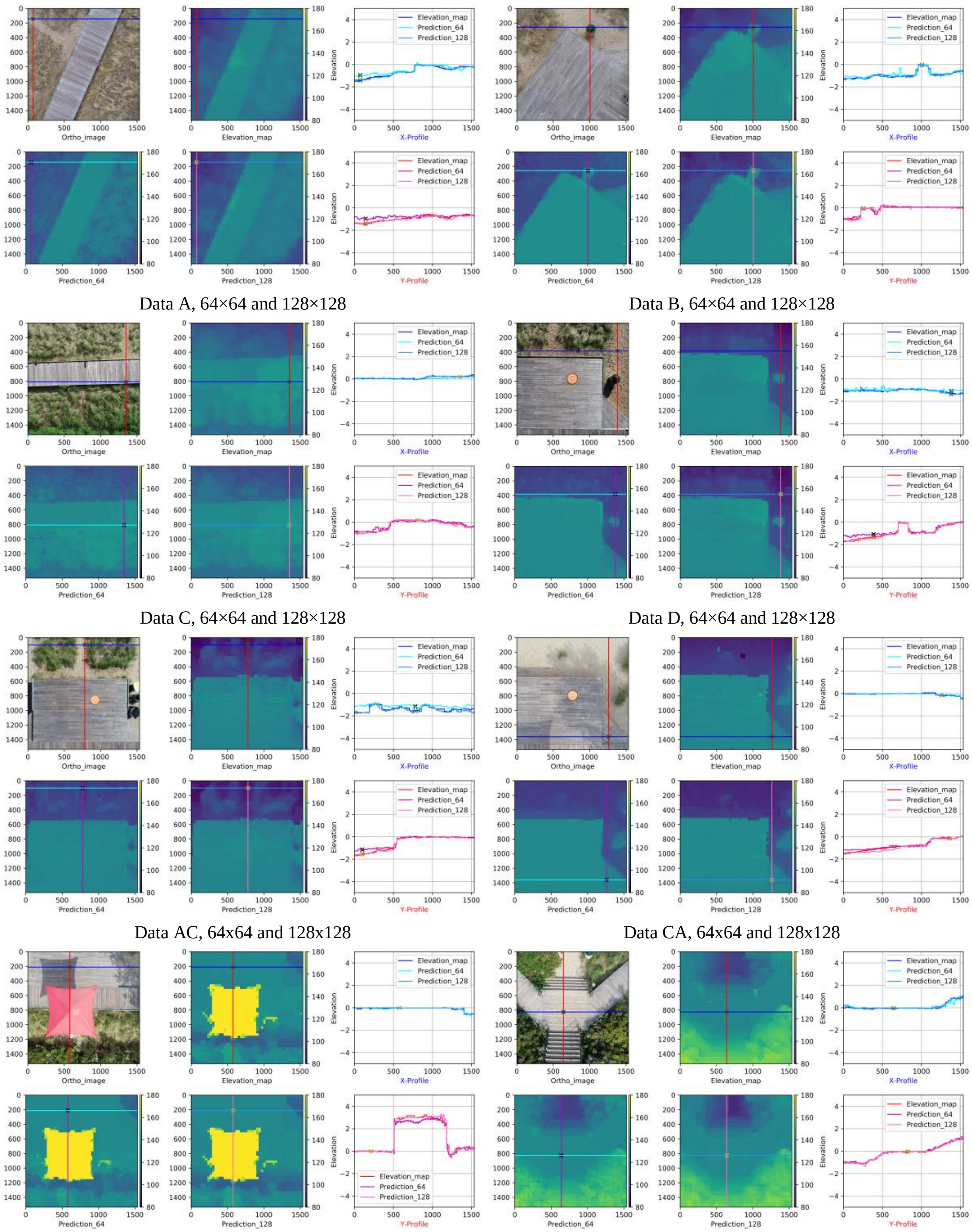
6.5 Elevation Estimation Discussions

6.5.1 Patch Size Comparison and Discussion

As the model training and testing results show, the 128×128-pixel patch and the 64×64-pixel patch are better than the other patch sizes in the “early stopping” trials. Figure 65 shows the overlapping assembly of model predictions with the “ground truth” elevation-map of the eight model training datasets between these two patch sizes. In addition, several interesting points were selected to show their X/Y-profiles elevation (unit: m) changes.

Each data in Figure 65 has ground surface, large objects or structures, and small objects. For the ground surface 3D-reconstruction, the 128×128-pixel patch has the best performance, as seen with the selected points in data A, the Y-profiles of data C, D and AC. What’s more, the tiny and sparse grass on the ground shows no impact to the 3D-reconstruction of the ground surface shape, such as the X-profiles of data B and D (see Figure 65). The trained model with 128×128-pixel patch correctly identifies that these regions are ground surface and not vegetation. For large object 3D-reconstruction, the 128×128-pixel patch also has the best performance, seen in the Y-profile of the umbrella in data CG, the Y-profile of the stairways in data CI, and the wooden platforms and wooden paths in all of the training datasets. For small objects, both the 128×128-pixel and 64×64-pixel patches have good performance in the 3D-reconstruction of small objects’ shapes, such as the X/Y-profiles of the garbage can in data B and D.

In general, the 128×128-pixel patch has a better performance with the “early stopping” setting at the 18th epoch than the 64×64-pixel patch trial with 27 epochs. However, training the developed model with the smallest 32×32-pixel patch has given a potential function to correct the elevation errors in the “ground truth”, such as the wooden path edge in the center region of data A, the wooden platform corner in data B, and the gap between the platform and the garbage can in data B (see Figure 66). However, the large patch size 256×256 and 512×512 trials retained these errors. Therefore, the median size 128×128-pixel patch is the best option for balancing the local features and global features, each elevation value in the 32×32-pixel patch and the connections between 32×32-pixel patches.



Data A, 64×64 and 128×128

Data B, 64×64 and 128×128

Data C, 64×64 and 128×128

Data D, 64×64 and 128×128

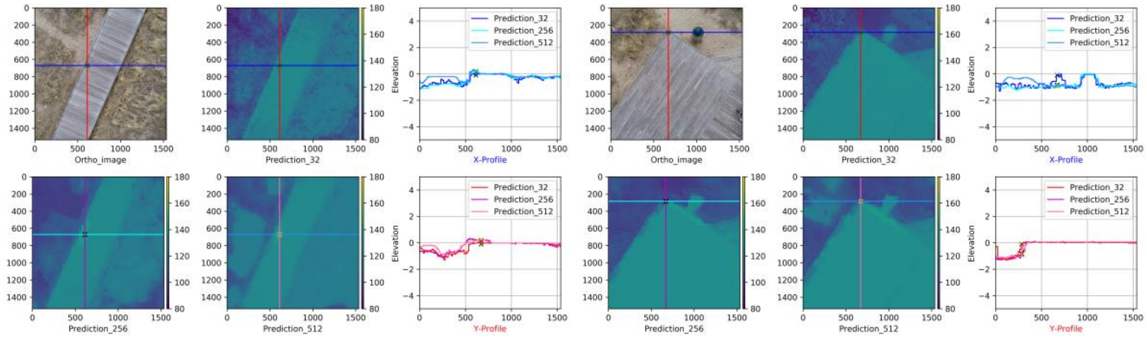
Data AC, 64x64 and 128x128

Data CA, 64x64 and 128x128

Data CG, 64x64 and 128x128

Data CI, 64x64 and 128x128

Figure 65 Patch size comparison I: predictions for each training dataset



Data A, 32×32, 256×256 and 512×512

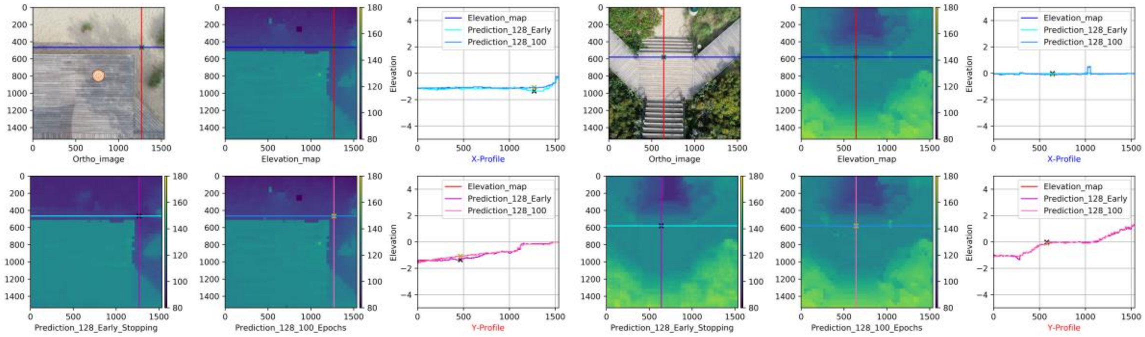
Data B, 32×32, 256×256 and 512×512

Figure 66 Patch size comparison II: additional predictions for data A and B

6.5.2 Texture Comparison and Discussion

Aside from the ground surfaces, the vegetation surfaces and wooden surfaces are the two major textures in the experiment site (see Figure 56). The vegetation surfaces were captured during different seasons; the vegetation blocks show different colors in data A, B, C, D, AC and CG (see Figure 65). In Figure 65, the selected point in data AC is on the ground surface. The neighboring vegetation blocks were 3D-reconstructed well in the X-profile of data AC (128×128-pixel patch), in which the real vegetation blocks' surface heights ranged from 0.6 m to 0.90 m on September 05th, 2019. The X-profiles of data A and B also crossed the withered vegetation blocks, in which the 128×128-pixel patch results are matched with the "ground truth". In additional, the data CG and CI contain denser foliage in the shrub blocks, which are different from the vegetation blocks. The Y-profile of data CG and X-profile of data CI are matched with the "ground truth". The wooden surfaces and ground surfaces were captured in different brightness environments and their colors varied in Figure 65. When creating the experiment datasets, all wooden surfaces (except the stairways) were set as elevation = ±0.00 m. They were all 3D-reconstructed well in the model predictions. Furthermore, the Y-profile of data CI shows the 3D-reconstructed stairways are matched with the "ground truth", and the selected point in data CA has the correct elevation differential to the wooden platform as well. Thus, the developed model that trained with 3-channel RGB ortho-images is robust in complex textured regions for the "early stopping" 128×128-pixel patch trial.

In addition, there are three kinds of poorly textured region in the experiment dataset, including shaded spots, shaded strips and shaded blocks. For small spots of shade, such as the garbage can's shade in data D (see Figure 65), the 128×128-pixel patch generated the correct predictions. For large shade blocks, such as the tree’s shade and umbrella’s shade on the wooden platform in data CA and CG respectively (see Figure 65), the 128×128-pixel patch has the correct predictions. The “early stopping” 128×128-pixel patch trial has inconsistent performance for the shaded strips. The selected point in data AC is on the shade of the vegetation block. The ground surface was identified as vegetation in the 64×64-pixel patch trial, but was correctly identified using the 128×128-pixel patch. However, the 64×64-pixel patch trials are more aligned with the “ground truth” than the 128×128-pixel patch trials, such as the Y-profiles of the shaded ground surface close to the wooden platform in data CA and the shaded area next to the bottom stairs in data CI(see Figure 65). Fortunately, adding model training epochs can improving the prediction accuracy (see Figure 67), which will be discussed in the next section. Therefore, using the 128×128×3 RGB ortho-image input patch and 128×128×1 grayscale elevation-map pair datasets to train the developed convolutional encoder-decoder network model has a good performance both in complex textured and poorly textured regions.



Data CA, 128×128 Early Stopping vs 100 Epochs

Data CI, 128×128 Early Stopping vs 100 Epochs

Figure 67 Epochs comparison I: predictions for early stopping vs 100 epochs

6.5.3 Epoch Comparison and Discussion

The validations of data CA and CI (see Figure 67) indicated that it is worth continuing to train the model after the “early stopping” point to improve the performance of the 128×128-pixel patch. This is due to the model not training well enough at the 18th epoch, though it still has the potential to narrow down the variations of the validation loss (see Figure 68). In addition, the comparison of testing results (see Figure 63) shows that the 128×128-pixel and 256×256-pixel are better and smoother than other patch sizes in the

100 epochs trials, and the comparison of the two 100 epochs validation loss curves in Figure 68 confirmed that the 128×128-pixel patch is more stable and can reach a stable trend earlier than the 256×256-pixel patch. Thus, the well-trained 128×128-pixel patch has both the best model training and prediction performance for the developed convolutional encoder-decoder network model.

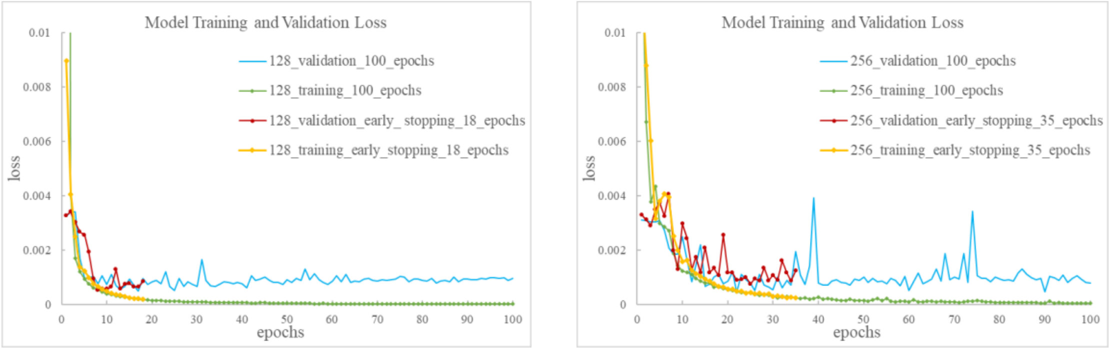


Figure 68 Epochs comparison II: training and validation loss (128×128-pixel vs 256×256-pixel)

Furthermore, the quantitative evaluation of the model validation accuracy and testing accuracy were conducted by measuring the point cloud (see Figure 64). In detail, for each validation result of 128×128-pixel patch "early stopping" trial and 128×128-pixel patch 100 epochs trial, 2,304 (48×48) points (the centers of each 32×32-pixel patch) are selected from the corresponding ortho-images, elevation-maps and overlapping assembled model predictions (1536×1536-pixel). Then the 3D point clouds were generated by Eq. 13 with the selected 2,304 points. For each model training and validation data from A to CI, the variable "ELE-DIFF-EARLY" was created as the elevation differential between "ground truth" and 128×128-pixel patch "early stopping", and variable "ELE-DIFF-100" was created as the elevation differential between "ground truth" and 128×128-pixel patch 100 epochs. Both variables have 18,432 (2,304×8) samples. For the testing data AO, the same variables were created and named as "AO-DIFF-EARLY" and "AO-DIFF-100" with 2,304 samples.

The descriptive statistics for the four variables are listed in Table 18. For the model training and validation results, the 99% Confidence Interval (CI) of elevation differential is reduced from (1.6, 2.1) cm to (0.6, 0.8) cm by adding the model training epochs. In addition, the trained model has a good result in predicting the elevations for the testing data AO, the elevation differential has a 99% CI (2.14, 4.08) cm. Therefore, the model training epochs have a positive effect in improving the model accuracy; after 100 epochs the developed model is a well-trained model.

Table 18 Elevation Differential Results

Sample	*N*	*Mean (unit: m)*	*StDev*	*SE Mean*	*99% CI for μ*	*Minimum*	*Q1*	*Median*	*Q3*	*Maximum*
ELE-DIFF-EARLY	18432	0.018495	0.133267	0.000982	(0.015966, 0.021024)	-2.31373	-0.03922	0	0.07843	1.84314
ELE-DIFF-100	18432	0.0073	0.058502	0.000431	(0.006190, 0.008410)	-1.05882	0	0	0.03922	0.94118
AO-DIFF-EARLY	2304	0.0424	0.17635	0.00367	(0.03293, 0.05187)	-1.17647	-0.03922	0.03922	0.11765	1.01961
AO-DIFF-100	2304	0.03111	0.18096	0.00377	(0.02140, 0.04083)	-1.05882	-0.03922	0	0.07843	1.21569

6.5.4 Accuracy Evaluation and Discussion

Figure 69 shows the distributions of the elevation differential between the “ground truth”, model validation results and model testing results. The histogram of “ELE-DIFF-100-CM” shows that 94% of points from the model training datasets have an elevation error less than 10 cm in the “well-trained” model (100 epochs). The two histograms “AO-DIFF-100-CM” and “AO-DIFF-EARLY-CM” of the testing data AO show that the “well-trained” model has a significant improvement over the “early stopping” model. The “well-trained” model prediction accuracy is 52.43% compared to the 47.05 % on the “early stopping” model, in which an accurate elevation measurement is defined as measurement error is equal to or less than 5.0 cm (Takahashi et al. 2017). The worst predictions (error > 25 cm or error < -25 cm) account for 9.64% and 12.37% in the “well-trained” model and “early stopping” model respectively.

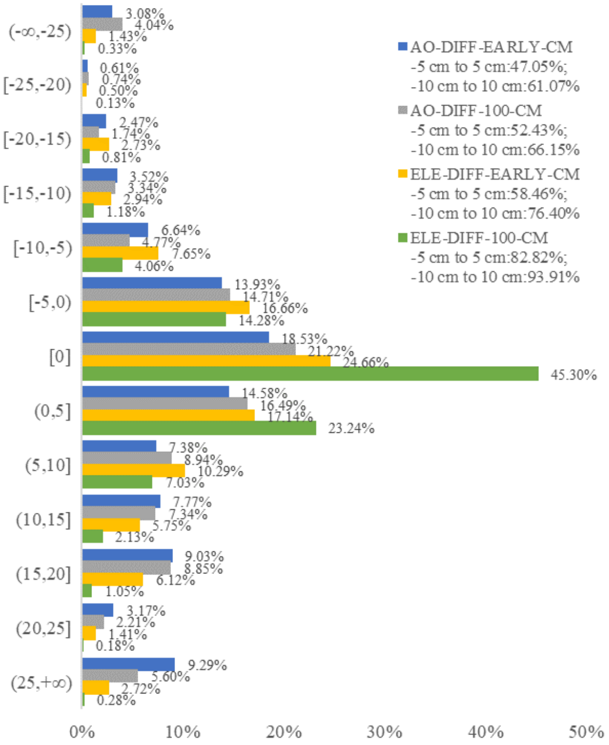
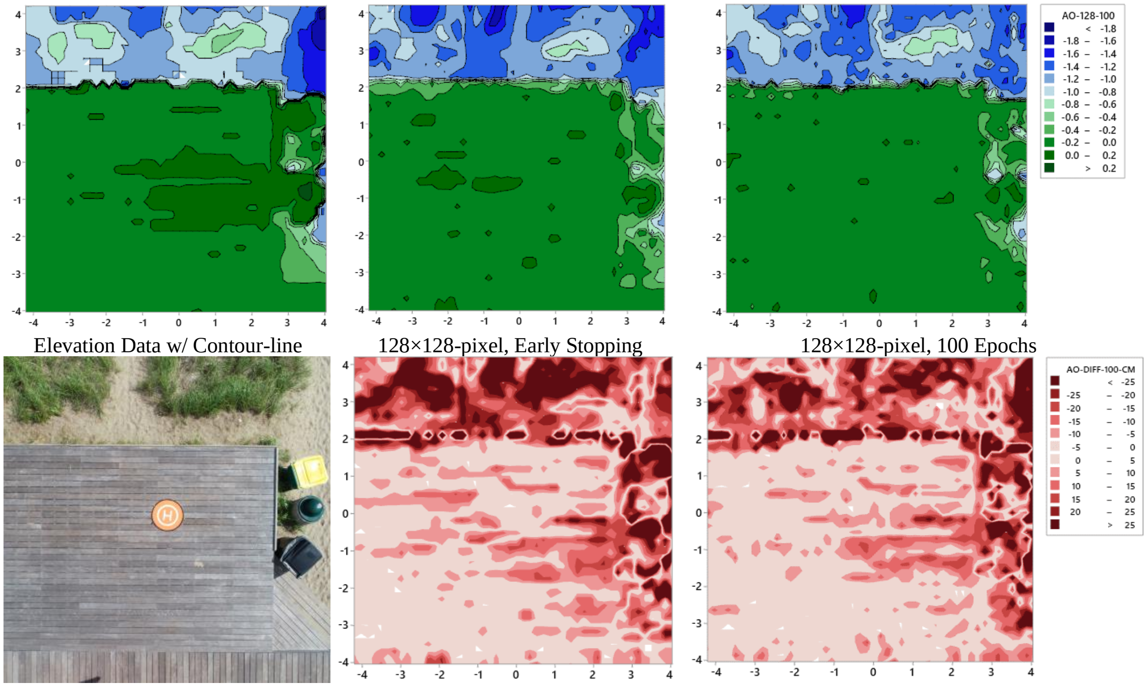


Figure 69 Distribution of elevation differential

In addition, the prediction contour-maps are show in Figure 70, and the elevation differentials were mapped as well. Most of the worst predictions of the “well-trained” model are on the edges of the wooden platform and garbage cans. This is because the “ground truths” on these locations are incorrect, the model predictions have corrected them. Excluding these errors, the model prediction accuracy will raise up. Thus, the “well-trained” model at least has a 52.43% accuracy in estimation the construction site elevations.



Elevation Data w/ Contour-line

128×128-pixel, Early Stopping

128×128-pixel, 100 Epochs

Ortho-image,
Data AO, 10 m

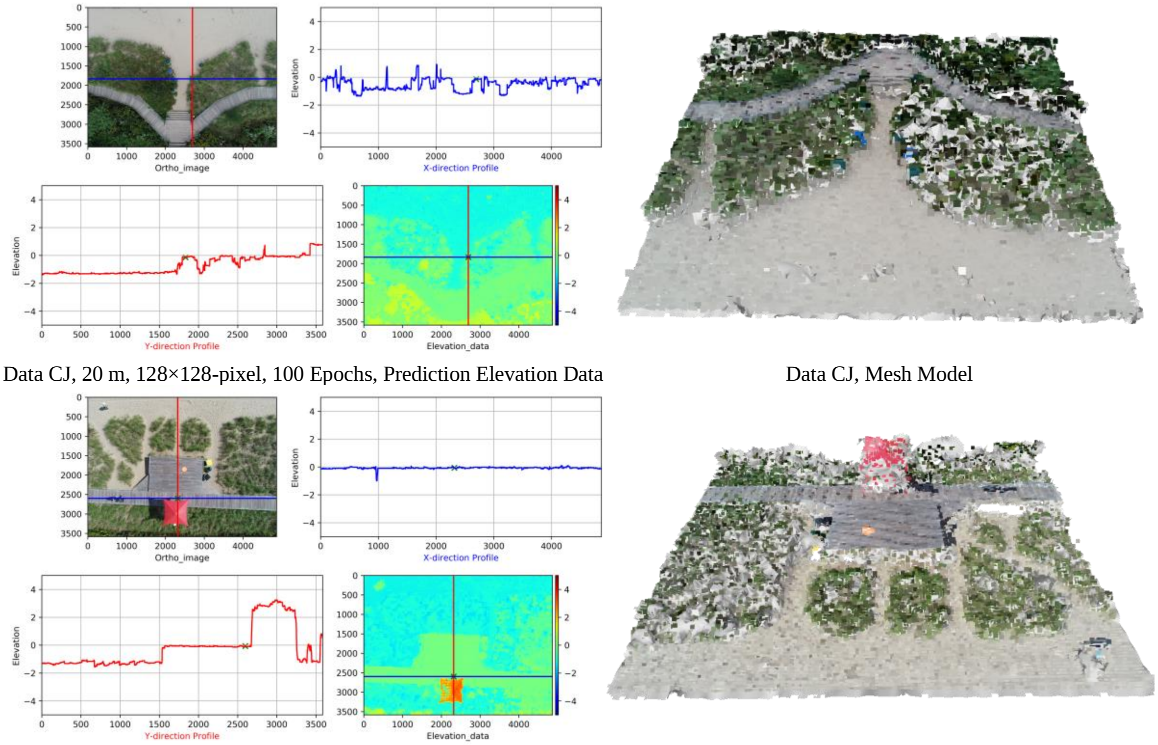
Elevation Differential,
128×128-pixel, Early Stopping

Elevation Differential,
128×128-pixel, 100 Epochs

Figure 70 Spatial distribution of elevation differential

6.5.5 Prediction Evaluation and Discussion

Figure 71 shows two 20m ortho-images which have the same GSD=0.54 cm/pixel as the model training dataset. The blue garbage can (17.65 cm lower than the wooden platform) is the new object not used in training the developed model and the original images were cut to 3584×4864-pixel without image resize. The Y-profile at the blue garbage can is -13.7 cm, which is close to the true value with the error 3.95 cm <5.0 cm. The Y-profile on the top of the umbrella is 3.196 m, which is accurately matched with its true value 3.20 m. Thus, training the developed model with the ortho-images captured at height 10 m can be used in 3D-reconstruction of ortho-images at height 20 m, the trained model is also able to generate the accurate elevations for the ortho-images at 20m as well. However, its performance worsens for the ortho-images at 40 m and above.



Data CJ, 20 m, 128×128-pixel, 100 Epochs, Prediction Elevation Data

Data CJ, Mesh Model

Data AN, 20 m, 128×128-pixel, 100 Epochs, Prediction Elevation Data

Data AN, Mesh Model

Figure 71 Elevation Predictions of data CJ and data AN

Furthermore, the top view of the experiment site in Figure 56 was captured at fight height 100 m. The elevation prediction results of the "well-trained" models with 128×128-pixel and 32×32-pixel patches are shown in Figure 72. The 32×32-pixel patch results show the ground surfaces, wooden surfaces, and shrub blocks are well reconstructed, but the vegetation blocks are assigned with incorrect elevations. The bad prediction of the 128×128-pixel patch occurs around (500, 2000), where the shaded wooden path was not included in the model training datasets. Thus, to make the model satisfied with complex construction site situations, a comprehensive dataset (ortho-image and elevation-map pairs) is required. This dataset should include different textures of the construction site, because the top layer materials of the construction sites are not limited to vegetation, water, snow, sand, rock, soil, concrete, asphalt, buildings and structures.

For training a precise deep learning model, the number of datasets should be large enough to cover the various construction site surfaces.

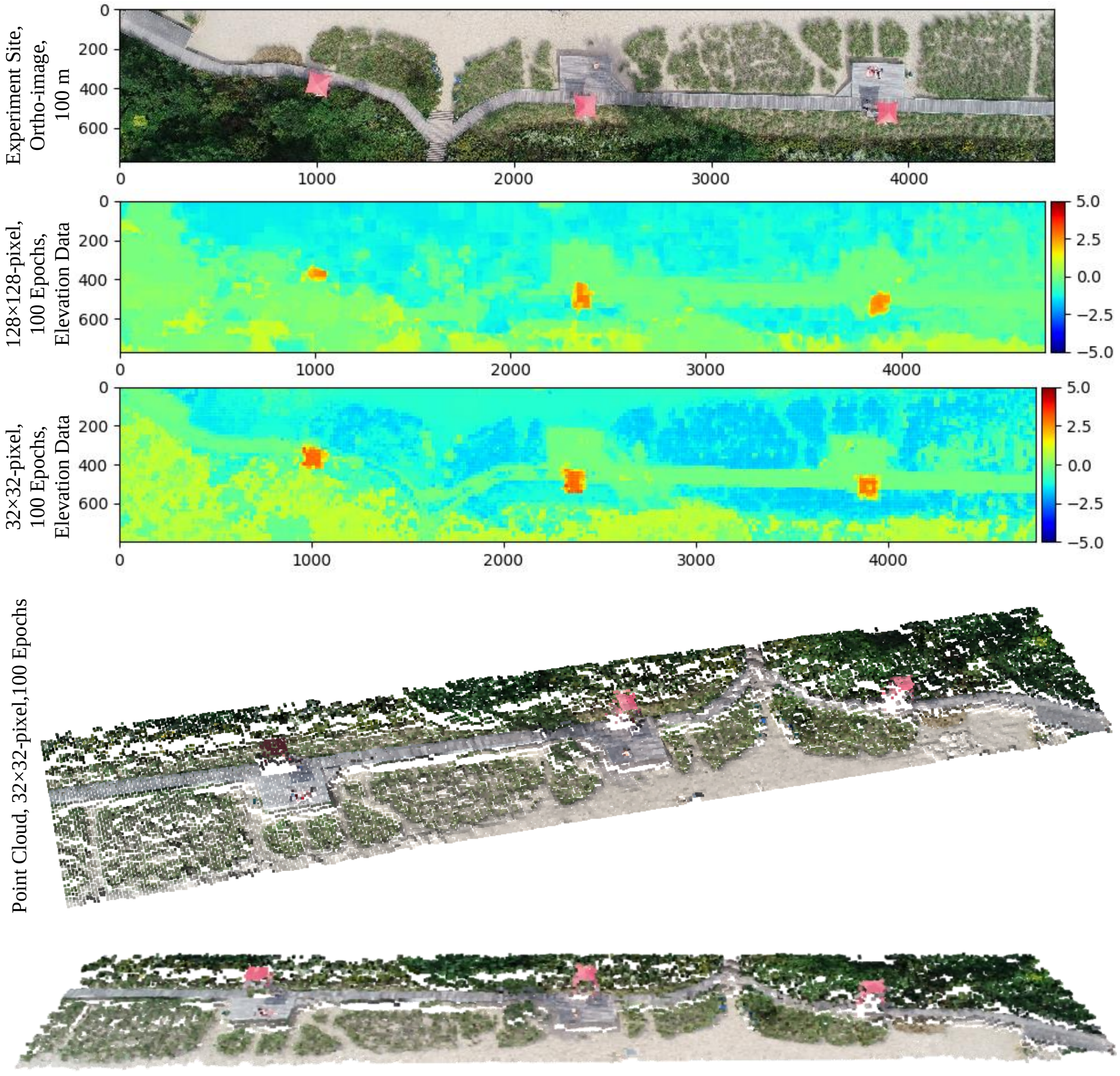


Figure 72 Elevation Predictions of the experiment site

CHAPTER 7: ORTHO-IMAGE AND DEEP LEARNING-BASED VEGETATION IDENTIFYING AND REMOVING ALGORITHM DESIGN AND TESTING

7.1 Introduction

The performance of the image-based elevation determination methods in Chapter 4 and Chapter 6 are affected by the plants and other ground covers on the construction site when determing the ground elevations. This is because the light rays are reflected on the surface of vegetation instead of the "real" ground surface. Therefore, to improve the effectiveness of the image-based surveying methods, automatically detecting and removing the vegetation and other obstacles from their raw surveying results and determining the "real" ground elevations, are necessary and important for construction professionals who heavily depend on elevation data in earthwork operations and facility layout.

In this chapter, a convolutional neural network (CNN) model is designed to classify the small sized image patch into vegetation categories or other object categories using a drone-based high-resolution construction site ortho-image (see Figure 73). Then, a vegetation removing algorithm is used to determine the ground elevations covered by vegetation from the elevation-map. Experiments are conducted to evaluate the effectiveness of the proposed method with high-resolution ortho-image and label-image pair datasets. The label-image is marked at each pixel with an 8-bit grayscale value [0,255] to represent up to 256 objects' categories. To explain how to determine the "real" ground elevation covered by vegetation from a drone-based high-resolution ortho-image and elevation-map pair, the rest of this chapter presents the research results of dataset acquisition and creation, model architecture designs, model training and testing, field experiments and result discussions.

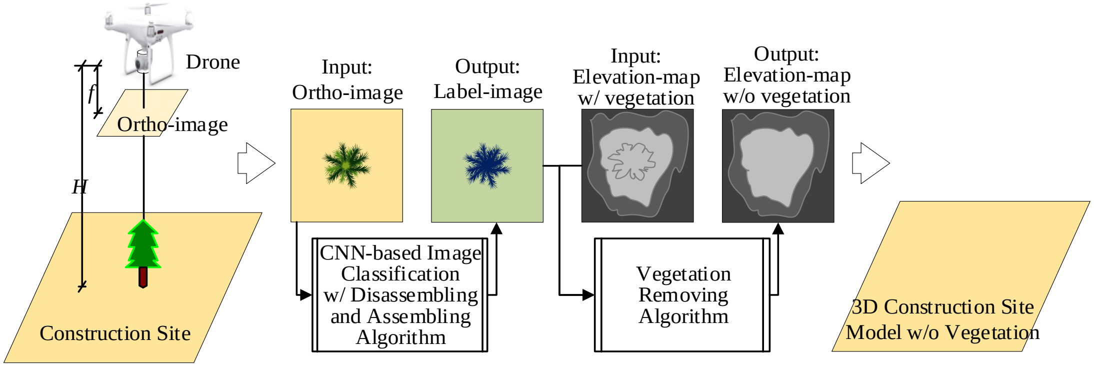
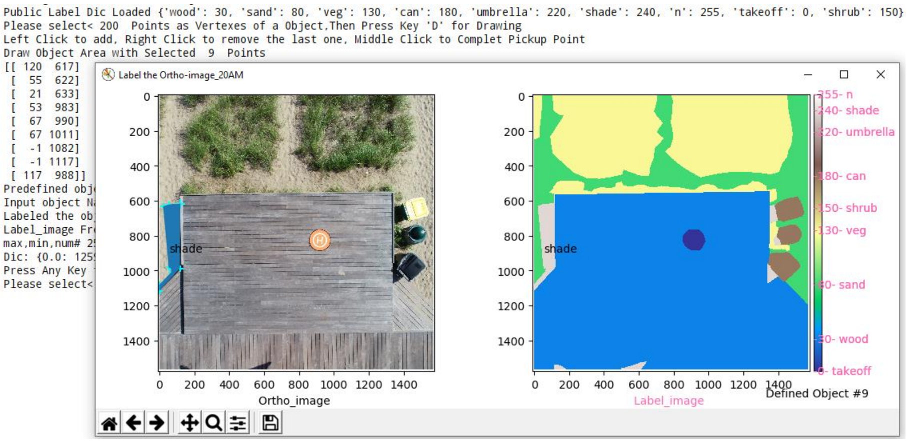

Drone
f
H
Ortho-image
Construction Site
Input:
Ortho-image
Output:
Label-image
Input:
Elevation-map
w/ vegetation
Output:
Elevation-map
w/o vegetation
CNN-based Image Classification w/ Disassembling and Assembling Algorithm
Vegetation Removing Algorithm
3D Construction Site Model w/o Vegetation

Figure 73 Workflows of vegetation identifying and removing method

7.2 Vegetation Identifying Dataset Creation

7.2.1 High-resolution Label-image Creation

Figure 74 shows the graphical user interface of the “Label-App” which is designed for labeling an ortho-image with 8-bit values [0, 255] and programmed using Python 3.6.8 and matplotlib 3.1.1 library. The computer mouse is used to select vertexes on the ortho-image for identifying each object. The keyboard is used to create a new class-label or select a predefined class-label such as “240-shade” in the left side of the label-image. The label-image is shown in “terrain” colormap for better visualization. Same as the ortho-image, the label-image also has the high-resolution 1568×1568-pixel, which is saved in two file-formats including a 1,568×1,568-pixel grayscale image file for visualization and a 1,568-row and 1,568-column spread sheet file for training the deep learning model. Saving as spread sheet file is necessary because the interpolation value appears on the edges of different objects in the image file.



```
Public Label Dic Loaded {'wood': 30, 'sand': 80, 'veg': 130, 'can': 180, 'umbrella': 220, 'shade': 240, 'n': 255, 'takeoff': 0, 'shrub': 150}
Please select< 200  Points as Vertexes of a Object,Then Press Key 'D' for Drawing
Left Click to add, Right Click to remove the last one, Middle Click to Complet Pickup Point
Draw Object Area with Selected  9  Points
[[ 120  617]
 [  55  622]
 [  21  633]
 [  53  983]
 [  67  990]
 [  67 1011]
 [  -1 1082]
 [  -1 1117]
 [ 117  988]]
Predefined obj
Input object N
Labeled the ob
Label_image Fr
max,min,num# 2
Dic: {0.0: 125
Press Any Key
Please select<
```

Figure 74 Example of ortho-image and label-image

Figure 75 shows a high-resolution ortho-image, a label-image and an elevation-map pair, all of which will be used as the testing dataset in this research project. The point cloud is generated using the selected central points of each 32×32-pixel patch of the ortho-image (textures) and elevation-map (elevation values). The high-resolution images used in this research project are 1,536×1,536-pixel, which are generated by removing 16 pixels on each margin of the 1,568×1,568-pixel images. This process allows each high-resolution image to be cropped into divisible integer numbers of 8×8-pixel, 16×16-pixel, 32×32-pixel or 64×64-pixel small-patch.

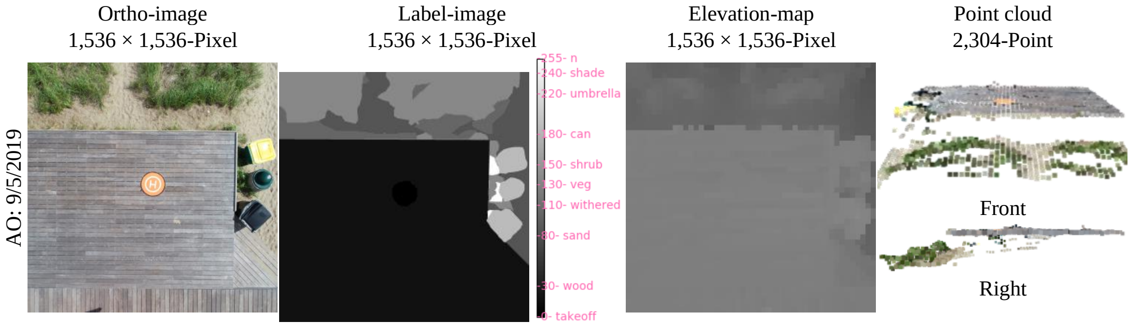


Figure 75 Testing dataset of ortho-image, label-image and elevation-map pair

7.2.2 Small-patch Dataset Creation

7.2.2.1 Patch size and number

A high-resolution 1,536×1,536-pixel is 6 times larger than a low-resolution 256×256-pixel. As a result, the high-resolution images cannot be directly used in training a deep learning model. The author proposed to disassemble the high-resolution ortho-image and label-image pair into multiple overlapping patch pairs with size 8×8-pixel, 16×16-pixel, 32×32-pixel or 64×64-pixel. Figure 76 shows the example of these four different small sized patch pairs of ortho-images and label-images.

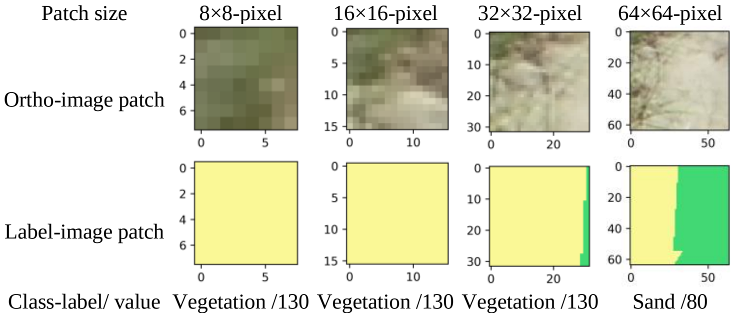


Figure 76 Examples of ortho-image and label-image patches

In addition, when cropping these small-patches, the strides are set as 4, 8, 16 and 32-pixel respectively. Moreover, in order to make the proposed deep learning model more robust in different image orientations, the high-resolution ortho-images and label-images are planned to rotate 90, 180 and 270 degrees to increase datasets by four times. Table 19 listed the number of small-patch datasets from a 1,536×1,536-pixel ortho-image and label-image pair.

Table 19 Dataset Parameters

Patch Sizes	*Strides*	*Rows*	*Columns*	*Num.*	*Num. after 4-rotation*
8×8	4	383	383	146,689	586,756
16×16	8	191	191	36,481	145,924
32×32	16	95	95	9,025	36,100
64×64	32	47	47	2,209	8,836

7.2.2.2 Dataset shape

An ortho-image acquired by the drone has RGB 3-channel. Considering that the color textures are important in distinguishing different objects on a construction site, the proposed deep learning model uses the color ortho-image patches as model input data. Therefore, using a high-resolution ortho-image can produce the model training datasets with $shape\ (586756,8,8,3)$, $shape\ (145924,16,16,3)$, $shape\ (36100,32,32,3)$, or $shape\ (8836,64,64,3)$, where the first number is the quantity of the small-patches.

A label-image generated from the "Label-App" only has one channel. Disassembling a high-resolution label-image can produce the small-patch datasets with $shape\ (586756,8,8,1)$, $shape\ (145924,16,16,1)$, $shape\ (36100,32,32,1)$, or $shape\ (8836,64,64,1)$. Thus, the maximum frequency class-label /value in each small-patch is determined and set as the class-label/value for each label-image patch. For example, in Figure 76 the "green" region is bigger than the "yellow" region of the 64×64-pixel label-image patch, thus, the class-label "sand" /value "80" is assigned for that small-patch. By doing that, the small-patch datasets are transformed into class vector (integers), such as $[130, 95, \dots, 130]$ with $shape\ (586756,1)$, $shape\ (145924,1)$, $shape\ (36100,1)$, or $shape\ (8836,1)$. Additionally, the class vector needs to be converted to binary class matrix with $shape\ (586756,256,1)$, $shape\ (145924,256,1)$, $shape\ (36100,256,1)$, or $shape\ (8836,256,1)$ as the model training datasets (Chollet 2015). For example, a integer "130" is converted to a binary class vector $[0.0_0, 0.0_2, \dots, 1.0_{130}, \dots, 0.0_{255}]$ with $shape\ (256, 1)$, and a class vector is converted to a binary class matrix with $shape\ (Num.\, of\ Small\_Patches, 256{,}1)$.

7.3 Vegetation Identifying and Removing Algorithm Design

7.3.1 Vegetation Identifying Deep Learning Model Architecture

7.3.1.1 Convolution Neural Network Architecture Design

The CNN-based image classification model architecture is presented in Figure 77, which includes a feature learning block and a classification block. In the feature learning block, three convolution layers learn the ortho-image patches (model input) as feature-maps (layer outputs). Three max pooling layers reduce feature-maps' (layer inputs) size to its half-size as their layer outputs without losing important features. For example, the 8×8-pixel, 16×16-pixel, 32×32-pixel and 64×64-pixel patches are resized down to 1×1-pixel, 2×2-pixel, 4×4-pixel and 8×8-pixel patches respectively after the 3rd max pooling layer.

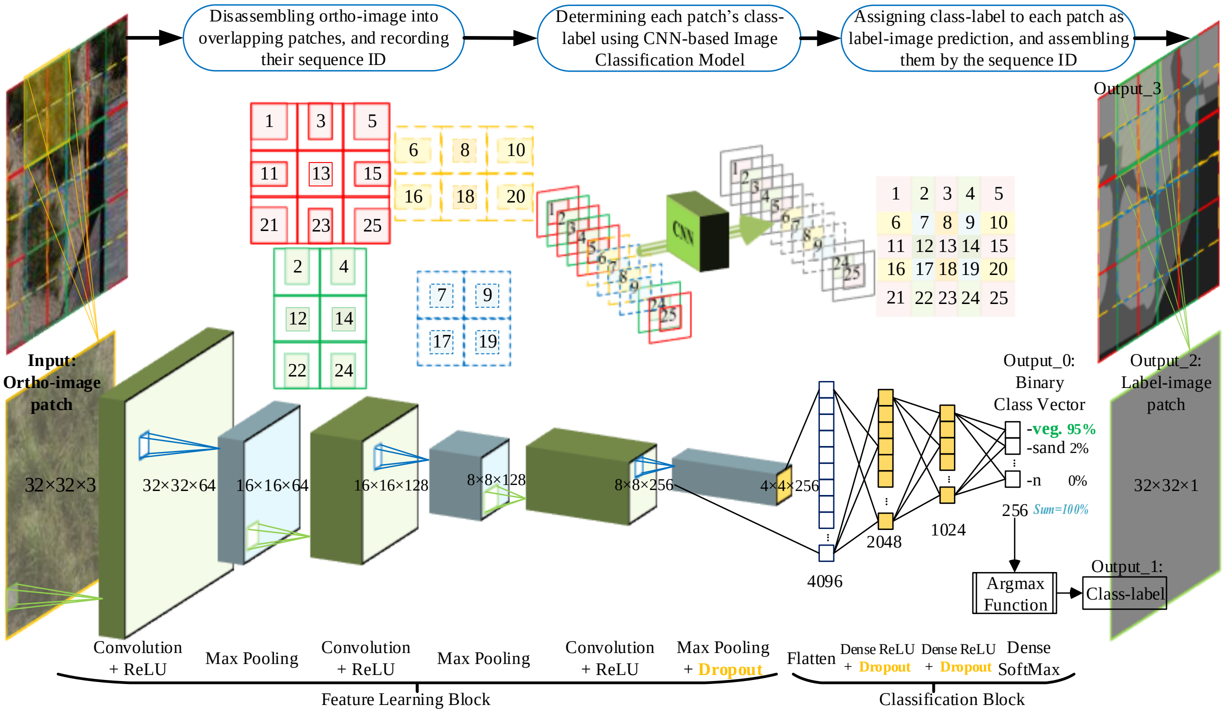


Figure 77 CNN-based image classification model with 32×32-pixel patch

In the classification block, the flatten layer transforms the feature-map (layer input) into a feature-vector (layer output), which can be used in the classification block. Three fully connected layers (also known as dense layers) transform feature-vectors (layer inputs) to a binary class vector as a model prediction output for each input ortho-image patch.

7.3.1.2 Convolutional Neural Network Model Layers Setup

The detailed model layers for the four different patch sizes are shown in Table 20, where the type of layers is described in the Keras 2.3 style (Chollet 2015). After each convolutional layer and dense layer, there is an activation function (layer) which performs the non-linear transformation of the input features from the previous convolutional layers or dense layers (Dettmers 2015). As the model input datasets normalize from value range [0,255] to [0.0,1.0] by dividing them by 255, the activation function should progressively change from 0.0 to 1.0 with no discontinuity. Therefore, the Rectified Linear Unit activation function (ReLU), $f(x) = max(0, x)$, is used in hidden layers. Because the ReLU function does not always output a non-zero value, which results in less neurons being utilized and less dependence between features (Nair and Hinton 2010), it is faster than the Sigmoid activation functions. In addition, the SoftMax activation function is used in the 3rd dense layer to calculate the probabilities of the 256 class-labels in the binary class matrix/vector. Finally, the dropout layers randomly set half of the input units to "0" at each update during training time which helps prevent model overfitting (Chollet 2015).

Table 20 Model Layer Parameters

Model Architecture for 8×8, 16×16, 32×32 and 64×64-pixel Patches					*Output Shapes for Each Patch*				
					Row × Column				
Blocks	***Layer (Type and filter size)***	***Stride***	***Padding***	***Activation***	***8×8***	***16×16***	***32×32***	***64×64***	***Channels***
Input	input_1 (Input Layer)	-	-	-	8×8	16×16	32×32	64×64	3
Feature learning block	conv2d_1 (64,Conv2D 3×3)	1	same	ReLU	8×8	16×16	32×32	64×64	64
	max_pooling2d_1 (Max Pooling 2×2)	2	-	-	4×4	8×8	16×16	32×32	64
	conv2d_2 (128,Conv2D 3×3)	1	same	ReLU	4×4	8×8	16×16	32×32	128
	max_pooling2d_2 (Max Pooling 2×2)	2	-	-	2×2	4×4	8×8	16×16	128
	conv2d_3 (256,Conv2D 3×3)	1	same	ReLU	2×2	4×4	8×8	16×16	256
	max_pooling2d_3 (Max Pooling 2×2)	2	-	-	1×1	2×2	4×4	8×8	256
	dropout_1 (Dropout 0.5)	-	-	-	1×1	2×2	4×4	8×8	256
Classification block	flatten_1 (Flatten)	-	-	-	256	1,024	4,096	16,384	-
	dense_1 (Dense)	-	-	ReLU	256	1,024	2,048	4,096	-
	dropout_2 (Dropout 0.5)	-	-	-	256	1,024	2,048	4,096	-
	dense_2 (Dense)	-	-	ReLU	256	512	1,024	1,024	-
	dropout_3 (Dropout 0.5)	-	-	-	256	512	1,024	1,024	-
Output	dense_3(Dense)	-	-	SoftMax	256				-

7.3.1.3 Convolutional Neural Network Compiling Configuration

For compiling the developed CNN-based model, the author use "adam" as the optimizer, and use "categorical_crossentropy" as the loss function (Chollet 2015). "Validation_split" is set to 0.05, which means that 95% of small-patch datasets are used for training the model and 5% of small-patch datasets are used for model validation. The “early stopping” configuration is set as “EarlyStopping(monitor='val_accuracy', patience=5)” which means the model training will be stopped as monitored quantity of validation accuracy had stopped improving during the past 5 epochs (Chollet 2015).

7.3.2 Ortho-image Disassembling and Label-image Assembling Algorithm

7.3.2.1 Small-patch Label-image Prediction

In the developed CNN-based image classification model (see Figure 77), for an ortho-image patch input, the model prediction output is a binary class vector (“Output_0”), which contains the probability values of the 256 unique class-labels. With this model prediction, three post-processes need to be conducted to get a high-resolution segmented label-image result.

First, the “Argmax” function returns the index of the maximum probability value of the binary class vector, this index is the class-label/value prediction (“Output_1”) for the input ortho-image patch. For example, the “veg” is the class-label prediction for the input ortho-image patch in Figure 77, because it has the maximum frequency/percent 95% among the 256 class-labels.

Second, the class-label /value prediction is assigned to each pixel of the small-patch as the label-image patch prediction (“Output_2”) for the corresponding input ortho-image patch.

Third, the small-patch will be used as a part of the high-resolution label-image prediction result (“Output_3”).

7.3.2.2 Ortho-image Disassembling and Label-image Assembling Algorithm Design

Figure 77 shows the workflow of the high-resolution ortho-image overlapping disassembling and high-resolution label-image assembling algorithm, which makes the proposed CNN-based image classification model works with the high-resolution image instead of resizing the original image down to the low-resolution.

This algorithm disassembles the ortho-image into several overlapping small-patches and records their locations in their sequence ID. The number of small-patches is determined by Eq. 14. When assembling the high-resolution label-image prediction, the small-patches are considered as corner patches, edge patches or regular patches, and only the selected region (marked as filled rectangles) of each patch will be used in the high-resolution label-image prediction. For example, 95-row and 95-column overlapping small-patches with size 32×32-pixel are produced from a high-resolution 1,536×1,536-pixel ortho-image for generating the 32×32-pixel label-image patch predictions; the same number of small-patches are required to assemble a high-resolution 1,536×1,536-pixel label-image prediction, and for each regular 32×32-pixel label-image patch prediction, the used region is only a quarter of the regular patch, which is the filled 16×16-pixel patch in Figure 77. In addition, with this developed algorithm, each 16×16-pixel ortho-image patch is linked with a 16×16-pixel label-image patch prediction through a class-label prediction. Therefore, using this method of CNN-based image classification model and the overlapping disassembling and assembling algorithm with 8×8-pixel, 16×16-pixel, 32×32-pixel or 64×64-pixel patches (useful region 4×4-pixel, 8×8-pixel, 16×16-pixel or 32×32-pixel in each regular patch) is similar to resizing a high-resolution 1,536×1,536-pixel ortho-image down to a 383×383-pixel, 191×191-pixel, 95×95-pixel or 47×47-pixel low-resolution image for image segmentation.

$$Num. of\ Small\_Patches = (2 \times \frac{Image\ Height}{Patch\ Size} - 1) \times (2 \times \frac{Image\ Width}{Patch\ Size} - 1) \qquad \textbf{Eq. 14}$$

7.3.3 Vegetation Removing Algorithm Using Label-image

There are two approaches for removing the vegetations’ height from the raw surveying result using the identified vegetation blocks in the label-image. One measures an average height of vegetation blocks on the construction site, and then directly subtract this value from the raw elevation values of the vegetation blocks. Another searches the neighboring ground blocks on the label-image, then interpolates these surroundings’ elevation values as the “real” ground elevation under the vegetations. In this research project, the vegetation removing algorithm (see Figure 78) is based on the second approach, because it is more convenient for automatically determining the ground elevation without any manual participation, and the result will be closer to the “true” ground elevation than the prior option.

In detail, the $\boldsymbol{VEG\_REMOVING\_IN\_ROW\_THEN\_COL\_TRAVERSE}$ algorithm traverse the high-resolution label-image in the row-column-row-loops (see Figure 79). In each row-loop, the $\boldsymbol{SEARCH\_VEG\_REPLACE\_GROUND}$ algorithm uses a size adjustable window, which can be extended in the row direction, to search the minimum number of “ground” class-labels. Similarly, in each column-loop, the adjustable window is changed in column direction only. When sufficient “ground” class-labels appear in the search window, the ***SEARCH_VEG_REPLACE_GROUND*** algorithm replaces the current “vegetation” patch’s elevation value with the average elevation from the searched neighboring “grounds.” In addition, the label-image patch is updated with the “ground” class-label, and the ortho-image patch is marked with a specific color as well.

```
$\boldsymbol{SEARCH\_VEG\_REPLACE\_GROUND}(row_{index}, col_{index}, ele\_map, label\_image, ortho\_img, stride, qsize, in\_row_{bool})$
$\boldsymbol{Initial}\quad win_{size\_max} = \boldsymbol{MIN}(image_{Width}, image_{Height}), ratio_q = 10, counter_{min} = qsize \times qsize \times ratio_q$
$\boldsymbol{if}\ label\_image[row\_index, col\_index]\ \boldsymbol{in}\ veg\_label\_list$:
  $\boldsymbol{for}\ win_{size}\ \boldsymbol{in}\ range(0, win_{size\_max}, qsize)$  #search in adjustable window $[\mathrm{row_{low_b}}: \mathrm{row_{up_b}}, \mathrm{col_{low_b}}: \mathrm{col_{up_b}}]$
    $row_{low_b} = row_{index} - qsize \times ratio_q - win_{size} \times (1 - in\_row_{bool})$# in Col. −loop, extend the windows in the Col. direction only
    $col_{low_b} = col_{index} - qsize \times ratio_q - win_{size} \times (in\_row_{bool})$# in Row. −loop, extend the windows in the Row direction only
    $row_{up_b} = row_{index} + qsize \times ratio_q + win_{size} \times (1 - in\_row_{bool}) + 1$
    $col_{up_b} = col_{index} + qsize \times ratio_q + win_{size} \times (in\_row_{bool}) + 1$
    $label\_image_{patch} = label\_image[row_{low_b}: row_{up_b}, col_{low_b}: col_{up_b}]$
    $ele\_map_{patch} \quad = \quad ele\_map[row_{low_b}: row_{up_b}, col_{low_b}: col_{up_b}]$
    $ground_{index} = (label\_image_{patch} == ground_{label})$#return the indexes of $\mathrm{ground_{label}}$ in the $\mathrm{label\_image_{patch}}$
    $\boldsymbol{if}\ \boldsymbol{SUM}(ground_{index}) >= \ counter_{min}$#return the num. of all True elements in $\mathrm{ground_{index}}$
      # if the num. of $\mathrm{ground_{label}}$ in the windows is enough, then remove current vegetation patch $[\mathrm{row_{index}}: \mathrm{row_{index}} + \mathrm{stride}, \mathrm{col_{index}}: \mathrm{col_{index}} + \mathrm{stride}]$
      $ground_{ele} = \boldsymbol{MEAN}(ele\_map_{patch}[ground\_index])$
      $ele\_map[row_{index}: row_{index} + stride, col_{index}: col_{index} + stride] = ground_{ele}$
      $label\_image[row_{index}: row_{index} + stride, col_{index}: col_{index} + stride] = ground_{label}$
      $ortho\_img[row_{index}: row_{index} + stride, col_{index}: col_{index} + stride] = (255{,}125{,}255)$# mark the ortho_image
      $\boldsymbol{break}$

$\boldsymbol{VEG\_REMOVING\_IN\_ROW\_THEN\_COL\_TRAVERSE}(ele\_map, label\_image, ortho\_img, qszie)$
$\boldsymbol{Initial}\quad row, col = 0{,}0;\ imageH, imageW = ele\_map.shape;\ H, W = 0{,}0;\ i = 0\ ;\ stride = qsize/4$ # traversal stride
$\boldsymbol{while}\ i < image_{Height} + image_{Width}$# max num. of Row & Col −  loops
  $\boldsymbol{while}\ col + W <= image_{Width}$:
    $\boldsymbol{if}\ col + W < image_{Width}$#process Row − loop
      $row_{index} = row + H;\ col_{index} = col + W$
      $\boldsymbol{SEARCH\_VEG\_REPLACE\_GROUND}(row_{index}, col_{index}, ele\_map, label\_image, ortho\_img, stride, qsize, in\_row_{bool} = \boldsymbol{True})$# remove in Row
      $W = W + stride$ # move to next element in the Row
    $\boldsymbol{else}$#completed the Row, move to Col − loop
      $W = 0;\quad H = stride$ # skip the already processed 1st element in the Row − loop
      $\boldsymbol{while}\ row + H < image_{Height}$: #process Col − loop
        $row\_index = row + H;\ \ col\_index = col + W$
        $\boldsymbol{SEARCH\_VEG\_REPLACE\_GROUND}(row_{index}, col_{index}, ele\_map, label\_image, ortho\_img, stride, qsize, in\_row_{bool} = \boldsymbol{False})$# remove in Col.
        $H = H + stride$ # to next element in the Col
      $row = row + stride$ # move to the next row
      $H = 0$
      $\boldsymbol{break}$# complete the current col
  $col = col + stride$ # move to the next col
  $i = i + 1$
```

Figure 78 Vegetation removing algorithm

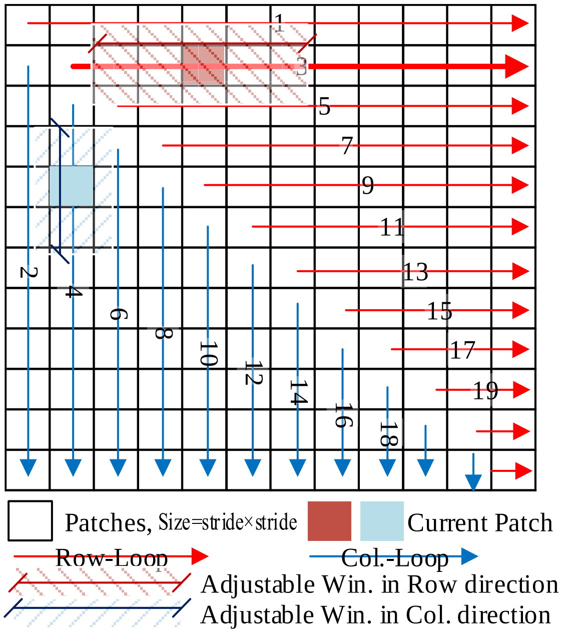

1
3
5
7
9
11
13
15
17
19
2
4
6
8
10
12
14
16
18
Patches, Size=stride×stride
Current Patch
Row-Loop
Col.-Loop
Adjustable Win. in Row direction
Adjustable Win. in Col. direction

Figure 79 Row-column-row-loop for traversing a label-image

7.4 Vegetation Identifying Experiments

7.4.1 Experiment Dataset

7.4.1.1 Experiment Site

Figure 80 shows the overall condition of the experimental site, which is located at Atwater Park (Shorewood, WI, USA).

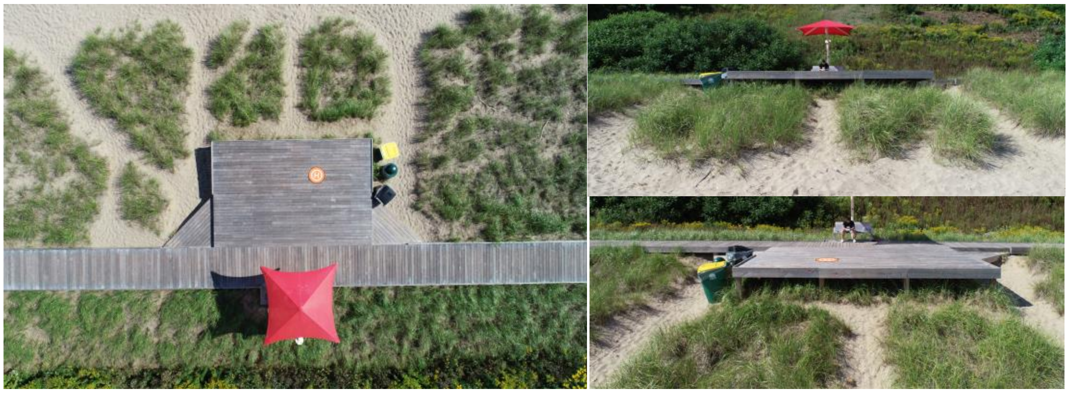


Figure 80 Vegetation identifying experiment site

7.4.1.2 Ortho-images and Label-images

Ten high-resolution ortho-image and label-image pairs are shown in Figure 81.

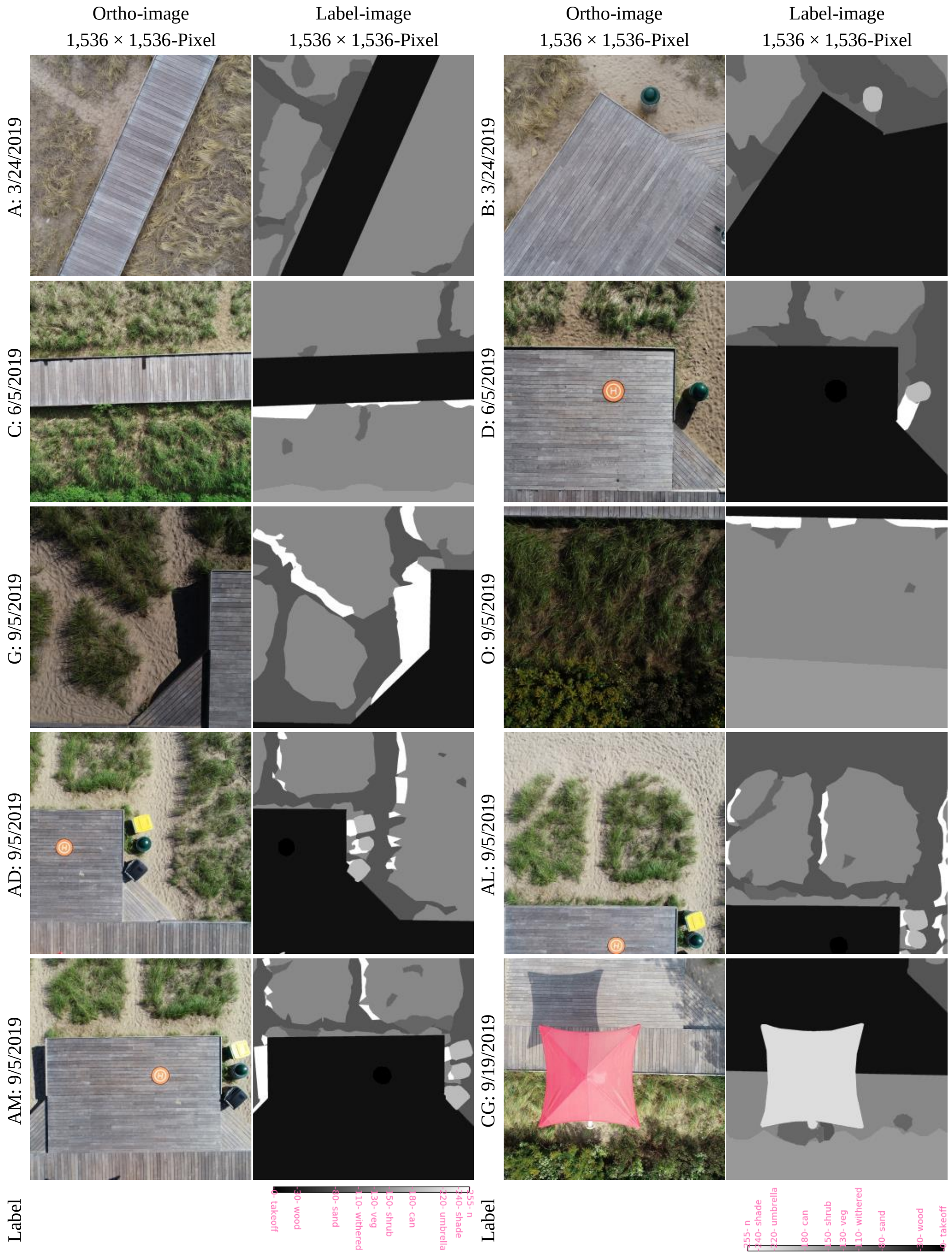
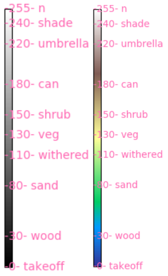


Figure 81 Training and validation datasets

These ortho-images were captured during different seasons and they contain ten categories of different objects and surfaces (see Table 21). For the vegetation blocks, in data A and B, the vegetation had not recovered yet; in data C and D, the vegetation was growing; and in data G, O, AD, AL, AM and CG, the vegetation was fully grown, and their heights were around 2 feet (60.96 cm). In addition, the ortho-image and label-image pair in Figure 75 is used to test the well trained model.

The small-patch datasets for training and testing the CNN-based image classification model are created followed by the rules stated in the dataset creation section. Furthermore, each label-image patch is assigned with a class-label. For example, a 32×32-pixel label-image patch has 1,024 elements in total. If 513 of them have the value “220”, then this patch has the “umbrella” class-label.

Table 21 Class-label Definitions

Class-label	*8-bit Grayscale value*	*Definitions*	*Gray/Terrain Colormap*
n	255	Other undefined objects	 255- n 240- shade 220- umbrella 180- can 150- shrub 130- veg 110- withered 80- sand 30- wood 0- takeoff 255- n 240- shade 220- umbrella 180- can 150- shrub 130- veg 110- withered 80- sand 30- wood 0- takeoff
shade	240	Shades on ground	
umbrella	220	Red umbrella surface	
can	180	Garbage cans	
shrub	150	Shrub surface	
veg	130	Vegetation surface	
withered	110	Withered vegetation surface	
sand	80	Ground surface, includes sand and soil	
wood	30	Wooden surface, includes platform and path	
takeoff	0	Drone takeoff and landing pad	

7.4.2 Vegetation Identifying Deep Learning Model Training and Validation

7.4.2.1 Model Training Configuration

The model training parameters including dataset numbers, batch sizes, and epochs which are listed in Table 22.

Table 22 Vegetation Identifying Model Training Parameters and Results

	Patch Size Trials				*Training Epoch Trials*	
Patch Sizes	*Datasets (Validation Split = 0.05)*			*Batch Sizes*	*EarlyStopping(monitor='val_accuracy', patience=5), Epochs=50*	
	Total Num.	*Training*	*Validation*		*w/ Early Stopping*	*w/o Early Stopping*
8	5,867,560	5,574,182	293,378	256	13	50
16	1,459,240	1,386,278	72,962	256	24	50
32	361,000	342,950	18,050	256	14	50
64	88,360	83,942	4,418	256	13	50

7.4.2.2 Training and Validation with Early Stopping

The results of training loss, training accuracy, validation loss and validation accuracy with “early stopping” for four different pixel size are shown in Figure 82, which were stopped at different epochs (see Table 22). The 64×64-pixel and 8×8-pixel patch trial stopped at the 13th epoch and were the earliest trials, and the 32×32-pixel patch stopped at the 14th epoch. The 16×16-pixel patch took the most epochs for the validation accuracy to reach stable.

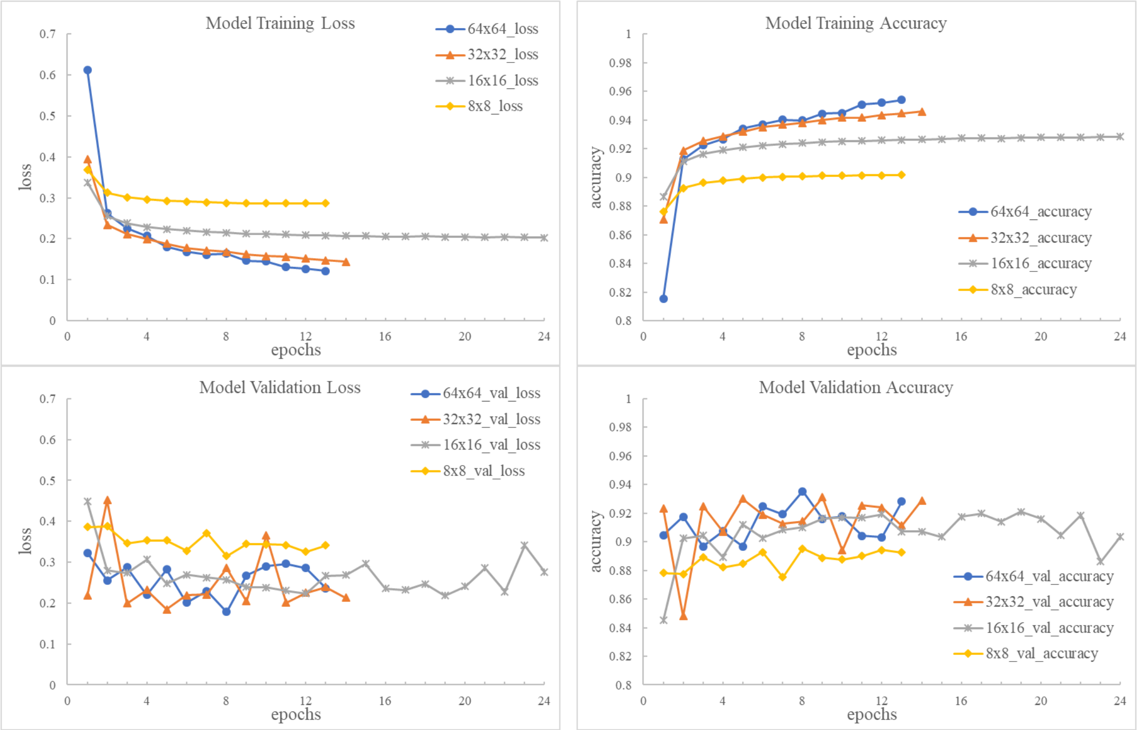


Figure 82 Training and validation results I: loss and accuracy w/ early stopping trials

Furthermore, The validation results are shown in Figure 83, where the “ground truths” are the class-labels used in training the model, the predictions are the class-label predictions generated from the trained CNN-based image classification model. The smaller patches 8×8-pixel and 16×16-pixel have more chance to have incorrect class-label prediction. The larger patches size 32×32-pixel and 64×64-pixel have more chance to form complex label-image patches with multiple class-labels in one label-image patch, and the class-label predictions are more likely correct. This result matches the training accuracy and validation accuracy results in Figure 82, where the 32×32-pixel and 64×64-pixel have the better training accuracy and validation accuracy than the 8×8-pixel and 16×16-pixel. However, it is hard to conclude either 32×32-pixel or 64×64-pixel has the best performance based on the “early stopping” loss and accuracy plots.

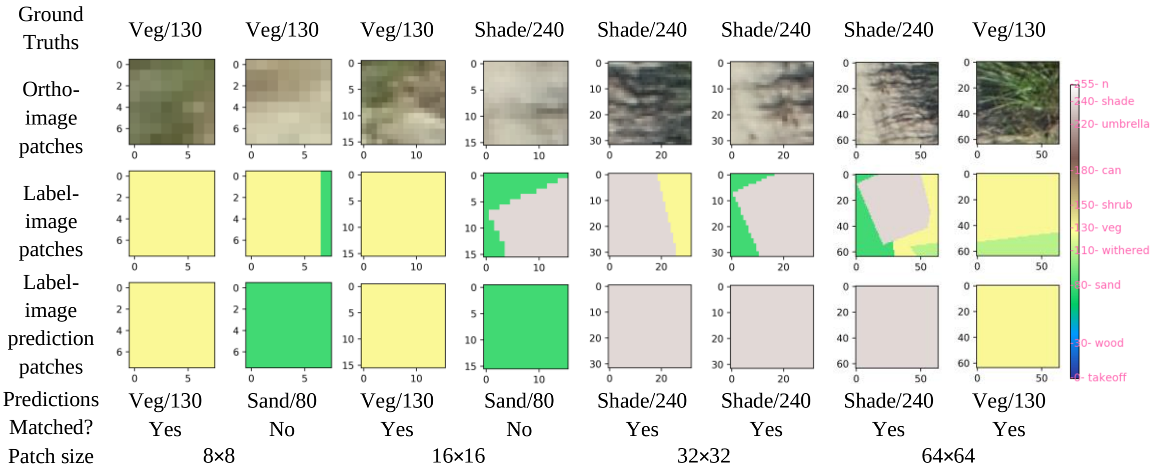


Figure 83 Training and validation results II: model predictions of data AM w/ early stopping trials

7.4.2.3 Training and Validation with 50-epoch

The model training was conducted without “early stopping” as well. The 50-epoch model training and validation results of the four different patch sizes are shown in Figure 84. The 64×64-pixel patch has the largest model training accuracy of 0.9908 at the 50th epoch, which is better than 0.9540 of the “early stopping” trial. However, the 32×32-pixel patch has the best validation accuracy of 0.9304 at the 50th epoch, which is better than 0.9288 at the “early stopping” trial and also better than the validation accuracy of 0.9219 for the 64×64-pixel patch at the 50th epoch. In addition, the 32×32-pixel patch has the smallest validation loss as well. Thus, the author concludes that using the 32×32-pixel patch for the developed CNN-based image classification model has the best model training and validation performance, followed by the 64×64-pixel patch and 16×16-pixel patch. The smallest 8×8-pixel patch, however, has the worst performance.

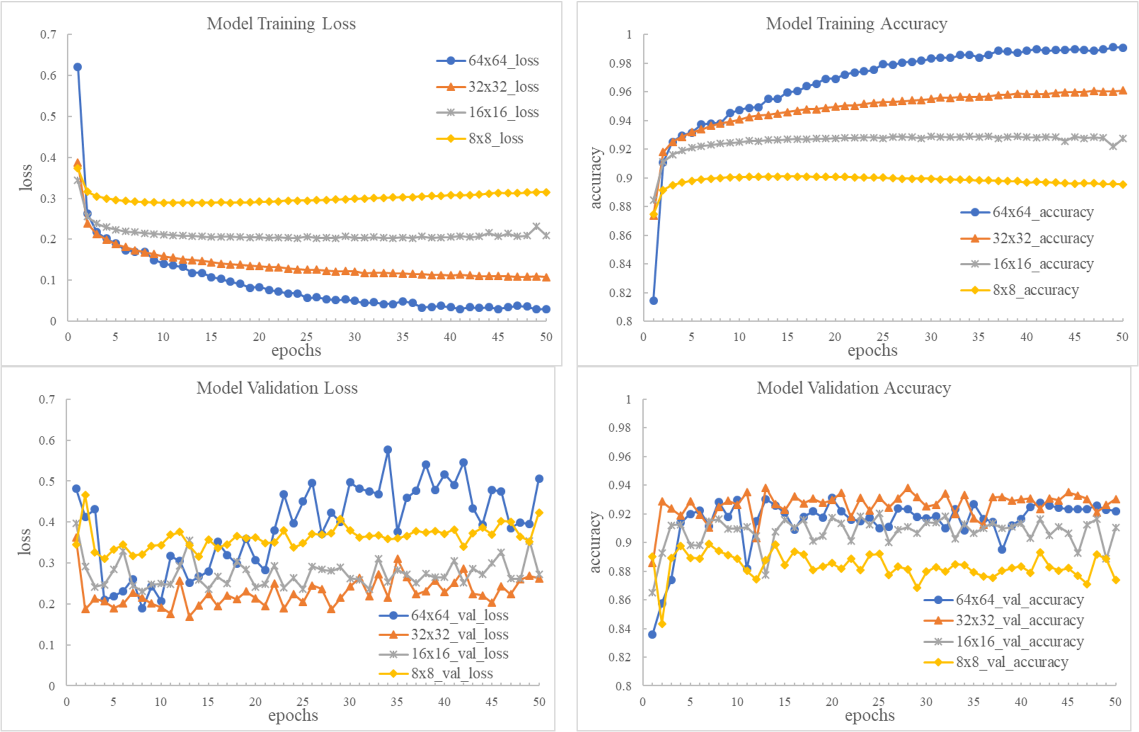


Figure 84 Training and validation results III: loss and accuracy of 50-epoch

7.4.2.4 Model Training Discussion

The extra training epochs after the “early stopping” point have no impact on the smaller 8×8-pixel and 16×16-pixel patches based on the training accuracy and training loss in Figure 84, but they have positive impacts on the 32×32-pixel and 64×64-pixel patches. However, the validation accuracy and loss have nonsignificant improvement as the training epochs increase in the 32×32-pixel and 64×64-pixel patch trials.

The cause of this issue can be explained from the assembled high-resolution model validation results in Figure 85. Compared to the “early stopping”, the 50-epoch has incorrect model predictions on the wooden platform of data AM and G, but it has better model predictions for the “withered” class-label in data A and CG. Thus, the overall model validation accuracy is maintained around 93% for the 32×32-pixel patch.

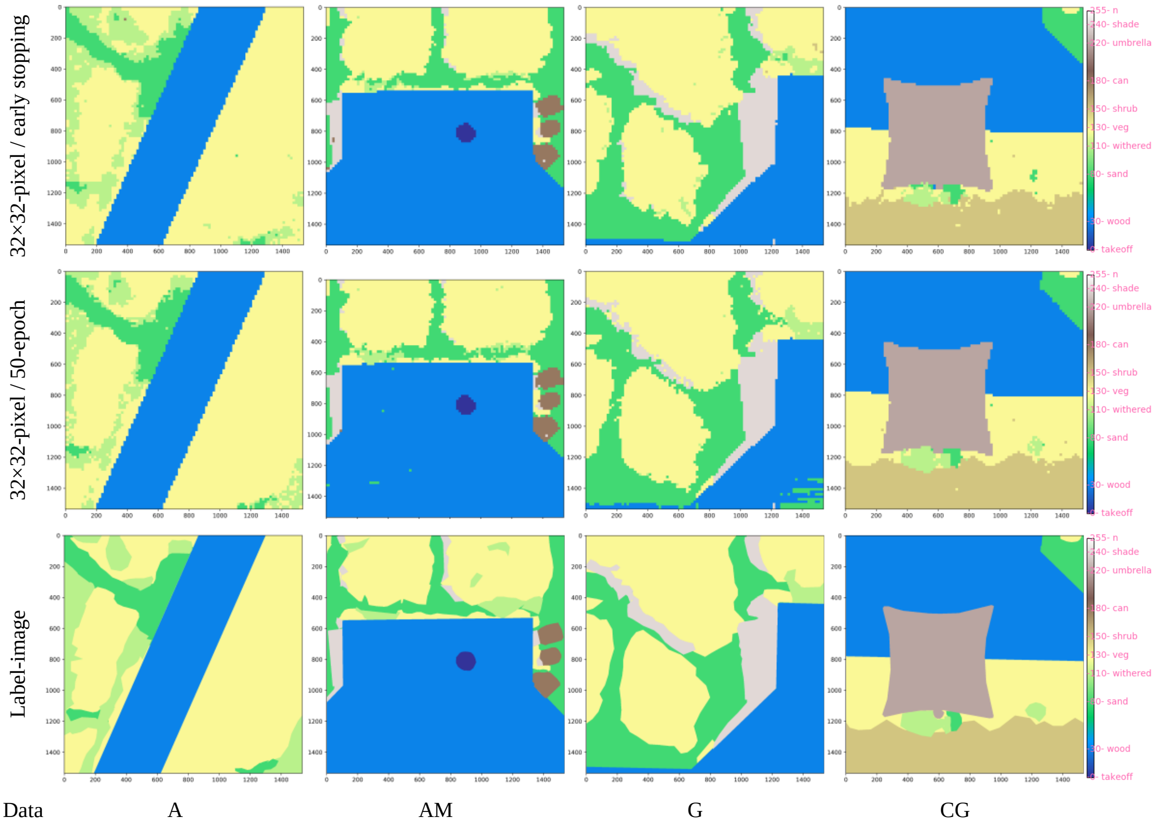
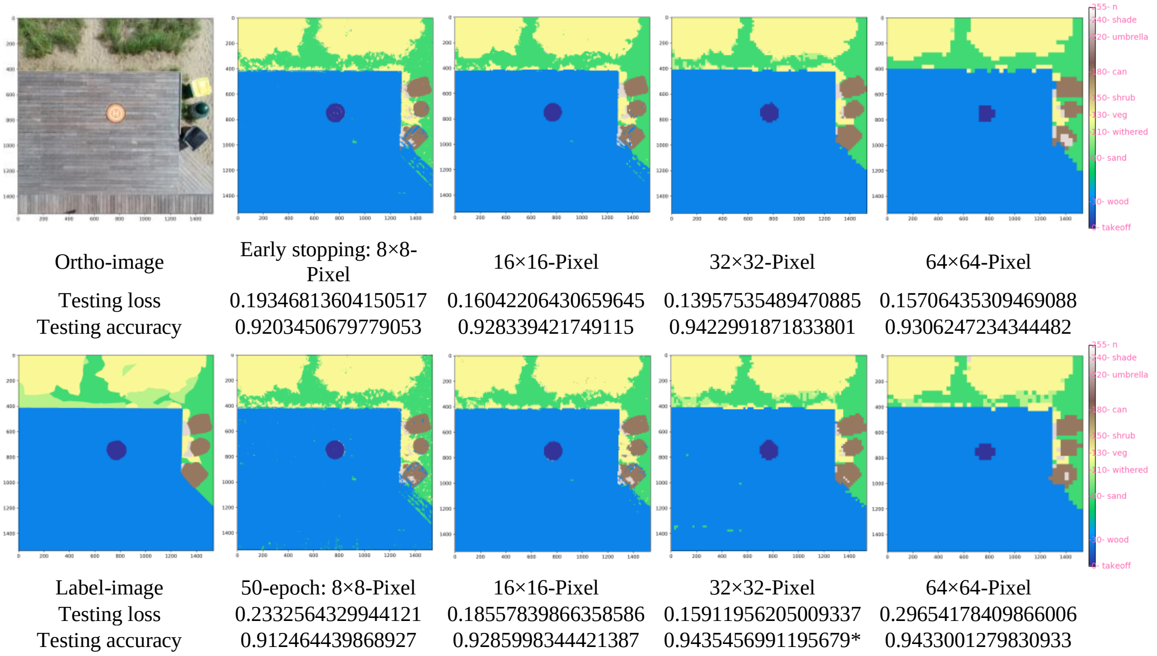


Figure 85 Training and validation results IV: assembly of model predictions

7.4.3 Vegetation Identifying Testing Results

The trained “early stopping” and 50-epoch models were tested with the high-resolution data AO in Figure 75, which was disassembled as the model training datasets as well. For example, the 32×32-pixel patch trial was tested with the 9,025 small-patch datasets. The results of model testing loss and testing accuracy and the assembled high-resolution label-image predictions are shown in Figure 86. The best testing accuracy of 0.9435 is the 32×32-pixel patch with 50-epoch, the second-best testing accuracy of 0.9433 is the 64×64-pixel patch with 50-epoch, and the third-best testing accuracy of 0.9423 is the 32×32-pixel patch with “early stopping.” These results were the same as the model validation results, because the “wood” class-label was getting worse, while the “withered” class-label was getting better when the training epoch was increasing.



Ortho-image	Early stopping: 8×8-Pixel	16×16-Pixel	32×32-Pixel	64×64-Pixel
Testing loss	0.19346813604150517	0.16042206430659645	0.13957535489470885	0.15706435309469088
Testing accuracy	0.9203450679779053	0.928339421749115	0.9422991871833801	0.9306247234344482
Label-image	50-epoch: 8×8-Pixel	16×16-Pixel	32×32-Pixel	64×64-Pixel
Testing loss	0.2332564329944121	0.18557839866358586	0.15911956205009337	0.29654178409866006
Testing accuracy	0.912464439868927	0.9285998344421387	0.9435456991195679*	0.9433001279830933

Figure 86 Vegetation identifying testing results

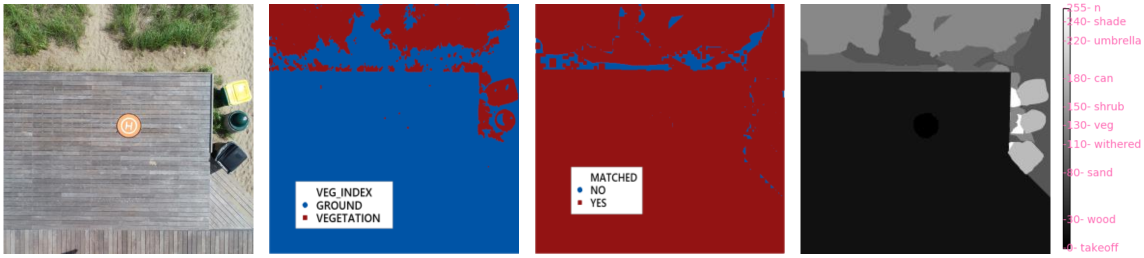
7.4.4 Vegetation Identifying Evaluation and Discussion

Visually, in Figure 86 the 32×32-pixel patch with “early stopping” had the best prediction result and followed by the 32×32-pixel patch with 50-epoch. The 64×64-pixel patch with the 50-epoch was a reasonable result, too, but the 32×32-pixel patch was more accurate in the objects’ boundaries. The number of each category of class-labels of the manually crafted label-image and the assembled high-resolution label-image prediction result (32×32-pixel patch with “early stopping”) are summarized in Table 23 , where 93.57% (2,207,641 of 2,359,296 pixels) class-labels were exactly matched between them. This accuracy is nonsignificant to the small-patch testing accuracy (94.23%). Thus, the developed overlapping small-patch disassembling and assembling algorithm was efficient as the result of directly processes high-resolution images.

Table 23 Class-label Statistic Summary

Class-label	*value*	*Label-image frequency/percent*		*Label-image Prediction frequency/percent*		*Label-image w/ vegetation removing frequency/percent*	
n	255	-	-	-	-	-	-
shade	240	10,983	0.47%	8,192	0.35%	-	-
umbrella	220	-	-	-	-	-	-
can	180	70,597	2.99%	73,856	3.13%	73,856	3.13%
shrub	150	-	-	256	0.01%	-	-
veg	130	376,330	15.95%	427,456	18.12%	-	-
withered	110	119,551	5.07%	26,624	1.13%	-	-
ground	95	-	-	-	-	728,832	30.89%
sand	80	223,045	9.45%	266,304	11.29%	-	-
wood	30	1,540,072	65.28%	1,539,456	65.25%	1,539,456	65.25%
takeoff	0	18,718	0.79%	17,152	0.73%	17,152	0.73%
Sum	-	2,359,296	100.00%	2,359,296	100.00%	2,359,296	100.00%

Additionally, the class-label prediction errors (6.43% of unmatched) were mapped in pixel coordinate as shown in Figure 87. The majority unmatched class-labels were appeared on the "withered" region of the manually crafted label-image. In this research project, the "withered" vegetation class-label is defined as a ground surface category between the "sand" and normal "vegetation." However, it is hard to distinguish the "withered" and "sand" in the ortho-image, and most "withered" blocks are "sand" blocks in reality in the manually crafted label-images (see Figure 81). In the early stage of this research project, the author obtained a 0.9646 validation accuracy and 0.9673 testing accuracy without adding the "withered" class-label to these label-image datasets. Thus, most class-label errors can be avoided by considering the "withered" and "sand" as the same ground surface class-label.



Ortho-image Vegetation index method Mapped prediction error Label-image

Figure 87 Vegetation identifying results: vegetation index and mapped prediction error

Furthermore, the vegetation index $ExG = 2G - R - B$ (Anders et al. 2019) result is shown in Figure 87, where 15.32 % (361,221 of 2,357,926) pixels had been identified as vegetation. That result is close to the 15.95% of "veg" class-label in the manually crafted label-image, but it also contains a large number of incorrect results from the yellow and green garbage cans. Thus, the vegetation index method is not suitable for the detailed vegetation detection at a complicatedly textured construction site with other green and yellow textured objects.

Therefore, the author concluded that the developed CNN-based image classification model with 32×32-pixel ortho-image patch input data had a good accuracy (93%) to identify the objects on the construction site using the drone-based high-resolution ortho-image.

7.5 Vegetation Removing Experiment

7.5.1 Vegetation Removing Algorithm Configuration

The $\boldsymbol{VEG\_REMOVING\_IN\_ROW\_THEN\_COL\_TRAVERSE}$ algorithm and $\boldsymbol{SEARCH\_VEG\_REPLACE\_GROUND}$ algorithm in Figure 78 were programmed using Python 3.6.8. The vegetation removing experiments were conducted with the 32×32-pixel "early stopping" prediction result. The initial search window size is $(2\,qszie \times ratio_q) \times (2\,qszie \times ratio_q)$=(2×32×10)×(2×32×10), where $qsize$ is the patch size of 32-pixel used in the CNN-based image classification model, and the $ratio_q$ is set as 10. The max search windows size is dependent on win_{size_max} , which is set as half of the image width=768-pixel. The minimum required number of "ground" class-label pixels in the search window is set as $qsize \times qsize \times ratio_q$=32×32×10. In addition, the label-image traversal $stride$ is set as $qsize/4$=8-pixel, which means the high-resolution 1,536×1,536-pixel label-image is disassembled into 192-row, 192-column and 36,864 patches with a size of 8×8-pixel.

7.5.2 Vegetation Removing Testing Results

The high-resolution label-image prediction result contains 2,359,296-pixel. It includes 8,192-pixel of "shade," 256-pixel of "shrub," 427,456-pixel of "veg," which are considering as vegetation blocks that need to be removed and replaced with class-label "ground"/ value "95." It also contains 266,304-pixel of "sand," 26,624-pixel of "withered," which are considered as ground blocks and needed to update class-label to "ground".

Table 23 shows the sum number of "shade," "shrub," "veg," "sand" and "withered" in the label-image prediction is equal to the number of "ground" in the vegetation removed label-image, which confirms that the developed algorithm had successfully traversed the high-resolution label-image. In addition, the label-image and ortho-image were successfully updated after removing the vegetation, where the new "ground" blocks were marked with pink color in the ortho-image (see Figure 88).

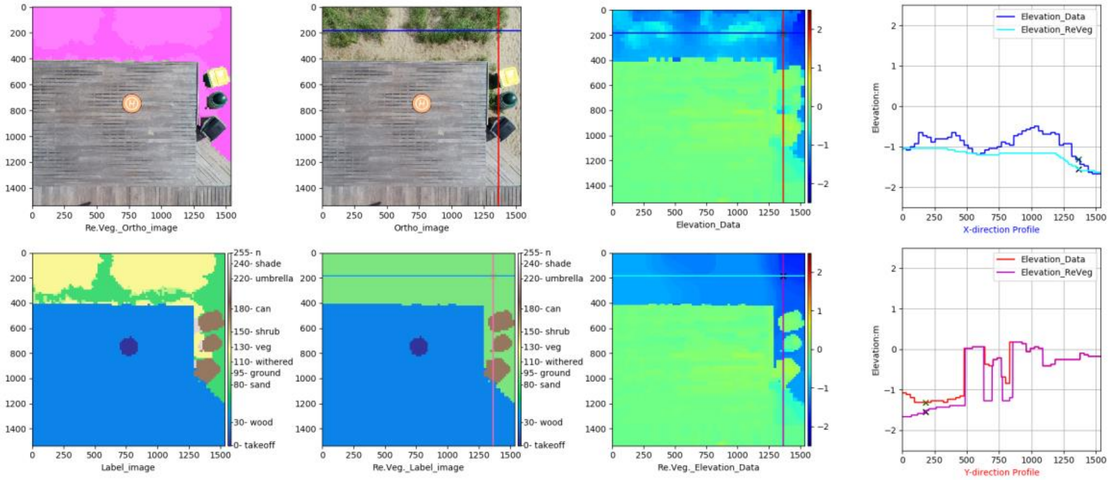


Figure 88 Vegetation removing results I: modified label-image and elevation-map

7.5.3 Vegetation Removing Evaluation and Discussion

Figure 89 shows the updated point cloud, which was generated using the selected central points of each 32×32-pixel patch of the vegetation removed ortho-image (textures) and vegetation removed elevation-map (elevation values). Among the selected 2,304-point, the vegetation category ("shade," "shrub" and "veg") has 447-point which accounts for 19.40%; the ground category ("withered" and "sand") has 267-point which accounts for 11.59%. The sample distribution of the selected 2,304-point is similar to the population distribution of the 2,359,296-pixel in the high-resolution label-image.

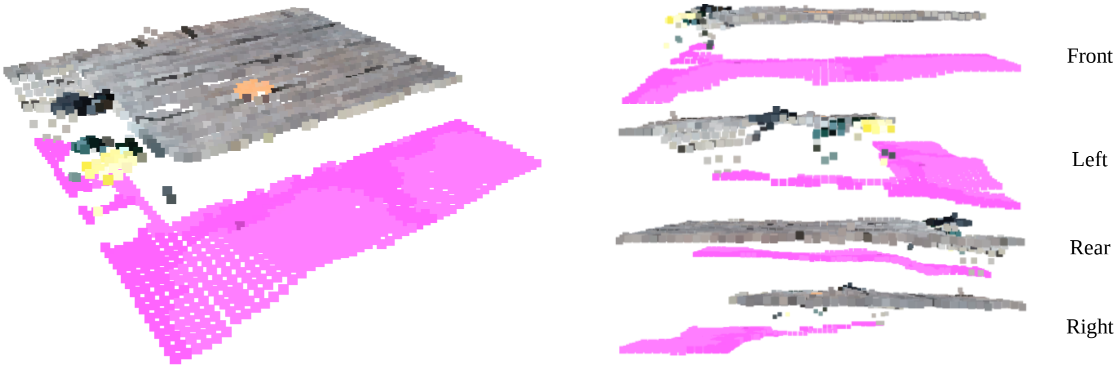


Figure 89 Vegetation removing results II: modified point cloud

Figure 90 shows the elevation differential between the original and updated elevations on ground and vegetation blocks. The non-ground and non-vegetation points (1,590) were excluded in the histogram. The larger elevation changes (≥0.7 m) occur on the edges of the wooden platform and garbage cans, where the updated elevations are more accurate than the elevations determined by the image-based method. The majority of the elevation changes on the vegetation blocks ranged from 0.1 to 0.6 m, and the maximum elevation changes on the vegetation block is 0.6-0.7 m, which is also shown as the peak point of X-profile in Figure 88. This result is similar to the measured vegetation height of 0.61 m on the experimental site.

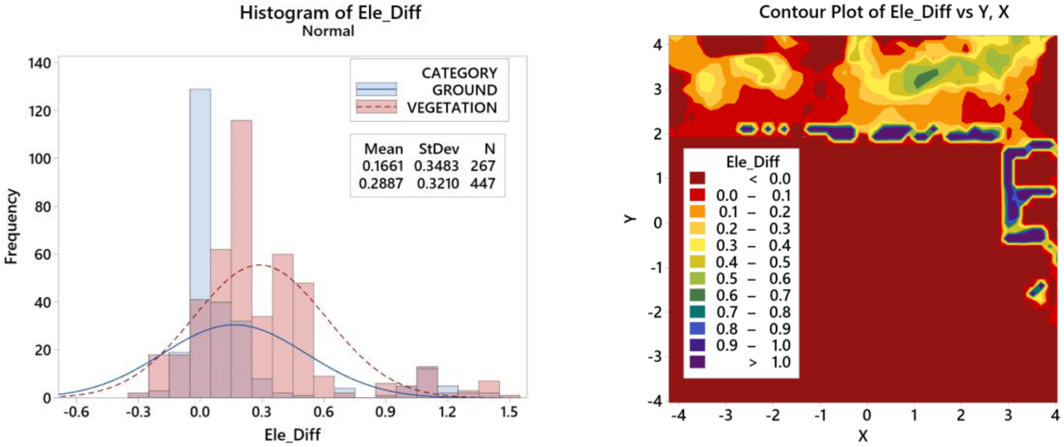


Figure 90 Vegetation removing results III: elevation differential statistic summary

There are some negative elevation changes -0.3-0.0 m on the left end of the vegetation histogram, which are the top-edge (Y>3.8 and -3.8<X< -2.2) vegetation block in the contour plot. This error appears due to this region being conflicted with the requirement of the vegetation block needing to be surrounded by ground blocks. Fortunately , this kind of error can be avoided by adding the necessary neighboring grounds, such as the central vegetation block in the data AL in Figure 81, or stitching with the other adjacent dataset. Because the ***SEARCH_VEG_REPLACE_GROUND*** algorithm can generate the correct ground elevations for the vegetation covered regions, the author concluded that the developed methods in this paper (see Figure 73) can automatically identify the vegetation and determine the ground elevation covered by the vegetations.

CHAPTER 8: CONCLUSIONS AND RECOMMENDATIONS

8.1 Summary

This research project utilized drone-based ortho-imaging to advance the image-based 3D-reconstruction method for determining the construction site elevations with the consideration of the affection of static vegetation. Major work and findings are presented in Chapters 4 to 7. A summary is provided below.

Chapter 4 presents an effective, rapid and easily-implementable two-frame-image-based 3D-reconstruction method for automatically determination of construction site elevations using drone technology. The method of input images is different from the traditional drone photogrammetry method and the classic stereo-vision method which are 2:1 scale ratio quadcopter drone-based low-high ortho-image pairs instead of the same scale image pairs. The general procedure of the developed low-high ortho-image pair-based elevation determination method includes:

1. Acquiring low-high ortho-image pairs on a construction site using drone ortho-imaging,
2. Matching pixel grid and determining elevations simultaneously by the low-high ortho-image pair pixel grid matching and elevation determination algorithms, and
3. Modeling and measuring with 2D elevation-map and 3D point cloud.

Chapter 5 discusses how to use the developed low-high ortho-image pair-based elevations determination method to acquire a construction site high-resolution ortho-image and elevation-map dataset for training the deep learning-based construction site elevation estimation model. Based on the acquired dataset, a single-frame image-based 3D-reconstruction method for construction site elevation estimation was developed and presented in Chapter 6, which only needs a drone-based ortho-image as the input. The general procedure of the ortho-image and deep learning based-elevation estimation method includes the following steps:

1. Using a drone to acquire construction site ortho-images,
2. Using overlapping disassembling algorithm to generate the overlapping small-patches and their sequence number,

3. Using the trained convolutional encoder-decoder network model to predict the elevation-map for each small-patch,
4. Assembling the prediction small-patches with the assigned sequence, and
5. Converting the elevation-map to elevation data or 3D point cloud.

Chapter 7 presents a deep learning-based method to identify vegetation objects on a construction site using drone-based ortho-image and determine the “real” ground surface elevations from the raw surveying results. The general procedure of the vegetation identifying and removing method includes the following steps:

1. Using a drone to acquire construction site ortho-images,
2. Disassembling the high-resolution ortho-image into overlapping small-patches,
3. Using the trained CNN-based image classification model to generate the small-patches of label-image and assemble them to a high-resolution label-image,
4. Searching and identifying the vegetation blocks in the high-resolution label-image,
5. Modifying their elevation values with the surrounding grounds’ elevations in the same coordinate elevation-map, and
6. Converting the modified elevation-map to elevation data or 3D point cloud of the construction site.

8.2 Contributions

The success of this research project contributes to the advancement of drone ortho-imaging and deep learning methods in construction site surveying. First, it advanced the multiple image-based 3D-surveying techniques with drone ortho-imaging, which is a flexible option for conditions where drones need to be kept away from target objects and a real-time as-build model is needed. The developed low-high ortho-image pair-based elevation determination method is suitable to create high-resolution ortho-images and elevation-maps datasets of construction sites for conducting deep learning-based research, which is highly dependent on elevation data. In addition, for the earthwork operations, the generated pixel grid results are easy to convert to a 2D site plan for updating the earthwork quantity and a 3D point cloud for documenting and visualizing the project progress.

Second, it explored single-frame image-based 3D-surveying with drone ortho-imaging, which is a convolutional encoder-decoder network model for estimating construction site elevations. The developed ortho-image and deep learning-based elevation estimation method can generate the elevation values as an elevation-map output using a single ortho-image input. The experiments were conducted to evaluate the effectiveness of the convolutional neural networks (CNNs) in the determination of construction site elevations. The developed input image disassembling and output image assembling algorithm provides the ideal training of a deep learning model with larger size images instead of shrinking images, which could result in losing image detail. The success of this research project makes it possible to generate elevation data on a construction site much faster than traditional survey methods, thus, speeding up the on-site construction operations.

Third, it provided and verified a feasible approach of using a CNN to segment a high-resolution ortho-image of construction sites. The developed model can be used for automatically identifying and locating multiple static object categories from the raw surveying results. In addition, the developed method can be extended to remove dynamic objects (i.e. moving objects like trucks, people, etc.) from the high-resolution ortho-imaging videos. With these advancements, this research project has proved that it is possible to use drone technologies to make the image-based construction surveying measurements of ground elevations much faster, more accurate and convenient.

8.3 Conclusions

Construction surveying plays a crucial role in determining construction sites’ elevations and locations, which are important in earthwork operations and critical for making decisions. However, accurately and quickly determining elevations of a construction site in real-time is still a challenge for the construction industry. This research project utilized the drone ortho-imaging, deep learning, computer vision, image processing and image classification methods to simplify and speed up the image-based 3D-reconstruction techniques in the construction site elevation determination. The major findings of this research project are:

1. By only using two frame ortho-images, the developed low-high ortho-image based elevation determination method focuses on 3D-reconstruction of the ground surface and excludes the

vertical side surfaces of any attached or sunken objects on a construction site, which makes it simpler than traditional drone photogrammetry. This method maximizes the overlap of the ortho-image pair, where the entire low ortho-image is contained in the overlap. It only took 2 to 5 minutes for determining elevations for 2500 points in the 10-20 m trial or 4761 points in the 20-40 m trial with Python 3, while using a fast programming language such as C++ this time could be reduced. In addition, the generated results, the ortho-image and elevation-map pairs, were easily stitched using a very narrow overlapping strip, which is much less than the 70% overlap ratio in traditional drone photogrammetry.

2. For the automatic matching of the 2:1 scale ortho-image pair, the four-scaling reference patch feature descriptors for the low ortho-image were designed first to have the same size as the target patch feature descriptor for the high ortho-image. Then, the NCC method was used to match the pixel grid and return the corresponding 0.5-pixel coordinate from the high ortho-image for the given pixel from the low ortho-image. The developed pixel grid matching and elevation determination algorithms were robust even for poorly textured surfaces and large sloped surfaces, and also effective in indirectly lit environments. It can give an accurate pixel grid match for the low-high ortho-image pair at least 92% of the time. It can produce an accurate elevation result for the strongly matched pixel grid within the acceptable measuring error of less than 5.00 cm.
3. The input image overlapping disassembling and output image assembling algorithm which ran in parallel with the deep learning models is developed, which made the workstation system more efficient to train a deep learning model with high-resolution images instead of shrinking images and losing image details. By disassembling the datasets into multiple small patches, the number of datasets was significantly increased as well. With the suitable patch size, such as the 128×128-pixel patch, the developed deep learning models balanced the global features and local features, and it can even be well-trained earlier than larger patch sizes. The experimental results showed that the 128×128-pixel patch trial stopped training the convolutional encoder-decoder network model at the 18th epoch, while the 256×256-pixel

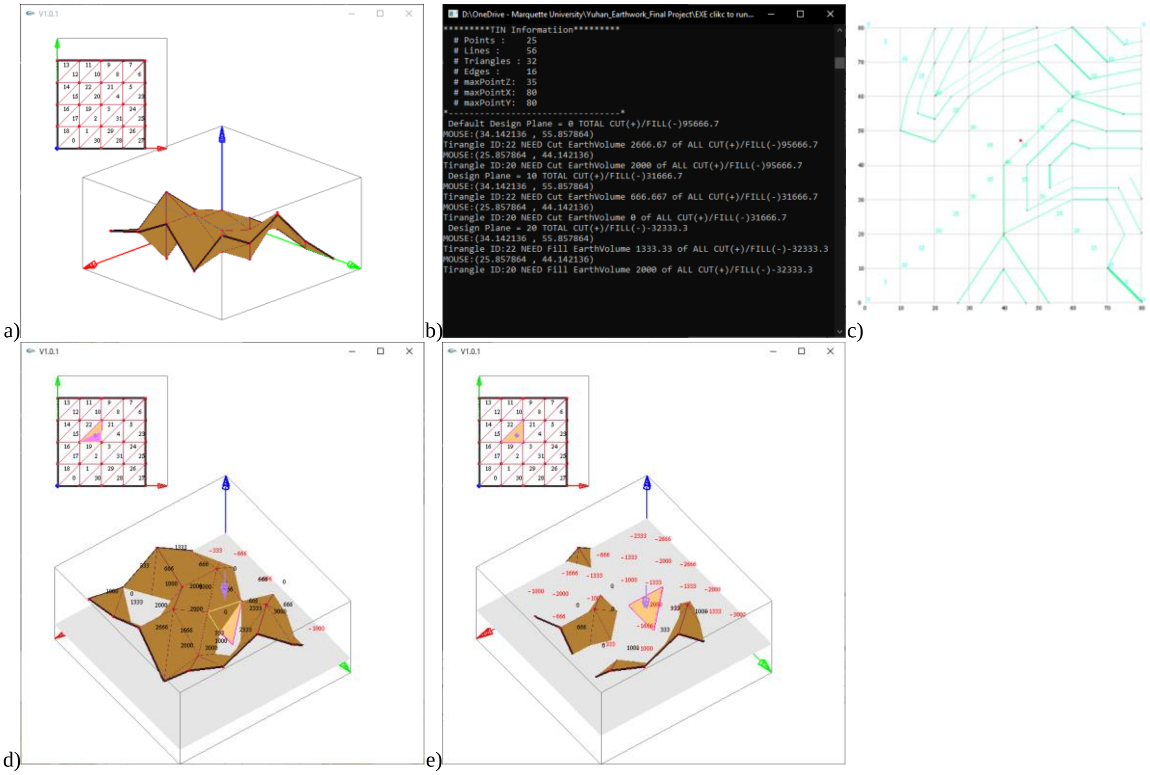
patch trial stopped at the 35th epoch . In addition, the smaller 32×32-pixel patch contains the maximum local features, which was important for changes in edges and corners.

4. Experiments were conducted to evaluate the effectiveness of the developed convolutional encoder-decoder network model for estimating construction site elevations. The results showed that the 128×128-pixel patch had the best prediction performance when the elevation values were shared in the elevation-map with a 32×32-pixel patch. Adding model training epochs had a positive relationship to the model prediction accuracy. The testing results showed that the "well-trained" model had a 52.43% accuracy in elevation estimation with a ± 5.0 cm error and 66.15% accuracy with a ± 10.0 cm error. Compared with the 94% accuracy (error ± 10.0 cm) in model training, it still has potential for improving the deep learning method for single-image-frame-based 3D-reconstruction of construction sites.
5. Experimental results showed the developed CNN-based image classification model using the 32×32-pixel patch had the best performance of 94% accuracy in identifying each main object's class-label from each small-patch ortho-image on the construction site. The testing results showed that the developed method, which disassembled the 1,536×1,536-pixel high-resolution image into 9,025 overlapping small-patches for image classification and assembled the label-image small-patch predictions to the 1,536×1,536-pixel high-resolution label-image prediction, was as effective as image segmentation algorithms because different object categories were marked with different colors in the assembled high-resolution label-image predictions with a high accuracy of 93%; and the edges of different objects were well determined.

8.4 Recommendations

The following recommendations are suggested for implementing the results of this research project and for future research on multiple/single ortho-image based 3D-reconsruction for construction site elevation determination and excavation operations. First, this research project focused on determining the elevation of a construction site, the transformation ability of converting the ortho-image and elevation-map results to 3D point cloud had been addressed in this research project, and the textured point cloud was a part of the generated results from the developed method as well. Transforming the point clouds results to the earthwork volume quantities can be implemented by using Autodesk Civil 3D to create a TIN (Triangulated Irregular Network) mesh model, and estimate the earthwork volume from the mesh model. As the selected and matched pixels/points were in the intersection points of the regular grid as the site plan formation, the volume can be estimated by the four-point-method (using four corners of each grid cell) or three-point-method (using three corners of each triangular cell) when the current elevation and the designed elevation of each intersection point are known. Figure 91 a and c show a 25-point 3D mesh model and 2D site plan demo (programmed with C++ and OpenGL), where the x/y-axis ranges from 0 to 80 m, and the current elevation ranges from 0 to 35 m. In this demo, 32 triangular cells were generated, each of them is a small ground area (Area=20×20/2=200 m^2) and its volume can be estimated by $(Ele_1 + Ele_2 + Ele_3) \times Area/3$, where Ele_i is the elevation for each point. Figure 91 d and e show two earthwork estimation demos, where the designed sites are two different flat planes (gray), and the volume estimation results for each triangular cell with these two different design planes were calculated and indicated in the 3D surface model. In detail, when the design plane is 10 m, part of the selected triangular cell (ID=20) needs to be cut (pink surface/yellow edges) and part of it needs to be filled (yellow surface/pink edges), and the total earthwork quantity is balanced in this triangular cell, as the volume=0 m^3 (see Figure 91 d); when the design plane is 20 m, the selected triangular cell needs to be filled with 2,000 m^3 (see Figure 91 e). Moreover, for monitoring and estimating the earthwork quantities at an active excavation site, capturing two low-high ortho-image pairs in two different times could be done. Then, the generated ortho-images and elevation-maps can be used to determine the elevation changes and calculate the volume changes between those two time points. The critical process is to align the two ortho-images and elevation-maps to the same coordinate. Fortunately, it can be easily handled by the 2D image rotation and translation (see Figure 92). The elevation changes for any selected point, can be determined by subtracting the latest elevation-map from the previous elevation-map. Figure 92 shows an elevation monitoring demo, where the ortho-images and elevation-maps were aligned to the same image center, and the x/y-profiles of the center point were overlapped to show the elevation changes. Furthermore, this research project utilized an experimental site at the lake beach, which simulated most cases of a real construction site including the larger sloped surface.

It is recommended that the developed methods and algorithms need to be evaluated at a real excavation site in the future.



```
D:\OneDrive - Marquette University\Yuhan_Earthwork_Final Project\EXE clikc to run...
*********TIN Informatiion*********
  # Points :    25
  # Lines :     56
  # Triangles : 32
  # Edges :     16
  # maxPointZ:  35
  # maxPointX:  80
  # maxPointY:  80
*---------------------------------*
 Default Design Plane = 0 TOTAL CUT(+)/FILL(-)95666.7
MOUSE:(34.142136 , 55.857864)
Tirangle ID:22 NEED Cut EarthVolume 2666.67 of ALL CUT(+)/FILL(-)95666.7
MOUSE:(25.857864 , 44.142136)
Tirangle ID:20 NEED Cut EarthVolume 2000 of ALL CUT(+)/FILL(-)95666.7
 Design Plane = 10 TOTAL CUT(+)/FILL(-)31666.7
MOUSE:(34.142136 , 55.857864)
Tirangle ID:22 NEED Cut EarthVolume 666.667 of ALL CUT(+)/FILL(-)31666.7
MOUSE:(25.857864 , 44.142136)
Tirangle ID:20 NEED Cut EarthVolume 0 of ALL CUT(+)/FILL(-)31666.7
 Design Plane = 20 TOTAL CUT(+)/FILL(-)-32333.3
MOUSE:(34.142136 , 55.857864)
Tirangle ID:22 NEED Fill EarthVolume 1333.33 of ALL CUT(+)/FILL(-)-32333.3
MOUSE:(25.857864 , 44.142136)
Tirangle ID:20 NEED Fill EarthVolume 2000 of ALL CUT(+)/FILL(-)-32333.3
```

Figure 91 Volume estimation demo

Aligned Ortho-images
1824×1824-Pixel

Aligned Elevation-maps
1568×1568-Pixel

Elevation Comparisons
* picked point on the center of the ortho-images; red curves are CG; green curves are CF.

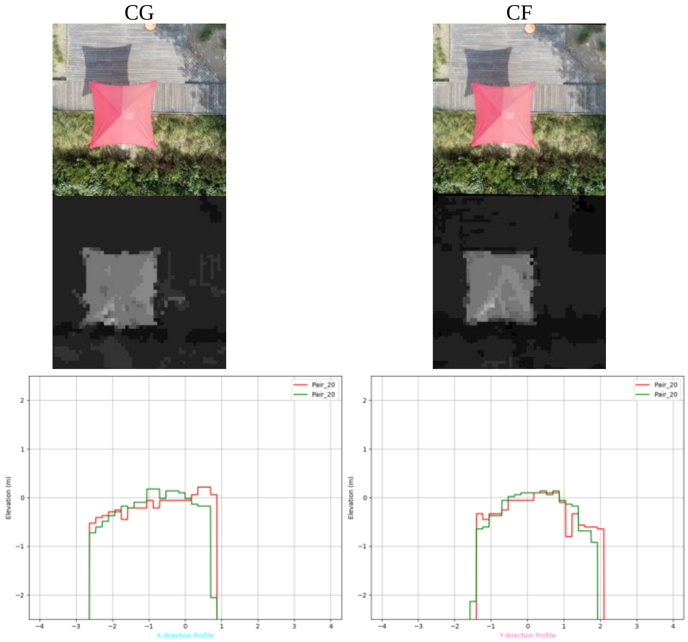


Figure 92 Elevation monitoring demo

Second, the image patch-based NCC matching approach can be improved in the developed low-high ortho-image pair-based elevation determination method. In this research project, the patch-based NCC method was used to determine whether the reference pixel/patch in the reference image was strongly matched with the candidate pixel/patch in the target image or not. The used reference and target patches were grayscale single-channel, while future research could use the RGB 3-channel reference patch and target patch to increase redundant features and enhance the matching accuracy. In addition, the developed four-scaling reference patch feature descriptors were used to make the reference patch had the same size with the target patch, while the developed convolutional encoder-decoder network model could be used to generate the predicted target patch for each reference patch by reducing an up-sample layer in the encoder block. Then the target patch prediction can be used to compare with the candidate patches of each virtual plane in the target image. For a single low-high ortho-image pair, this proposed approach can be used in dense pixel grid matching after getting the initial pixel grid matching results. When the training datasets are big enough, the well-trained model can be used to generate the target patch prediction for new low-high ortho-image pairs.

Third, the developed drone-based ortho-image and deep learning-based method can be used to estimate construction site elevations if the convolutional encoder-decoder network model is well-trained with datasets of similarly textured objects at sites. In this ortho-image-based 3D-reconstruction method, the model training datasets are the reference information to estimate the construction site elevations. Thus, the performance of the developed method relies on the quality and quantity of the model training datasets. The quality means more comprehensive texture features and geometry shape features while the quantity helps to build the ability to ignore incorrect elevation values in the dataset and noise in the model predictions. In this research project, the elevation estimation model training dataset was limited to 10-m drone-based ortho-images, which only contain a few objects in a single image frame. In addition, the formation of the elevation-map only contains single elevation values in each 32×32-pixel patch. Therefore, adding the sixth convolution layer or adding the filters in the fifth convolution layer for the developed CNN encoder model has nonsignificant improvement in the model prediction. Future research can assign more elevation values to each 32×32-pixel patch by the dense pixel grid matching, and then, the developed model may need additional CNN encoder layers and CNN decoder layers to connect the added elevation features. In addition, increasing the drone flight altitude can enlarge an image's spatial resolution and include more objects. Therefore, future research can train the developed model with more datasets at different altitudes other than the 10 m ortho-images. Furthermore, to increase the accuracy of the elevation estimation, future research would use image classification to assign a class-label for each patch (32×32-pixel). The class-label can be used as the additional reference information (feature-map) to increase the accuracy of the elevation prediction.

Fourth, this research project only considered removing the static vegetation blocks from the image-based construction site surveying results. The model training datasets only contain the static objects on a construction site, such as the static vegetation block and static structures. The CNN-based image classification model was developed to identify the predefined static objects using the drone-based still ortho-images. The experimental results confirmed that the developed classification model can be applied in construction site surveying. However, there are additional works that need to be done until it can be used for monitoring earthwork operations on a construction site in real time. The active excavating construction site is much more complex than the still construction site. The dynamic objects such as excavators, dozers, trucks and workers on the construction site have impacts on accurately determining the site elevations using the non-contact surveying methods. For example, a moving dozer may be included in overlapping ortho-image pairs with different locations, which will lead to the incorrect image pair matching using the traditional drone photogrammetry method. If the dozer is stopped during the capture of the ortho-image pair, it can still have an impact on the results of construction site surveying because they contain the height of construction equipment. Thus, future research is needed to extend the CNN-based image classification model training dataset to include all the potential static and dynamic objects at not only a still construction site but also an active construction site.

BIBLIOGRAPHY

Aguilar, R., Noel, M. F., and Ramos, L. F. (2019). “Integration of reverse engineering and non-linear numerical analysis for the seismic assessment of historical adobe buildings.” *Automation in construction*, 98, 1-15.

Anders, N., Valente, J., Masselink, R., and Keesstra, S. (2019). “Comparing Filtering Techniques for Removing Vegetation from UAV-Based Photogrammetric Point Clouds.” *Drones*, 3(3), 61.

Arias, P., Herraez, J., Lorenzo, H., and Ordonez, C. (2005). “Control of structural problems in cultural heritage monuments using close-range photogrammetry and computer methods.” *Computers & structures*, 83(21-22), 1754-1766.

Ashour, R., Taha, T., Mohamed, F., Hableel, E., Kheil, Y. A., Elsalamouny, M., ... and Cai, G. (2016). “Site inspection drone: A solution for inspecting and regulating construction sites.” *Proc. 2016 IEEE 59th International Midwest Symposium on Circuits and Systems (MWSCAS)*, IEEE, Abu Dhabi, doi:10.1109/mwscas.2016.7870116.

Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017). “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation.” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12), 2481-2495.

Bang, S., Kim, H., and Kim, H. (2017). “UAV-based automatic generation of high-resolution panorama at a construction site with a focus on preprocessing for image stitching.” *Automation in construction*, 84, 70-80.

Barazzetti, L., Remondino, F., and Scaioni, M. (2010). “Automation in 3D reconstruction: Results on different kinds of close-range blocks.” *Proc. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, ISPRS, Newcastle upon Tyne, 55-61.

Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). “Speeded-up robust features (SURF).” *Computer vision and image understanding,*110(3), 346-359.

Bernardini, S., Fox, M., and Long, D. (2014). “Planning the Behaviour of Low-Cost Quadcopters for Surveillance Missions.” *Proc. 24th International Conference on Automated Planning and Scheduling(ICAPS)*, AAAI, Portsmouth, 445-453.

Chan, A. P. C., and Owusu, E. K. (2017). “Corruption Forms in the Construction Industry: Literature Review.” *Journal of Construction Engineering and Management*, 143(8), 4017057.

Chen, K., Lu, W., Xue, F., Tang, P., and Li, L. H. (2018). “Automatic building information model reconstruction in high-density urban areas: Augmenting multi-source data with architectural knowledge.” *Automation in construction*, 93, 22-34.

Chen, W., Fu, Z., Yang, D., and Deng, J. (2016). “Single-image depth perception in the wild.” *Proc. Conference on 30th Neural Information Processing Systems (NIPS 2016)*, NeurIPS, Barcelona, 730-738.

Chollet, F. (2015). “Keras: The Python Deep Learning library.” <https://keras.io/> (Aug. 7, 2019).

Cunliffe, A. M., Brazier, R. E., and Anderson, K. (2016). “Ultra-fine grain landscape-scale quantification of dryland vegetation structure with drone-acquired structure-from-motion photogrammetry.” *Remote Sensing of Environment*, 183, 129-143.

Daftry, S., Hoppe, C., and Bischof, H. (2015). “Building with drones: Accurate 3D facade reconstruction using MAVs.” *Proc. 2015 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, Seattle, 3487- 3494.

Dalal, N., and Triggs, B. (2005). “Histograms of oriented gradients for human detection.” *Proc. 2015 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005)*, IEEE, San Diego, doi: 10.1109/CVPR.2005.177

Deng, J., Dong, W., Socher, R., Li, L., Li, K. and Li., F. (2009). “ImageNet: A large-scale hierarchical image database.” *Proc. 2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Miami, 248-255.

Dettmers, T. (2015). “Deep Learning in a Nutshell: Core Concepts” <https://devblogs.nvidia.com/deep-learning-nutshell-core-concepts/> (Aug. 7, 2019).

Du, J. C., and Teng, H. C. (2007). “3D laser scanning and GPS technology for landslide earthwork volume estimation.” *Automation in construction*, 16(5), 657-663.

Eigen, D., Puhrsch, C., and Fergus, R. (2014). “Depth map prediction from a single image using a multi-scale deep network.” *Proc. 28th Conference on Neural Information Processing Systems (NIPS 2014)*, NeurIPS, Montréal, 2366-2374.

Ellenberg, A., Kontsos, A., Moon, F., and Bartoli, I. (2016). “Bridge deck delamination identification from unmanned aerial vehicle infrared imagery.” *Automation in construction*, 72, 155-165.

Engelcke, M., Rao, D., Wang, D. Z., Chi Hay Tong, and Posner, I. (2017). “Vote3Deep: Fast object detection in 3D point clouds using efficient convolutional neural networks.” *Proc. 2017 IEEE International Conference on Robotics and Automation (ICRA)* , IEEE, Singapore, 1355-1361.

Erickson, M. S., Bauer, J. J., and Hayes, W. C. (2013). “The accuracy of photo-based three-dimensional scanning for collision reconstruction using 123D catch.” *Proc. SAE 2013 World Congress & Exhibition*, doi:10.4271/2013-01-0784

Freimuth, H., and König, M. (2018). “Planning and executing construction inspections with unmanned aerial vehicles.” *Automation in construction*, 96, 540-553.

Furukawa, Y., Ponce, J. (2010). “Accurate, Dense, and Robust Multiview Stereopsis.” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32 (8), 1362-1376.

Garg, R., BG, V. K., Carneiro, G., and Reid, I. (2016). “Unsupervised CNN for single view depth estimation: Geometry to the rescue.” *Proc. 14th European Conference on Computer Vision (ECCV 2016)*, Springer, Amsterdam, 740-756.

Gheisari, M., and Esmaeili, B. (2016). “Unmanned Aerial Systems (UAS) for Construction Safety Applications.” *Proc. Construction Research Congress 2016* , ASCE, Historic San Juan, 2642-2650.

Gheisari, M., Irizarry, J., and Walker, B. N. (2014). “UAS4SAFETY: The potential of unmanned aerial systems for construction safety applications.” *Proc. Construction Research Congress 2014* , ASCE, Atlanta, 1801-1810.

Guo, H., Yu, Y., Ding, Q., and Skitmore, M. (2018). “Image-and-Skeleton-Based Parameterized Approach to Real-Time Identification of Construction Workers’ Unsafe Behaviors.” *Journal of Construction Engineering and Management*, 144(6). doi: 10.1061/(ASCE)CO.1943-7862.0001497.

Guo, Q., Su, Y., Hu, T., Zhao, X., Wu, F., Li, Y., ... and Zheng, Y. (2017). “An integrated UAV-borne lidar system for 3D habitat mapping in three forest ecosystems across China.” *International Journal of Remote Sensing*, 38(8-10), 2954-2972.

Gwak, H. S., Seo, J., and Lee, D. E. (2018). “Optimal cut-fill pairing and sequencing method in earthwork operation.” *Automation in Construction*, 87, 60-73.

Hamledari, H., McCabe, B., and Davari, S. (2017). “Automated computer vision-based detection of components of under-construction indoor partitions.” *Automation in Construction*, 74, 78-94.

Han, J., Zhang, D., Cheng, G., Guo, L. and Ren, J. (2015). “Object Detection in Optical Remote Sensing Images Based on Weakly Supervised Learning and High-Level Feature Learning.” *IEEE Transactions on Geoscience and Remote Sensing*, 53(6), 3325-3337.

Han, K., Degol, J. and Golparvar-Fard, M. (2018). “Geometry- and Appearance-Based Reasoning of Construction Progress Monitoring.” *Journal of Construction Engineering and Management*, 144 (1), 04017110.

Han, K. K., and Golparvar-Fard, M. (2017). “Potential of big visual data and building information modeling for construction performance analytics: An exploratory study.” *Automation in Construction*, 73, 184-198.

Hassner, T., and Basri, R. (2006). “Example based 3D reconstruction from single 2D images.” *Proc. 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW’06)*, IEEE, New York, doi: 10.1109/CVPRW.2006.76.

Haur, C. J., Kuo, L. S., Fu, C. P., Hsu, Y. L., and Da Heng, C. (2018). “Feasibility Study on UAV-assisted Construction Surplus Soil Tracking Control and Management Technique.” *Proc. 5th Annual International Conference on Material Science and Environmental Engineering (MSEE2017)*, IOP Publishing, Xiamen, doi:10.1088/1757-899X/301/1/012145.

Hearn, D., Baker, M. P., and Baker, M. P. (2004). *Computer Graphics with OpenGL (3rd edition)*, Pearson Prentice Hall, Upper Saddle River, NJ.

Hola, B., and Schabowicz, K. (2010). “Estimation of earthworks execution time cost by means of artificial neural networks.” *Automation in Construction*, 19(5), 570-579.

Holz, D., Holzer, S., Rusu, R. B., and Behnke, S. (2011). “Real-time plane segmentation using RGB-D cameras.” *Proc., 15th RoboCup International Symposium 2011*, Springer, Istanbul, pp. 306-317.

Huang, A. S., Bachrach, A., Henry, P., Krainin, M., Maturana, D., Fox, D., and Roy, N. (2017). “Visual odometry and mapping for autonomous flight using an RGB-D camera.” *Proc., the 15th International Symposium of Robotic Research (ISRR)*, Springer, Flagstaff, 235-252.

Huang, Q., Luzi, G., Monserrat, O., and Crosetto, M. (2017). “Ground-based synthetic aperture radar interferometry for deformation monitoring: a case study at Geheyan Dam, China.” *Journal of Applied Remote Sensing*, 11(3). doi: 10.1117/1.JRS.11.036030.

Hubbard, B., Wang, H., Leasure, M., Ropp, T., Lofton, T., Hubbard, S., and Lin, S. (2015). “Feasibility study of UAV use for RFID material tracking on construction sites.” *Proc., 51st ASC Annual International Conference*. ASC, College Station.

Inzerillo, L., Di Mino, G., and Roberts, R. (2018). “Image-based 3D reconstruction using traditional and UAV datasets for analysis of road pavement distress.” *Automation in Construction*, 96, 457-469.

Irizarry, J., Gheisari, M., and Walker, B. N. (2012). "Usability assessment of drone technology as safety inspection tools." *Journal of Information Technology in Construction*, 17, 194-212.

Josephson, P. E., and Hammarlund, Y. (1999). "The causes and costs of defects in construction: A study of seven building projects." *Automation in construction*, 8(6), 681-687.

Kaehler, A., and Bradski, G. (2016). *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library* ,O'Reilly Media, Sebastopol, CA.

Kang, K., Li, H., Yan, J., Zeng, X., Yang, B., Xiao, T., Zhang, C., Wang, Z., Wang, R., Wang, X., and Ouyang, W. (2018). "T-CNN: Tubelets With Convolutional Neural Networks for Object Detection From Videos." *IEEE Transactions on Circuits and Systems for Video Technology*, 28(10), 2896-2907.

Kim, D., Liu, M., Lee, S., and Kamat, V. R. (2019). "Remote proximity monitoring between mobile construction resources using camera-mounted UAVs." *Automation in Construction*, 99, 168-182.

Kim, H., and Kim, H. (2018). "3D reconstruction of a concrete mixer truck for training object detectors." *Automation in construction*, 88, 23-30.

Kim, K., Kim, H., and Kim, H. (2017). "Image-based construction hazard avoidance system using augmented reality in wearable device." *Automation in Construction*, 83, 390-403.

Kim, S., Irizarry, J., and Costa, D. B. (2016). "Potential factors influencing the performance of unmanned aerial system (UAS) integrated safety control for construction worksites." *Proc., Construction Research Congress 2016*,ASCE, Historic San Juan 2614-2623.

Kirscht, M., and Rinke, C. (1998). "3D Reconstruction of Buildings and Vegetation from Synthetic Aperture Radar (SAR) Images." *Proc., IAPR Workshop on Machine Vision Application (MVA'98)* , IAPR, Makuhari, 228-231.

Kraig, K., Clifford, J. S., Christine, F., Richard, E. M. (2008). *Construction Management Fundamentals*, McGraw-Hill, New York, NY.

Kwon, S., Park, J. W., Moon, D., Jung, S., and Park, H. (2017). "Smart Merging Method for Hybrid Point Cloud Data using UAV and LIDAR in Earthwork Construction." *Procedia Engineering*, 196, 21-28.

Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., and Navab, N. (2016). "Deeper depth prediction with fully convolutional residual networks." *Proc., 4th International Conference on 3D Vision (3DV 2016)*, IEEE, Stanford, 239-248.

Lewis, J.P. (1995). "Fast Template Matching." Proc., Vision Interface 95, *Canadian Image Processing and Pattern Recognition Society*, Quebec City, 120-123.

Li, D., and Lu, M. (2018). "Integrating geometric models, site images and GIS based on Google Earth and Keyhole Markup Language." *Automation in Construction*, 89, 317-331.

Li, F., Zlatanova, S., Koopman, M., Bai, X., and Diakité, A. (2018). "Universal path planning for an indoor drone." *Automation in Construction*, 95, 275-283.

Li, R. Ma, F. , Xu, F., Matthies, L.H., Olson, C.F., Arvidson, R.E. (2002). "Localization of Mars rovers using descent and surface-based image data." *Journal of Geophysical Research: Planets*, 107, doi: 10.1029/2000JE001443.

Li, Z., and Snavely, N. (2018). "MegaDepth: Learning single-view depth prediction from internet photos." *Proc., 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, 2041-2050.

Litomisky, K. (2012). "Consumer rgb-d cameras and their applications." <http://alumni.cs.ucr.edu/~klitomis/files/RGBD-intro.pdf> (Feb. 7, 2019)

Liu, F., Shen, C., and Lin, G. (2015). "Deep convolutional neural fields for depth estimation from a single image." *Proc., 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2015),* IEEE, Boston,5162-5170.

Lowe, D. G. (1999). "Object recognition from local scale-invariant features." *Proc., 7th IEEE International Conference on Computer Vision (ICCV'99)* , IEEE, Kerkyra, 1150-1157.

Lowe, D. G. (2004). "Distinctive image features from scale-invariant keypoints." *International Journal of Computer Vision*, 60(2), 91-110.

Maghiar, M., and Mesta, D.(2018). "Measurement comparison of city roadway intersection models obtained via laser-scanning and photogrammetry." *Proc., 54th ASC Annual International Conference* ,ASC, Minneapolis, 568-576.

Matthies, L., Maimone, M., Johnson, A., Cheng, Y., Willson, R., Villalpando, C., Goldberg, S., Huertas, A., Stein, A., Angelova, A. (2007). "Computer vision on Mars." *International Journal of Computer Vision,* 75 (1),67-92.

Matthies, L.H., Olson, C.F., Tharp, G., Laubach, S. (1997). "Visual localization methods for mars rovers using lander, rover, and descent imagery." <https://trs.jpl.nasa.gov/bitstream/handle/2014/22227/97-0695.pdf?sequence=1> (May 8, 2018).

Memarzadeh, M., Golparvar-Fard, M., and Niebles, J. C. (2013). "Automated 2D detection of construction equipment and workers from site video streams using histograms of oriented gradients and colors." *Automation in Construction*, 32, 24-37.

Meng, C., Zhou, N., Xue, X., and Jia, Y. (2013). "Homography-based depth recovery with descent images." *Machine Vision and Applications*, 24(5), 1093-1106.

Metni, N., and Hamel, T. (2007). "A UAV for bridge inspection: Visual servoing control law with orientation limits." *Automation in Construction*, 17(1), 3-10.

Moon, D., Chung, S., Kwon, S., Seo, J., and Shin, J. (2019). "Comparison and utilization of point cloud generated from photogrammetry and laser scanning: 3D world model for smart heavy equipment planning." *Automation in Construction,* 98, 322-331.

Morgenthal, G., Hallermann, N., Kersten, J., Taraben, J., Debus, P., Helmrich, M., and Rodehorst, V. (2019). "Framework for automated UAS-based structural condition assessment of bridges." *Automation in Construction*, 97, 77-95.

Nair, V. and Hinton, G. (2010). "Rectified linear units improve restricted boltzmann machines." *Proc., 27th international conference on machine learning(ICML 2010),* International Machine Learning Society (IMLS), Haifa, 807-814.

Nasirian, A., Arashpour, M., and Abbasi, B. (2019). "Critical Literature Review of Labor Multiskilling in Construction." *Journal of Construction Engineering and Management*, 145(1), 4018113.

Nassar, K., and Jung, Y. (2012). "Structure-From-Motion Approach to the Reconstruction of Surfaces for Earthwork Planning." *Journal of Construction Engineering and Project Management*, 2 , 1-7.

Nex, F., and Remondino, F. (2014). "UAV for 3D mapping applications: a review." *Applied Geomatics*, 6(1), 1-15.

Nichols, H., and Day, D. (2010). *Moving the Earth: The Workbook of Excavation (6th edition)*, McGraw-Hill, New York, NY.

Nico, G., Leva, D., Fortuny-Guasch, J., Antonello, G., and Tarchi, D. (2005). "Generation of digital terrain models with a ground-based SAR system." *IEEE Transactions on Geoscience and Remote Sensing*, 43(1), 45-49.

Noferini, L., Pieraccini, M., Mecatti, D., Macaluso, G., Atzeni, C., Mantovani, M., ... and Tagliavini, F. (2007). "Using GB-SAR technique to monitor slow moving landslide." *Engineering Geology*, 95(3-4), 88-98.

Noh, H., Hong, S., and Han, B. (2015). "Learning Deconvolution Network for Semantic Segmentation." *Proc., 2015 IEEE International Conference on Computer Vision (ICCV 2015)*, IEEE, Santiago, 1520–1528.

Nunnally, S. W. (2010). *Construction Methods and Management (8th edition)*, Pearson Prentice Hall Upper Saddle River, NJ.

Omar, T., and Nehdi, M. L. (2017). "Remote sensing of concrete bridge decks using unmanned aerial vehicle infrared thermography." *Automation in Construction*, 83, 360-371.

Park, J., Kim, P., Cho, Y. K., and Kang, J. (2019). "Framework for automated registration of UAV and UGV point clouds using local features in images." *Automation in Construction*, 98, 175-182.

Peurifoy, R. L., and Garold D. O. (2014). *Estimating Construction Costs (6th edition)*, McGraw-Hill, New York, NY.

Phung, M. D., Quach, C. H., Dinh, T. H., and Ha, Q. (2017). "Enhanced discrete particle swarm optimization path planning for UAV vision-based surface inspection." *Automation in Construction*, 81, 25-33.

Remondino, F. (2003). "From point cloud to surface: the modeling and visualization problem". *Proc., International Workshop on Visualization and Animation of Reality-based 3D Models*, ISPRS, Engadin.

Remondino, F., and El-Hakim, S. (2006). "Image-based 3D modelling: a review." *The Photogrammetric Record*, 21(115), 269-291.

Roca, D., Lagüela, S., Díaz-Vilariño, L., Armesto, J., and Arias, P. (2013). "Low-cost aerial unit for outdoor inspection of building façades." Automation in Construction, 36, 128-135.

Rusu, R. B., and Cousins, S. (2011). "3D is here: Point Cloud Library (PCL)." *Proc., 2011 IEEE International Conference on Robotics and Automation*, IEEE, Shanghai, doi: 10.1109/ICRA.2011.5980567

Saxena, A., Chung, S. H., and Ng, A. Y. (2008). "3-D Depth Reconstruction from a Single Still Image." *International journal of Computer Vision*, 76(1), 53-69.

Schenk, T. (1999). *Digital photogrammetry: Vol. I: Background, fundamentals, automatic orientation produceres*. Terra Science, Laurelville, OH.

Schneider, S., Taylor, G. W., and Kremer, S. (2018). “Deep Learning Object Detection Methods for Ecological Camera Trap Data.” *Proc., 2018 15th Conference on Computer and Robot Vision (CRV)*, IEEE, Toronto, 321-328.

Seo, J., Duque, L., and Wacker, J. (2018). “Drone-enabled bridge inspection methodology and application.” *Automation in Construction*, 94, 112-126.

Seo, J., Lee, S., Kim, J., and Kim, S. K. (2011). “Task planner design for an automated excavation system.” *Automation in Construction*, 20(7), 954-966.

Shewchuk, J. R. (2002). “Delaunay refinement algorithms for triangular mesh generation.” *Computational Geometry*, 22(1-3), 21-74.

Siebert, S., and Teizer, J. (2014). “Mobile 3D mapping for surveying earthwork projects using an Unmanned Aerial Vehicle (UAV) system.” *Automation in Construction*, 41, 1-14.

Solem, J. E. (2012). *Programming Computer Vision with Python: Tools and algorithms for analyzing images*, O'Reilly Media, Sebastopol, CA.

Sophian, A., Sediono, W., Salahudin, M. R., Shamsuli, M. S. M., and Za’aba, D. Q. A. A. (2017). “Evaluation of 3D-Distance Measurement Accuracy of Stereo-Vision Systems.” *International Journal of Applied Engineering Research*, 12(16), 5946-5951.

Spence, W. P., and Kultermann, E. (2016). *Construction Materials, Methods and Techniques (4th edition)*, Cengage Learning, Alexandria, VA.

Sung, C., and Kim, P. Y. (2016). “3D terrain reconstruction of construction sites using a stereo camera.” *Automation in Construction*, 64, 65-77.

Szeliski, R. (2010). *Computer vision: algorithms and applications*, Springer, New York, NY.

Takahashi, N., Wakutsu, R., Kato, T., Wakaizumi, T., Ooishi, T., and Matsuoka, R. (2017). “Experiment On UAV Photogrammetry And Terrestrial Laser Scanning For ICT-Integrated Construction” *Proc., 2017 International Conference on Unmanned Aerial Vehicles in Geomatics*, ISPRS , Bonn, 371-377.

Toole, T. M. (2002). “Construction Site Safety Roles.” *Journal of Construction Engineering and Management*, 128(3), 203-210.

Tsai, V. J. (1993). “Delaunay triangulations in TIN creation: an overview and a linear-time algorithm.” *International Journal of Geographical Information Science*, 7(6), 501-524.

Tulldahl, H. M., and Larsson, H. (2014). “Lidar on small UAV for 3D mapping.” *Proc., The International Society for Optical Engineering*, SPIE, doi: 10.1117/12.2068448

Ullman, S. (1979). “The Interpretation of Structure from Motion.” *Proceedings of the Royal Society of London. Series B, Biological Sciences*, 203(1153), 405-426.

Van Blyenburgh, P. (1999). “UAVs: an overview.” *Air and Space Europe*, 1(5-6), 43-47.

Van den Heuvel, F. A. (1998). “3D reconstruction from a single image using geometric constraints.” *ISPRS Journal of Photogrammetry and Remote Sensing*, 53(6), 354-368.

Wang, J., Sun, W., Shou, W., Wang, X., Wu, C., Chong, H. Y., Liu, Y. and Sun, C. (2015). "Integrating BIM and LiDAR for Real-time Construction Quality Control." *Journal of Intelligent and Robotic Systems*, 79(3-4), 417-432.

Wang, J., Zhang, S., and Teizer, J. (2015). "Geotechnical and safety protective equipment planning using range point cloud data and rule checking in building information modeling." *Automation in Construction*, 49, 250-261.

Wang, L., Chen, F., and Yin, H. (2016). "Detecting and tracking vehicles in traffic by unmanned aerial vehicles." *Automation in Construction*, 72, 294-308.

Westoby, M. J., Brasington, J., Glasser, N. F., Hambrey, M. J., and Reynolds, J. M. (2012). "'Structure-from-Motion'photogrammetry: A low-cost, effective tool for geoscience applications." *Geomorphology*, 179, 300-314.

Wing, C. K. (1997). "The ranking of construction management journals." *Construction Management and Economics*, 15(4), 387-398.

Xiong, Y., Olson, C.F., Matthies, L.H. (2005). "Computing depth maps from descent images." *Machine Vision and Applications*, 16 (3), 139-147.

Yang, C., Tsai, M., Kang, S., and Hung, C. (2018). "UAV path planning method for digital terrain model reconstruction – A debris fan example." *Automation in Construction*, 93 214-230.

Yi, C., and Lu, M. (2016). "A mixed-integer linear programming approach for temporary haul road design in rough-grading projects." *Automation in Construction*, 71, 314-324.

Zakeri, H., Nejad, F. M., and Fahimifar, A. (2016). "Rahbin: A quadcopter unmanned aerial vehicle based on a systematic image processing approach toward an automated asphalt pavement inspection." *Automation in Construction*, 72, 211-235.

Zhang, S., Teizer, J., Pradhananga, N., and Eastman, C. M. (2015). "Workforce location tracking to model, visualize and analyze workspace requirements in building information models for construction safety planning." *Automation in Construction*, 60, 74-86.

Zhao, W. Q., and Lin, Z. (2016). "SfM Precise Surface Measurement: Evaluation of Resolution and Accuracy and Error Analysis." *Geography and Geo-Information Science*, 32(6), 25-31. (in Chinese).

Zhong, X., Peng, X., Yan, S., Shen, M., and Zhai, Y. (2018). "Assessment of the feasibility of detecting concrete cracks in images acquired by unmanned aerial vehicles." *Automation in Construction*, 89, 49-57.

Zhou, X., Zhong, G., Qi, L., Dong, J., Pham, T. D., and Mao, J. (2017). "Surface height map estimation from a single image using convolutional neural networks." *Proc., 8th International Conference on Graphic and Image Processing (ICGIP 2016)*, Society of Photo-Optical Instrumentation Engineers (SPIE), Tokyo, 1022524.