RESEARCH & DEVELOPMENT

DeepHyd: A Deep Learning-based Artificial Intelligence Approach for the Automated Classification of Hydraulic Structures from LiDAR and Sonar Data

Wenwu Tang, Ph.D.
Shen-En Chen, Ph.D.
John Diemer, Ph.D.
Craig Allan, Ph.D.
Tianyang Chen, Graduate Research Assistant
Zachery Slocum, Graduate Research Assistant
Tarini Shukla, Graduate Research Assistant
Vidya Shubhash Chavan, Ph.D., Former Graduate Research Assistant
Navanit Sri Shanmugam, Graduate Research Assistant

Center for Applied Geographic Information Science
Department of Geography and Earth Sciences
Department of Civil and Environmental Engineering
School of Data Science
University of North Carolina at Charlotte

NCDOT Project 2019-03

FHWA/NC/2019-03

January 2022

DeepHyd: A Deep Learning-based Artificial Intelligence Approach for the Automated Classification of Hydraulic Structures from LiDAR and Sonar Data

Final Report

(Report No. FHWA/NC/2019-03)

To: North Carolina Department of Transportation

(Research Project No. RP 2019-03)

Submitted by

Wenwu Tang[1,2,3,*], Shen-En Chen[4,*], John Diemer[2,*], Craig Allan[1,2,*], Tianyang Chen[1,2,**], Zachery Slocum[1,2,**], Tarini Shukla[1,4,**], Vidya Shubhash Chavan[4,**], Navanit Sri Shanmugam [4,**]

[1] Center for Applied Geographic Information Science

[2] Department of Geography and Earth Sciences

[3] School of Data Science

[4] Department of Civil and Environmental Engineering

University of North Carolina at Charlotte, Charlotte, NC 28223

*: PIs

**: Graduate Research Assistants

Contact

Wenwu Tang, Ph.D.

Phone: (704) 687-5988;

Fax: (704) 687-5966;

Email: wtang4@uncc.edu

January 2022

Technical Report Documentation Page

1. Report No. ***FHWA/NC/2019-03***	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle DeepHyd: A Deep Learning-based Artificial Intelligence Approach for the Automated Classification of Hydraulic Structures from LiDAR and Sonar Data		5. Report Date January 30th, 2022
		6. Performing Organization Code
7. Author(s) Wenwu Tang, Shen-En Chen, John Diemer, Craig Allan, Tianyang Chen, Zachery Slocum, Tarini Shukla, Vidya Shubhash Chavan, Navanit Sri Shanmugam		8. Performing Organization Report No.
9. Performing Organization Name and Address Center for Applied Geographic Information Science Department of Geography and Earth Sciences, University of North Carolina at Charlotte, Charlotte, NC 28223		10. Work Unit No. (TRAIS)
		11. Contract or Grant No.
12. Sponsoring Agency Name and Address North Carolina Department of Transportation Research and Development Unit 104 Fayetteville Street Raleigh, North Carolina 27601		13. Type of Report and Period Covered Final Report 07/01/2018-12/31/2021
		14. Sponsoring Agency Code PR 2019-03
Supplementary Notes:		

16. Abstract

Monitoring the conditions of hydraulic structures such as bridges and culverts is essential in warranting the safety and sustainability of transportation infrastructure. This is particularly important for North Carolina as more than 8 percent of NC bridges have been found in poor conditions and need immediate maintenance. Lidar and sonar technologies have been increasingly applied to support this monitoring need. However, the processing and classification of point cloud data generated from LiDAR and sonar techniques represents a challenge as hydraulic structures are often complicated in geometric characteristics and considerable labor and time are needed for the processing and classification.

To address this challenge, in this project, we developed DeepHyd, a deep learning-based 3D modeling framework and software tools for the automated classification of point cloud data of hydraulic structures. We collected field data from 11 sites in the Greater Charlotte Metropolitan region for training and validation of the deep learning algorithms. The field data collection combines the use of terrestrial LiDAR, sonar, total station, survey-grade GPS, and drone. The deep learning algorithm that we used for point cloud classification is a state-of-the-art 3D artificial intelligence technique. We used a two-tiered modeling approach to train deep learning algorithms using annotated point cloud data: classification of bridges from vegetation and ground, and classification of specific bridge components including beam, pier, railing, and retaining walls. We implemented scientific workflows to automate the classification of point cloud data of hydraulic structures using deep learning. Our major findings are: 1) our 3D deep learning algorithms in DeepHyd achieve high classification performance on point cloud data of hydraulic structures. 2) deep learning can effectively handle the classification of large volumes of point cloud data, but the training of deep learning algorithms requires large amounts of annotated data. 3) annotated point cloud data serve as a foundation database for the automated classification of hydraulic structures using artificial intelligence techniques. More annotated point cloud data, which cover alternative types of hydraulic structures, are needed for further improving classification performance.

17. Key Words Point cloud classification, Hydraulic structures, Deep learning, Artificial Intelligence	18. Distribution Statement		
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 60	22. Price

Form DOT F 1700.7 (8-72) **Reproduction of completed page authorized**

Disclaimer

Acknowledgements

This project is supported by the North Carolina Department of Transportation. Specifically, the authors owe thanks to the Steering and Implementation Committee: including Matthew Lauffer (Chair), John W. Kirby, Tom Langan, Gary Thompson, Paul Jordan, Mark Swartz, Mark Ward, Derek Bradner, Brian Radakovic, Kevin Fischer. We thank Matthew Macon, Rodney Hough, Donald Early, from UAS program and Photogrammetry Unit, NCDOT for their help on UAS testing. We thank NCDOT IT for their help on deploying and testing the software.

Executive Summary

Monitoring the conditions of hydraulic structures such as bridges and culverts is essential in warranting the safety and sustainability of transportation infrastructure. This is particularly important for North Carolina as more than 8 percent of NC bridges have been found in poor conditions and need immediate maintenance. LiDAR and sonar technologies have been increasingly applied to support this monitoring need. However, the processing and classification of point cloud data generated from LiDAR and sonar techniques represents a challenge as hydraulic structures are often complicated in their geometric characteristics and considerable labor and time are needed for the processing and classification of large point cloud datasets.

To address this challenge, in this project, we developed DeepHyd, a deep learning-based 3D modeling framework and software tools for the automated classification of point cloud data of hydraulic structures. We collected field data from 11 sites in the Greater Charlotte Metropolitan region for the training and validation of the deep learning algorithms. The field data collection combines the use of a variety of survey instruments, including terrestrial LiDAR, sonar, total station, survey-grade GPS, and drone-based photogrammetry. The deep learning algorithm that we utilized for the point cloud classification is a state-of-the-art 3D artificial intelligence technique based on convolutional neural networks. We used a two-tiered modeling approach to train deep learning algorithms using annotated point cloud data: classification of bridges from vegetation and ground, and classification of specific bridge components including beam, pier, railing, and retaining walls. We implemented scientific workflows to automate the processing and classification of point cloud data of hydraulic structures using deep learning.

Considering the unique geographical divisions in North Carolina from the mountain ridges in the Appalachian to the Atlantic coastal plain, a great diverse of highway bridge types interconnected the state and the automated bridge component classification tool represents a paradigm shift in transportation management. Our major findings are summarized below:

1) Our 3D deep learning algorithms in DeepHyd achieve high classification performance on point cloud data of hydraulic structures. The two-tiered geospatial modeling design can effectively support 1) the classification of hydraulic structures, vegetation, and ground surfaces, and 2) the classification of specific bridge components.
2) Transfer learning using pre-trained models and hyperparameter analysis as two approaches at the deep learning algorithm level can significantly enhance the point cloud classification using state-of-the-art artificial intelligence techniques.
3) 3D deep learning can effectively handle the classification of large volumes of point cloud data, but the training of deep learning algorithms requires large amounts of annotated data.
4) Annotated point cloud data serve as a foundation database for the automated classification of hydraulic structures scanned by LiDAR using artificial intelligence techniques. More annotated point cloud data, which cover a wider range of hydraulic structures, are needed for further improving classification performance.

Table of Content

List of Figures

List of Tables

List of Acronyms

AGL	Above Ground Level
AI	Artificial Intelligence
CNN	Convolutional Neural Networks
CPU	Central Processing Unit
DEM	Digital Elevation Model
DSM	Digital Surface Model
FAA	Federal Aviation Administration
GCP	Ground Control Points
GIS	Geographic Information System
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GPU	Graphics Processing Units
IO	Internal Orientation
IoU	Intersection over Union
LiDAR	Light Detection and Ranging
MLP	Multi-layered perception
NAD	North America Datum
RMSE	Root Mean Square Error
RTK	Real-Time Kinematic Positioning
SfM	Structure from Motion
Sonar	Sound Navigation and Ranging
UAS	Unmanned Aerial Systems
VRS	Virtual Reference Station

1. INTRODUCTION

1.1. Background

The study of hydraulic structures such as bridges and culverts has received considerable attention in particular as the upgrade of physical infrastructure has become a national priority for the United States (ASCE 2021). Monitoring the condition of hydraulic structures such as bridges plays a pivotal role in warranting the safety of transportation infrastructure and their sustainability. The monitoring of hydraulic structures in NC has become an urgent need, in particular for the systematic management of NCDOT's assets, the development of guidelines for roadway drainage and highway stormwater management, and documentation of compliance with NCDOT and federal standards from, e.g., FEMA and FHWA. In NC, each of the State's approximately 13,500 bridges needs to be inspected by NCDOT every two years or less to ensure their structural stability and health for public safety (NCDOT 2022). Approximately 8.2% of the NC bridges are evaluated as in poor condition (by March 2021) and in need of immediate maintenance.

A suite of techniques such as LiDAR and sonar (Watson et al. 2013, Burguera and Oliver 2016) have been extensively used for the detection and measurement of hydraulic structures. For example, LiDAR techniques (typically including airborne, terrestrial, and mobile) have been recognized as a powerful and high-resolution approach for the documentation of 3D shapes of hydraulic structures and their surrounding environments (Feroz and Abu Dabous 2021). At the same time, sonar techniques can delineate 3D characteristics of underwater topography. The combination of these two techniques provides support for the monitoring of hydraulic structures for both above- and under-water conditions. Their capabilities in the quantification of 3D characteristics of hydraulic structures have been well recognized, especially when compared to traditional visual inspection methods that are often subjective and labor intensive (Prendergast and Gavin 2014).

The use of LiDAR and sonar techniques leads to the generation of large 3D point cloud datasets. These point cloud data are of great help for representing 3D characteristics of hydraulic structures, which are often fed into hydraulics models for the in-depth investigation of hydraulic structures. However, these point cloud data are unstructured and typically in large volumes. The processing of these large point cloud data tends to be both labor- and computation-intensive, which poses a significant challenge in the classification of these data. Further, different types of hydraulic structures exist and their geometric characteristics may be sophisticated and change over time. This further complicates the classification of these point cloud data for the monitoring of hydraulic structures. In other words, the classification of point cloud data collected from LiDAR and sonar techniques represents a big data-driven challenge (Tang and Feng 2017).

Artificial intelligence holds great potential in resolving the challenges associated with the classification of point cloud data from LiDAR and sonar. Over the past few years, artificial intelligence techniques, represented by deep learning, have been increasingly developed and applied to real-world problem solving (Goodfellow et al. 2016, LeCun et al. 2015). This trend will continue as artificial intelligence has become a nation-wide priority. Deep learning techniques have been applied to various studies (e.g., unmanned driving, natural language processing, remote sensing, and medical studies) to support the needs of classification, pattern recognition, and computer vision (Guo et al. 2016, Goodfellow et al. 2016). Deep learning techniques have been highly touted because of their superior performance over conventional modeling approaches. The use of deep learning techniques often leads to significant savings in labor and costs.

1.2. Research Need Definition

According to NCDOT Research Need Statement (RNS#: 9102), the NCDOT Hydraulic Unit is interested in utilizing high-resolution LiDAR and bathymetric sonar data for the detection and classification of hydraulic structures and their as-built conditions. This requires the use of artificial intelligence methods to support the automated classification of point cloud data for the detection and evaluation of hydraulic structures. An AI-based point cloud classification solution will bring significant benefits for NCDOT when LiDAR and sonar techniques are blended and used to facilitate the development of guidelines for highway stormwater management, roadway drainage, and hydraulic design and evaluation.

1.3. Research Objectives

The overall objective of this project is to develop a spatially explicit 3D modeling framework and software package that are based on deep learning as a cutting-edge artificial intelligence approach for automated and reliable classification of hydraulic structures from point cloud data (DeepHyd; see **Figure 1.1**). The deep learning-based artificial intelligence solution (DeepHyd, including framework and software tools) can help resolve the challenges associated with the extraction and classification of hydraulic features from LiDAR and sonar data while also having the flexibility and potential to incorporate additional manual survey data and information from digital photography.

To address the NCDOT research needs, this project has four goals established for the DeepHyd modeling framework:

- **Goal 1**: to collect a combination of LiDAR, sonar, GPS, aerial photometric and plane survey data from 11 NCDOT hydraulic structures in Cabarrus, Gaston, Iredell and Mecklenburg counties, NC
- **Goal 2**: to pre-process the data collected from Goal 1.
- **Goal 3:** develop a deep learning-based artificial intelligence approach for the classification of the point cloud data into hydraulic structures of interest.
- **Goal 4:** automate this entire classification effort (training, validation, and prediction) using scientific workflows.

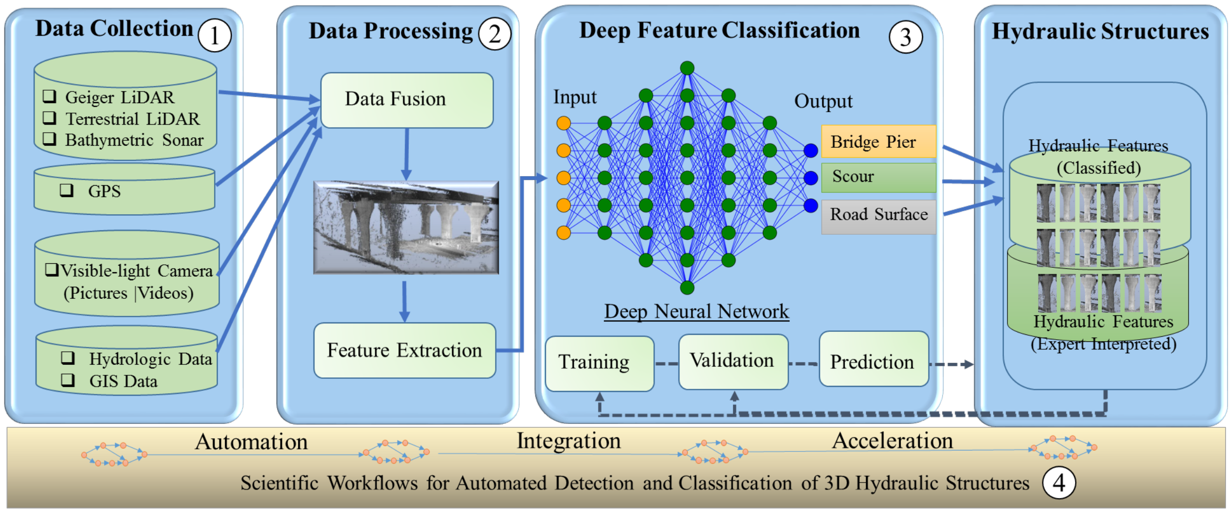


Figure 1.1. Design of DeepHyd: A deep learning-based 3D modeling framework for the automated classification of hydraulic structures from LiDAR and sonar data.

1.4. Report Organization

The rest of this report is organized in the following structure. Section 2 presents a literature review examining the role of deep learning methodologies in the classification of large volumes of point cloud data generated from LiDAR and sonar scans. Section 3 focuses on discussing the research methodology employed in this study, including field data collection, deep learning-based point cloud classification (training and validation of deep learning algorithms, model inferencing or prediction for point cloud classification), scientific workflows for the automation of point cloud classification, and software implementation. Section 4 discusses findings and conclusions from the DeepHyd research project. Section 5 presents recommendations for utilizing the DeepHyd system and suggestions for future research efforts in this area.

2. LITERATURE REVIEW

2.1. Point Cloud Data from LiDAR and Sonar

LiDAR (Light Detection and Ranging), also known as laser radar system, is an optical remote sensing technology developed for range detection (Chen 2012). By determining the heterodyne laser beam phase shifts, scanning LiDAR can detect the distance information from a plane of data points, called point cloud. The point cloud information, which basically consists of the physical positions of any surface that the laser "sees", can then be used to detect useful critical information about a structure including the elevation (underclearance), surface defects (damage quantification) and deformation under loading (deflection measurements), etc. Contrast to conventional analysis of photographic images, relatively simple algorithms can be used to manipulate the geometric point cloud data to retrieve the afore-mentioned information.

In early 2000, the Federal Highway Administration (Fuchs et al. 2004) first introduced terrestrial LiDAR for bridge testing, specifically for the static load displacement measurements. At the same time, Pieraccini et al. (2006) used laser scanning to quantify urban site creep induced by a landslide. In recent years, the utilization of LiDAR scanning has rapidly gained popularity in applications within the civil construction industry—The digital survey and graphic representation capabilities of scanning LiDAR make it an attractive and logical technology for quantifying physical features of large, massive structures such as buildings and bridges (Bisio 2017). In the late 2000s, the USDOT extended LiDAR applications to the quantification of damage on existing bridges (Watson et al. 2013) and bathymetric applications such as the mapping of stream channels and drainages (Prendergast and Gavin 2014).

Most terrestrial LiDARs are eye-safe (typical wavelength range 1040-1060 nm) and do not penetrate through water, hence, are limited to above water scans. For underwater scans, side scan sonar can be used to supplement the LiDAR data for a complete basin mapping. Character et al. (2021) demonstrated a underwater shipwreck detection application such system. More recently, bathymetric LiDAR, utilizing a green or blue-green wavelength (500 nm – 600 nm) laser, has been used for the detection of underwater bathymetry or submerged objects. The green laser can penetrate through the water column and be reflected from the bottom of the water body. As a result, a point cloud of 3D bathymetry can be obtained through this LiDAR technology. Depending on the murkiness of water, bathymetric LiDARs can penetrate a couple meters of water depth. Over the past few years, airborne bathymetric LiDAR has been increasingly applied to characterize 3D bathymetry (Mandlburger et al. 2015). For example, Kinzel and Legleiter (2019) conducted bathymetric surveys of a river in Colorado by using ASTRALite Edge bathymetric Lidar on a DJI Matrice 600 Pro drone in combination with wading and sonar techniques. They stressed that the utility of bathymetric LiDAR is affected by environmental conditions and LiDAR post-processing algorithms. Airborne topo-bathymetric LiDAR technologies have been applied to monitor the fluvial topography of Pielach River in Austria and the approach is found highly feasible for delineating riverbed topography at high resolutions (over 20 points per m^2) with low measurement error (Mandlburger et al. 2016). Table 2.1 lists bathymetric LiDARs currently available for UAS-borne surveys. Based on what has been reported to date, the UAS-borne bathymetric LiDAR laser can detect bottom features up to 3x secchi disk depth visibility (SD) (Table 2.1).

Table 2.1. List of UAS-compatible bathymetric LiDARs (SD: secchi disk depth).

Bathymetric Lidar	**Vendor**	**Weight**	**Pulse Repetition Rate (PRR)**	**Depth penetration**
Edge	ASTRALite	5kg	20 kHz	>1.5 SD
RAMMS	Fugro	14kg	27 kHz	3SD
GREEN	TDOT	2.6kg	60 kHz	Up to 13.5 meter (flying altitude)
VQ-840-G Lidar	RIEGL	12kg	50-200 kHz	2SD

High density sonar (sound navigation and ranging) techniques supplement the LiDAR above-water scans and provide similar geometric data for underwater structures. Recently, sonar has been used extensively in bridge scour mapping (Prendergast and Gavin 2014). Both LiDAR and sonar measure 3D geospatial environments and measurement errors can be introduced due to scanning angle, range, edge effects, and surface reflectivity. For deployment at sites with limited access, these uncertainty factors may be easily introduced into the collected point cloud data.

2.2. Deep Learning

The past decade has witnessed a rapid advance in Artificial Intelligence, stimulated by deep learning. Deep learning is inherently based on deep neural networks that rely on a collection of network layers for problem-solving instead of multiple layers (3 or 4) as in traditional neural networks for machine learning (Goodfellow et al. 2016). A typical neural network architecture consists of three types of layers: input, hidden, and output. Deep neural networks, with support from deep learning techniques, allow for the use of many hidden layers for computation. These hidden layers are connected to learn a hierarchical representation of the problem of interest. There exist different types of deep neural networks (Goodfellow et al. 2016, Guo et al. 2016, Shrestha and Mahmood 2019): stacked autoencoder, deep belief networks, convolutional networks, and recurrent neural networks. Among them, convolutional neural networks (CNN) are representative of deep neural networks that have been widely developed and used for computer vision, pattern and image recognition, and classification (Goodfellow et al. 2016, Guo et al. 2016). CNN for computation is inspired by the study of visual cortex (vision). CNN uses a combination of convolution (filters) and pooling operations (e.g., max pooling) to learn features hidden in data. Via a series of combined convolution and pooling operations, low-level features are extracted and organized into higher-level features that correspond to the objects to be detected. Deep learning algorithms have been increasingly proposed and used for a variety of applications (Goodfellow et al. 2016, Guo et al. 2016). A suite of deep learning software platforms or libraries are now available (Shrestha and Mahmood 2019), represented by Caffe (https://caffe.berkeleyvision.org/), Tensorflow (https://www.tensorflow.org/), Keras (https://keras.io/), and PyTorch (https://pytorch.org/). Most of these deep learning software platforms are implemented in C/C++ and Python. To increase the efficiency of deep learning algorithms, Graphics Processing Units (GPUs) have been used for acceleration by leveraging their multiple-core computing power (Shi et al. 2016).

When we use deep learning (or any machine learning algorithms) for problem-solving, three steps are involved: training, testing (also known as validation), and inference (also known as prediction). For classification-type problems (with categorical data), annotated data are needed for training and testing of the deep learning algorithm. Once a deep learning algorithm is trained and tested, we could use the trained deep learning algorithm for prediction—i.e., model inferencing. To evaluate the model performance during training and testing, a suite of performance metrics are available. Performance metrics for classification problems are typically based on the use of confusion matrix and include, but are not limited by averaged accuracy, Intersection over Union (IoU), precision, and recall (Hossin and Sulaiman 2015, Rahman and Wang 2016).

2.3. Deep Learning for Point Cloud Classification

A series of methods have been developed to use deep learning techniques for point cloud classification. These methods mainly include three types: 1) multi-view based, 2) volumetric-based, and 3) point-based (Guo et al. 2020, Qi et al. 2017). Multi-view methods basically convert 3D point clouds into 2D images that leverage 2D CNN for classification (Ibrahim et al. 2021). Once 2D images are classified, they are projected back to point clouds in the 3D space. The voxel-based method relies on the use of a volumetric grid: point clouds are aggregated into a volumetric grid (similar to the rasterization of vector-based GIS data into a 2D grid), then 3D CNN are applied for classification. However, the use of 3D CNN is computationally demanding, which limits the resolution of a volumetric grid (coarse resolution is often used) (Guo et al. 2020). As a result, fine-scale geometric information from points may be lost due to the use of this coarse resolution. The third method is directly based on 3D points instead of converting point clouds into different representations (2D images or 3D voxels). The direct use of 3D points for deep learning-based classification is nontrivial because points in point clouds are unstructured and unordered, compared to regular and ordered representations in 2D images and 3D voxels. Point-based methods for 3D deep learning for point cloud classification mainly include pointwise multi-layered perception (MLP), convolution-based, graph-based, and recurrent neural network-based (see Guo et al. 2020). We focus our discussion here on two methods, pointwise MLP and convolution-based, through the use of two state-of-the-art 3D deep learning frameworks: PointNet and ConvPoint. Please refer to Guo et al. (2020) for other 3D deep learning methods.

PointNet (Qi et al. 2017) is a pioneering 3D deep learning architecture that directly uses points for deep learning-based classification, and is representative of pointwise MLP methods. The fundamental approach of PointNet is that 3D points are transformed into canonical space via a symmetric function. By this means, the trained networks are permutation-invariant to the input point clouds (the re-ordering of points that are unordered will not influence the predicted results of deep learning algorithms). Multi-layered perception (MLP) combined with max-pooling are used to approximate this permutation-invariant transformation. This network design provides an elegant way of coping with the unstructured and unordered characteristics of points in a point cloud and can effectively learn features directly from a point cloud. PointNet and its improved version PointNet were implemented in Tensorflow (deep learning platform from Google). A number of 3D deep learning approaches for point cloud classification stem from the PointNet framework (Guo et al. 2016).

ConvPoint (Boulch 2020) is representative of using a convolution-based approach for 3D deep learning. ConvPoint is based on the architecture of CNN for 3D deep learning directly on point clouds. ConvPoint relies on the use of continuous convolutions (as a generalized version of discrete convolutions). A geometric weighting function learned by a MLP is used to transform characteristics of discrete points to kernels that are distributed within a unit sphere. Each of these kernels for convolutions covers a subset of points (neighbors) within a radius of the location of the kernel—i.e., a neighborhood search is needed (implemented using a k-d tree). In ConvPoint, a kernel takes into account the information from both a feature domain and spatial domain. The ConvPoint architecture is invariant to the permutation of points as it uses the summation operation (symmetric) in the geometric weighting function. Further, ConvPoint uses relative coordinates between points in point clouds and locations of kernels, which makes it invariant to geometric transformations such as rotations. Based on the building block of continuous convolution, ConvPoint supports the point cloud classification by combining five layers of convolutional layers and one fully connected layer. These convolutional layers extract features from points and then assemble them into higher-level features. The segmentation of a point cloud is implemented by using an encoder-decoder design in which convolutional layers in the classification architecture (feature extraction) are connected with another set of five convolution layers for decoding operations (segmentation). ConvPoint was implemented using PyTorch, a deep learning library. Based on testing on a series of point cloud benchmark datasets, ConvPoint, in general, achieved model performances (for classification and segmentation) that are higher than those of other 3D deep learning algorithms (including PointNet).

3. RESEARCH METHODOLOGY

In this section, we present research methods that have been used for field data collection and for DeepHyd system development for deep learning-based point cloud classification of hydraulic structures. These methods include 1) field data collection, 2) deep learning-based point cloud classification, and 3) scientific workflows for model automation. The products from each method is then integrated into the DeepHyd system.

3.1. Field Data

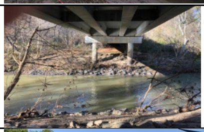
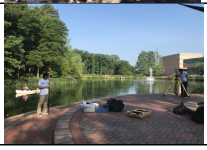
3.1.1. Field Sites

We identified a collection of potential sites for field surveys in this project. With consultation with NCDOT, eleven sites were selected to conduct field surveys (see Table 3.1). These sites are located in several counties in the Greater Charlotte Metropolitan region, including Gaston County, Cabarrus County, Iredell County, and Mecklenburg County. For each of these selected sites, we conducted terrestrial LiDAR scans. We also conducted sonar-based bathymetric surveys at three sites. To ensure the scans can be geo-referenced, total stations and high-resolution GPS surveys were conducted at eight of the eleven sites. We also collected RGB (red-green-blue) imagery using drones at two of the sites, and digital camera images at the eight sites. In addition, we opted to use a bridge (Philips Road Bridge) and a lake (Hechenbleikner Lake) on the campus of UNC Charlotte as experimental sites. Terrestrial LiDAR and sonar scanning data (lake only) were collected from Phillips Road Bridge and Hechenbleikner Lake.

In response to a request from NCDOT, we set up a Web GIS portal (based on Esri ArcGIS Online; see Figure 3.1) to guide our field work and facilitate the sharing of information. The Web GIS portal is hosted on a web server at the Center for Applied GIScience at UNC Charlotte, and the URL of the portal is: https://cybergis.uncc.edu/deephyd/index.php/study-sites/

Table 3.2 summarizes the field trips that we conducted for this project. These field trips were used for: 1) calibration of survey instruments (including total station, LiDAR, sonar, and UAS), and 2) LiDAR and sonar surveys for the collection of point cloud data from the various study sites.

Table 3.1. List of survey sites and data collected for the project.

Site #	Snapshot	Coordinate (WGS84)	Type	Description
Site 2		35.280754 - 81.016666	Bridge	Tuckaseege Road over Fites Creek, Mt Holly, NC
Site 3		35.262183 -81.063021	Culvert	Eastwood Drive over an unnamed branch of the South Fork Catawba River, Belmont, NC
Site 5		35.289614 -81.192872	Bridge	Tulip Drive over Kaglor Branch, Gastonia, NC
Site 6		35.279093 -81.187093	Bridge	Caldwell Street over an unnamed rivulet of Long Creek, Gastonia, NC
Site 7		35.251552 -81.217522	Bridge	West 5th Avenue over Blackwood Creek, Gastonia, NC
Site 8		35.333853 -80.668114	Bridge	Morehead Road over Mallard Creek, Harrisburg, NC.
Site 11		35.427483 -80.956422	Bridge	Highway 73 over Catawba River immediately downstream of Cowan's Ford Dam
Site 14		35.318134 -80.737441	Bridge	North Tryon Street over Mallard Creek, Charlotte, NC
Site 15		35.306664 -80.738421	Bridge	Philips Road, over Toby Creek, UNC Charlotte Campus, Charlotte, NC
Site 16		35.329582 -80.617986	Bridge	Pharr Mill Road over Rocky River, Harrisburg, NC
Site 17		35.304216 -80.731512	Lake	Hechenbleikner Lake, UNC Charlotte Campus, Charlotte, NC

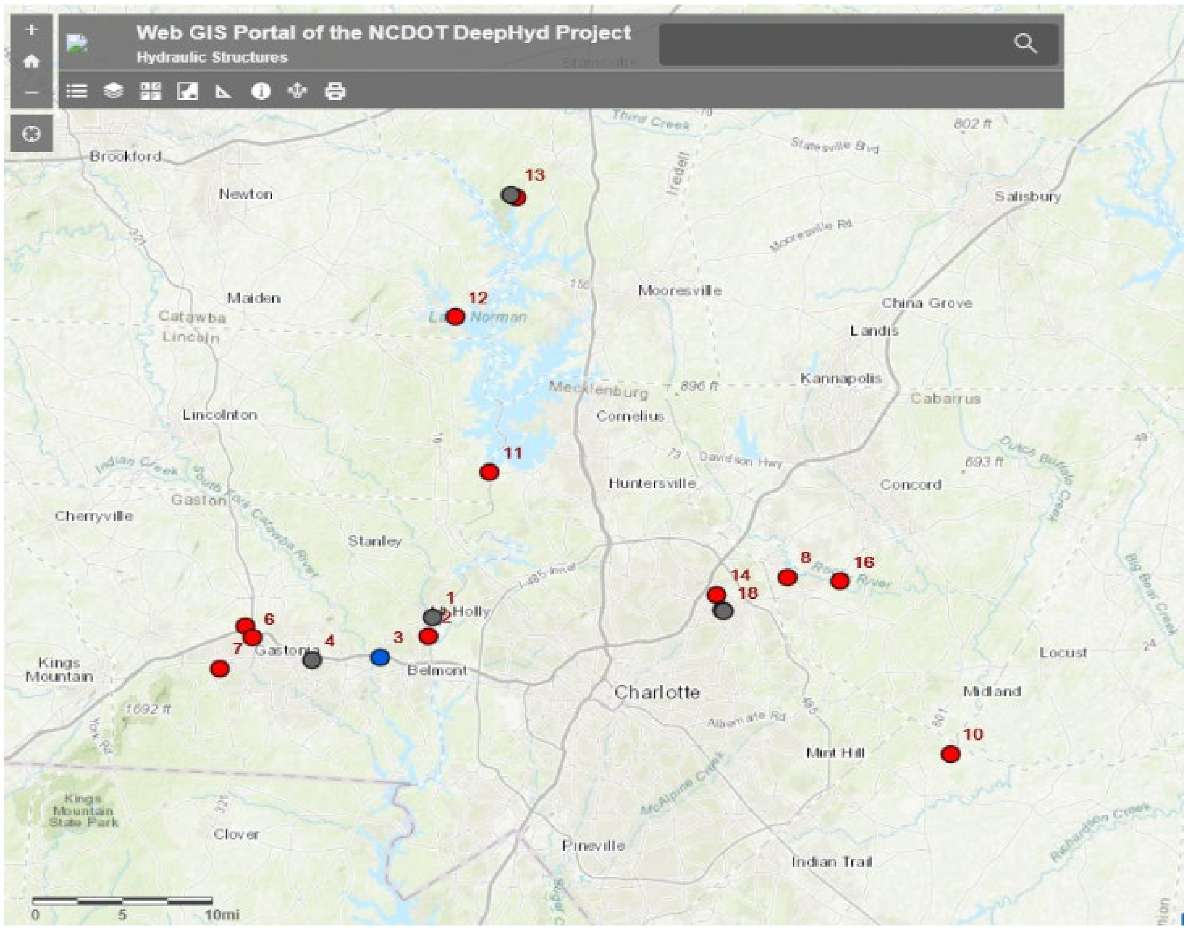


Figure 3.1. Web GIS portal for the field surveys in the project (number labels are IDs of sites).

Table 3.2. Summary of field work conducted for the project.

Survey Dates	Survey Purposes (calibration, LiDAR, sonar)
November 16th, 2018 December 1st, 2018	Sonar Survey and Lidar survey
January 16th, 2019	Calibration of total station at NCGS Statesville EDM Calibration Base Line (Statesville Regional Airport)
May 16th, 2019 January 7th, 2020	UAS calibration at NCGS/NCDOT UAS Test Site at Butner, NC
May 30th, 2019	Sonar and Lidar test at the Hechenbleikner Lake at UNC Charlotte
January 26th, 2020 February 1st, 2020 February 2nd, 2020	Lidar and Sonar Survey at study sites

3.1.2. Field Data Collection Methods

In this project, we collected point cloud data using terrestrial LiDAR and sonar techniques. We also used a Topcon total station, a survey-grade GPS, and a Phantom drone to collect additional field site data together with LiDAR and sonar instruments. Table 3.3 lists the specific information for the instruments used in this project. The survey-grade GPS we employed was a Trimble R10 GNSS receiver (rented) with support of network RTK (horizontal resolution: 8mm+0.5ppm; vertical resolution: 15mm+0.5ppm). The survey-grade GPS requires the use of a Virtual Reference Station (VRS) network, which is the North Carolina VRS network supported by NC Geodetic Survey. The GPS and total station were used to collect information for ground control points (GCPs) for georeferencing purposes. A Lowrance sonar system (single beam) mounted on a canoe was used for the bathymetric survey, supplemented by the total station and survey-grade GPS to collect information about underwater topography.

To ensure the survey quality of our work, we conducted calibration trials of the various instruments used in this project. For terrestrial LiDAR, it was calibrated by the vendor (certified by FARO). We calibrated the Topcon total station on the Statesville EDM Calibration Base Line (established by NCGS in 2016). Our Topcon DS-226 total station has the following accuracies: angle accuracy: 6”; distance accuracy: ±(2mm+2ppm*distance). Our drone was calibrated at the Butner, NC, NCGS/NCDOT UAS Test Site.

Table 3.3. Summary of data collection instruments.

Data Collection Method	Instrument Information
Terrestrial LiDAR	FARO Focus S350
Bathymetric sonar	Lowrance HDS-12 Gen3 with StructureScan^{TM}
UAS (drone)	DJI Phantom 4 Pro V2.0
Total Station	Topcon DS-226
GPS	Trimble R10 GNSS receiver
GPS Camera	Sony DSC-HX400V (Image resolution: 5,184 * 3,888)

3.1.2.1. Terrestrial Lidar Survey

The terrestrial LiDAR surveys relied on the combined use of terrestrial LiDAR, a total station, and a survey-grade GPS. Multiple LiDAR scans were conducted for each site, and the collected point cloud data were stitched together. Once the data were collected, we applied a series of post processing steps to these data. The processing of the collected data included the following four steps: 1) stitch data collected by the total station and LiDAR for each site, 2) evaluate the accuracy of the stitching operation (if there are more than three points observed), 3) identify more than three feature points in the stitched point clouds to be surveyed by the identified instruments and then use those feature points for georeferencing, and 4) evaluate the accuracy of the georeferencing operations. As an example, Figure 3.2 shows the map of stitched LiDAR point cloud for Site 16 Pharr Mill Road over Rocky River, Harrisburg, NC.

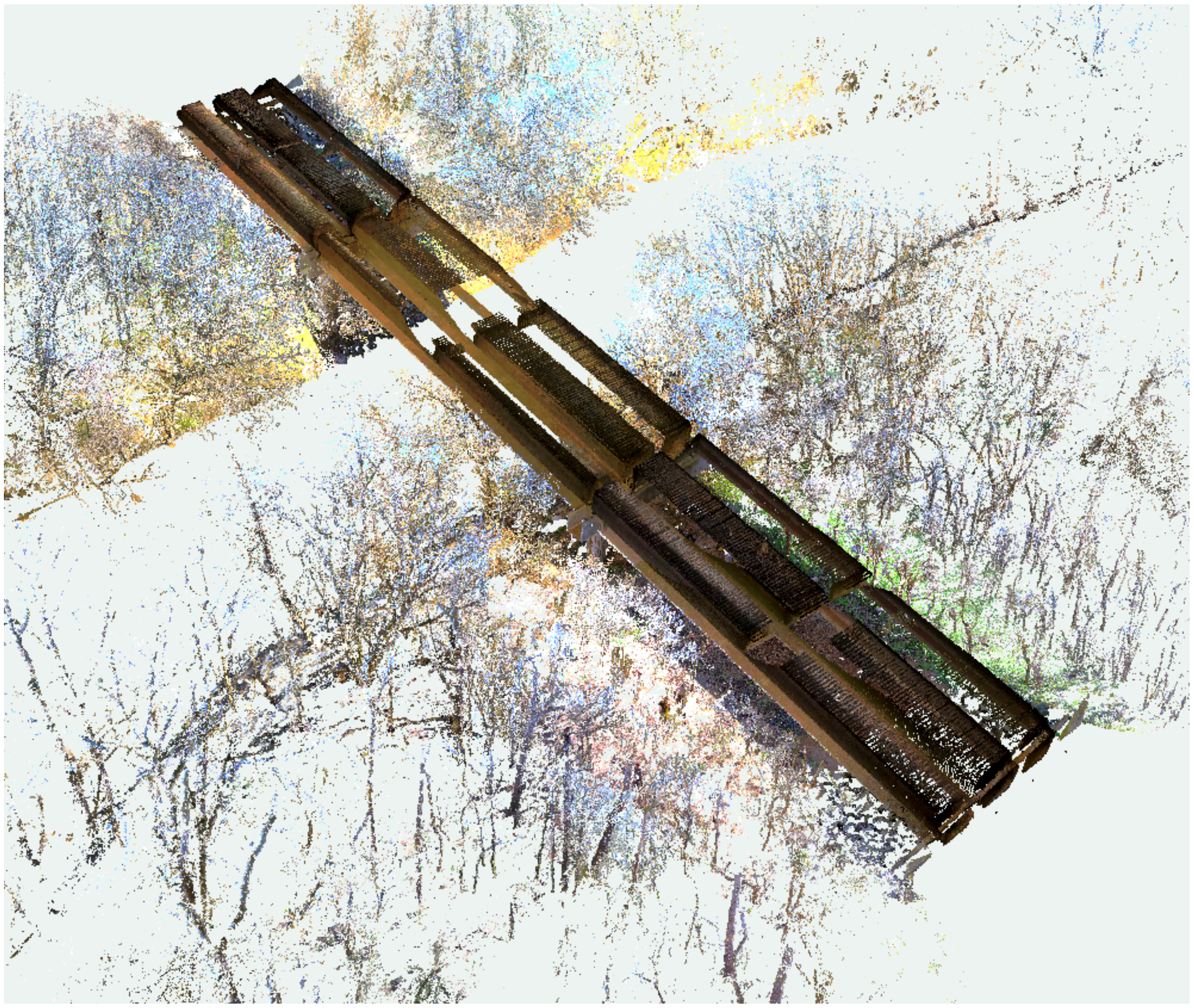


Figure 3.2. Illustration of collected LiDAR point cloud (Site# 16, stitched from two scans with a total of 147,166,173 points; see Table 3.1 for 2D image of the site).

3.1.2.2. Bathymetric Survey: Sonar and Total Station Data Collection

We used a single beam sonar (Lowrance HDS-12 Gen3 with StructureScan^{TM}) to collect bathymetric data mainly at one site (Site 16). We tested our sonar system at the Hechenbleikner Lake experimental site on UNC Charlotte campus. Then, with recommendation from NCDOT, we focused on Site 16 to collect bathymetric data using the sonar system.

Site 16 is the Pharr Mill Road site (see Table 3.1 as needed), where Highway 1158 crosses the Rocky River (see Figure 3.3). The channel at that site has sufficient water depth (>1.0 ft) to permit the use of sonar for most of its width. The sonar transects were designed to provide coverage of the channel bathymetry for a reach extending about 270 feet (90 m) at the bridge. Multiple, flow-parallel sonar runs were made which provided bank-to-bank coverage of the channel reach. The sonar system collected bathymetric data and stored it in slg (sonar data only), sl2 (sonar and structure scan), sl3 (sonar and 3D structure scan), and .gpx (X, Y coordinates of waypoints and tracks) formats. The single beam echo sounder depth data collection was synchronized with positioning information. We used a frequency of 200 kHz for the data collection and open source C# API, SonarLogAPI (https://github.com/risty/SonarLogApi), for the conversion of data (from binary to .csv format) and extraction of depth data.

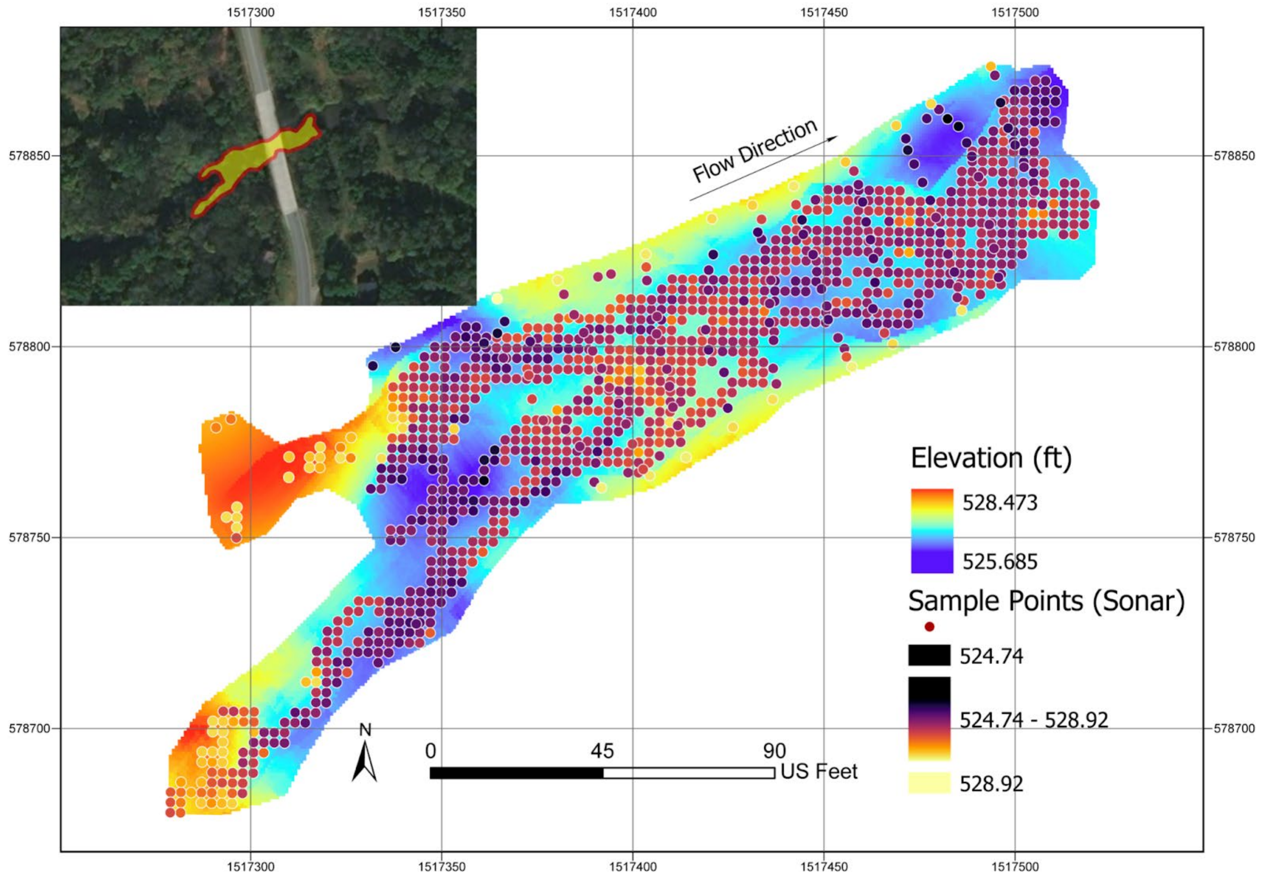


Figure 3.3. Bathymetry at Site 16, Pharr Mill Bridge, Cabarrus County, NC. Data were collected using sonar and a survey-grade GPS using Virtual Reference Station. The seven transects were transverse to the river and spaced about 5 meters (15 feet) apart. Three transects were upstream of the bridge, one transect was beneath the bridge, and three transects were downstream of the bridge. The sonar data were collected during multiple flow-parallel runs along the 90 meter (270 foot) reach illustrated in the map.

In addition to sonar data, we collected bathymetric data using the survey-grade GPS leveraging VRS to validate the accuracy of the data. The average water edge elevation measured with the VRS served as a reference to calculate the elevation of the sample points (depth data) collected from the sonar as shown in Table 3.4. For the validation of the sonar data at Site 16, we selected 82 points collected by GPS, which were also measured by the sonar. The depth values measured from sonar were subtracted from the reference elevation (528.6 ft) to estimate the stream bottom elevation of each point.

Table 3.4. Estimation of average water edge elevation at Site #16 (7 transects were used). The transects were flow-transverse and spaced about 5 meters (15 feet) apart. There were 3 transects upstream of the bridge, one underneath the bridge, and 3 located downstream of the bridge, all within the area mapped using the sonar system. See Table 3.1 for site information.

Transect number	Water Edge elevation (ft) (Home bank, far bank)	Average water edge elevation (ft)
1	528.62, 528.68	528.65
2	528.70, 528.66	528.68
3	528.52, 528.52	528.52
4	528.58, 528.46	528.52
5	528.74, 528.68	528.71
6	528.52, 528.40	528.46
7	528.64, 528.46	528.69
Average water edge elevation (ft): 528.60		
Standard deviation (ft): 0.101		

3.1.2.3. 3D image reconstruction using drone and RGB camera

The UAS operations in this project fall under the public operation category and the drone used in this project was a DJI Phantom 4 Pro v 2.0. We adhered to all FAA and NCDOT mandatory operational requirements along with UNCC UAS policy. The UAS was registered under FAA and a certificate of authorization was obtained before carrying out the operations along with NCDOT UAS knowledge test certification. The UAS in this project was covered by a liability insurance and all flight missions were carried out below 400 feet AGL (Above Ground Level) and the obstacle sensors were tested by maneuvering the UAS in the vicinity of different obstacles. All UAS missions were conducted via automated flight plan under the responsibility of remote pilot and presence of visual observers.

The digital photo image acquisition from small UAS (sUAS) and subsequent data processing involved four components: 1) planning, 2) flight/data collection, 3) data processing, and 4) data analysis. The ground control points (GCP) are established before carrying out sUAS operations. These GCPs assist in georeferencing the images during subsequent analysis. The point cloud generated from sUAS photogrammetry is used to generate a high-resolution digital elevation model (DEM) or digital surface model (DSM).

In this project, we established 4 to 5 ground control points (GCPs) all around the hydraulic structures at a site before carrying out the flight mission for the site. The GCPs were measured using a Trimble R-10 VRS system. As UAS are prone to wind gusts and turbulence, we made sure that the UAS ground speed was lower than 5 m/s, which may lead to low quality images due to motion blur. The blurriness results from either the object moving or excessive speed of the platform to which the sensor/camera is attached, or both. For sUAS platforms, the blurriness can be effectively reduced by increasing flight altitude and/or decreasing sUAS ground speed. In addition, before each mission we calibrated the camera and gimbal to minimize the motion blur. Below we presented the UAS operations that we conducted at two sites (Site #5, and Site #16).

UAS operation at Site #5

This UAS operation was conducted on February 1st, 2020 and a total of 226 images were collected and 5 ground control points (GCPs) were established. The GCPs were measured using the Trimble R10 integrated GNSS system. As the flight altitude was kept at 200 ft, the ground speed of UAS was set lower at 3m/s to reduce motion blur. For the weather conditions for this flight, please see Table A1 in Appendix A. The flight operated along the stream at an altitude of 200ft and the planning parameters are shown in Table A2 in Appendix A, with the flight plan shown in Figure 3.4. The compass, camera and gimbal were all calibrated before the operation. A photogrammetry software Pix4DMapper (https://www.pix4d.com/product/pix4dmapper-photogrammetry-software) was used for the post processing using the Structure from Motion (SfM) technique.

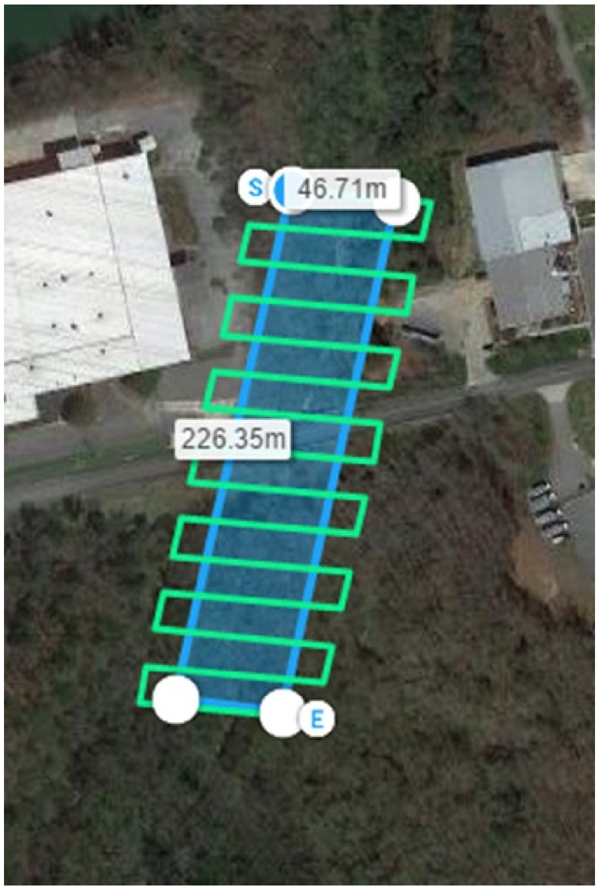


Figure 3.4. Flight plan for unmanned aerial system operation at Site #5 (see Table 3.1 for site information).

The five control points measured using Trimble R-10 GPS with VRS are shown in Table 3.5. The GCPs are marked manually in the selected images. The localization error per GCP and mean error along X, Y and Z coordinate directions are shown in Table 3.6. Table 3.7 shows the mean and standard deviation of errors between the initial and computed image positions. The key points details in alignment are reported in Table 3.8. The orthomosaic with 3D information generated for Site #5 is shown in Figure 3.5. The camera calibration information is shown in Table 3.9.

The internal orientation (IO) parameters define the origin of the image space coordinate system. These parameters are focal length, principal point, radial and tangential lens distortions. The length from the principal point to the perspective center is called the focal length. While the principal point is the image position where the optical axis intersects the image plane. The camera calibration information (Table 3.9) provides the detail on IO parameters. R1, R2, R3 are the radial distortions along X, Y and Z axes while T1 and T2 are tangential distortions in the vertical plane.

Table 3.5. List of ground control points for Site #5.

Location	Point ID	Easting NAD83(2011) US Survey Feet	Northing NAD 83(2011) US Survey Feet	Orthometric Height NAVD88 US Survey Feet
Site #5	G83	1345671.988	567497.151	686.318
Site #5	G84	1345729.220	567591.108	687.625
Site #5	G85	1345722.823	567474.747	691.910
Site #5	G86	1345685.136	567608.737	687.687
Site #5	G87	1345590.718	567656.860	696.856

Table 3.6. Localization accuracy per ground control point (unit: US Survey feet).

GCP Name	Error X	Error Y	Error Z	Images Marked
G83	-0.022	-0.023	0.098	20
G84	0.027	0.011	0.017	20
G85	-0.007	-0.002	-0.070	20
G86	-0.013	0.018	0.022	20
G87	0.008	-0.020	-0.025	13
Mean	-0.001583	-0.003005	0.008294	
RMSE	0.017269	0.016621	0.056500	

Table 3.7. Statistics of errors between initial and computed image positions (std: standard deviation; unit for X, Y and Z: foot; unit for Omega, Phi, and Kappa: degree).

	X	Y	Z	Omega	Phi	Kappa
Mean	0.017	0.016	0.028	0.015	0.008	0.002
Std	0.004	0.003	0.015	0.008	0.004	0.001

Table 3.8. Summary of key points and matched key points per image.

	#2D key points per image	#Matched 2D key points per image
Median	64,822	7,190
Min	29,897	310
Max	79,640	20,280
Mean	62,148	7,575

Table 3.9. Information on camera calibration.

	Focal Length (mm)	Principal Point X (mm)	Princip al Point Y (mm)	R1	R2	R3	T1	T2
Initial Values	8.600	5.704	4.278	0.004	-0.017	0.019	0.000	0.000
Optimized Values	9.283	5.702	4.266	0.004	-0.023	0.027	-0.001	0.000
Uncertainties	0.011	0.000	0.000	0.000	0.001	0.001	0.000	0.000

Note: Focal length: distance from the front nodal point of the lens to the plane of the photograph/image. Principal point: the point where the perpendicular dropped from the center of lens meets/strikes the plane of the photograph/image. R1, R2, and R3 are the radial distortions along X, Y and Z axes while T1 and T2 are tangential distortions in the vertical plane.

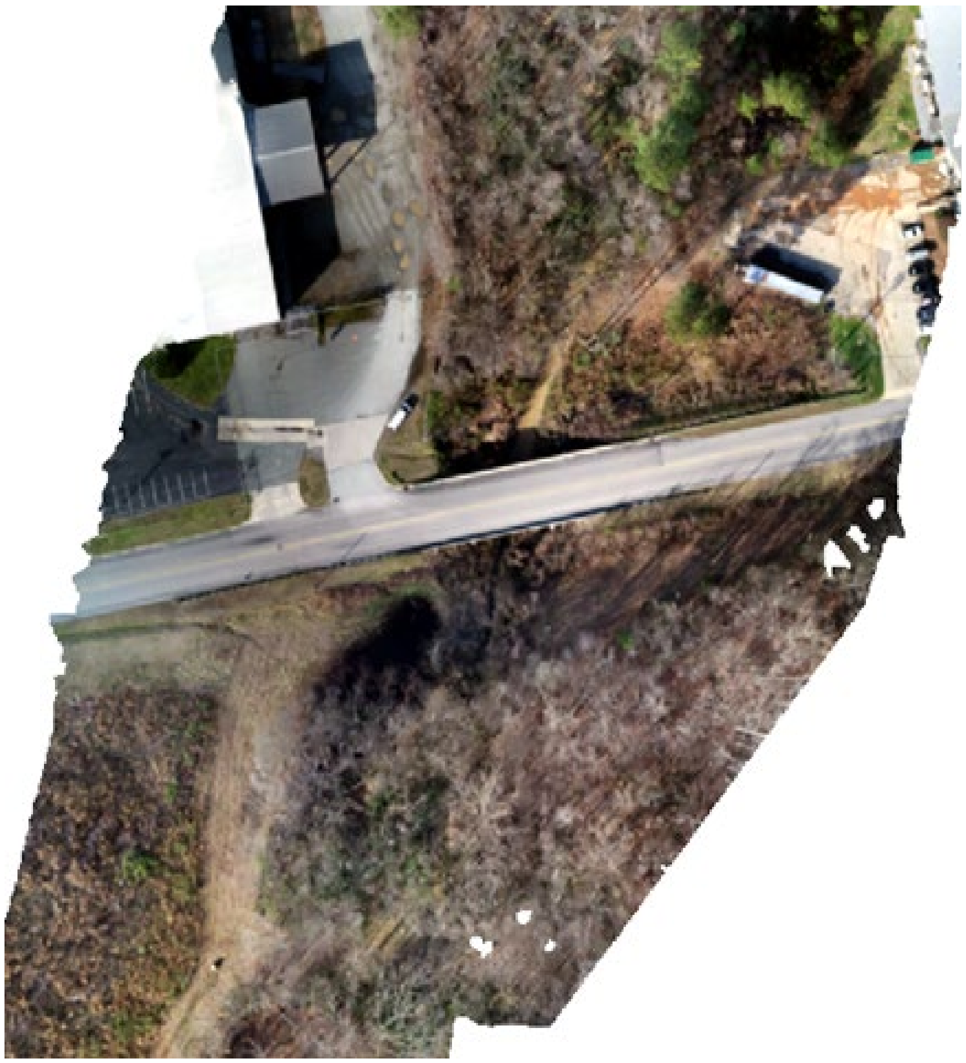


Figure 3.5. Orthomosaic (with 3D information) generated using Structure from Motion technique for Site #5 (see Table 3.1 for site information).

UAS operation for Site #16

This flight operation was conducted on January 26th, 2020 and a total of 105 images were collected. A total of four GCPs were established using the Trimble R10 GPS with VRS support. Two flight operations were conducted at a mission altitude of 200ft with a UAS ground speed of 5m/s. The flight plans are shown in Figure 3.6 and 3.7. Please see Table A3 and A4 in Appendix A for the weather details for the flight and the planning parameters for flight operation.

The four control points measured using the Trimble R10 GPS are shown in Table 3.10. The GCPs are marked manually in the selected images. The localization error per GCP and mean error along the X, Y and Z coordinate directions are shown in Table 3.11.

Table 3.12 shows the mean and standard deviation of errors between the initial and computed image positions, where Mean and Sigma are the mean value of and the standard deviation of errors in X/Y/Z direction in feet and Omega/Phi/Kappa the angle in degrees. The alignment details are displayed in Table 3.13. The camera calibration information is shown in Table 3.14. The orthomosaic (with 3D information) generated for Site #16 is as shown in Figure 3.8.

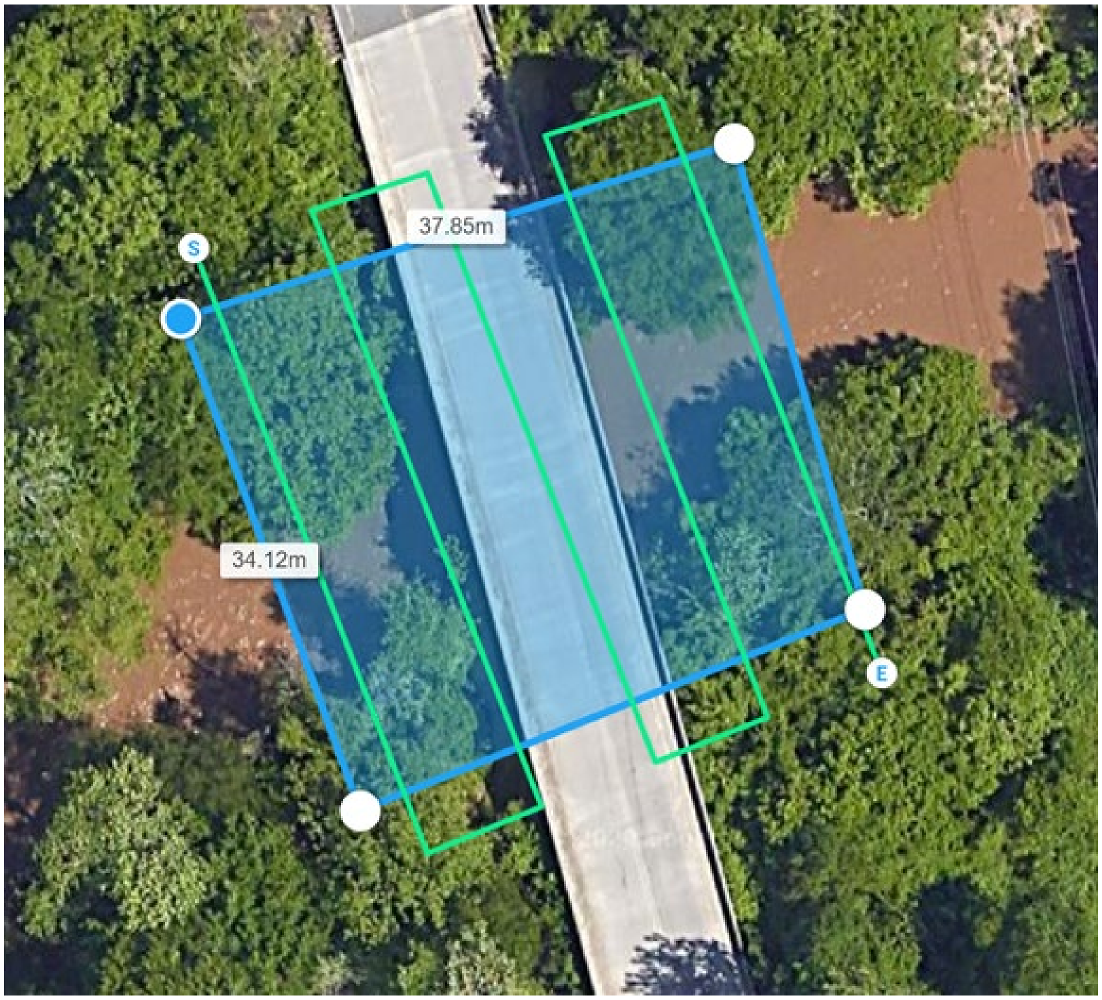


Figure 3.6. Flight plan I for UAS operation at site 16 (see Table 3.1 for site information).

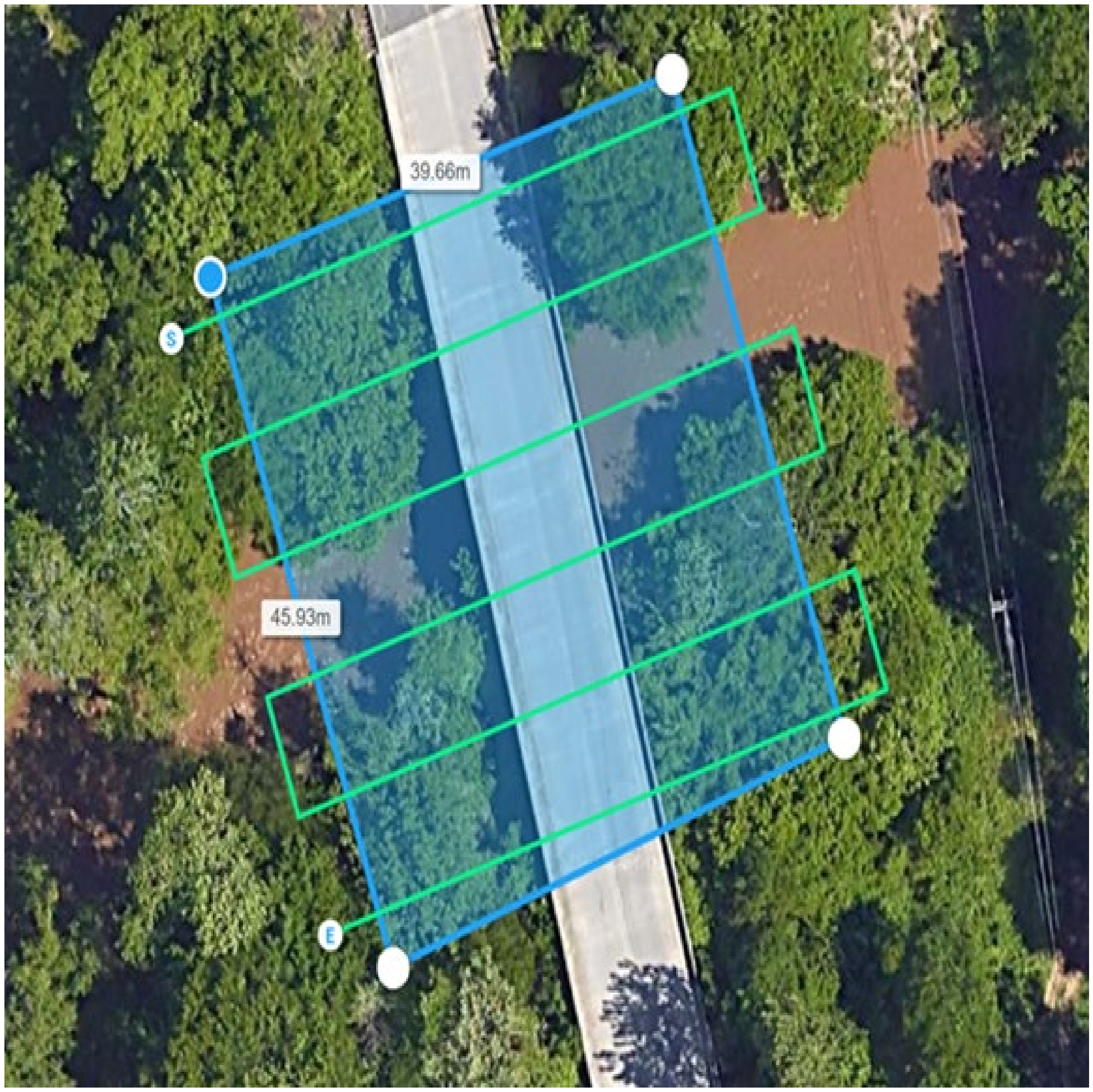


Figure 3.7. Flight plan II for UAS operation at site16 (see Table 3.1 for site information).

Table 3.10. List of ground control points for Site #16.

Location	Point ID	Easting NAD 83(2011) US Survey Feet	Northing NAD 83(2011) US Survey Feet	Orthometric Height NAVD88 US Survey Feet
Site #16	Drone6	1,517,394.289	578,719.393	537.632
Site #16	Drone8	1,517,550.486	578,807.289	537.54
Site #16	Drone14	1,517,503.65	578,941.251	539.71
Site #16	Drone15	1,517,385.129	578,860.992	540.987

Table 3.11. Localization accuracy per ground control point (std: standard deviation; unit: US survey feet).

GCP Name	Error X	Error Y	Error Z	Images Marked
Drone6	0.012	-0.006	0.020	22
Drone8	-0.008	-0.013	-0.012	24
Drone14	-0.024	-0.031	0.126	11
Drone15	0.006	0.018	-0.021	25
Mean	-0.003600	-0.007988	0.028123	
Sigma (std)	0.013957	0.017594	0.058376	
RMSE	0.014414	0.019323	0.064797	

Table 3.12. Statistics of initial and computed image positions (std: standard deviation).

	X (foot)	Y (foot)	Z (foot)	Omega (degree)	Phi (degree)	Kappa (degree)
Mean	0.022	0.023	0.297	0.006	0.006	0.001
Sigma (std.)	0.005	0.004	0.002	0.001	0.001	0.000

Table 3.13. Summary of 2D key points and matched 2D key points per image.

	# 2D key points per image	# matched 2D key points per image
Median	56,041	7,760
Min	51,293	4,837
Max	62,020	12,020
Mean	56,214	7,731

Table 3.14. Information on camera calibration.

	Focal Length (mm)	Principal Point X (mm)	Principal Point Y (mm)	R1	R2	R3	T1	T2
Initial Values	8.600	5.704	4.278	0.004	-0.017	0.019	0.000	0.000
Optimized Values	9.128	5.694	4.293	-0.004	-0.015	0.019	-0.001	0.000
Uncertainties	0.013	0.000	0.000	0.000	0.001	0.001	0.000	0.000

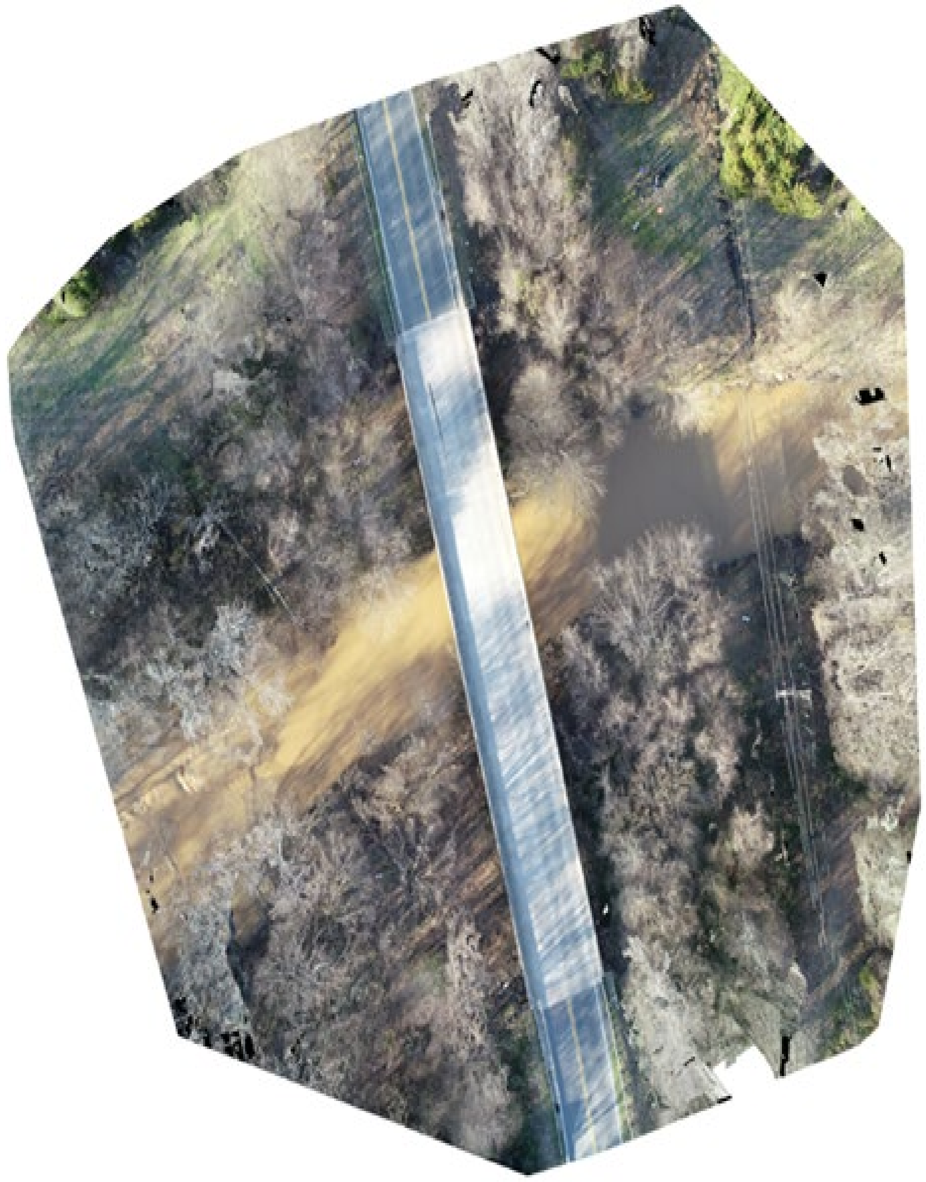


Figure 3.8. Orthomosaic (with 3D information) generated using SfM technique for Site #16 (see Table 3.1 for site information).

3.1.3. Summary of Field Data Collected

Table 3.15 summarizes the data collected in this project, including LiDAR, sonar, total station, drone images, and camera images. Table 3.16 shows the file formats of these data. As an example, Figure 3.9 shows a map of 3D point cloud with LiDAR and sonar data merged. These data provide empirical support for the development of the deep learning algorithm for point cloud classification. The horizontal and vertical coordinate systems that we used are NAD 1983 (2011) StatePlane North Carolina FIPS 3200 (US feet) and GEOID12B as suggested by NCDOT.

Table 3.15. List of survey sites and data collected for the project (see Table 3.1 for site information).

Site #	# LiDAR Scanning	# Sonar Points	# Total station points	# Drone images	# Camera images
Site 2	1		86		308
Site 3	2		98		157
Site 5	1		241		220
Site 6	2		101		363
Site 7	1		95		251
Site 8	3		168		398
Site 11	5	824			
Site 14	1		205		420
Site 15	1			181	213
Site 16	4	1095	127	109	
Site 17	4	3,180			

Table 3.16. Summary of field data types collected.

Data Type	Data Format	File Format
LiDAR Point Cloud	XYZ, RGB	.fls (Faro format)
Sonar Data	XY, depth	.sl2, .sl3 and .slg
Total station Data	Latitude, longitude, elevation	CSV
Drone images	RGB image	JPEG
Camera images	RGB image	JPEG

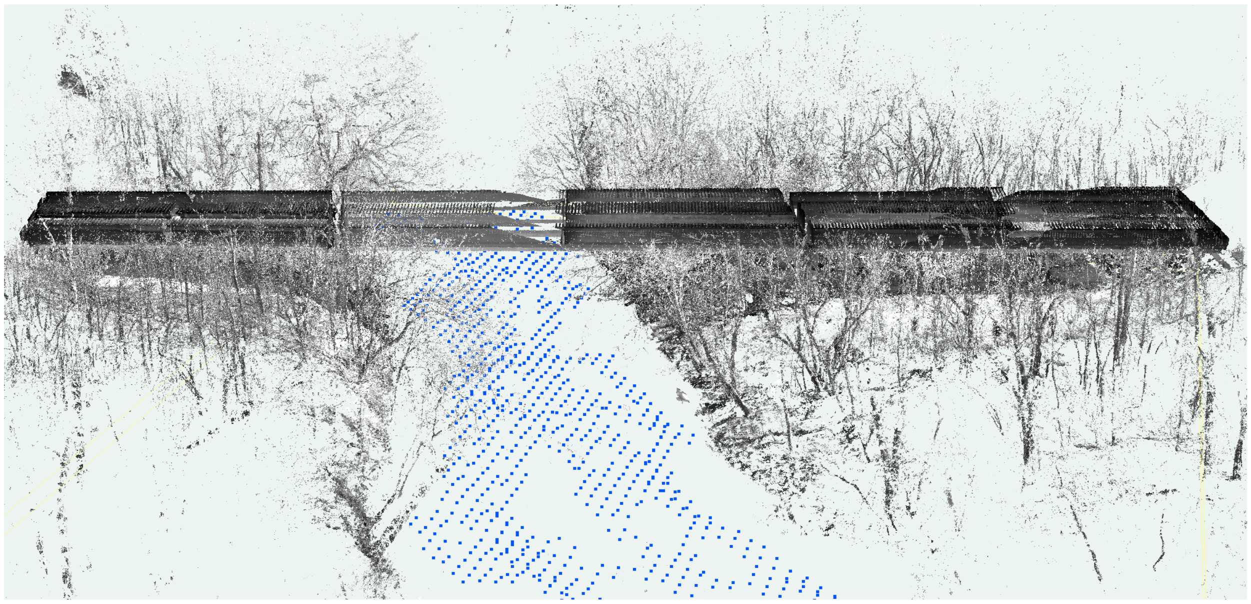


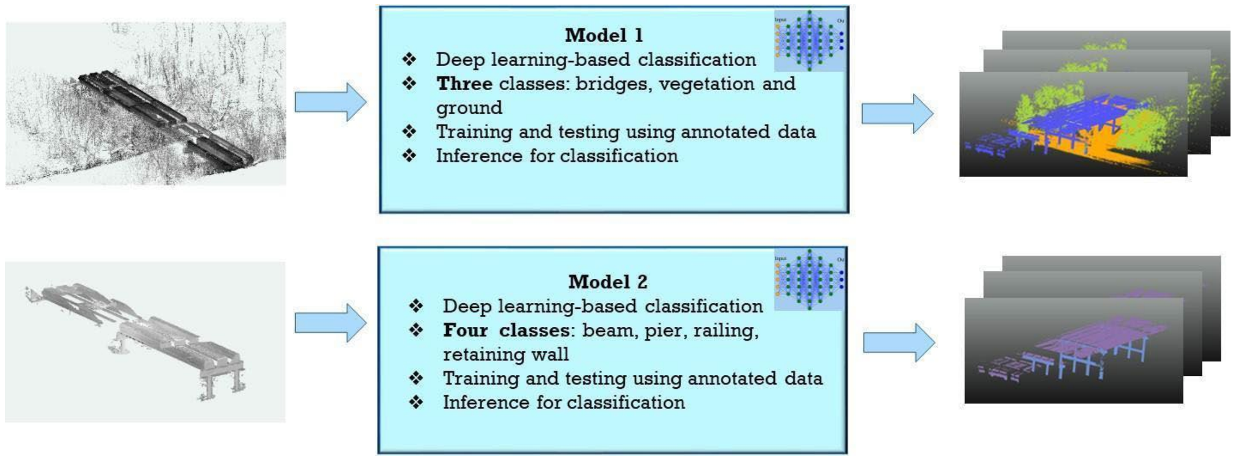
Figure 3.9. Illustration of map of fused Lidar and sonar data (Lidar data are in gray; sonar data are in blue; site#: 16; Pharr Mill Road site; see Table 3.1 for site information).

3.2. Deep Learning-based Point Cloud Classification

3.2.1. Model Design

The DeepHyd system is designed in a two-tiered spatial modeling framework for the point cloud classification of hydraulic structures (see Figure 3.10). The first tiered model, referred to as Model 1, is designed for the classification of bridges, vegetation, and ground. The second tiered model, denoted as Model 2, is used to classify bridge components from a point cloud of a bridge. Specific bridge components include beam, pier, railing, and retaining wall. The combination of the two tiers of the model allows for the classification of hydraulic structures from point cloud data.

The use of a deep learning algorithm for the classification of point cloud data requires the following steps: 1) annotation of point cloud data as training data; 2) training and testing of the deep learning algorithm using annotated data, and 3) inferences (prediction) of point cloud data for classification using the trained deep learning algorithm. In this section, we discuss these specific steps based on the two-tiered spatial modeling framework.



Model 1
❖ Deep learning-based classification
❖ Three classes: bridges, vegetation and ground
❖ Training and testing using annotated data
❖ Inference for classification
Model 2
❖ Deep learning-based classification
❖ Four classes: beam, pier, railing, retaining wall
❖ Training and testing using annotated data
❖ Inference for classification

Figure 3.10. Tiered spatial modeling framework for deep learning-based point cloud classification of hydraulic structures.

3.2.2. Annotation of Point Cloud Data

We collected 11 point cloud datasets of hydraulic structures from our study sites via field work. We also have 30 point cloud datasets of bridges from previous projects (provided by Dr. Shen-En Chen). In total, there are 41 point cloud datasets that we used for annotation.

Because the original point clouds collected from field work are very dense, we randomly resampled them with 1-cm cut-off distance between points (the minimum between-point distance of the output will be not less than 1cm). In this way, it will reduce the size of point clouds and preserve the shape of structures, which makes it more tangible for annotation which is an exceedingly labor-intensive task. Furthermore, we omitted the RGB color information of the point cloud collected for our study sites to make the data consistent with the bridge point cloud datasets collected from previous projects (without RGB information).

We annotated these point cloud data using a typology with 16 labels (see Figure 3.11). Figure 3.12 and 3.13 show snapshots of annotated point clouds scanned from this project and previous projects (by Dr. Shenen Chen). Then, we aggregated these labels into higher levels to support the two-tiered modeling framework for point cloud classification of hydraulic structures (see Figure 3.14). In other words, two types of point cloud datasets were generated for the two-tiered models: bridge-vegetation-ground datasets and bridge component datasets. In the bridge-vegetation-ground datasets, each point in a point cloud is labelled with bridge, vegetation, or ground classes. For the bridge component datasets, a point in a point cloud is labelled with any of these following classes: retaining wall, pier, beam, and railing.

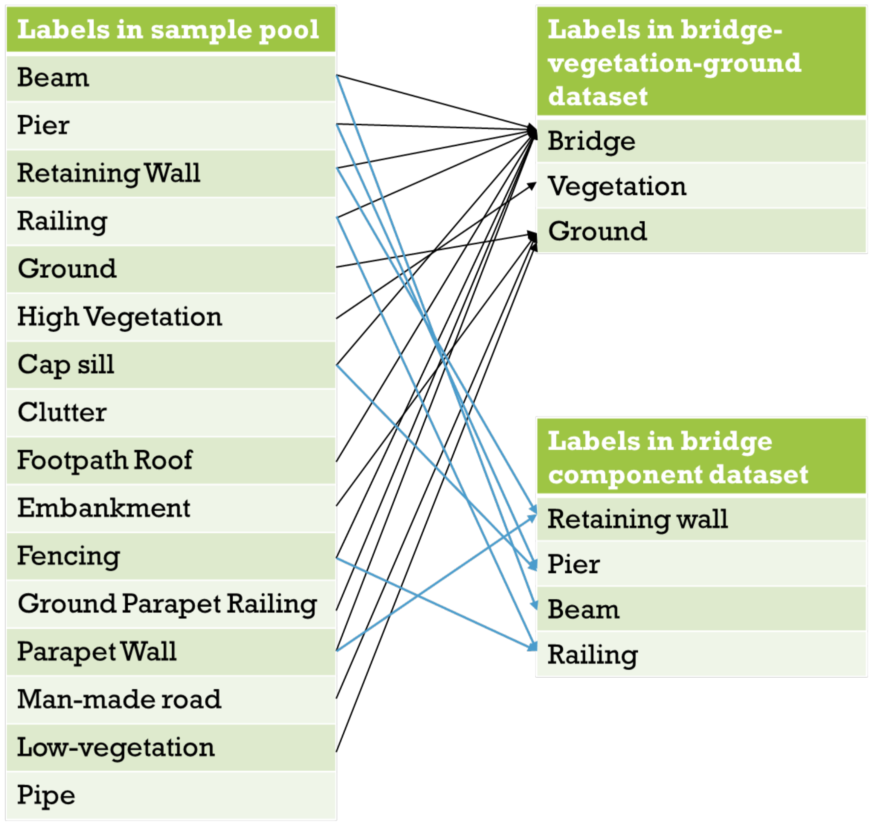


Figure 3.11. Typology of annotation of point cloud data of hydraulic structures.

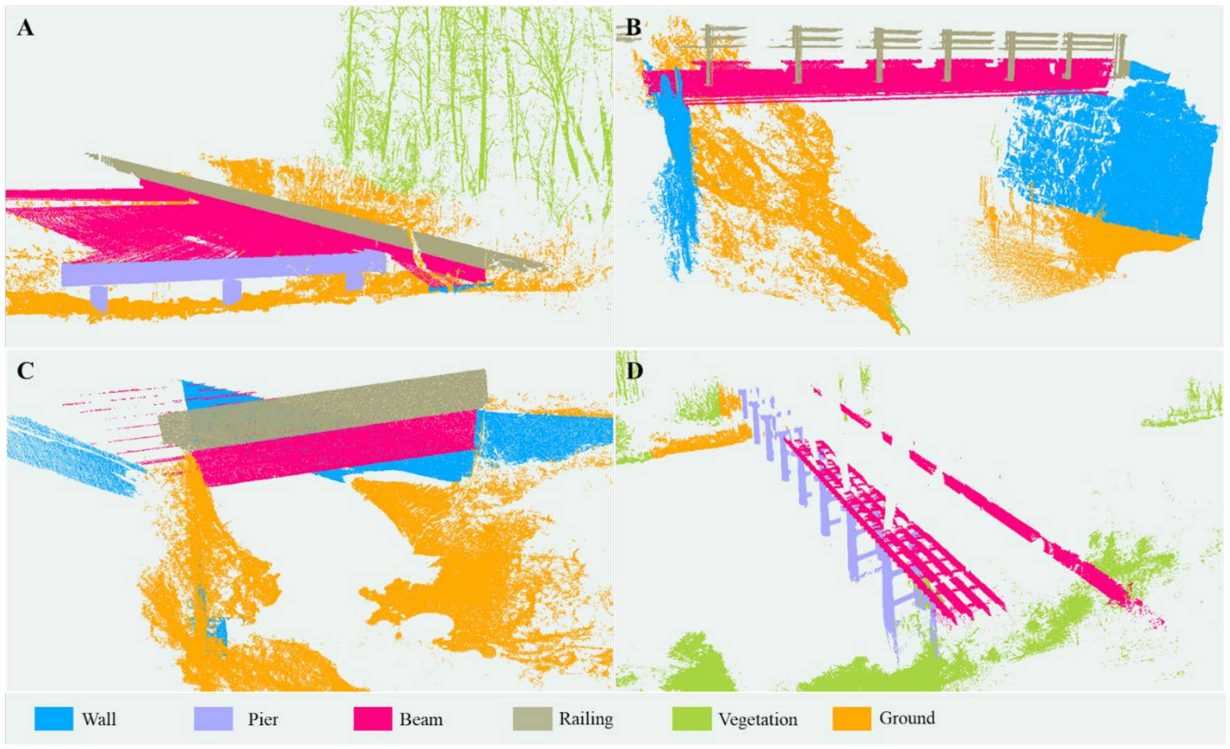


Figure 3.12. Illustration of annotated point clouds collected from the field work in this project (A: Site # 5. B: Site# 2. C: Site # 6. D: Site # 11; See Table 3.1 for more information about these sites).

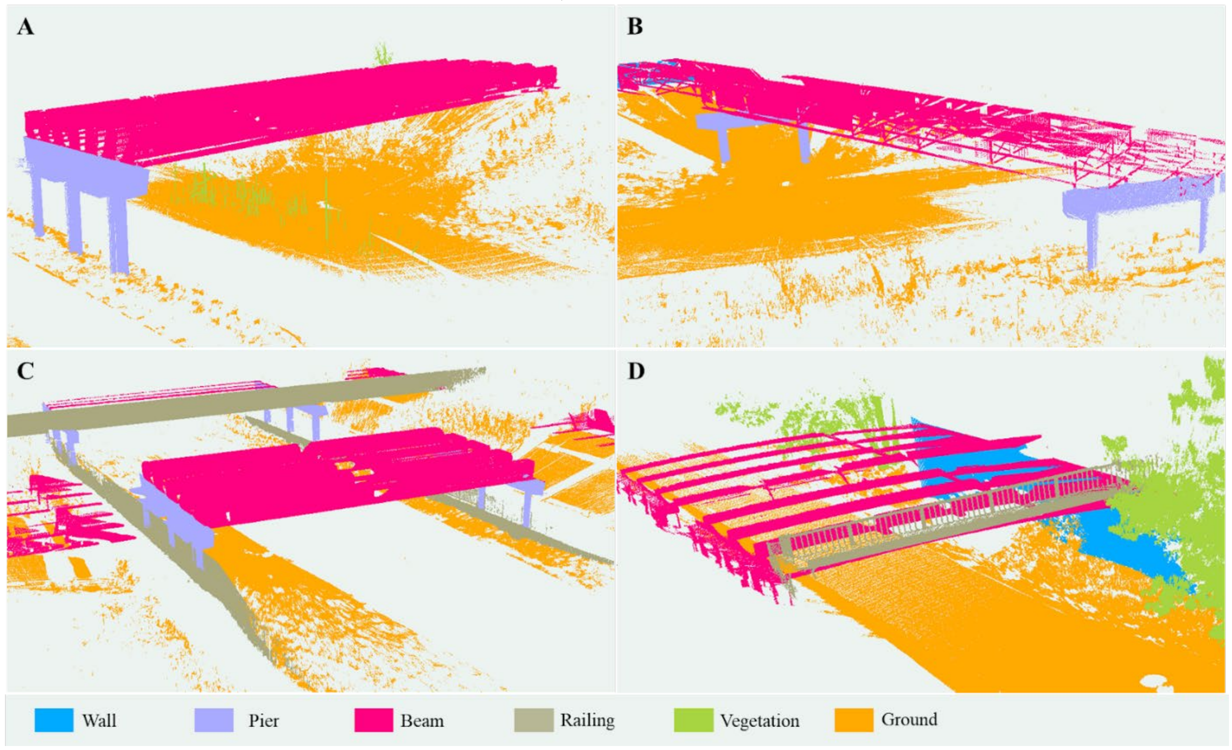


Figure 3.13. Illustration of annotated point clouds scanned from previous projects collected prior to this study (Chen, unpublished data).

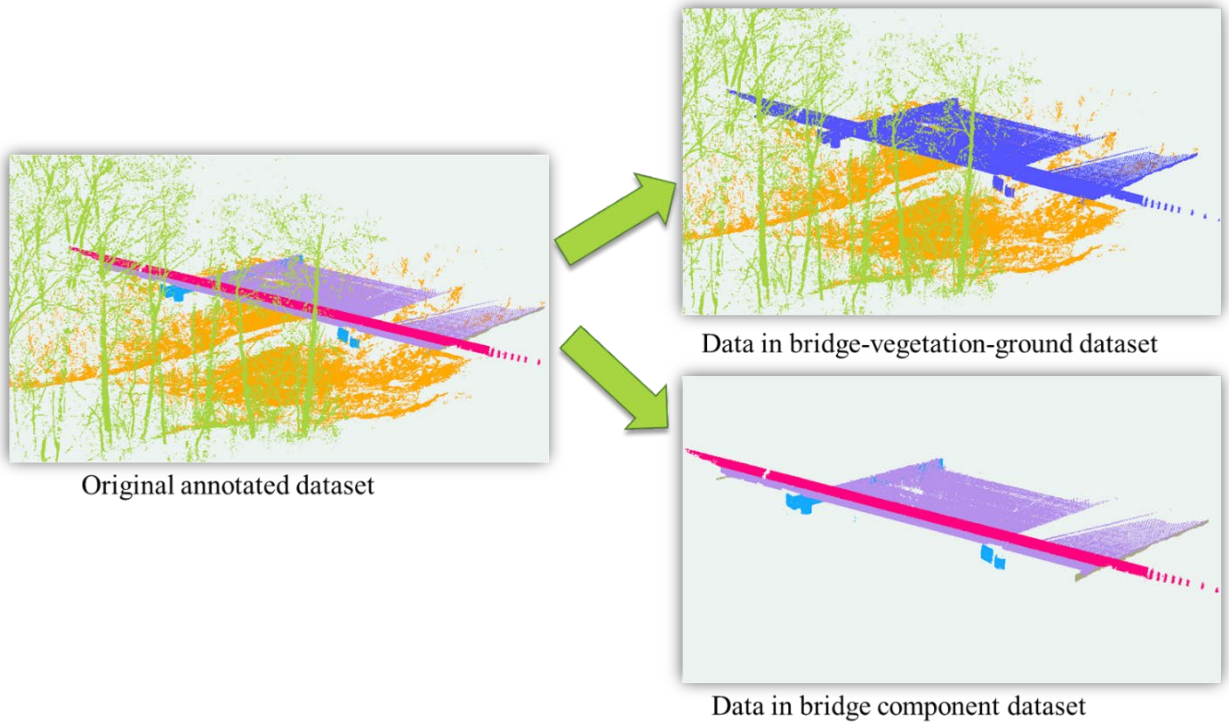

Original annotated dataset
Data in bridge-vegetation-ground dataset
Data in bridge component dataset

Figure 3.14. Aggregation of annotated point cloud data for deep learning-based classification (One annotated scan from Site # 5; see Table 3.1 for site information as needed).

Table 3.17 and 3.18 summarize the information of the annotated point cloud datasets. In total, there are about 207 million points labelled in the 41 datasets. 52.71% of points belong to the bridge class, followed by the ground class (30.36%). Only 16.93% of points are in the class of vegetation. Within the bridge component dataset, beam class ranked first, followed by pier class. Wall and railing classes comprise only 6.54% and 4.54%, respectively.

Table 3.17. Bridge-vegetation-ground dataset.

Statistics/Labels	**Bridge**	**Vegetation**	**Ground**	**Total**
Total	109,354,102	35,122,404	62,993,247	207,469,753
Percentage	52.71%	16.93%	30.36%	100.00%

Table 3.18. Bridge-component dataset.

Statistics/Labels	**Wall**	**Pier**	**Beam**	**Railing**	**Total**
Total	6,949,996	17,673,431	76,778,145	4,818,671	106,220,243
Percentage	6.54%	16.64%	72.28%	4.54%	100.00%

3.2.3. Selection of 3D Deep Learning Algorithms for Point Cloud Classification

As discussed in the literature review section, there are a suite of 3D deep learning methods available for point cloud classification. In this project, we implemented our DeepHyd system using ConvPoint, a state-of-the-art 3D deep learning platform based on continuous convolutions (Boulch 2020). This selection is based on the consideration of the reported classification performance of various 3D deep learning algorithms on a point cloud benchmark dataset, Semantic 3D (http://www.semantic3d.net/view_results.php?chl=1). Table 3.19 compares the classification performance of these 3D deep learning algorithms (top 5 deep neural networks were selected). As shown in this Table, ConvPoint has the best classification performance on the Semantic 3D benchmark. This guided us to use ConvPoint for the deep learning engine of our DeepHyd system for point cloud classification of hydraulic structures. For more detail, please refer to Guo et al. (2020) for 3D deep learning algorithms and Boulch (2020) for the ConvPoint deep learning platform.

Table 3.19. Accuracy performance of the top 5 deep neural networks on the Semantic 3D benchmark (source: http://www.semantic3d.net/view_results.php?chl=1; Table is adapted from the table of semantic-8 results on the website of Large Scale Classification Benchmark; IoU: Intersection over Union; IoU and overall accuracy are model performance metrics defined in Section 3.2.5).

Deep neural networks	**Average IoU**	**Overall Accuracy**	**Reference**
ConvPoints	0.777	0.950	Boulch (2020)
WreathProdNet	0.771	0.946	Wang et al. (2020)
SPGraphs	0.762	0.929	Landrieu and Simonovsky (2018)
FKAConv	0.746	0.941	Boulch et al. (2020)
PointCE	0.710	0.923	Liu et al. (2020)
Pointnet	0.521	0.825	Qi et al. (2017)

3.2.4. Transfer learning for improved 3D Deep Learning for Point Cloud Classification

While ConvPoint claims top rank for the benchmark dataset, the use of ConvPoint for the classification of point clouds from LiDAR is nontrivial. This is because the use of deep learning often requires a large number of labelled data for training and testing. However, we only have 41 labelled point clouds. Therefore, we opted to use the transfer learning technique to cope with this issue. Transfer learning is one of the methods that have been developed to improve deep learning when the amount of available training data is limited. It is suggested in the literature that learned weights in a neural network (pre-trained on similar data) can be used as prior information to train (and thus improve the performance of) deep neural networks (Goodfellow et al. 2016). The use of transfer learning relies on the identification of similar labelled data. For our case, outdoor 3D point cloud benchmark datasets are similar to our point cloud datasets of hydraulic structures as both datasets contain the information of vegetation, ground, and man-made structures.

We studied a set of currently available outdoor 3D point cloud benchmarks (see Table 3.20). Unlike the 2D image benchmarks, there are only a few benchmark datasets for 3D point cloud classification. Moreover, not all the datasets are suitable for our case. For example, there are less than 2 million labeled points in the Oakland data set. Both Sydney Urban Objects and the IQmulus & TerraMobilita Contest use a LiDAR mounted on a car that has much lower point density than a static scanner (see Table 3.24 for links to various 3D point cloud datasets). The same applies to the airborne Vaihingen 3D benchmark. We identified the Semantic 3D dataset (also known as the large-scale point cloud classification benchmark, see Hackel et al. (2017)) as more suitable for use in our case, as it provides a labeled 3D point cloud dataset of the outdoor scene with over 4 billion points collected by static terrestrial LiDAR. It covers a range of eight classes: man-made terrain, natural terrain, high vegetation, low vegetation, building, hard scape, scanning artifacts, cars, and clutter (unclassified). The labels within this benchmark are similar to our data. Transfer learning can thus be used to transfer the knowledge (features) learned by the deep neural network on this benchmark dataset (e.g. natural terrain, vegetation, and buildings) to benefit the deep learning on our task (separating bridge components from the natural materials including vegetation and the ground surface).

The pre-trained deep neural network is retrieved from the repository that the author of Convpoint[1] published, which has eight classes as the output (e.g., man-made terrain, natural terrain, vegetation, car, etc.). However, the model that we utilized for transfer learning embraces only three classes (bridge, vegetation, and ground). Therefore, we omitted the output layer of the pre-trained deep neural network and established an output layer with random initial weights that fits our situation. Furthermore, we gave a relatively low initial learning rate (10^{-4}) to those loaded weights from the pre-trained network to keep the weight change slow so that the pre-trained weights can be better utilized. We then tuned the learning rate of the output layer (see the training of Model 1 in Section 3.2.5.1).

Table 3.20. Labeled 3D point cloud benchmark.

Benchmark name	Data collector	Links
Semantic3D	Static LiDAR	http://semantic3d.net/
Oakland data set	Static LiDAR	http://www.cs.cmu.edu/~vmr/datasets/oakland_3d/cvpr09/doc/
Sydney Urban Objects data set	Mobile LiDAR	http://www.acfr.usyd.edu.au/papers/SydneyUrbanObjectsDataset.shtml
Virtual KITTI 3D Dataset	Mobile LiDAR	https://github.com/VisualComputingInstitute/vkitti3D-dataset
IQmulus & TerraMobilita Contest	Mobile LiDAR	http://data.ign.fr/benchmarks/UrbanAnalysis/
Vaihingen3D airborne benchmark	Airbone LiDAR	http://www2.isprs.org/commissions/comm3/wg4/3d-semantic-labeling.html

[1] Source: https://github.com/aboulch/ConvPoint/tree/master/examples/semantic3d

3.2.5. Training and Validation of 3D Deep Learning for Point Cloud Classification

We have a total of 41 labeled samples in our annotated point cloud data pool, where there are approximately 200 million points in the bridge-vegetation-ground dataset (for Model 1) and 100 million points in the bridge component dataset (for Model 2). However, not all of these point cloud datasets are independent in terms of the hydraulic structures since some of them are different scans of the same hydraulic structure. We thus selected 17 independent point cloud samples, in which we randomly selected six of them as validation samples. We used the rest of the 35 point cloud samples for the training dataset. We applied this split strategy on training and validation samples through all experiments in the following analysis so that their results are comparable. Figure 3.15 shows the network architecture of ConvPoint that we used for 3D point cloud classification.

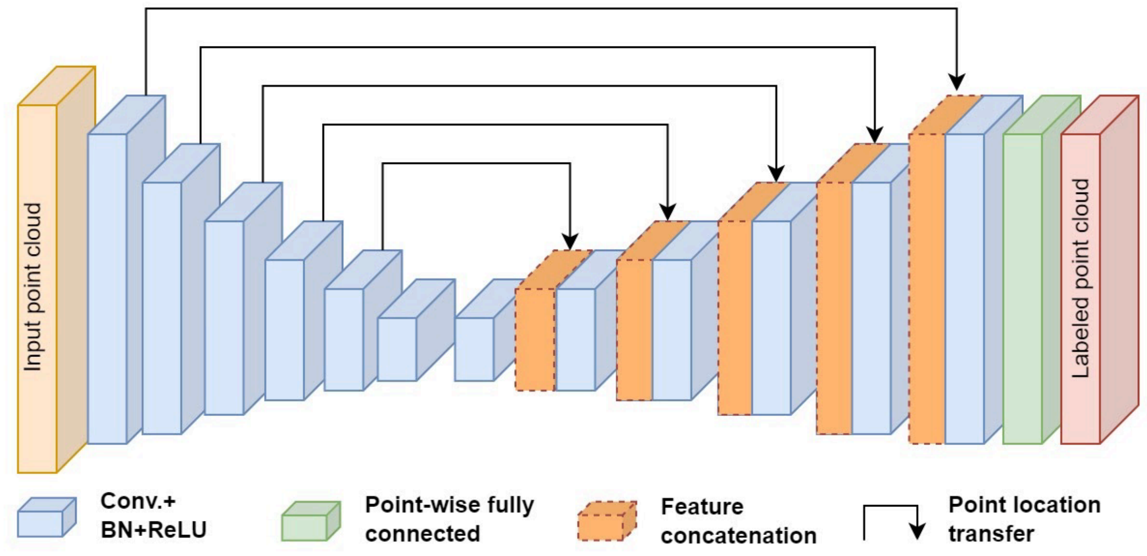


Figure 3.15. Architecture of ConvPoint segmentation networks for 3D deep learning-based point cloud classification (adapted from Boulch, 2020, Figure 5). The structure consists of an encoder reducing the size of the point cloud and a decoder getting back to the initial size of the point cloud. Skip connections represented by black arrows transfer the information from encoder to decoder. (Conv.: Convolutional layer. BN: Batch normalization. ReLU: An activation function).

Tuning of deep learning algorithms is required when we use deep learning for point cloud classification. For deep neural networks, parameters are referred to those weights assigned to connections between neurons. While the learning process is applied to train deep neural networks, a set of other parameters, so-called hyperparameters, are used to guide the learning process to optimize these weights. There are a series of hyperparameters for deep learning algorithms such as learning rate and momentum. These hyperparameters need to be optimized for the training of deep learning algorithms. For our DeepHyd system, we focus on tuning four hyperparameters: learning rate, number of iterations, block size, and number of points per block (see Table 3.21). The first two hyperparameters (learning rate and number iterations) are related to the deep learning algorithm per se. The last two hyperparameters are specific to ConvPoint for 3D point cloud classification. In this section, we first discuss performance metrics and computing resources used for hyperparameter tuning for DeepHyd. Then, we present the tuning of the two types of hyperparameters for Model 1 and Model 2.

Table 3.21. List of hyperparameters for the DeepHyd system.

Hyperparameter	**Explanation**
Learning rate	A hyperparameter to control the rate of learning of the deep neural network in the training process.
Learning rate schedules	A schedule to adjust learning rate in the training process.
# iterations	The number of batches in one epoch, working with # epoch to identify the training steps
# epochs	Number of epochs in the training process, working with # iterations to identify the training steps.
Block size	Data related hyperparameter: The edge length of the square moving window in x-y plane, generating a subset of the point cloud falling into this window, used to feed data to the deep neural network.
# points per block	Data related hyperparameter: Number of points to be sampled from one block.
Hold out	Data related hyperparameter: Number of independent scans used for validation in the training process.
Inference step	The step length of the moving window generating point in the inference process.

Performance Metrics for Model Evaluation: To evaluate the classification performance of our DeepHyd system, performance metrics are needed. We used three performance metrics for the evaluation of deep learning-based models in this project, including Intersection over Union (*IoU*; see Eq. 1), overall accuracy (*OA*; see Eq. 2), and average accuracy (*AA*; see Eq. 3). For more information about these metrics, please refer to Hossin and Sulaiman (2015) and Rahman and Wang (2016).

$$IoU = \frac{TP_i}{TP_i+FN_i+FP_i} \quad (1)$$

$$OA = \frac{TP_{all}}{TP_{all}+FP_{all}+FN_{all}} \quad (2)$$

$$AA = average(\frac{TP}{TP+FN}) \quad (3)$$

where *TP*, *FN*, and *FP* are true positive (hit), false negative (missing; type II error), and false positive (false alarm; type I error); subscript *all* indicates all points and *i* indicates the class number of or one of the labels. Below explains hit, missing, false positive:

- Hit: The label is correctly predicted (also true positive) by the model. For example, points belonging to the bridge were predicted as a bridge by the model.
- Missing: The point belongs to a target class but is predicted as another class (also false negative). For example, points should belong to the bridge but were predicted as ground.
- False Alarm: From the view of model performance for a specific class, the points belonging to other classes were predicted as this specific class (false positive).

For the hyperparameter tuning, we used IoU as the performance metric to evaluate the classification performance of the DeepHyd system.

High Performance Computing Resources for Model Acceleration: The use of 3D deep learning for point cloud classification is highly computationally demanding. To address this computational challenge, we used a high-performance computing cluster with GPUs at University Research Computing at the University of North Carolina at Charlotte to accelerate the parameter tuning of our DeepHyd system. A series of GPUs are available on this high-performance computing cluster, including Nvidia GTX 1080 Ti (#cores: 3,584), Nvidia Tesla K80 (#cores: 4,992), Nvidia Titan RTX (#cores: 4,608), and Nvidia Titan V (#cores: 5,120). We wrapped computing tasks (training of 3D deep learning algorithm) in treatments of the experiments conducted in this project to the GPU cluster and these computing tasks are executed in parallel while each of them leverages GPU devices for further acceleration.

3.2.5.1. Hyperparameter Tuning: Learning Rate and Number of Iterations

Learning rate and number of iterations are two hyperparameters of the deep learning algorithms. The learning rate is a hyperparameter that is essential in developing deep learning-based models. A range of learning rates as suggested in the literature (Goodfellow et al. 2016) is from 10^{-7} to 10^{-2}. The number of iterations is a deep learning parameter that represents the number of batches (subsets of training samples) to be processed within one learning epoch.

We applied the Learning Rate Range Test (LRRT) per Smith et al. (2017) to identify an appropriate range of learning rates to train the 3D deep learning model on our data. A learning rate scheduler was adopted to amplify the learning rate ten times after every 100 iterations with an initial learning rate as $1e^{-7}$, which is the lower bound of the suggested learning rate range by Goodfellow et al. (2016). LRRTs were tested separately to Model 1 and 2. As a result, the ranges of Model 1 and Model 2 are $[10^{-4}, 10^{-6}]$ and $[10^{-2}, 10^{-4}]$. Further parameter tuning was conducted to identify the best learning rates in these ranges.

<u>Training of Model 1: Classification of Bridges, Vegetation, and Ground</u>

We conducted an experiment (experiment 1) with 11 treatments (Table 3.26) to evaluate how learning rate and number of iterations impact model performance. Figure 3.16 shows the impact of the iteration number on the model performance indicating about 10% increase in the averaged IOU as the number of iterations increases from 500 to 1,500. Correspondingly, computing time will increase significantly when the number of iterations increases. Since we have sufficient computing resources, we chose to use 1,500 as the number of iterations in the latter experiments.

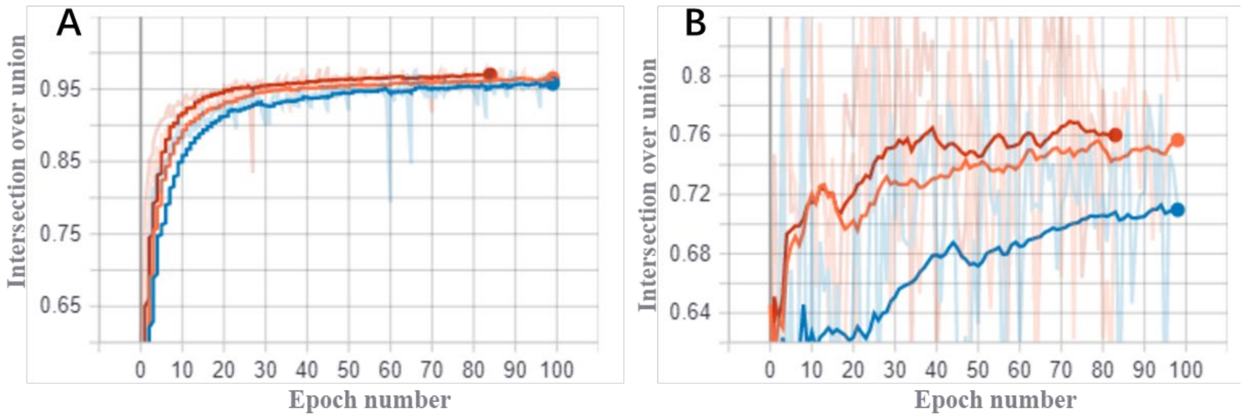


Figure 3.16. Illustration of learning curves for training and validation of deep learning algorithm for Model 1 in response to number of iterations (A: training; B: validation; performance metric: Intersection over Union; Three treatments were used: learning rate =10^{-2} and # iteration = 500 (blue), 1,000 (orange), and 1,500 (red).

As shown in Figure 3.16, the model appeared hard to converge where the divergence of the validation curve was still dramatic (around 10%) even at the end of the training. In this situation, it was hard to capture the best model since the indicated best model was always the last one. Therefore, in experiment 2 we further applied a cosine learning rate scheduler to reduce the learning process from an initial learning rate to 0 following a cosine curve. Experiment 2 has three treatments (treatment 12 to 14 in Table 3.22) within each of which a learning rate scheduler was used for training. Figure 3.17 shows the learning curve of averaged IOU. After utilizing the learning rate scheduler, the divergence became relatively small (around 1%). The learning curves were visually similar, so we chose the one with the highest averaged IoU, whose learning rate is 10^{-5}. Therefore, the learning rate and number of iterations for Model 1 on our current dataset was identified as 10^{-5} and 1,500.

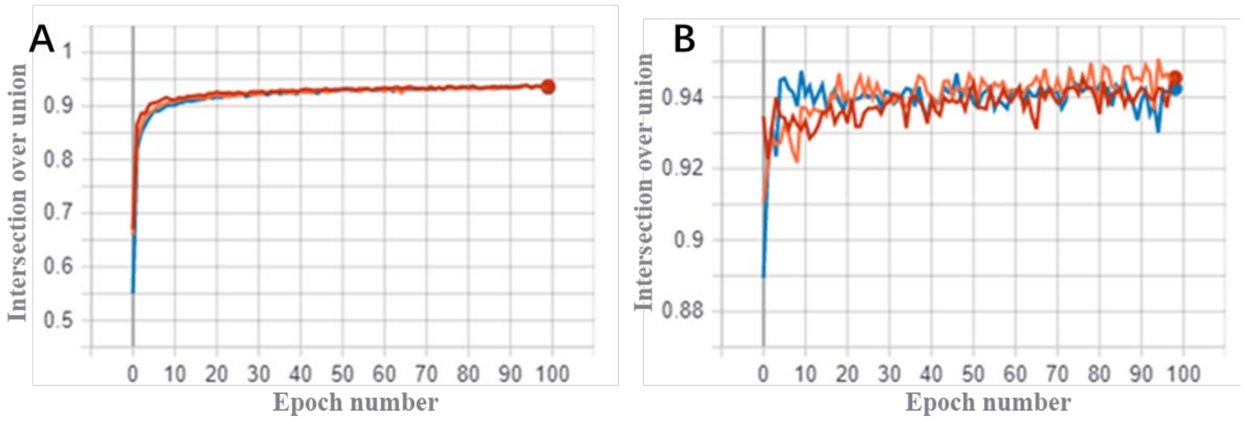


Figure 3.17. Illustration of learning curves for training and validation of deep learning algorithm for Model 1 in response to learning rate (A: training; B: validation; performance metric: Intersection over Union; Three treatments were used: learning rate =10^{-4} (blue), 10^{-5} (orange), and 10^{-6} (red)).

Table 3.22. Summary of experimental design of Model 1 as well as GPU computing performance.

	Treatment #	**Learning Rate**	**# iteration**	**GPU**	**Computing Time (hours)**
Experiment 1	Treatment 1	10^{-2}	500	1080Ti	42
	Treatment 2	10^{-2}	1,000	1080Ti	52
	Treatment 3	10^{-2}	1,500	K80	70
	Treatment 4	10^{-3}	500	1080Ti	42
	Treatment 5	10^{-3}	1,000	K80	53
	Treatment 6	10^{-3}	1,500	K80	70
	Treatment 7	10^{-4}	500	1080Ti	42
	Treatment 8	10^{-4}	1,000	1080Ti	51
	Treatment 9	10^{-4}	1,500	K80	69
	Treatment 10	10^{-5}	1,000	1080Ti	51
	Treatment 11	10^{-5}	1,500	1080Ti	68
Experiment 2	Treatment 12	10^{-4}	1,500	Titan RTX	67
	Treatment 13	10^{-5}	1,500	Titan V	68
	Treatment 14	10^{-6}	1,500	Titan V	68
				Total	813

Training of Model 2: Classification of Bridge Components

The experimental design for training of Model 2 is similar to that of Model 1, includes two experiments. The first experiment (Experiment 1) includes 11 treatments (Treatments 1 to 11 in Table 3.23) that examine how learning rate and number of iterations influence the Model 2 performance. Results observed in the parameter tuning for Model 2 are similar to those for Model 1: an increase in number of iterations would increase the model performance but would slow down the training process. This finding was also confirmed by the author of ConvPoint (https://github.com/aboulch/ConvPoint/issues/32). Thus, we used experiment 2 that has three treatments (Treatments 12 to 14) each using a learning rate scheduler for further parameter tuning. Figure 3.18 illustrates the learning curves of this experiment. It appeared that the best learning rate for Model 2 was 10^{-4} (red), which outperformed other treatments. Therefore, for Model 2, we used 10^{-4} as the learning rate and the number of iterations was also set to 1,500.

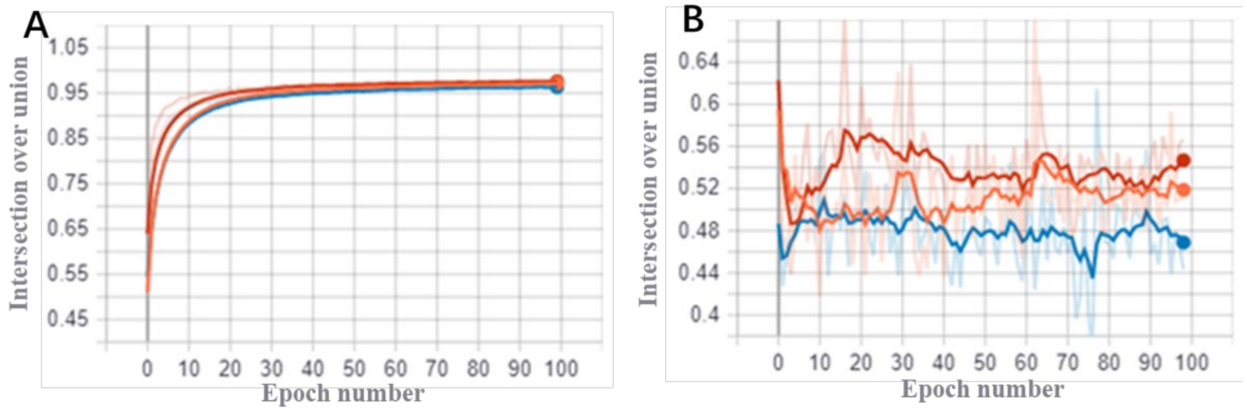


Figure 3.18. Illustration of learning curves for training and validation of deep learning algorithm for Model 2 in response to learning rate (A: training; B: validation; performance metric: Intersection over Union; Three treatments were used: learning rate =10^{-2}(blue), 10^{-3} (orange), and 10^{-4}(red)).

Table 3.23. Summary of experimental design of Model 2 as well as computing time.

	Treatment #	Learning Rate	# iteration	GPU	Computing Time (hours)
Experiment 1	Treatment 1	10^{-2}	500	1080Ti	31
	Treatment 2	10^{-2}	1,000	1080Ti	41
	Treatment 3	10^{-2}	1,500	1080Ti	47
	Treatment 4	10^{-3}	500	1080Ti	29
	Treatment 5	10^{-3}	1,000	1080Ti	41
	Treatment 6	10^{-3}	1,500	1080Ti	47
	Treatment 7	10^{-4}	500	1080Ti	31
	Treatment 8	10^{-4}	1,000	1080Ti	41
	Treatment 9	10^{-4}	1,500	K80	43
	Treatment 10	10^{-5}	1,000	K80	32
	Treatment 11	10^{-5}	1,500	K80	43
Experiment 2	Treatment 12	10^{-2}	1,500	Titan V	41
	Treatment 13	10^{-3}	1,500	Titan V	41
	Treatment 14	10^{-4}	1,500	Titan V	41
				Total	549

3.2.5.2. Hyperparameter Tuning: Block Size and Number of Points per Block

Once we tune the hyperparameters associated with the deep learning algorithm (learning rates and number of iterations here), we need to tune hyperparameters associated with ConvPoint, which is the deep learning core of our DeepHyd system. We focus on tuning two hyperparameters of ConvPoint: block size and number of points per block. To cope with potentially large volumes of points in a point cloud, blocks are used in ConvPoint to extract a subset of points for deep learning. Block size refers to the size of a block that defines the subset of point clouds. Within each block, a specific number of points are randomly sampled that are fed to ConvPoint for inputs of deep learning algorithms—the number of points per block. The range of block size is set to 1 meter to 256 meter, and the number of points per block is in the range of 512-16,384. Such a setting is based on our practical knowledge and the computing capacity of our hardware environment. Note that the value 16,384 as the number of points per block does not work for training Model 1 due to the capacity of our GPUs. We used a grid-based approach to systematically search the hyperparameter space defined by these two hyperparameters. Table 3.24 and 3.25 report hyperparameters used for Model 1 and Model 2. For Model 1, we varied block size within the range of 1 meter to 256 meters while the number of points per block was changed from 512 to 8,192. The combination of varying the two parameters leads to 45 treatments (9 by 5) for the experiment of tuning Model 1. Likewise, for Model 2, there are in total 54 treatments (9 by 6).

Table 3.24. Hyperparameters of 3D deep learning algorithm for Model 1.

Parameter	Value
Block size (meter)	1; 2; 4; 8; 16; 32; 64; 128; 256
#points per block	512; 1,024; 2,048; 4,096; 8,192
#epochs	100
#classes	3
#iterations	1500 (1500 blocks are generated in each epoch)
Holdout number	6 (6 independent samples as validation data)
Scheduler	a cosine learning rate scheduler is applied
Learning rate	10^{-5}

Table 3.25. Hyperparameter configurations of 3D deep learning algorithm for Model 2.

Parameter	Value
Block size (meter)	1; 2; 4; 8; 16; 32; 64; 128; 256
#points per block	512; 1,024; 2,048; 4,096; 8,192; 16,384
#epochs	100
#classes	4
#iterations	1,500 (1,500 blocks are generated in each epoch)
Holdout number	6 (6 independent samples as validation data)
Scheduler	a cosine learning rate scheduler is applied
Learning rate	10^{-4}

Computing Performance

By leveraging the GPU cluster, we completed this hyperparameter tuning exercise in around one month (30 days) of computing time in total. The sequential computing time is around 5,400 hours (~ 225 days). An approximated speedup (a computing performance metric calculated as sequential computing time divided by parallel computing time) is 7.5, which is close to the number of GPUs (8 GPUs) we can request from the GPU cluster. The GPUs used in this experiment are based on availability when computing tasks are realized on the cluster; therefore, different GPUs might be used for different treatments in this experiment.

Classification Performance Evaluation

Once the training of 3D deep learning algorithms in each of the treatments for Model 1 and Model 2 is completed, we used a response surface approach (Box and Draper 2007) to find the optimal hyperparameter set. Figure 3.19 and 3.20 are results of response surfaces of validation model performance for Model 1 and Model 2 in terms of IoU. Inverse Distance Weighting (Burrough et al. 2015) is used for interpolation of model performance into continuous response surface and Root Mean Square Error (RMSE) was used for cross validation of the interpolation. Figure 3.21 and 3.22 show maps of annotated and classified point clouds for Model 1 and Model 2. Table 3.26 and 3.27 depict the corresponding confusion matrices in terms of classification.

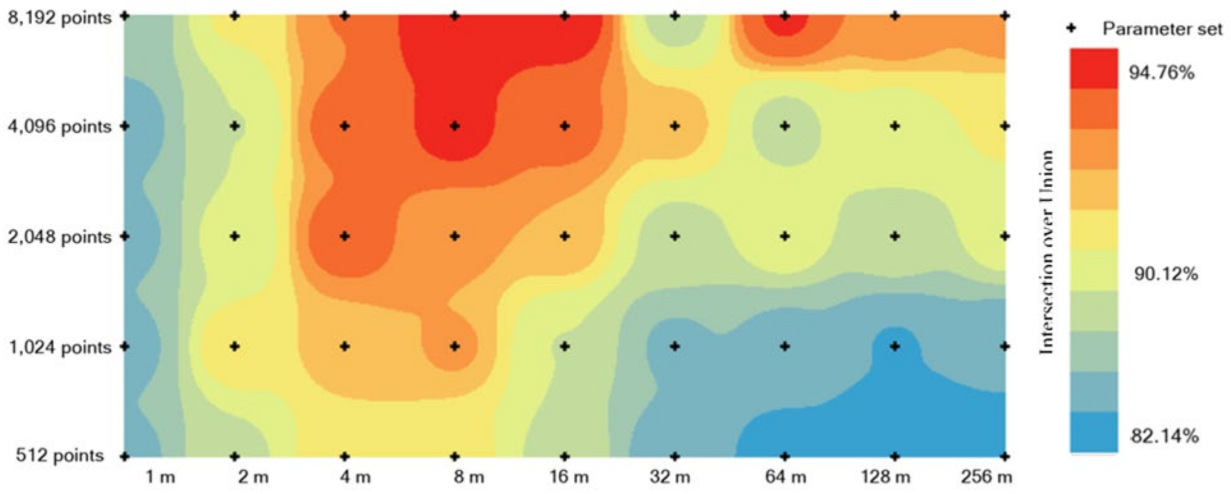


Figure 3.19. Response surface of Intersection over Union for Model 1 between block size and number of points per block (Inverse Distance Weighting is used for estimating the surface with RMSE=0.017).

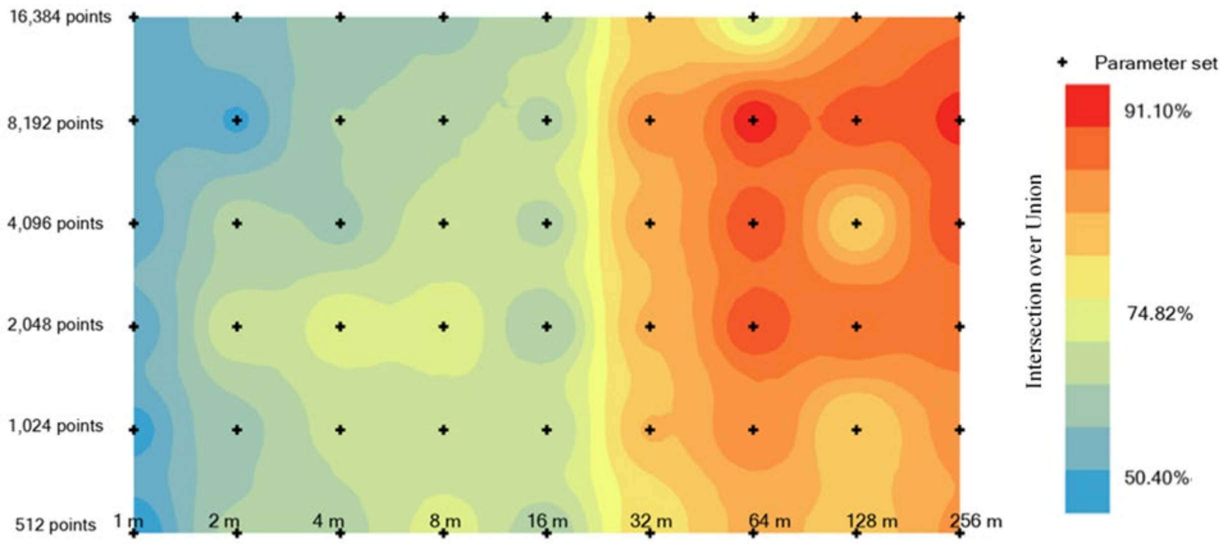


Figure 3.20. Response surface of Intersection over Union for Model 2 between block size and number of points per block (Inverse Distance Weighting is used for estimating the surface with RMSE=0. 058)

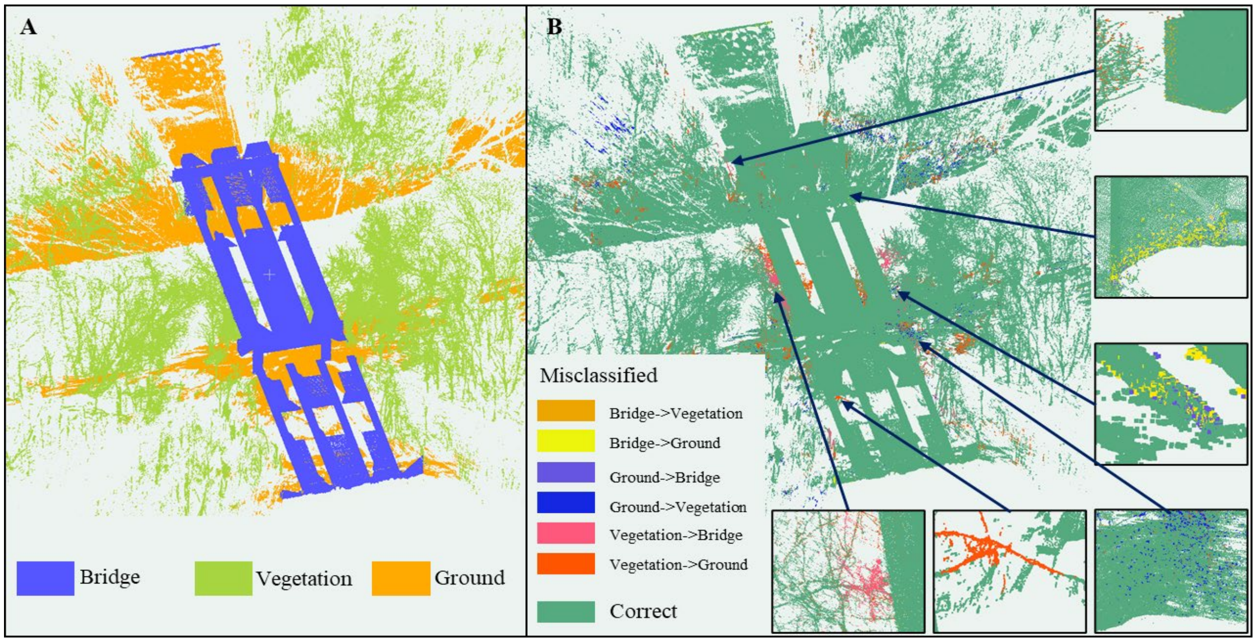


Figure 3.21. Comparison between annotated point cloud and predicted point cloud from Model 1 (One scan from Site # 16 was used. A: map of annotated point clouds. B: map of predicted point cloud showing misclassified points. see Table 3.1. for site information)

Regarding Model 1, we can observe from Figure 3.19 that the highest value of IOU (94.76%) converged to the parameter set in which block size and number of points per block as 8 m and 8,192. Therefore, it suggests that a block size of 8 m is an appropriate value for the deep learning-based classification of vegetation, ground, and bridge for hydraulic structures in our project. There is a rising trend of IOU in response to the increase in the number of points per block.

However, 8,192 is the largest value of the number of points per block that we can set due to the current computing capacity of our hardware environment.

Table 3.26. Confusion matrix of point cloud classification using a point cloud scan from a field site (Site #16: Pharr Mill site was used).

	Predicted bridge	Predicted vegetation	Predicted ground	Total
Actual bridge	33.47%	0.01%	0.09%	33.56%
Actual vegetation	0.38%	25.01%	2.18%	27.57%
Actual ground	0.02%	0.76%	38.09%	38.87%
Total	33.86%	25.78%	40.36%	100.00%

As for Model 2, it appears that there is an improvement in IoU from 16m to 32 m in block size, as shown in Table 3.27. The high values of IOU converged when the block size is 64m and the number of points per block is 8,192. The root-mean-square error is approximately three times that for Model 1. We attribute these results to lack of data, where not all the validation samples have all of the categories (some bridges do not have piers, and some of their railings may not be scanned due to the under-bridge operation). We need to notice that both IOU from Models 1 and 2 are estimated based on our current validation dataset, including six bridges. It may be biased due to the lack of validation data, but it is a trade-off between the number of training samples and the number of validation samples. If we obtain more LiDAR data in the future, the classification performance of our deep learning algorithm can be further improved.

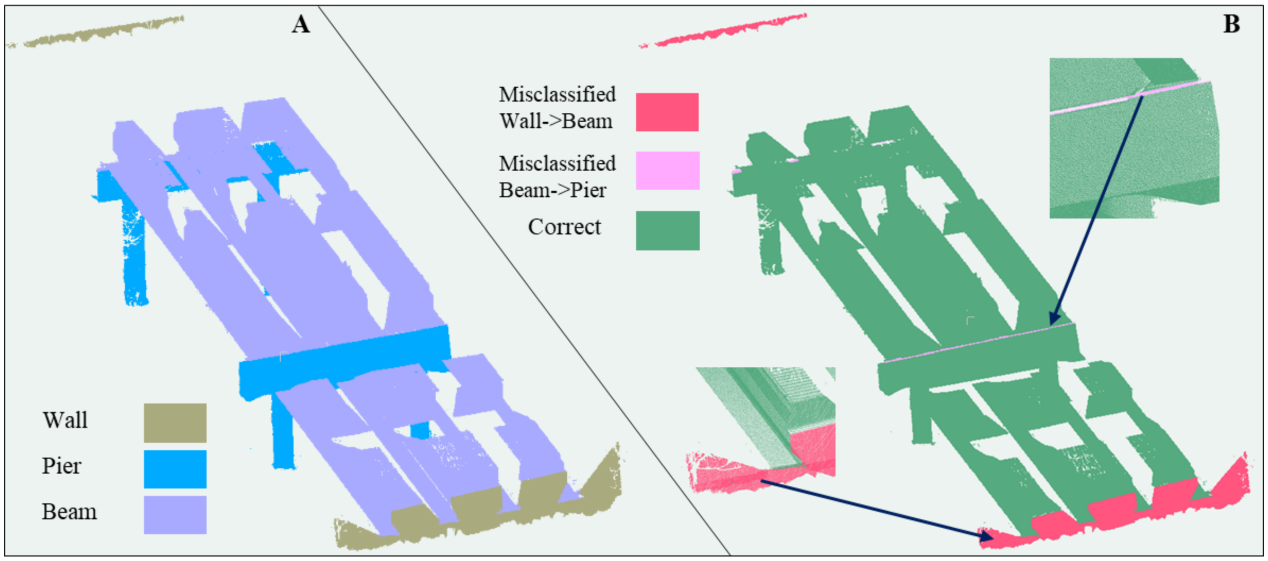


Figure 3.22. Comparison between annotated point cloud and classified point cloud by Model 2 (one scan from site #16 was used. A: map of annotated point clouds. B: map of classified point cloud showing misclassified points; see Table 3.1 for site information).

Table 3.27. Confusion matrix of classification results by Model 2 (a scan from Site #16: Pharr Mill site was used).

	Predicted wall	Predicted pier	Predicted beam	Predicted railing	Total
Actual wall	0.00%	0.00%	3.20%	0.00%	3.20%
Actual pier	0.00%	12.10%	0.00%	0.00%	12.10%
Actual beam	0.00%	0.25%	84.46%	0.00%	84.71%
Actual railing	0.00%	0.00%	0.00%	0.00%	0.00%
Total	0.00%	12.34%	87.66%	0.00%	100.00%

3.2.6. Model Inferencing for the Prediction of Point Cloud Classification

In this section, we present the model inferencing (or prediction) results for the point cloud classification using the DeepHyd system. We conducted model inferencing using six validation datasets.

3.2.6.1. Inferencing Results of Model 1 for the Classification of Bridges, Vegetation, and Ground

Table 3.28 depicts the results of classification performance of Model 1. Table 3.29 reports the confusion matrix of the classification performance for Model 1 (based on six validation datasets). The IoU for the bridge is 98.63%, which is a relatively high compared with training results reported from the benchmark dataset (see ConvPoint performance on the Large-Scale Point Cloud Classification benchmark; see http://www.semantic3d.net/view_results.php?chl=1). Figure 3.23 illustrates the snapshots of the predicted point clouds. The prediction results look reasonable, where most points were well labeled. However, the estimation may also be somewhat biased due to the small size of the validation dataset. We found the model may confuse vegetation and pier when the pier was not in a cylindrical shape, but rather a rack as shown in the mid-left bridge.

Table 3.28. Results of point cloud classification performance of Model 1.

Measure	**Value**
Overall Accuracy	98.55%
Average Accuracy	97.63%
Averaged IoU	94.76%
IoU for bridge	98.63%
IoU for vegetation	89.40%
IoU for ground	96.36%

Table 3.29. Confusion matrix of point cloud classification for Model 1 in terms of percentage.

	Predicted Bridge	**Predicted Vegetation**	**Predicted Ground**	**Total**
Actual Bridge	61.24%	0.30%	0.32%	61.86%
Actual Vegetation	0.02%	7.75%	0.35%	8.12%
Actual Ground	0.21%	0.24%	29.57%	30.02%
Total	61.47%	8.29%	30.24%	100.00%

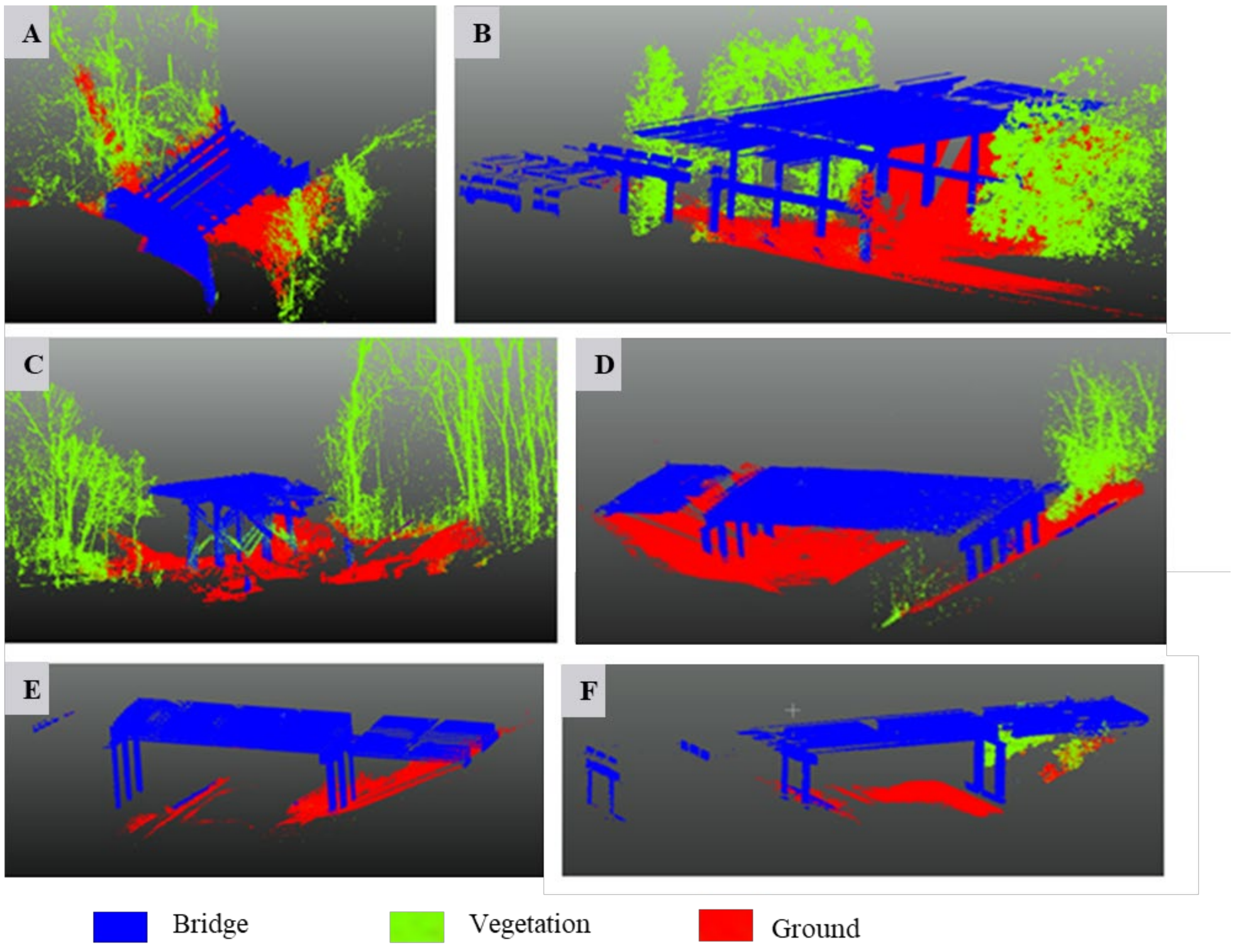


Figure 3.23. Demonstration of prediction results of Model 1 trained on bridge-vegetation-ground dataset. A: Site# 2. D: Site # 5. B, C, E, and F are from previous projects (Chen, unpublished data) (see Table 3.1. for the information of sites from this project).

3.2.6.2. Inferencing Results of Model 2 for the Classification of Bridge Components

Table 3.30 and 3.31 report the classification performance results of Model 2. Figure 3.24 shows the prediction results of point clouds. From Table 3.31, we can see the distribution of categories in the validation dataset for Model 2 is imbalanced, where the largest category, beam, represents 81.14% of the validation data and the smallest category, retaining wall only represents 2.94%. Therefore, those small categories' performance (IoU of Wall 2.94% and IoU of Railing 1.15%) might not be well estimated. Beam and Pier classes appear to have relatively higher proportions in the validation dataset, and they also have a relatively good performance as per their IOUs. We can also tell the long-tail of the data in Figure 3.24, where the pier and beam are represented well in comparison to the railing and wall bridge components. From Figure 3.24, we can tell different components of a bridge, where the pier and beam appear to be well detected via visual inspection. The only bridge with a retaining wall (Panel A) shows that the pier and wall might confuse the model a little bit when there is no pier for the bridge. It seems our validation dataset does not have an actual railing. Those points were predicted as railing because they were similar to the shape of the railing. However, there were some points on the ground (e.g., fencing). We originally had a fencing category in our dataset, but we aggregated it to the railing class due to their similarity. This can help the model to learn the shape of railing-like objects so that it can contribute to detecting railing in different types.

Table 3.30. Results of Model 2 performance for the point cloud classification of bridge components.

Measure	**Value**
Overall Accuracy	98.86%
Average Accuracy	93.11%
Averaged IoU	91.44%
IoU for Retaining Wall	80.08%
IoU for Pier	93.15%
IoU for Beam	99.34%
IoU for Railing	93.18%

Table 3.31. Confusion matrix of classification results for Model 2 (Wall: retaining wall).

	Predicted Wall	**Predicted Pier**	**Predicted Beam**	**Predicted Railing**	**Total**
Actual Wall	2.38%	0.53%	0.03%	0.00%	2.94%
Actual Pier	0.00%	14.50%	0.27%	0.00%	14.77%
Actual Beam	0.03%	0.20%	80.90%	0.00%	81.14%
Actual Railing	0.00%	0.07%	0.00%	1.07%	1.15%
Total	2.42%	15.30%	81.20%	1.08%	100.00%

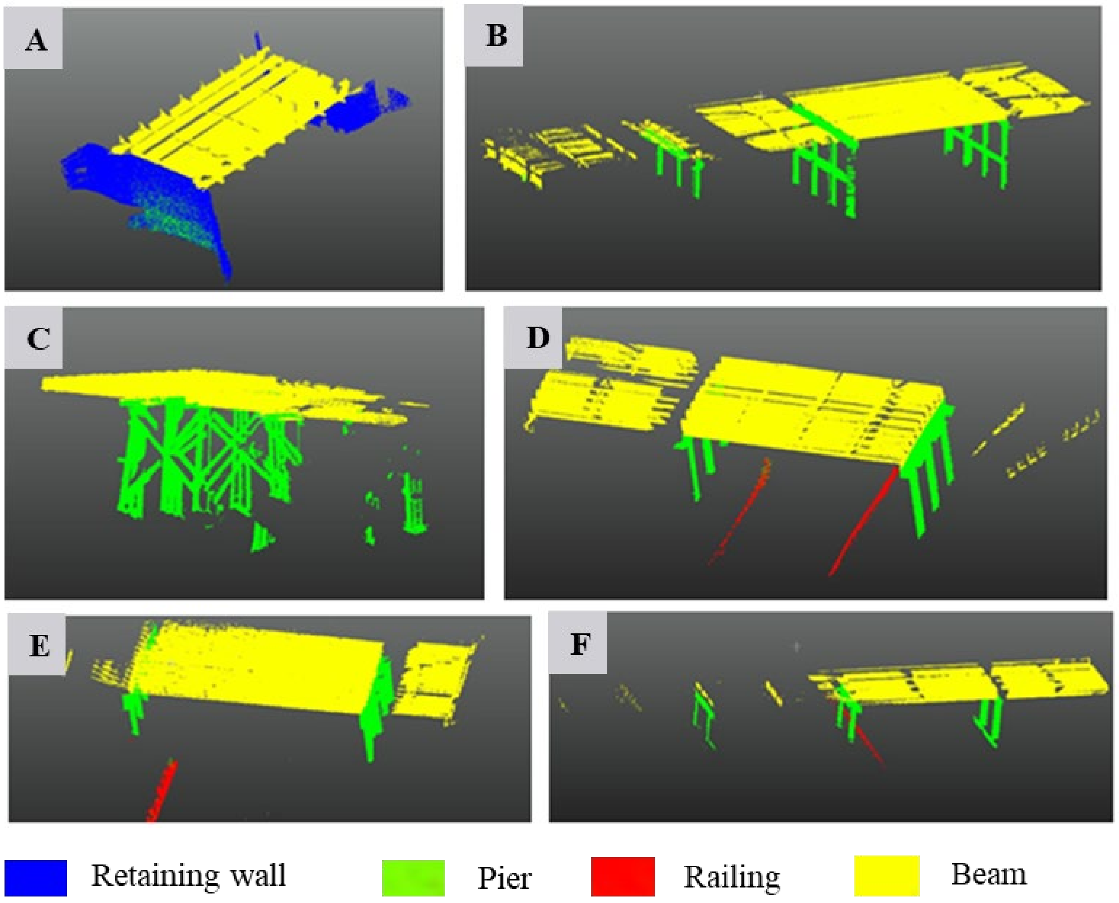


Figure 3.24. Demonstration of prediction results of Model 2 trained on bridge component dataset. A: Site# 2. D: Site # 5. B, C, E, and F are from previous projects (Chen unpublished data) (see Table 3.1. for the information of sites from this project).

3.3. Scientific Workflows for Model Automation

The classification of point cloud data of hydraulic structures is involved with a collection of steps (including pre-processing, model training, model validation, model inference for classification, and post-processing). These steps are often connected together. While some processing steps (e.g., point cloud annotation) may require direct human interaction (a manual approach), a series of steps can be incorporated into scientific workflows for automation. In this project, we used a scientific workflow approach (Taylor et al. 2007) to implement the automated point cloud classification using deep learning techniques. Figure 3.25 shows the scientific workflows that we developed for the automation of point cloud classification using 3D deep learning.

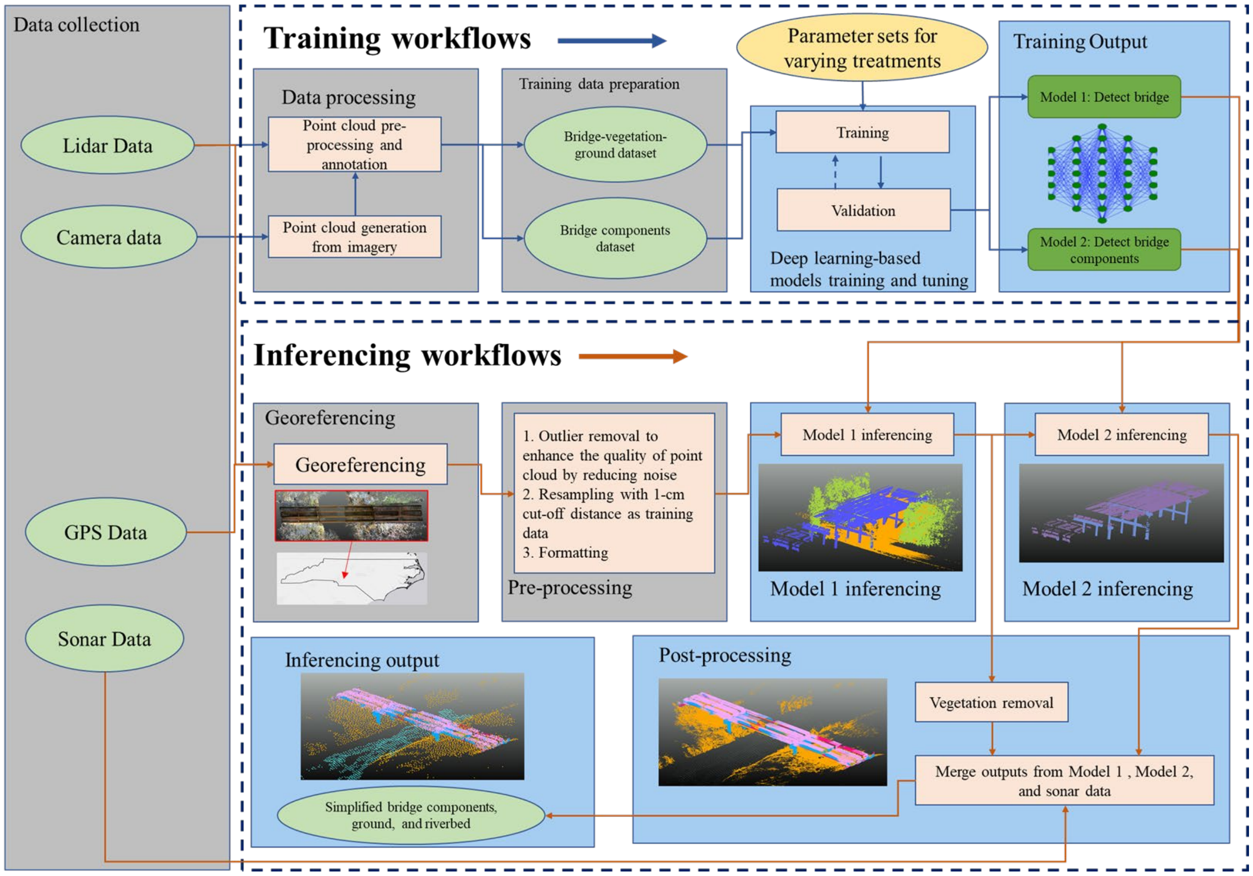

Data collection
Lidar Data
Camera data
GPS Data
Sonar Data
Training workflows
Parameter sets for varying treatments
Training Output
Data processing
Point cloud pre-processing and annotation
Point cloud generation from imagery
Training data preparation
Bridge-vegetation-ground dataset
Bridge components dataset
Training
Validation
Deep learning-based models training and tuning
Model 1: Detect bridge
Model 2: Detect bridge components
Inferencing workflows
Georeferencing
Georeferencing
1. Outlier removal to enhance the quality of point cloud by reducing noise
2. Resampling with 1-cm cut-off distance as training data
3. Formatting
Pre-processing
Model 1 inferencing
Model 1 inferencing
Model 2 inferencing
Model 2 inferencing
Inferencing output
Simplified bridge components, ground, and riverbed
Post-processing
Vegetation removal
Merge outputs from Model 1 , Model 2, and sonar data

Figure 3.25. The framework of the DeepHyd scientific workflows (manual modules are in gray and automated modules are in blue).

The scientific workflows of the DeepHyd system are implemented in Jupyter Notebooks (https://jupyter.org/) with input and output parameters. The notebook for each step is configured using these parameters, set by the user in the web interface. The notebooks provide a record of the code executed during each workflow step, as well as the corresponding output. In the case of an error, the notebook contains all code and parameters necessary to arrive at the error to allow us to fix problems which may arise. The notebooks are executed in Docker containers (a light-weighted cloud computing approach; https://www.docker.com/) defined explicitly for use in the DeepHyd system. Each instance of a scientific workflow is executed in its own container which provides the necessary system environment to execute the processing or classification.

3.3.1. Pre-processing of data

The pre-processing of point clouds in our project includes the following steps: 1) georeferencing of the point cloud, 2) outlier removal to enhance the quality of point clouds, 3) resampling of point clouds to remove redundant points, and 4) format conversion of the data for DeepHyd utilization. Outlier removal, resampling, and formatting are the same preprocessing steps for training and inferencing workflows, while georeferencing is only necessary for inferencing workflow. They are explained as follows:

1) Outlier removal is performed to improve the quality of point cloud by removing scattered points, which are noise, potentially misleading the classifier. This processing task is commonly implemented automatically in the official software for a specific LiDAR instrument, when the raw LiDAR scans were pre-processed (e.g., stitching scans, noise reduction, filtering with a cut-off distance) for further utilization. For example, we used FARO SCENE[2], the manufacture's official software to read and process LiDAR data collected by FARO LiDAR instruments.
2) Resampling with a cut-off distance (e.g., 1cm) was applied to training data as a pre-processing step, omitting redundancies and improving the efficiency of annotation. We also suggest the same resampling be applied to point clouds when inferencing; CloudCompare (or the official processing software from the vendor of a LiDAR instrument) can be a good option for this processing. In this project, we use CloudCompare for the resampling of point clouds.
3) Georeferencing is necessary for the inference workflow as the output may need to have specific horizontal and vertical coordinate systems, which can be conducted manually in the official software for a specific LiDAR instrument. The point cloud for above-water point clouds (e.g., LiDAR data for bridge and ground) needs to be in the same horizontal and vertical coordinate systems as those of under-water point cloud (sonar data) so that they can be fused directly.
4) The format of the point cloud data in DeepHyd includes at most four columns and each row represents one point (see Figure 3.26 for file formats of inputs and outputs), where the first three columns are spatial coordinates, x, y, z, and the last column is a label (necessary for training workflow). Table 3.32 shows the list of labels and their corresponding classes used in this project. Format conversion is to convert the input point cloud data from an ASCII text file (.txt) to a binary numpy file[3] (.npy) that serves as input files of both training and inferencing workflows.

---

[2] https://www.faro.com/en/Products/Software/SCENE-Software

[3] https://numpy.org/doc/stable/index.html

Input file format for model training workflow:

x, y, z, label

Output file format for model training workflow:

x, y, z, label

Input file format for model inference workflow:

x, y, z

Output file format for model inference workflow:

x, y, z, label

Figure 3.26. Input and output file formats for scientific workflows of model training and inference for point cloud classification (input and output files are ASCII text files; file format shows the columns in each line of data file; output file format for model training is the same as that for model inferencing).

Table 3.32. List of labels for classes used by the DeepHyd system.

Model 1 classes	Model 2 classes	Labels
Bridge		1
	Wall	11
	Pier	12
	Beam	13
	Railing	14
Vegetation		2
Ground		3

3.3.2. Post-Processing of Data

Post-processing of point cloud in DeepHyd is necessary for inference workflow, comprising 1) merging prediction results and sonar data, 2) Point cloud simplification for linking with hydraulic models, and 3) Format conversion of point cloud for web-based visualization.

1) Merging prediction results and sonar data: Prediction results of the two models in the inference workflow will be merged so that the input point cloud will be shown as labeled bridge components and ground (vegetation removed by the request of NCDOT). Georeferenced sonar data can be further manually integrated by the users or automatically merged by the workflow (input required for georeferenced sonar data).
2) Point cloud simplification for linking with hydraulic models: Point cloud simplification is used to make point clouds suitable as input into hydraulic models. The classified point cloud data may be in large volumes that are not suitable for being used as inputs for hydraulic models (per communication with NCDOT professionals). We implemented this point cloud simplification in CloudCompare.

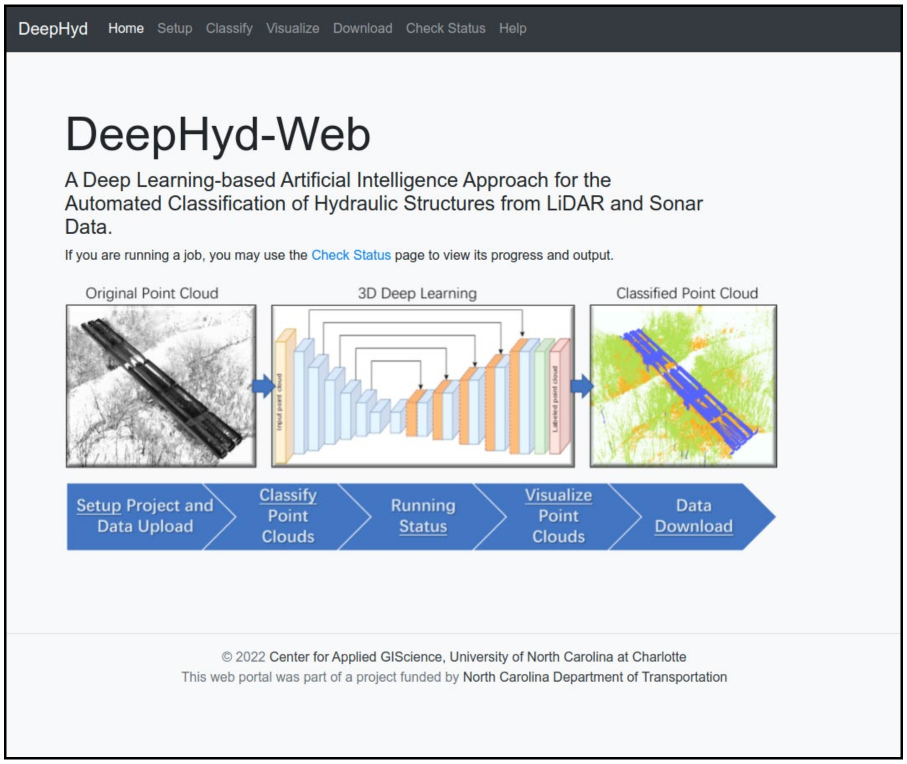
3) Format conversion of point cloud for web-based visualization: We post-process the inference output (classified point cloud) in order to visualize it in the Potree (http://potree.org/) web visualization interface. This step is necessary because the output of ConvPoint is not compatible with the Potree software. Every valid result from ConvPoint is post-processed in this way to allow immediate review of the results.

3.3.3. Use of DeepHyd for point cloud classification.

We implemented two user interfaces for the use of DeepHyd for point cloud classification: command line interface (DeepHyd-CLI) and web interface (DeepHyd-Web). Both interfaces now support the use of CPUs and GPUs for point cloud classification. Users can deploy the command line interface of DeepHyd to their own desktop computers. If GPU device is not available, then users can choose to use CPU environment for point cloud classification (though relatively slower than GPUs). Users just need to specify the parameter configuration file and point cloud data.

The web interface of DeepHyd, i.e., DeepHyd-Web, is a web portal that is capable of receiving datasets, preparing them for classification, running classification, and visualizing and downloading the resulting datasets (see Figure 3.27 for the main web page of the portal). These functions are provided in separate web pages meant to provide a chained scientific flow from one task to the next. The web portal was custom-built for the DeepHyd system in Python so as to match the programming language used for inference and other workflow steps. The web portal is run in its own container separated from the workflow worker processing and job queue database. This intentional separation provides benefits such as resiliency to crashes, performance despite running multiple tasks, scalability across GPU and CPU resources, and asynchronous execution. Once the point cloud is classified, it can be visualized in the web portal (based on Potree; see Figure 3.28).

The computing time for using DeepHyd for point cloud classification is affected by multiple factors such as point cloud size, characteristics of point cloud, or hardware configuration. Table B.1 in the Appendix B reports the computing time information for point clouds in varied sizes, which can provide insights into the estimation of computing time needed for point cloud classification using DeepHyd.



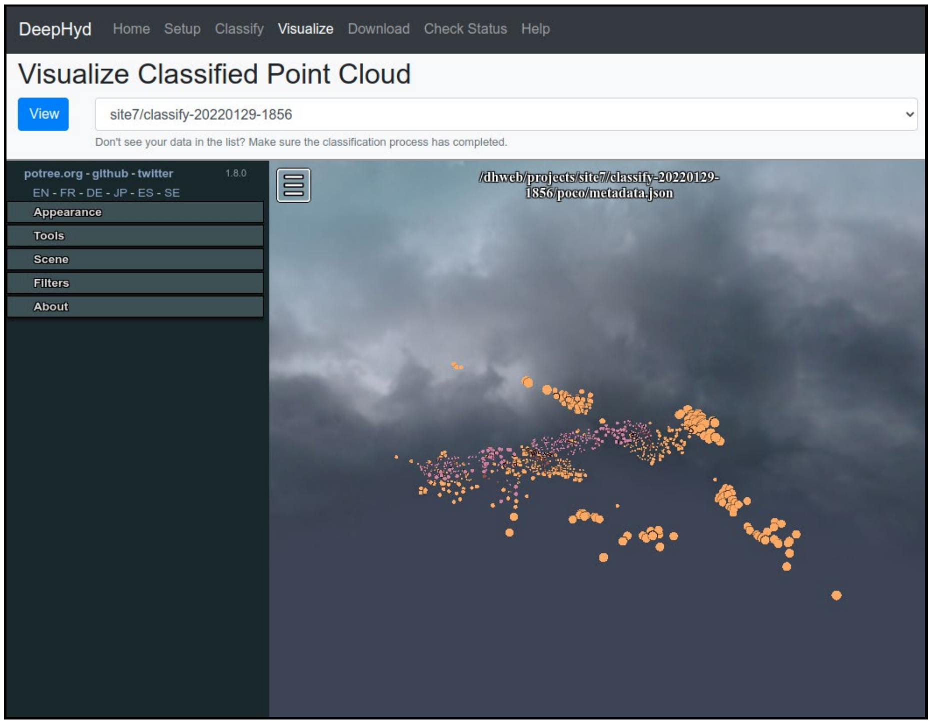
DeepHyd-Web

A Deep Learning-based Artificial Intelligence Approach for the Automated Classification of Hydraulic Structures from LiDAR and Sonar Data.

If you are running a job, you may use the Check Status page to view its progress and output.

© 2022 Center for Applied GIScience, University of North Carolina at Charlotte
This web portal was part of a project funded by North Carolina Department of Transportation

Figure 3.27. Snapshot of the main web page of the Web portal for using DeepHyd for point cloud classification.



Visualize Classified Point Cloud

site7/classify-20220129-1856

Don't see your data in the list? Make sure the classification process has completed.

/dhweb/projects/site7/classify-20220129-1856/poco/metadata.json

Figure 3.28. Snapshot of web-based visualization of classified point clouds in DeepHyd.

3.4. Software Implementation

The DeepHyd system involves the use of a suite of software packages and libraries. Table 3.33 documents the list of main software or libraries used in this project for the development of the DeepHyd system.

Table 3.33. List of key software or libraries used for the implementation of DeepHyd.

Utilization	**Software/Library**	**URL**
Point cloud stitching	Faro Scene	https://www.faro.com/en/Products/Software/SCENE-Software
Point cloud annotation	CloudCompare	https://www.cloudcompare.org/
Deep learning framework	PyTorch	https://pytorch.org/
Deep learning method	ConvPoint	https://github.com/aboulch/ConvPoint
Programming language	Python	https://www.python.org/
Workflow step execution	Jupyter Notebooks	https://jupyter.org/
Database for job queue	Redis	https://redis.io/
Point cloud web viewer	Potree	https://github.com/potree/potree
Containerization	Docker	https://www.docker.com/

4. Findings and Conclusions

Deep learning is a state-of-the-art artificial intelligence technique. The 3D deep learning approach that we used in DeepHyd provides substantial support for the automated classification of point cloud data collected from LiDAR. The use of the 3D deep learning for point cloud classification is inherently associated with a suite of steps, including field data collection, data processing, model training and validation, model inference for point cloud classification. The DeepHyd framework and system that we developed here can efficaciously handle these steps. Specifically, we have the following findings from this project:

1) LiDAR techniques provide a powerful approach for the collection of point cloud data in large volumes and high resolutions to document hydraulic structures and their surrounding environments. In particular, terrestrial LiDAR is well suited to the collection of point cloud data under bridges or inside culverts.

2) The use of sonar techniques supports the collection of underwater topography data particularly for those regions with deep water or turbid water. The relatively small streams examined in this study required the Lowrance sonar system to be mounted on a canoe and ground truthing had to be performed manually with a total and RTK station which was time- and labor-intensive. The resolution of point cloud data from sonar techniques employed in this study is not comparable with terrestrial LiDAR techniques, which suggests that deep learning-driven point cloud classification trained on terrestrial LiDAR data may not be suitable for the classification of sonar data directly. An autonomous water surface drone equipped with sonar and GPS, likely would have generated a denser, more suitable dataset for deep learning-driven point cloud classification. However, suitable training datasets would need to be developed.

3) The fusion of point cloud data from different sources such as terrestrial LiDAR, sonar, GPS, and total station can be achieved once these data are georeferenced based on ground control points. The integrated point clouds from LiDAR and sonar scans allow us to construct 3D models of hydraulic structures and their surrounding environments both above- and under-water.

4) The UAS used in this project has a battery life of less than 25 minutes, so the UAS missions were planned considering the battery life. In addition, FAA regulations and local weather conditions are required to be known before carrying out the flight missions. Lower flight altitude (60m) with a ground speed of less than 5 m/s was set to decrease motion blur and generate high-resolution DSM.

5) The training of deep learning algorithms in our project uses 41 point cloud datasets (11 collected from this project and others from previous projects). However, these point cloud data may not be enough. This is because 1) large amounts of training data are often required by deep learning algorithms, and 2) there are different types of hydraulic

structures. To (further) improve the classification performance would need more training data, while the annotation of point cloud data for training is labor intensive and expensive.

6) Transfer learning provides substantial support for the use of deep learning algorithms for 3D point cloud classification. Deep learning algorithms require significant amounts of training data to achieve high model performance. This poses a challenge for point cloud classification using 3D deep learning as there are not many annotated point cloud data of hydraulic structures. We used transfer learning to address this challenge by using deep neural networks pretrained on benchmark point cloud data similar to the study of hydraulic structures.

7) Our trained deep neural networks can serve as the pre-trained model for the classification of hydraulic structures when more point cloud data are collected and annotated. Our deep learning models can be reused when more training data are available instead of being trained from scratch every time. Further, these trained deep neural networks can be shared and reused for the point cloud classification of hydraulic structures not only in NC but also in other regions.

8) The tuning of deep learning algorithms is fundamental in the use of these algorithms for point cloud classification. The tuning process (including training and validation) is inherently associated with the analysis and optimization of hyperparameters of deep neural networks. Computational experiments need to be well crafted to identify combinations of optimal hyperparameters to ensure high classification performance of deep learning algorithms in our DeepHyd system. In this project, we focus on tuning two types of hyperparameters: one type is related to the deep learning per se (e.g., learning rate and number of iterations), the other type defines how point cloud data are used by deep learning algorithms (e.g., block size and number of points per block). These two types of hyperparameters would need to be analyzed and tuned if we are to achieve high model performance using deep learning algorithms for point cloud classification.

9) In general, the 3D deep learning algorithms used in our DeepHyd system achieves relatively high accuracies for the classification of point cloud data of hydraulic structures. The overall classification accuracy for Model 1 approaches 98.55%. The classification accuracy in terms of IoU for all three classes (bridge, vegetation, and ground) are high (the lowest is 89.40% for vegetation). For Model 2, while the wall component achieves 80.08% of accuracy in terms of IoU, all other three components (pier, beam, and railing) are more than 93%. The classification accuracies for both Model 1 and 2 tend to be high as experimental results indicated. However, we only used 6 annotated point clouds for validation.

10) Distribution of point cloud data has an influence on the training and validation performance of our point cloud classification using deep learning algorithms. In Model 1, for example, vegetation only comprised 16.93% in our training point cloud data, compared to 52.71% for bridge and 30.36% for ground. The classification accuracy of Model 1 in

terms of IoU is 89.40% for vegetation, as compared to 98.63% for bridge and 96.36% for ground. As classification results from Model 2 (the classification of bridge components) indicate, bridge components with low representation in the training datasets (e.g., retaining wall 6.54% and railing 4.54%) tend to have low classification performance. Deep learning algorithms typically require a large number of training data to achieve their superior modeling performance. While a suite of approaches (e.g., transfer learning) have been developed to cope with the data availability issue, our Model 2 for the classification of bridge components is affected by unbalanced data distributions within point cloud data. This renders the information of specific bridge components insufficient for our deep learning algorithm to learn a reasonable representation of all bridge components.

11) The use of deep learning for 3D point cloud classification is extremely computationally intensive. In this project, we introduced GPUs to accelerate the training, validation, and inference of point cloud data using deep learning algorithms. We trained our deep learning algorithms on a high-performance computing cluster with GPUs available. This saved significant amounts of training time for deep learning algorithms. Further, the use of deep learning algorithms for classification does not require a computing cluster. The trained deep learning algorithm can be deployed on workstations or servers with GPU devices available.

5. Recommendations

Based on the findings from this project as reported in Section 4, we have the following recommendations for future directions regarding AI-driven point cloud classification of hydraulic structures:

1) To cope with the low resolution of point cloud data collected from sonar techniques, bathymetric LiDAR techniques (both air-and UAV-borne platforms are available) can be considered in the future. Bathymetric LiDAR is based on the use of green laser light that can penetrate through water. For the survey of small-scale hydraulic structures, UAV-borne bathymetric LiDAR is an alternative solution for constructing topography and hydraulic structures underwater. The point cloud data collected from bathymetric LiDAR will be in high resolution and are suited to the use of the DeepHyd system. However, given the current limitations of bathymetric LiDAR to collect information from many freshwater and marine environments with variable levels of turbidity, the use of sonar on unmanned and crewed boat platforms equipped with high resolution is recommended.
2) It is recommended that more point cloud data of hydraulic structure be collected and annotated to further improve the classification performance of the DeepHyd system driven by deep learning. A wider variety of hydraulic structures should be surveyed. Once more annotated point cloud data are incorporated into the geodatabase of hydraulic structures, deep learning algorithms can be further trained and validated for higher classification performance.
3) It is recommended that camera, gimble and compass be calibrated before carrying out the UAS missions to ensure the generation of high-resolution DSM and DEM. Moreover, a visual observer along with a remote pilot is recommended for conducting flight operations to ensure the safety of people, infrastructure and wildlife if a UAS approach is to be used.
4) It is recommended to routinely collect point cloud data for the sites in this project over time. These point cloud data over time should be analyzed and compared to document potential changes in hydraulic structures and identify critical parts of hydraulic structures in need of maintenance or repair.
5) It is recommended that 3D models of hydraulic structures (e.g., BrIM of bridges) can be used to address the challenge of point cloud annotation. The existing 3D models of hydraulic structures will be particularly helpful for generating annotated point clouds for specific bridge components. This will save considerable time and labor for point cloud data collection using a field survey approach.
6) It is recommended that the balance in the proportion of alternative structure classes in point cloud data needs more attention when more point cloud data are to be collected. A systematic plan and analysis would be needed to cover alternative types of hydraulic structures and balance the information of different bridge component categories.

6. Implementation and Technology Transfer Plan - Developed in conjunction with the StIC

We plan to disseminate the research outcome of this project via 1) delivery of research products to NCDOT, including transfer (including upgrade) of the software, data, and models, 2) publication of research findings, 3) presentation at national and regional conferences, and 4) training workshops. During our project period (2018-2021), we have delivered 26 presentations at national/regional conferences such as NCDOT Summit and Annual Meeting of the American Association of Geographers. We will have one to be presented in Spring 2022. By the end of 2021, we have one peer-reviewed paper published, three manuscripts are in preparation, and one Ph.D. dissertation completed using the work from this project. There are two additional Ph.D. students who will use aspects of this project for their dissertation projects. Please refer to Appendix 2 for detail in these academic presentations and publications.

Technology Transfer: Our DeepHyd system can effectively conduct the classification of point cloud data of hydraulic structures. While the model training of the DeepHyd system is time consuming, the use of the DeepHyd for the point cloud classification of hydraulic structures is relatively straightforward. We developed a web-based dashboard to support the AI-driven point cloud classification using DeepHyd. But this dashboard needs further improvement and tuning to be best deployed and used by NCDOT. We will thus further improve the web-based dashboard for the automated LiDAR point cloud classification driven by deep learning techniques. The dashboard provides a web-based interface for the collective use of DeepHyd system, a deep learning-driven LiDAR point cloud classification software platform. This web-based dashboard will 1) allow users (e.g., NCDOT professionals) to conduct point cloud classification on their own data in a collective manner, 2) post-process the classified point clouds to the formats required by, e.g., specific hydraulic models, and 3) integrate and update more point cloud data to the training database of DeepHyd to further improve the classification performance of LiDAR point cloud of hydraulic structures. We will use cutting-edge containers as service technologies for the rapid deployment of the web-based dashboard and the underlying deep learning-based point cloud classification algorithms. We have submitted this technology transfer application to NCDOT.

7. References

ASCE, 2021. *A comprehensive assessment of America's infrastructure: 2021 report card for America's infrastructure.* https://infrastructurereportcard.org/wp-content/uploads/2020/12/National_IRC_2021-report-2.pdf.

Bisio, R. 2017. Automated modeling--Smart, fast, flexible. *LiDAR,* 7(4), 12-17.

Boulch, A. 2020. ConvPoint: Continuous convolutions for point cloud processing. *Computers & Graphics,* 88, 24-34.

Boulch, A., Puy, G. and Marlet, R., FKAConv: Feature-kernel alignment for point cloud convolution. ed. *Proceedings of the Asian Conference on Computer Vision*, 2020.

Box, G. E. and Draper, N. R., 2007. *Response surfaces, mixtures, and ridge analyses.* John Wiley & Sons.

Burguera, A. and Oliver, G. 2016. High-resolution underwater mapping using side-scan sonar. *PloS one,* 11(1), e0146396.

Burrough, P. A., McDonnell, R. A. and Lloyd, C. D., 2015. *Principles of geographical information systems.* Oxford university press.

Character, L.*, et al.* 2021. Archaeologic Machine Learning for Shipwreck Detection Using Lidar and Sonar. *Remote Sensing,* 13(9), 1759.

Chen, S.-E. 2012. Laser scanning technology for bridge monitoring. *Laser Scanner Technology*, 71.

Feroz, S. and Abu Dabous, S. 2021. UAV-Based Remote Sensing Applications for Bridge Condition Assessment. *Remote Sensing,* 13(9), 1809.

Fuchs, P.*, et al.* 2004. Applications of laser-based instrumentation for highway bridges. *Journal of Bridge Engineering,* 9(6), 541-549.

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

Guo, Y.*, et al.* 2016. Deep learning for visual understanding: A review. *Neurocomputing,* 187, 27-48.

Guo, Y.*, et al.* 2020. Deep learning for 3d point clouds: A survey. *IEEE transactions on pattern analysis and machine intelligence.*

Hackel, T.*, et al.* 2017. Semantic3d. net: A new large-scale point cloud classification benchmark. *arXiv preprint arXiv:1704.03847.*

Hossin, M. and Sulaiman, M. N. 2015. A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process,* 5(2), 1.

Ibrahim, M.*, et al.* 2021. Exploiting structured cnns for semantic segmentation of unstructured point clouds from lidar sensor. *Remote Sensing,* 13(18), 3621.

Kinzel, P. J. and Legleiter, C. J. 2019. sUAS-based remote sensing of river discharge using thermal particle image velocimetry and bathymetric lidar. *Remote Sensing,* 11(19), 2317.

Landrieu, L. and Simonovsky, M., Large-scale point cloud semantic segmentation with superpoint graphs. ed. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, 4558-4567.

LeCun, Y., Bengio, Y. and Hinton, G. 2015. Deep learning. *nature,* 521(7553), 436-444.

Liu, H.*, et al.* 2020. Semantic context encoding for accurate 3D point cloud segmentation. *IEEE Transactions on Multimedia.*

Mandlburger, G., *et al.* 2015. Topo-bathymetric LiDAR for monitoring river morphodynamics and instream habitats—A case study at the Pielach River. *Remote Sensing,* 7(5), 6160-6195.

Mandlburger, G., *et al.* 2016. Evaluation of a novel UAV-borne topo-bathymetric laser profiler. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences,* 41, 933.

NCDOT, 2022. *https://www.ncdot.gov/initiatives-policies/Transportation/bridges/Pages/default.aspx* [online].

Pieraccini, M., *et al.* 2006. Integration of radar interferometry and laser scanning for remote monitoring of an urban site built on a sliding slope. *IEEE Transactions on geoscience and remote sensing,* 44(9), 2335-2342.

Prendergast, L. J. and Gavin, K. 2014. A review of bridge scour monitoring techniques. *Journal of Rock Mechanics and Geotechnical Engineering,* 6(2), 138-149.

Qi, C. R., *et al.*, Pointnet: Deep learning on point sets for 3d classification and segmentation. ed. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, 652-660.

Rahman, M. A. and Wang, Y., Optimizing intersection-over-union in deep neural networks for image segmentation. ed. *International symposium on visual computing*, 2016, 234-244.

Shi, S., *et al.*, Benchmarking state-of-the-art deep learning software tools. ed. *2016 7th International Conference on Cloud Computing and Big Data (CCBD)*, 2016, 99-104.

Shrestha, A. and Mahmood, A. 2019. Review of deep learning algorithms and architectures. *IEEE Access,* 7, 53040-53065.

Smith, S. L., *et al.* 2017. Don't decay the learning rate, increase the batch size. *arXiv preprint arXiv:1711.00489*.

Tang, W. and Feng, W. 2017. Parallel map projection of vector-based big spatial data: Coupling cloud computing with graphics processing units. *Computers, Environment and Urban Systems,* 61, 187-197.

Taylor, I. J., *et al.*, 2007. *Workflows for e-Science: scientific workflows for grids.* Springer.

Wang, R., Albooyeh, M. and Ravanbakhsh, S., 2020. Equivariant maps for hierarchical structures. *34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.* Vancouver, Canada.

Watson, C., *et al.* 2013. LiDAR scan for blasting impact evaluation on a culvert structure. *Journal of performance of constructed facilities,* 27(4), 460-467.

Appendix

Appendix A.

Table A1. Weather conditions for UAS operation on Site #5 (see Table 3.1 for site information)

Date	Temperature	Dew Point	Visibility	Wind Speed	Wind Gust	Wind Direction	Weather
Feb. 01st, 2020	52°F	38°F	10 miles	6 mph	0 mph	220°	Mostly Cloudy

Table A2. Planning parameters for the flight plan for Site #5.

Planning Parameters	Flight Plan
Route Distance	1,404 m
Side Overlap ratio	80 %
Front Overlap ratio	90 %
Course angle	95°
Mission Altitude	61 m
Speed	3 m/s
Flight Time	10 min 28 s
Photos Count	241
Area Covered	0.01km^2

Table A3. Weather details of UAS operation on Site #16 (see Table 3.1 for site information).

Date	Temperature	Dew Point	Visibility	Wind Speed	Wind Gust	Wind Direction	Weather
January 26th, 2020	54°F	28°F	10 miles	9 mph	0 mph	240°	Sunny

Table A4. Planning parameters for the flight plan on Site #16.

Planning Parameters	Flight Plan-I	Flight Plan-II
Route Distance	238 m	326 m
Side Overlap ratio	90 %	90 %
Front Overlap ratio	90 %	90 %
Course angle	159°	251°
Mission Altitude	60 m	60 m
Speed	2.7 m/s	2.7 m/s
Flight Time	2 min 28 s	3 min 10 s
Photos Count	45	64
Area Covered	1200.60 m^2	1778.42 m^2

Appendix B.

Table B.1. Computing time for using DeepHyd for point cloud classification on sample datasets (bounding box size: the size or area of the bounding box of a point cloud on the XY dimension, calculated as (xmax-xmin)*(ymax-ymin)).

Point Cloud File Name	Data volume	# points	Bounding box size (x-y plane)	GPU time (seconds)	CPU time (seconds)
Scan_042	1.2GB	36,747,384	284,622	406	498
Scan_041	930MB	29,317,676	60,856	95	122
Scan_045	740 MB	22,636,227	125,460	143	187
Scan_044	569MB	17,259,928	261,448	123	201
Scan_015	198MB	6,214,410	19,818	13	24
Small_sample1	2.1 MB	100,000	208,089	17	26
Small_sample2	1.1MB	52,019	756	3	3

Hardware configuration:

CPU: 10-core Intel Xeon CPU E5-2687W v3 with clock rate of 3.10 GHz; Memory: 251 GB
GPU: NVIDIA Tesla K40c with 2880 cores, 12GB memory, and base clock speed of 745 MHz

Appendix C.

Academic activities:

Publications:

1. Chavan, V.S., 2021, Finite Element Modeling of A Pier-on-bank Bridge Scour, Ph.D Dissertation, Department of Civil and Environmental Engineering, University of North Carolina at Charlotte.
2. Chavan, V.S., Chen, S.E., Shanmugam, N.S., Tang, W., Diemer, J., Allan, C., Braxtan, N., Shukla, T., Chen, T. and Slocum, Z., 2022. An Analysis of Local and Combined (Global) Scours on Piers-on-Bank Bridges. *CivilEng*, *3*(1), pp.1-20.

Presentations:

1. Chen, T., Tang, W., Chen, S., Allan, C., Diemer, J., Shukla, T., Slocum, Z., Shanmugam, N., Chavan, V., and Lauffer, M.S. 2022. Empirical knowledge related to deep learning-based 3D point cloud classification in 3D GIS. American Association of Geographers Annual Meeting 2022, February 25 to March 1, 2022.
2. Chavan, V., Chen, S., Tang, W., Allen, C., Diemer, J., Shanmugam, N., Chen, T., Shukla, T., Slocum, Z., Lauffer, M. 2021. Effect of Scour on Stability of Drilled Pier (Pile) Foundation Using Three-Dimensional Finite Element Analysis Method. NCDOT Research & Innovation Summit 2.0. Virtual. October 5-6, 2021.
3. Chen, T., Tang, W., Chen, S., Allan, C., Diemer, J., Shukla, T., Shanmugam, N., Chavan, V., Lauffer, M.S. 2021. Automated semantic segmentation of point cloud data driven by deep learning and 3D GIS. NCDOT Research & Innovation Summit 2.0. Virtual. October 5-6, 2021.
4. Slocum, Z., Tang, W., Allan, C., Diemer, J., Chen, T., Chavan, V., Shanmugam, N., Shukla, T., Lauffer, M.S. 2021. A Web-based approach for the application of deep learning to the automated point cloud classification of hydraulic structures. NCDOT Research & Innovation Summit 2.0. Virtual. October 5-6, 2021.
5. Shukla, T., Tang, W., Allan, C., Chen, S., Diemer, J., Chen, T., Slocum, Z., Shanmugam, N., Chavan, V., and Lauffer, M.S., 2021. Scour Monitoring of Hydraulic Structures using Unmanned Aerial System and Sonar. NCDOT Summit 2021, October 6th, 2021
6. Tang, W., Chen, S., Diemer, J., Allan, C., and Lauffer, M., 2021, Deep learning-based detection of 3D hydraulic structures from point cloud data: Acceleration via a cyberinfrastructure-enabled high-performance computing approach, INES (Infrastructure and Environmental Systems) Seminar: Advanced Infrastructure Systems, the University of North Carolina at Charlotte, September 7, 2021.
7. Shukla, T., Tang, W., Allan, C., Diemer, J., Chen, S., Chen, T., Slocum, Z., and Lauffer, M.S., 2021. The fusion of Unmanned Aerial System and sonar data for the assessment of scours of hydraulic structures, STRATUS, May 17th-19th, 2021.
8. Chen, T., Tang, W., Chen, S., Allan, C., Diemer, J., Shukla, T., Shanmugam, N., Chavan, V., Lauffer, M.S. 2021. Deep learning-based 3D semantic segmentation: a practical case in hydraulic structures, Annual Meeting of the American Association of Geographers, Virtual Conference., April 7th, 2021.

9. Tang, W., Chen, S., Diemer, J., Allan, C., and Lauffer, M.S., 2021, Deep learning-based detection of 3D hydraulic structures from point cloud data: Acceleration via a cyberinfrastructure-enabled high-performance computing approach, Research Seminar of College of Computing and Informatics, UNC Charlotte, NC., January 29th, 2021.
10. Chavan, V., Chen, S., Allan, C., Diemer, J., Tang, W., Shanmugam, N., Slocum, Z., Shukla, T., Chen, T., and Lauffer, M.S., 2020, Bridge Pier Scour-Induced Loading Effect Using Finite Element Modeling, NCDOT Research & Innovation Virtual Summit. Virtual. October 13th-14th, 2020.
11. Chen, T., Tang, W., Chen, S., Allan, C., Diemer, J., Shukla, T., Slocum, Z., Shanmugam, N., Chavan, V., and Lauffer, M.S., 2020. 3D Point Cloud Semantic Segmentation: A practical case in bridge detection. USGIF GEOINTegration Summit. Virtual. September 28th, 2020.
12. Chen, T., Tang, W., Chen, S., Chavan, V., Shanmugam, N., Slocum, Z., Shukla, T., Allan, C., Diemer, J., and Lauffer, M.S., 2020. Deep Learning-based Semantic Segmentation of 3D Point Clouds: A Case Study for Hydraulic Structures and its components. NCDOT Research & Innovation Virtual Summit. Virtual. October 13th-14th, 2020.
13. Shanmugam, N., Chen, S., Allan, C., Diemer, J., Tang, W., Chavan,V., Slocum, Z., Shukla, T., Chen, T., and Lauffer, M.S., 2020, Quantification of Bridge Pier Scour from 3D Point Cloud Data Using Spatial Interpolation, NCDOT Research & Innovation Virtual Summit, October 13th-14th, 2020.
14. Shukla, T., Allan, C., Tang, W., Chen, S., Diemer, J., Chen, T., Slocum, Z., Shanmugam, N., Chavan, V., and Lauffer, M.S., 2020, Bathymetric Surveys of Hydraulic Structures using unmanned aerial systems and sonar, NCDOT Research & Innovation Virtual Summit, October 13th-14th, 2020.
15. Tang, W., Chen, S., Diemer, J., Allan, C., and Lauffer, M.S., 2020, The automation and acceleration of deep learning-based detection of 3D hydraulic structures from point cloud data: A cyberinfrastructure-enabled approach, NCDOT Research & Innovation Virtual Summit, October 13th-14th, 2020.
16. Tang, Wenwu, Chen, Shen-en, Diemer, John, Allan, Craig, Lauffer, Matthew S., 2019, Deep learning-based classification of point cloud data for the automated detection of hydraulic structures driven by cyberinfrastructure-enabled high-performance computing., invited talk for the Colloquium of the Department of Geography at Virginia Tech on October 25th, 2019.
17. Chen, T., (2019). A case study to evaluate clusterODM toolkit for unmanned aerial vehicle mapping. GIS Day Lighting Talk, Charlotte, NC., November 13th, 2019.
18. Shukla, T. (2019). Spatial analysis of sonar data for scour assessment. GIS Day Lighting Talk, Charlotte, NC., November 13th, 2019.
19. Shanmugam, N. & Chavan, V. (2019). LiDAR for bridge monitoring applications. GIS Day Lighting Talk, Charlotte, NC., November 13th, 2019.
20. Slocum, Z. (2019). UAS based data collection for GIS workflows. GIS Day Lighting Talk, Charlotte, NC., November 13th, 2019.
21. Shanmugam, N., Chen, S., Tang, W., Allan, C., Diemer, J., Chavan, V., Shukla, T., Chen, T., and Lauffer, M.S., 2019, Use of terrestrial LiDAR system for bridge condition assessment, International Highway Engineering Exchange Program, September 9th, 2019, Asheville, NC.

22. Shanmugam, N., 2019, Introduction to LiDAR and Basic Point Cloud Analysis, GIAN Course on Forensic Engineering and Failure Analysis, National Institute of Technology Tiruchirappalli, Tiruchirappalli, Tamil Nadu, India, June 10th-21st, 2019.
23. Tang, W., Chen, S., Diemer, J., Allan, C., and Lauffer, M.S., 2019, DeepHyd: A deep learning-based artificial intelligence approach for the automated classification of hydraulic structures from LiDAR and Sonar data, NCDOT Research & Innovation Summit, Greensboro, NC., May 7th, 2019.
24. Chen, T., Tang, W., Chen, S., Allan, C., Diemer, J., Shukla, T., Shanmugam, N., Chavan, V., Lauffer, M.S., 2019, Massive 3D scene reconstruction of hydraulic structures accelerated using high-performance computing, NCDOT Research & Innovation Summit, Greensboro, NC., May 7th, 2019.
25. Shukla, T., Chen, T., Tang, W., Chen, S., Allan, C., Diemer, J., Shanmugam, N., Chavan, V., and Lauffer, M.S., 2019, Bridge inspection based on unmanned aerial systems and photogrammetry techniques, NCDOT Research & Innovation Summit, Greensboro, NC., May 7th, 2019.
26. Chen, T., and Tang, W. (2019). When geospatial big data meets high performance computing in 3D GIS, Annual Meeting of the American Association of Geographers, Washington D.C., April 3-7th, 2019.
27. Shukla, T., Chen, T., and Tang, W., 2019, Topographical surveying using UAS photogrammetry, Charlotte-Metropolitan GIS User Group Meeting, Charlotte, NC, USA., February 15th, 2019.