SSI-DT46205029

Doctoral Dissertation

A STUDY ON STACKED OBJECT RECOGNITION AND STACKING OPERATION PLANNING COMBINING 3D POINT CLOUD REPRESENTATION, DEEP-LEARNING AND PHYSICS ENGINE

Yajun Xu

HOKKAIDO
UNIVERSITY

January, 2023

Course of Systems Science and Informatics
Graduate School of Information Science and Technology
Hokkaido University

Doctoral Dissertation
submitted to Graduate School of Information Science and Technology,
Hokkaido University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy.

Yajun Xu

Dissertation Committee: Professor Satoshi Kanai (Chief examinar)
Professor Masahiko Onosato
Professor Atsushi Konno
Associate Professor Hiroaki Date

A STUDY ON STACKED OBJECT RECOGNITION AND STACKING OPERATION PLANNING COMBINING 3D POINT CLOUD REPRESENTATION, DEEP-LEARNING AND PHYSICS ENGINE*

Yajun Xu

Abstract

Technically, three-demission (3D) data which provides richer geometric, shape, and scale information than 2D data, make it easier for machines to understand and interact with their surrounding environment. Typical 3D data include depth images, point clouds, meshes, and volumetric grids. Among them, point clouds are widely used in various fields, such as robotics, autonomous driving, and civil engineering, to preserve the original geometric information in 3D space without discretization. In some specific scenes, many objects are stacked on each other. For instance, in a robotic bin-picking scene, wherein heavily piled up parts occlude each other; on the coast, a large number of wave-dissipating blocks are stacked together in order to protect the embankment. Recognizing individual objects in these cluttered scenes poses a problem. Adding new objects based on the state of the stacked objects to address engineering requirements is an even more considerable challenge.

In this dissertation, we design a 3D instance segmentation framework for stacked objects scenes using a deep neural network and then develop a system that simulates object stacking using a physics engine and deep learning to complete the object stacking plan based on our recognized results.

Increasingly, deep learning on point clouds has attracted attention in recent years. 3D instance segmentation networks for indoor scenes have made some breakthroughs but still face several significant challenges. Several non-real-time deep learning-based 3D recognition frameworks for indoor scenes have been developed recently. However, deep learning of 3D point clouds still faces several significant challenges, such as data annotation, the memory required to process large-scale point clouds, and time-consuming processing. We propose a fast point cloud clustering-based deep neural network, FPCC, for the instances segmentation of stacked objects. The network simultaneously predicts the similarity of points and the likelihood of being centroids. Based on the predicted results, this study designs a novel clustering algorithm that can quickly generate the final segmentation results. Experimental results on public datasets show that the proposed method has excellent performance, reaching the current state-of-the-art precision and processing speed.

Then, we extend the application scenario of this 3D instance segmentation scheme to the recognition of wave-dissipating blocks, a structural unit of breakwaters. Compared with the current methods that minor the whole structure of the breakwater, our method can minor the blocks at the instance level. The recognition consists of three main steps:

*Doctoral Dissertation, Course of Systems Science and Informatics, Graduate School of Information Science and Technology, Hokkaido University, SSI-DT46205029, January 10, 2023.

point cloud instance segmentation of the blocks, pose estimation, and classification. A novel point cloud feature extractor is designed to replace the original feature extractor of FPCC, which can process more points faster with the same computational overhead. The new feature extractor employs an attention-pooling mechanism, which allows the neural network to learn richer local information. Then, the block-wise 6D pose is estimated using a three-dimensional feature descriptor, point cloud registration, and CAD models of blocks. Finally, the type of each segmented block is classified using model registration results. The pose estimation results on real-world data showed that the fitting error between the reconstructed scene and the scene point cloud ranged between 30 and 50 mm, which is below 2% of the detected block size. The accuracy in the block-type classification on real-world point clouds reached about 95%. These block detection performances demonstrate the effectiveness of our approach.

Finally, based on the recognized results of wave-dissipating blocks, a system is developed to simulate the block stacking plan utilizing a physics engine and deep learning, which can predict the additional block amounts and their stacking poses and provide pre-visualization of their stacking operations. Deep learning was used to estimate the additional block poses that better fit the stacked blocks. The simulation was applied to an actual block-stacking operation in a local port at Hokkaido. The final construction results in the real world verified the accuracy and usefulness of the simulation.

This dissertation generally makes three major contributions to object recognition and object stacking simulation. The first one is to propose a fast framework for point cloud instance segmentation called FPCC. The second major contribution is improving FPCC and its use for stacked wave-dissipating block scenes. Combined with pose estimation, this enables us to accurately retrieve the majority of the blocks in a 3D scene, minoring the blocks at the instance level. The third major contribution is the development of a simulation system for simulating the block supplementation project, which provides customizable pre-visualization results and blocks stacking solutions according to different construction requirements.

Keywords: Instance segmentation, Wave-dissipating blocks, Point cloud, Physics engine, Deep learning

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter describes the background of the research, related works, objectives, proposed methods and their features, and the structure of this dissertation.

1.1 Background and Scope of This Study

In recent years, computer vision based on deep learning has made breakthroughs in classification [1, 2], object detection [3, 4], and semantic and instance segmentation [5–7]. The performance on specific tasks is even comparable to that of human experts [1, 8]. These achievements have been deployed in autonomous driving [9, 10], robot navigation [11], medical image processing [12], and many other areas. Although deep learning-based models simplify the process of human-designed features, many crucial issues still need to be addressed in some specific scenarios or tasks.

Stacked object recognition is one of them. In industry, a scene where multiple identical objects are stacked in a container is called the bin-picking scene, which is very common [13]. The lack of information due to occlusion and the vast number of identical objects are obstacles to recognition. Such a scene exists not only in the industrial field but also in fruits [14], wave-dissipating blocks, Etc (see Figure 1.1).

In addition, the need for large-scale, high-quality labeled data for deep learning models severely impedes the deployment of the models in practical applications. Transfer learning and unsupervised or semi-supervised learning are essential directions to explore to alleviate the dataset collection. However, a more feasible solution nowadays is to use synthetic data instead of manually labeled data [16].

After finishing the recognition of stacked objects, continuously stacking new objects to reach a target height or shape based on the current stacking condition of the objects is a new topic that we propose based on practical construction. One of its application scenarios is the maintenance project of breakwaters.

Wave-dissipating blocks placed around breakwaters are often constructed along the coastline to stop wave erosion, but the wave-dissipating blocks, which are the structural units of breakwaters, sink or even break as the waves erode (see Figure 1.2). Therefore, it is a long and tedious project to replenish the breakwater blocks regularly to reach the required height. The current conventional practice can only roughly estimate the required number of dissipation blocks by volume ratio. However, it cannot provide more specific guidance, such as where and how a new block should be inserted.

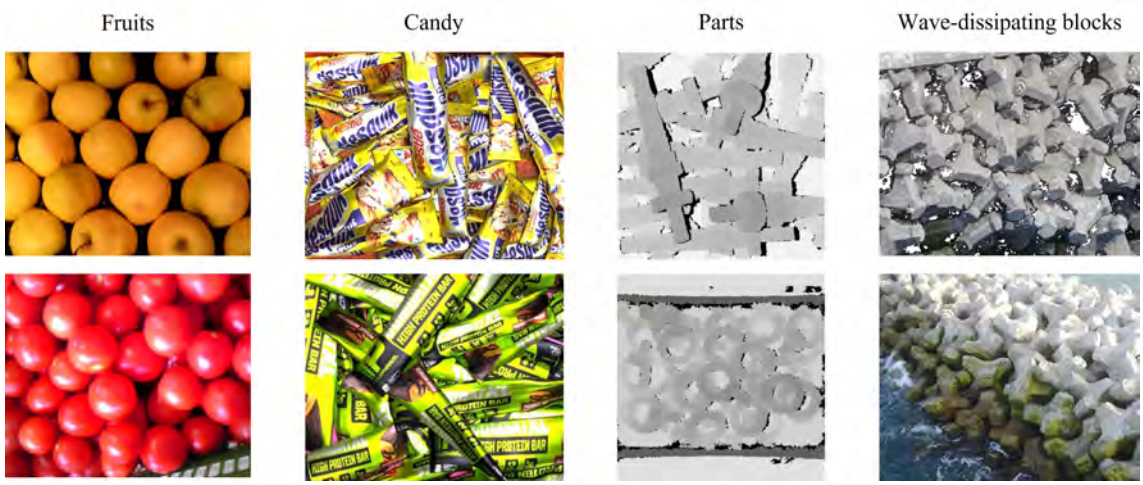


Figure 1.1: Examples of stacked scenes from FruitVeg-81 [14], Mikado [15], IPA [13] and wave-dissipating blocks.

Figure 1.2: Breakwater with its units—wave-dissipating blocks.

Based on this much-needed problem, this thesis mainly focuses on object identification and stacking plan in stacking scenarios to save human and financial resources. The proposed deep learning-based point cloud segmentation framework can be applied to numerous stacked object scenarios, such as bin-picking. The object stacking plans developed based on the physical engine can be highly customized to provide forward-looking construction guidance and can be applied to different construction needs, leading to efficiency gains and reduced production costs.

1.2 Related Works

1.2.1 3D Point Cloud Processing for Robotic Bin Picking

Computer vision empowers the interaction between robots and the real world. The availability of point clouds in a low-cost way has become possible with the development of 3D sensors. Point clouds are almost essential in robots' simultaneous localization and mapping (SLAM) due to the richer geometric, shape, and scale information they provide than images [17]. Another vital application of point cloud is grasping task, which generally consists of four main steps: target object localization, object pose estimation, grasp

Figure 1.3: An experimental real-world robot cell from Fraunhofer IPA [13]. A dual-arm robot is taking parts out of a bin and dropping them into another.

detection (synthesis), and grasp planning [18]. Numerous algorithms have been developed based on point clouds. However, object recognition and 6D pose estimation in cluttered scenes are still challenging.

Industrially, a scene where multiple objects are stacked in a container is called a bin-picking scene (see Figure 1.3). The robot must accurately grasp a target object from the container before completing a series of tasks [13, 19, 20]. In this scenario, pose-invariant descriptors like Point Pair Feature (PPF) have been widely used to estimate the 6D poses of objects from measured point clouds to locate the object. However, the occlusion of the stacking objects seriously lowers the discrimination performance among individual object instances. Therefore, the development of point cloud instance segmentation robust against the occlusion should be developed.

1.2.2 3D Point Cloud Processing for as-is Status Recognition of Existing Wave-Dissipating Blocks for Supplementary Work.

Figure 1.4: Regular supplementary works are required to maintain the height of the stack of blocks.

Wave-dissipating blocks made of large concrete slabs are essential components of the armor layer of breakwaters that protect their core from direct wave attacks. However, long-term wave motion and erosion damage the blocks irreversibly, causing them to sink and

even break off [21]. Therefore monitoring of breakwaters is a key aspect of maintaining the quality of breakwaters [22] as shown in Figure 1.4. To this end, constructors must estimate the number of new blocks to be supplemented as precisely as possible. Unfortunately, it is difficult to estimate the number of blocks required due to the quality of data and the limitations of current algorithms. Every year, a national budget of about 100 billion yen is invested in the wave-dissipating block maintenance project. It turned out that the cost could be reduced by about 10%~20% if the required number of blocks was accurately estimated.

In this study, we recognize individual wave-dissipating blocks from the point cloud so that an accurate reconstruction of the current scene can be performed in the physics engine. This will be the foundation for some subsequent developments and offers the following benefits for the administrators:

- 6D poses of individual blocks faithfully reproduce the as-built status of each block to improve the accuracy of estimates of new blocks' quantities and their stacking plan in the supplemental work;
- The as-built status can be grasped block by block after the construction and supplemental works. Thus, the construction results can be recorded and visualized comprehensively compared to recording only the measured point clouds of existing block surfaces;
- By providing the pose and attribute information to each block model, it is possible to check the long-term change in the blocks, such as missing, sinking, and damaged blocks, and implement a more precise and sustainable maintenance activity.

UAVs equipped with 3D sensors are a flexible and low-cost means of obtaining geometric data on breakwaters. To date, several studies compared the 3D point clouds acquired at different times to evaluate possible changes in breakwaters within a certain period [22–27]. Few studies have focused on the changes or movements of individual wave-dissipating blocks [21, 28, 29].

1.2.3 Object Stacking Operation Planning

Figure 1.5: Examples of packing problem from Y. Ma et al. [30]. 3D Irregular objects need to be packed as many as possible into a given container.

Object stacking operation planning can be regarded as an extension of the packing problem. Object stacking operation planning not only expects to fill as many objects as

possible in a given volume or container but also needs to meet the physical aspects of feasibility.

Packing irregular 3D rigid objects into a container with pre-defined dimensions is a well-known Np-hard problem that is faced in practical applications such as the construction industry, 3D printing, and dry stacking. No algorithm can find a globally optimal solution for the problem in polynomial time [31].

Due to the limitation of computational power, early approaches have approximated irregular objects into some regular shape primitives, such as bounding boxes or bounding cylinders, to reduce the computational requirements for geometric analysis and processing [32, 33]. However, these were inefficient and not current to the application's needs [34].

Researchers have previously developed methods for packing objects based on mathematical models [35]. However, their mathematical modeling methods suffered from complexion and a large number of vertices, resulting in poor accuracy for typical irregular objects. Furthermore, some studies [30, 31] have emphasized the solution's optimality at the expense of the balance and stability of object stacking, which is particularly important in construction.

1.3 Objectives, Proposed Approaches and Contributions

1.3.1 Objectives of this study

The research aims to design and propose an efficient and effective method for stacked object recognition and stacking operation planning combining 3D point cloud representation, deep learning, and a physics engine. In detail, the objectives of the research are summarized as follows:

- For the object recognition of stacked objects, a deep-learning-based fast point cloud instance segmentation method is newly developed to accurately distinguish individual objects in stacking scenes, such as the industrial bin-picking scene.
- The deep neural networks for fast point cloud instance segmentation are employed to segment the point clouds of existing stacked wave-dissipating blocks placed oversea and undersea levels at the instance level, estimate the 6D poses and classify the type of the segmented blocks.
- To generate reasonable stacking plans of additional wave-dissipating blocks, an algorithm is designed to automatically find the space where additional blocks need to be inserted. Also the optimum initial pose of an block before it falls is predicted using another deep neural network, as the initial poses of blocks have a crucial impact on the construction results.
- To achieve the first three goals, we effectively used the physics engine to develop systems that can automatically generate physically feasible scenes of stacked objects, providing sufficient synthetic data for training the neural network and validating the method's performance.

1.3.2 Proposed Approaches

1) Deep-learning-based instance segmentation from point cloud of stacked objects

We proposed a novel neural network for block segmentation from point cloud. We use our-original category-agnostic instance segmentation network FPCCv2 to segment the input point cloud into the subsets of points corresponding to individual block instances from an input point cloud measured by UAV and MBES. FPCCv2 is a kind of deep neural networks and is pre-trained by synthetic point clouds that mimic the point clouds of stacked blocks measured by UAV and MBES, respectively, using the stacked block CAD models and surface point sampling. FPCCv2 extracts features of each point while inferring the centroid of each instance. After that, the remaining points are clustered to the closest centroids in the feature embedding space.

2) Model-based 6D pose estimation and type classification of objects

After segmenting the input point cloud into a set of points corresponding to individual block instances, the 6D pose of an individual block is estimated from each segmented point cloud using Point Pair Feature descriptor and the best fit point cloud alignment by Iterative Closest Points (ICP), which match the scene points with the surface points sampled from the CAD model of the blcok. Moreover, the block type is classified based on the pose estimation results, i.e., the fits between the model point cloud and the scene point cloud.

3) Block stack-up simulation for object instance segmentation

The simulation software is implemented on a python module for physics engine Pybullet. We use synthetic point cloud data that mimics the stacking poses of wave-dissipating blocks to train our instance segmentation network FPCCv2, and evaluate the performance of the segmentation of our method. The synthetic data that is easier to synthesize in an acceptable period, makes the network more robust and avoids the interference of environmental factors.

4) Deep-learning-based physically-feasible and semi-compact object stack-up simulation

After reconstructing the scene in a physical engine based on the recognition result, we designed an algorithm to detect the space between the existing blocks. And then we employ a network to predict a reasonable pose of new block to make the whole breakwater more compact and stable. The combination of the physical engine and deep learning enables physically-feasible and semi-compact planning of stack-up operation planning.

1.3.3 Contributions

The contributions of this study are listed as follow.

1) Fast 3D instance segmentation for robotic bin-picking

- A high-speed instance segmentation scheme for a 3D point cloud is proposed.
- Designed a convolutional network and an algorithm for point cloud instance segmentation.
- Experiments show that FPCC trained by synthetic data demonstrates excellent performance on real-world data compared with existing methods.
- FPCC achieved 55% precision on IPA Bin-Picking dataset and 80% precision on XA Bin-Picking dataset.

- The proposed 3D instance segmentation scheme can be applied in other similar stacked objects scenes. Our method will significantly promote the application of deep learning in industrial production and make robots more intelligent.

2) Detection of wave-dissipating block from large-scale point cloud

- Proposed a novel feature extractor for point cloud. By stacking graph convolution layers and constructing local information of the point cloud, we can expand the perceptual field of each point at the cost of a small computational overhead.
- Developed a simulation system based on Pybullet for providing the training data sample for our network, so that we can train the network without any manually labeled data.
- Experiments shown that our method could retrieve 70% 95% blocks in a point cloud scene.
- Due to the efficiency of sparse convolution, a neural network built with sparse convolution should be able to process larger point clouds than the current. Our simulation system can provide richer labeling for a wider variety of scenarios.

3) Deep learning-based construction planning of wave-dissipating block

- Designed an algorithm to detect the spaces between blocks for inserting a new block.
- Made the new block inserted in a reasonable pose, so that the whole structure of the breakwater is more stable.
- Provided previsualization of the construction work and the reconstruction plan.
- The strategy of how to insert a new block need to be complete and be more flexible.

1.4 Structure of the Dissertation

The dissertation is organized and divided into five chapters:

- **Chapter 1: Introduction.** This chapter introduces the background, objective, relative works, and the construction of the dissertation.
- **Chapter 2 Deep learning-based object instance segmentation from point clouds of stacked industrial parts for robotic bin picking**
  This chapter introduces the bin-picking scene and explains why segmentation is necessary for 6D pose estimation work about the bin-picking scene at first. Then the chapter summaries some related works about point cloud instance segmentation. Next, the chapter narratives the principles of a CNN for point-cloud instance segmentation, FPCC. A simulation system is developed to simulate blocks stacking with bullet physical engine to generate point clouds and instance label. The experimental results show that the performance of the network trained with synthetic data is acceptable.

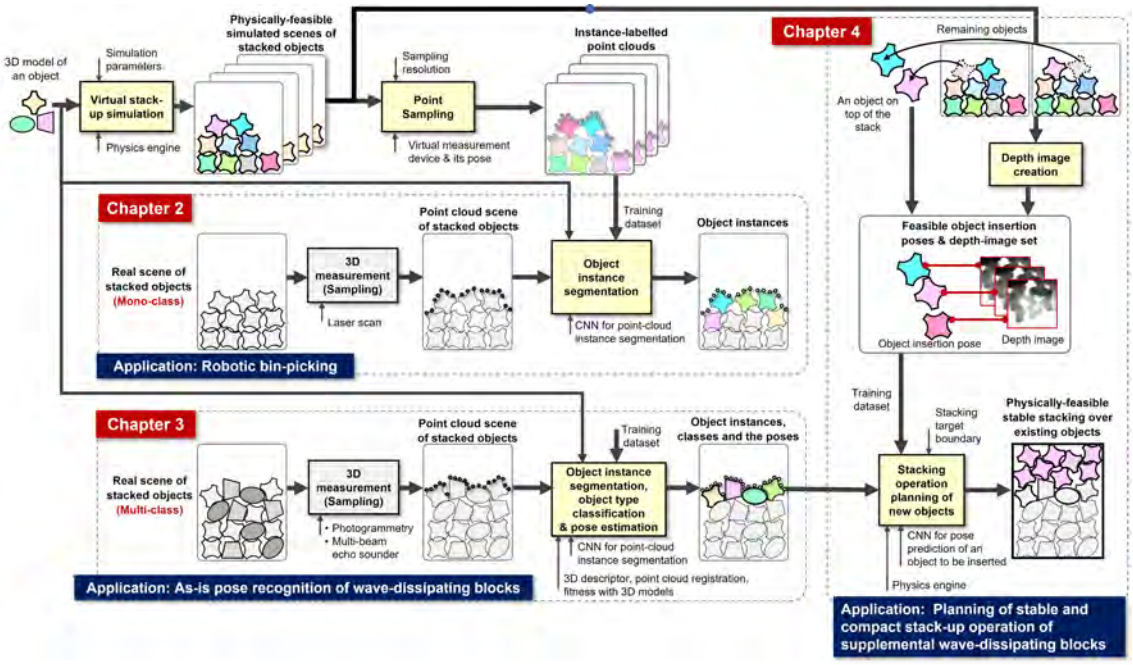


Figure 1.6: The main structure of the dissertation, except for the introduction (Chapter 1) and the conclusion (Chapter 5). In this dissertation, Chapter 2 narratives the principles of our 3D point cloud instance segmentation method. Chapter 3 develops a framework for block recognition, pose estimation, and block type classification. Based on the recognized results from the Chapter 3, Chapter 4 pictures a deep learning-based pose prediction method to make the new blocks drop from a fine initial pose so that the breakwater structure is steady and compacted.

- **Chapter 3 Deep learning-based object instance segmentation and pose estimation from point clouds of stacked wave-dissipating blocks**
  At first, this chapter introduces the breakwater and wave-dissipating blocks and the current progress in monitoring breakwater and recognizing wave-dissipating blocks. The scene point clouds are measure by photogrammetry and Multi-beam echo sounder. Next, the chapter presents our recognition scheme for wave-dissipating blocks, analyzes the limitation of FPCC, and prose a new point-wise feature extractor for FPCC. Next, the chapter picture the simulation system for synthetic data generation, which are used for training network and validation experiment. Finally, this chapter conducts experiments on synthetic and real-world data to validate our scheme's effectiveness in recognition, pose estimation, and block type classification.

- **Chapter 4 Deep learning-based object stacking operation planning for replenishing wave-dissipating blocks**
  This chapter first introduces the construction process of the wave-dissipating block supplementation project. Next, the chapter reconstructs the current block scene in a physic engine with the block CAD model and recognized poses based on the results of chapter 3. Next, the chapter presents a space detection algorithm for finding where the new blocks need to be inserted. And then, the chapter narratives a deep learning-based pose prediction method to make the new blocks drop from a

fine initial pose so that the breakwater structure is steady and compacted. At last, deep learning-based supplement work is compared with traditional one to estimate its effectiveness.

- **Chapter 5 Conclusions and Future Work** This chapter summarizes the results and the conclusions that could be drawn from the previous chapters. The limitations of the proposed approach are discussed. Future research directions are prospected.

Chapter 2

Deep Learning-based Object Instance Segmentation from Point Clouds of Stacked Industrial Parts for Robotic Bin-Picking

2.1 Background and Objectives of this Chapter

Acquisition of three-dimensional (3D) point cloud is no longer difficult due to advances in 3D measurement technology, such as passive stereo vision [36–39], phase shifting method [40], gray code [41], and other methods [42, 43]. As a consequence, efficient and effective processing of 3D point cloud has become a new challenging problem. Segmentation of 3D point cloud is usually required as a pre-processing step in real-world applications, such as autonomous vehicles [44], human-robot interaction [45–47], robotic bin-picking [3, 19, 20, 48], pose estimation [49–53], and various types of 3D point cloud processing [54–58]. In the field of robotics, bin-picking scenes have received a wide range of attention in the past decade. In this scene, many objects of the same category are stacked together. The difficulty of bin-picking scenes in logistics warehouses is that there are too many categories and unknown objects [59–61], while the problem of industrial bin-picking scenes is that it is difficult to distinguish the same objects and make datasets. At present, an application of convolutional neural networks (CNNs) to instance segmentation of 3D point cloud is still far behind its practical use. The technical key points can be summarized as follows: 1) convolution kernels are more suitable for handling structured information, while raw 3D point cloud is unstructured and unordered; 2) the availability of high-quality, large-scale image datasets [1, 62, 63] has driven the application of deep learning to 2D images, but there are fewer 3D point cloud datasets; and 3) instance segmentation on 3D point cloud based on CNNs is time-consuming.

For key point 1), PointNet [64] has been proposed as the first framework which is suitable for processing unstructured and unordered 3D point clouds. PointNet does not transform 3D point cloud data to 3D voxel grids such as [2, 65], but uses multi-layer perceptions (MLPs) to learn the features of each point and has adopted max-pooling to obtain global information. The pioneering work of PointNet has prompted further research, and several researchers have introduced the structure of PointNet as the backbone

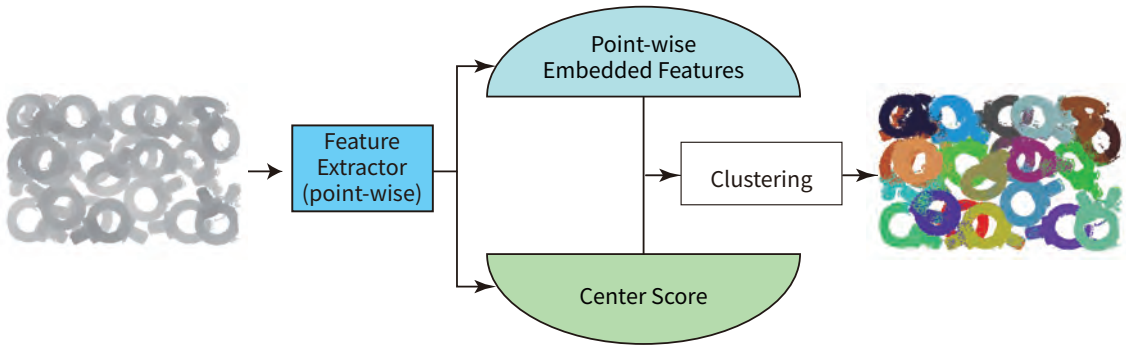


Figure 2.1: Instance segmentation results using FPCC-Net. FPCC-Net has two branches: the embedded feature branch and the center score branch.

of their network [7, 66, 67]. It is known that PointNet processes each point independently and it results in learning less local information [66, 68]. To enable learning of the 3D point cloud's local information, the methods proposed in [68–75] have increased the network's ability to perceive local information by exploring adjacent points. Following our previous work [16], we employ DGCNN [68] as our feature extractor because DGCNN is flexible and robust to process point clouds with only coordinates.

For key point 2), some well-known 3D point cloud datasets include indoor scene datasets such as S3DIS [76] and SceneNN [77], driving scenario datasets such as KITTI dataset [78] and Apollo-SouthBay dataset [79], and single object recognition dataset likes ShapeNet dataset [2]. For robotic bin-picking, it is a huge and hard work to provide a general training dataset of various industrial objects and there is no such dataset currently. Synthesizing training data through simulation provides a feasible way to alleviate the lack of training dataset [13, 16, 80–83]. At this stage, we argue that training the network with synthetic data is an economical and feasible strategy. Our network is trained by synthetic dataset and shows acceptable results on real data.

For key point 3), the reasons why instance segmentation on 3D point cloud by CNNs is time-consuming are described as follows. Instance segmentation locates different instances, even if they are of the same class. As instances in the scene are disordered and their number is unpredictable, it is impossible to represent instance labels with a fixed tensor. Therefore, the study of instance segmentation includes two methods: the proposal-based method requiring an object detection module and the proposal-free method without an object detection module. Proposal-based methods require complex post-processing steps to deal with many proposal regions and have poor performance in the presence of strong occlusion. For the instance segmentation of 3D point cloud, most researchers adopt the proposal-free method [7, 67, 84–88]. The proposal-free method usually performs semantic segmentation at first and then distinguishing different instances via clustering or metric learning [7, 16, 67, 84]. The current clustering methods first generates multiple candidate groups and then merge them, which is a very time-consuming process. In contrast, our clustering algorithm does not generate candidate groups, but directly generates instances based on the feature distance between the object's center point and the rest of the points. This way dramatically improves the speed of instance generation and avoids the case that a point belongs to multiple instances at the same time.

This chapter aims to design and propose a fast point cloud clustering for instance seg-

mentation method named FPCC consisting of FPCC-Net and a fast clustering algorithm based on the output of FPCC-Net. FPCC-Net is a graph convolutional neural network that can effectively segment the 3D point cloud at instance-level without training by any manually annotated data. FPCC-Net involves mapping all points to a discriminative feature embedding space, which satisfies the following two conditions: 1) points of the same instance have similar features, 2) points of different instances are widely separated in the feature embedding space. Simultaneously, FPCC-Net finds center points for each instance, and the center points are used as the reference point of the clustering process. After that, the fast clustering is performed based on the center points as shown in Figure 2.1.

The main contributions of this work are listed below.

- A high-speed instance segmentation scheme for 3D point cloud is proposed.
- The proposed scheme consists of a novel network of 3D point cloud for instance segmentation named FPCC-Net and a novel clustering algorithm using the found center points.
- A hand-crafted attention mechanism is introduced into the loss function to improve the performance of FPCC-Net, and its effectiveness is verified in an ablation study.
- Experiments show that FPCC-Net trained by synthetic data demonstrates excellent performance on real-world data compared with existing methods.
- We annotate instance information for parts that have not been labeled in XA Bin-Picking dataset [16]. The completed dataset is available at https://github.com/xyjbaal/FPCC.

The remainder of this Chapter is organized as follows. Section 2.2 discusses the progress of instance segmentation on images and 3D point cloud. Section 2.3 shows the structure and principle of FPCC-Net. Experimental analyses are provided in Section 2.4. Section 2.5 discusses the Chapter. Finally, Section 2.6 gives the conclusion of the Chapter.

We use the following notations in this Chapter. A real number set is represented by $\mathbb{R}$. A coordinate of point i is denoted by $p_i = (x_i, y_i, z_i) \in \mathbb{R}^3$. Point cloud containing N points is denoted by $\mathbb{P} = \{p_1, p_2, ..., p_N\}$. Distance function is denoted by

$$d(a, b) = \|a - b\|_2, \tag{2.1}$$

where $d(a, b)$ denotes Euclidean distance between $a \in \mathbb{R}^n$ and $b \in \mathbb{R}^n$. For a matrix $A \in \mathbb{R}^{n \times m}$, (i, j)-th element of A is denoted by $a_{(i,j)}$.

2.2 Related Works

With the emergence of CNNs, the methods of feature extraction from images and 3D point cloud have been changing from manual design to automatic learning [89–91]. Instance segmentation is one of the most basic tasks in the field of computer vision and receives much attention. Segmentation on two-dimensional (2D) images has been almost fully developed [92, 93], but 3D point cloud segmentation has remained underdeveloped.

2.2.1 2D Instance Segmentation

Current 2D instance segmentation methods can be roughly divided into two categories, two-stage method, and one-stage method. Two-stage means they perform object detection first by utilizing proposal generation, then followed by mask prediction. Mask R-CNN [6] is one of the representative two-stage instance segmentation methods. Mask R-CNN decomposes the problem of instance segmentation into object detection and pixel-level segmentation of a single object. Mask R-CNN has added a branch to predict object masks based on Faster R-CNN [94]. Based on the Mask R-CNN [6] and FPN [95], PANet [96] enhanced mask prediction by improving the information propagation between lower and higher levels. MaskLab [97] predicted the direction of each pixel to its corresponding instance center for instance segmentation of the same class. TensorMask [98] regards dense instance segmentation as a prediction task over 4D tensor and proposes a general framework for operations on 4D tensors. The one-stage methods are usually faster than the two-stage methods because the one-stage methods do not contain the proposal generation and pooling step. YOCLAT [99] generates instance masks by linearly combining prototype and mask coefficients, sacrificing a little performance for computational speed. BlendMask [100] combined instance-level information with semantic information to enhance mask prediction, and CenterMask [101] added a spatial attention-guided mask branch to the object detector to predict masks. PolarMask [102] describes the mask of each instance by its center and rays emitted from the center to the contour. Other approaches which are not categorized into one-stage nor two-stage methods, such as clustering or metric learning, do not perform well on 2D instance segmentation [103, 104]. Although the deep learning-based methods on 2D instance segmentation utilizing a large number of high-quality and manually labeled datasets [62, 105] have great achievements, they still have drawbacks in industrial bin-picking scenes, where labeled datasets are often difficult to obtain. To overcome the difficulties of making datasets, SD Mask R-CNN [83] has been proposed as an extension of Mask R-CNN. The paper [83] has presented a method to generate synthetic datasets rapidly, and their network was trained only with the generated synthetic depth images instead of RGB images. However, SD Mask R-CNN does not show enough performance, especially for multiple object instances with occlusion in the scene.

2.2.2 3D Instance Segmentation

Some researchers have adopted proposal-based methods that detect objects and predict the instance masks. 3D-SIS [106] have combined 2D images and 3D geometric information to infer the bounding boxes of objects in 3D space and the corresponding instance mask. The demand for 2D images limits the application of this method because it is expensive to train the network with 2D images. GSPN [107] continued the idea of Mask R-CNN. They first predicted the candidate regions, and then refined the candidate regions with R-PointNet to obtain the result of instance segmentation. However, the region proposal network of 3D-SIS and GSPN consumes a long computing time. 3D-MPA [88] has used an object-centric voting scheme to generate instance proposals, which are robust to potential outliers in the instance proposal stage. 3D-BoNet [108] provided by B. Yang et al. directly regresses the bounding box of each instance in the 3D point cloud and simultaneously predicts point-level masks for each instance. 3D-BoNet performs well on the completed point cloud because there is little overlap of objects' bounding box in these datasets e.g., S3DIS [76], ScanNet [109]. In other words, the overlap and incompleteness of objects in

the bin-picking scenes make 3D-BoNet difficult to regress reasonable bounding boxes.

Proposal-free methods extract the features of the points at first and then group points into instances. SGPN [7] is the first direct 3D instance segmentation method that assumed the points of the same instance should have similar features. Three sub-networks predict semantic labels, confidence score, and point-wise features, respectively. The confidence score of each point indicates the confidence of the reference point of clustering. SGPN [7] have highlighted an interesting phenomenon that the clustering confidence scores of the points located in the boundary area are lower than others. Inspired by this, FPCC takes only one point that is most likely to be the geometric center of an object as the reference point of clustering for the object. Some methods combining semantic with instance information have been proposed to improve performance [67, 84, 110]. Q. Phm et al. [67] have used a Multi-Value Conditional Random Field to learn semantic and instance labels simultaneously. J. Lahoud et al. [110] have performed instance segmentation by clustering 3D points and mapping the features of points to the feature embedding space according to relationships of point pairs. However, the objects in the bin-picking scene are all of the same types. Thus the semantic information of each point is the same. The way of combining semantic and instance information with each other loses its effectiveness in this case. PointGroup [85] and HAIS [86] employ a similar strategy. They predict the offset vector for moving each point towards its instance center and then cluster the moved points in Euclidean space. OccuSeg [87] has constrained the clustering based on predicted occupancy size and the clustered occupancy size, which help to correctly cluster hard samples and avoid over-segmentation. B, Zhang, et al. [111] have presented a probabilistic embedding framework to encode the features of each point and a novel clustering step.

Although previous proposal-free methods have used various ways to extract features of points, they all need to find points far greater than the number of instances as reference points for clustering, and each reference point corresponds to a potential group. Each group is merged by intersection over union (IoU). The process of merging takes a lot of time. The reason for time-consuming is described in more detail in Section 2.4.4. In contrast, FPCC does not need the process of merging. In addition, all these methods are based on public datasets such as S3DIS [76] and SceneNN [77] with rich annotations. Making such a dataset for bin-picking scenes is a time-consuming and laborious task [16]. FPCC shows an acceptable performance on real-world data even trained by synthetic data.

2.2.3 Instance segmentation for Bin-Picking Scenes

Overall, the existing instance segmentation methods for bin-picking scenes can be divided into two groups: logistics-oriented methods and industrial-oriented methods. The former is multi-class, multi-instance learning. The latter tends to be one-class, multi-instance learning from cluttered scenes without predicting semantic labels. Researchers [61, 83, 112–115] have adopted the mainstream 2D detection or 2D instance segmentation network for logistics scenes to locate objects. However, logistics scenes are usually less cluttered than industrial scenes.

Currently, few works have focused on instance segmentation for industrial bin-picking scenes. Because industries primarily use point cloud, and current 3D instance segmentation is far slower than 2D instance segmentation and challenging to annotate. D. Liu et al. [51] have located the bounding boxes of unoccluded objects from RGB image by RetinaNet and then projected the bounding boxes into 3D space. Further, M. Grard et

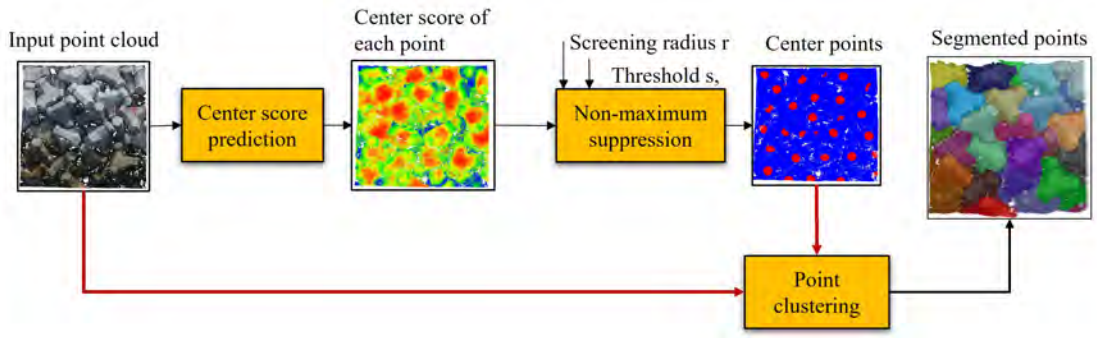


Figure 2.2: Instance segmentation principle of FPCC. The geometric centers of each object is clustered with other points based on the feature distance.

al. [15] proposed a residual encoder-decoder design to predict the masks of unoccluded objects. They did not segment industrial objects in 3D space and required 2D image annotation. W. Abbeloos et al. [116] used point pair features to match model points and scene points. They have reduced the complexity of the problem by reducing the search space using simple heuristics. However, the number of objects found in this way is affected by the degree of clutter, and the computation time is sensitive to the number of points. PPR-net [50] has regressed the 6D transform of the instances corresponding to each point, and then a density-based clustering method clustered these transformed points in the pose space. The primary goal of PPR-net is to regress the 6D transform of the objects, and the results of instance segmentation are generated based on pose regression.

2.3 Proposed Method

This chapter proposes a novel clustering method for instance segmentation on the 3D point cloud. The training data are 3D point cloud without color and can be automatically generated in simulation by using a 3D shape model of the target object. The main idea of fast clustering is to find geometric centers of each object, and then use these points as reference points for clustering. Figure 2.2 shows the principle of FPCC. Compared with the existing cluster-based methods, we have two advantages. The first is that we can achieve an acceptable result by synthetic point cloud without color information. The second is that, theoretically, we can find center points equivalent to the number of instances in the scene as reference points for clustering, so there is no need for redundant merged algorithms [7, 84], which consumes a lot of computing time and easily introduce errors in severely overlapping scenes.

2.3.1 Structure of Fast Point Cloud Clustering (FPCC)

In the first step, coordinates of original 3D point cloud $p_i = (x_i, y_i, z_i)$, $i = 1, 2, ..., N$ is converted to new coordinate system by

$$\begin{aligned} \overline{x}_i &= x_i - \min\{x_1, x_2, ..., x_N\} \\ \overline{y}_i &= y_i - \min\{y_1, y_2, ..., y_N\} \\ \overline{z}_i &= z_i - \min\{z_1, z_2, ..., z_N\}, \end{aligned} \tag{2.2}$$

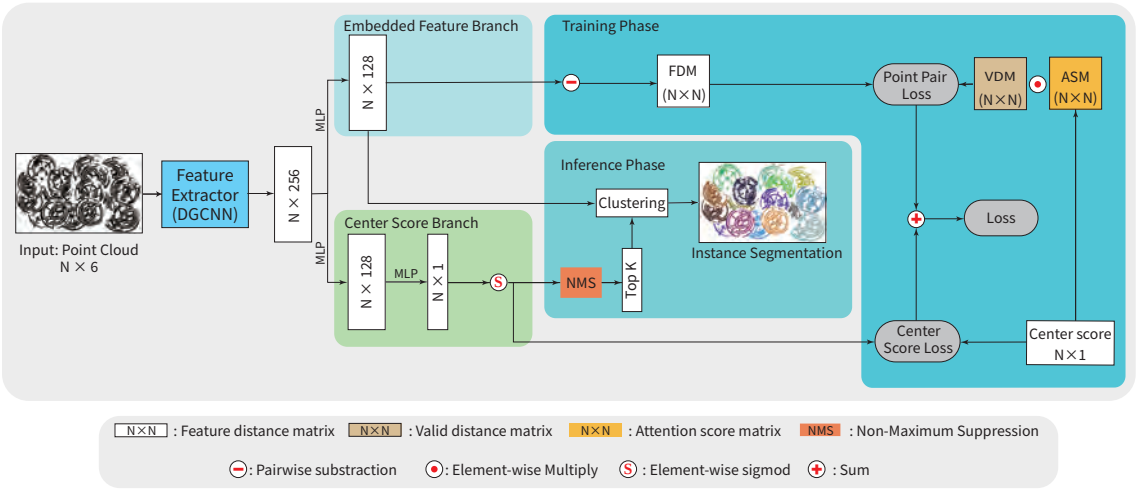


Figure 2.3: Network architecture of FPCC-Net. The represented 3D point cloud $[\overline{x}_i, \overline{y}_i, \overline{z}_i, n_{i,x}, n_{i,y}, n_{i,z}]^\top$ $(i = 1, 2, \cdots, N)$ is fed into the network and the instance label is outputted for each point. The features of each point are extracted using a feature extractor and then sent to two respective branches. The embedded feature branch extracts 128-dimensional features of each point and the center score branch predicts the center score of each point. For supervising FPCC-Net, the following matrices are introduced. Valid distances matrix defined by (2.4) is a binary matrix used to ignore some point pairs whose distance is greater than a certain threshold. Attention score matrix defined by (2.6) is used to increase the weight of point pairs closer to the center position.

Then converted each point is represented by a six-dimensional (6D) vector of $\overline{x}$, $\overline{y}$, and $\overline{z}$ and a normalized location (n_x, n_y, n_z) as to the whole scene (from 0 to 1). The represented 3D point cloud is fed into the network and outputs are 128-dimensional features and the center score of each point.

As shown in Figure 2.3, FPCC-Net has two branches that encode the feature of each point in the feature embedding space and infer each point's center score. First, the point-wise features of N points are extracted through a feature extractor. In FPCC-Net, DGCNN [68] without the last two layers is adopted as the feature extractor. DGCNN has better performance than PointNet in extracting the features of point cloud without color [16]. The extracted point-wise features with size $N \times 256$ are fed into two branches, **embedded feature branch** and **center score branch**.

In the embedded features branch, the extracted features pass through an MLP to generate an embedded feature with size $N \times 128$. The center score branch is parallel to the embedded feature branch and used to infer the center score of each point. In the center score branch, the point-wise features generated by the feature extractor are activated by a sigmoid function after passing through two MLP. Then, predicted center score $\widehat{s}_{\text{center}}$ with size $N \times 1$ is obtained. After the prediction of the center scores, We use algorithm 1 to find the points most likely to be the geometric centers of each object, and the found points are taken as reference points in the clustering process.

2.3.2 Training phase

The loss of the network is a combination of two branches: $L = L_{EF} + \alpha L_{CS}$, where L_{EF} and L_{CS} represent the losses of the embedded feature branch and the center score branch, respectively. The symbol α is a constant that makes L_{EF} and L_{CS} terms are roughly equally weighted. We introduce three matrices, feature distance matrix, valid distance matrix (VDM), and attention score matrix (ASM) for the learning of embedded features.

We explain our design for the training phase in the following order: feature distance matrix, valid distance matrix, center score, attention score matrix, embedded feature loss, and center score loss. Attention score matrix is obtained from the center score of each point.

Feature distance matrix

In the feature embedding space, the points belonging to the same instance should be close, while the points of different instances should be apart from each other. To make features of the points in the same instance similar, we introduce the following feature distance matrix $D_F \in \mathbb{R}^{N\times N}$. The (i, j)-th element of D_F is represented by

$$d_{F(i,j)} = \|e_F^{(i)} - e_F^{(j)}\|_2. \tag{2.3}$$

Valid distance matrix

The valid distance matrix $D_V \in \mathbb{R}^{N\times N}$ is a binary matrix in which each element is 0 or 1. The purpose of introducing D_V is to make the network focus on distinguishing whether point pairs within a certain Euclidean distance belong to the same instance or not. In the inference phase, points are clustered based on the feature distance and the Euclidean distance of the point pair at the same time. If the Euclidean distance of two points exceeds twice the maximum distance $d_{\max}$, the two points cannot belong to the same instance. Therefore, we ignore these point pairs with too far distance so that they do not contribute to the loss. The (i, j)-th element of D_V is defined by

$$d_{V(i,j)} = \begin{cases} 1 & \text{if } \|p_i - p_j\|_2 < 2d_{\max} \\ 0 & \text{otherwise} \end{cases} \tag{2.4}$$

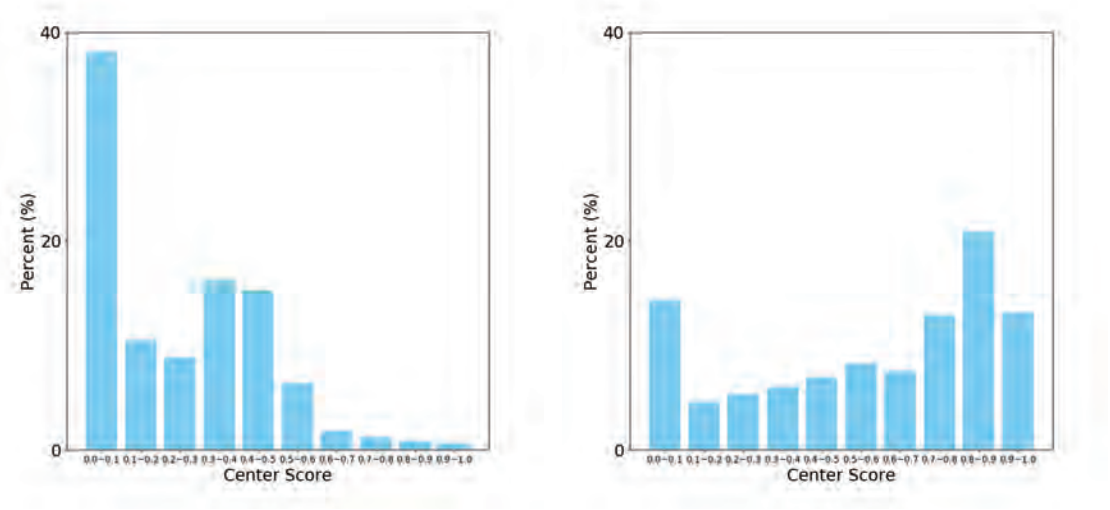
(2.4) indicates whether the Euclidean distance between point p_i and p_j is within a reasonable range or not.

Center score

The center score is designed such that it should reflect the distance between a point and its corresponding center. The points near the center of object have higher scores than the points on the boundary. Based on this concept, the center score of p_i is introduced by

$$s_{\text{center}(i)} = 1 - \left(\frac{\|p_i - c_i\|_2}{d_{\max}}\right)^{\beta}, \tag{2.5}$$

where β is positive constant and c_i is the coordinate of the geometric center of the instance to which point p_i belongs. The value of $s_{\text{center}(i)}$ is in the range $[0, 1]$. If $\beta = 1$, the distribution of the center score will lead to imbalances, as shown in Figure 2.4: only a



(a) $\beta = 1$ (b) $\beta = 2$

Figure 2.4: Distribution of the center score in the same scene. It is apparent that $\beta = 2$ makes the scores more uniform in the 0-1 interval.

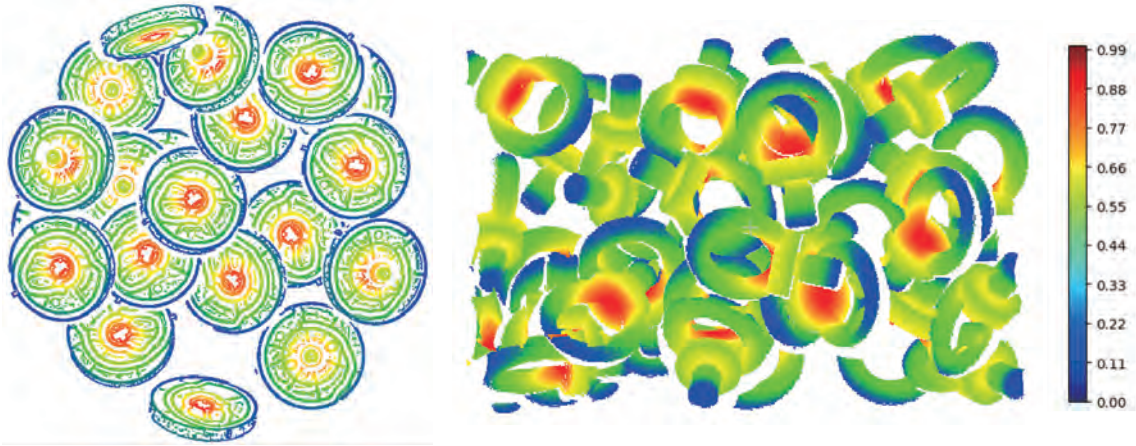


Figure 2.5: Visualization of center score ($\beta = 2$). Red indicates a higher score. The points at the boundary are mostly scored 0.

very small number of points have higher scores, while most points have lower scores. This causes the center score branch to fail to effectively predict the center scores (all scores are biased towards zero). Figure 2.4 shows that $\beta = 2$ leads more uniform balance than $\beta = 1$. Thus, β is set to 2 in our implementation. The center scores are visualized in Figure 2.5. Figure 2.5 shows that the points on the boundary area are mostly scored 0, and those near the center are approximately scored 1.

Attention score matrix

We introduce attention score matrix $S_A \in \mathbb{R}^{N \times N}$ to increase the weight of important point pairs. The (i, j)-th element of S_A represents the weight of a point pair for point i

Ring Screw Gear Shaft Object A Object B Object C

Figure 2.6: Models of objects used in the experiment. The gear shaft and ring screw are from the Fraunhofer IPA Bin-Picking dataset [13], and Object A, B, C are from XA Bin-Picking dataset [16].

(a)

(b)

Figure 2.7: (a) Synthetic 3D point cloud scenes. (b) Center score computed by (2.5).

and j. Because the center point is used as the reference point for clustering in inference phase, the point pair closer to the center position should have a higher weight. In this dissertation, $s_{A(i,j)}$ is computed by

$$s_{A(i,j)} = \min(1, s_{\text{center}(i)} + s_{\text{center}(j)}). \tag{2.6}$$

Embedded feature loss

A point pair (p_i, p_j) has two possible relationships as follows: 1) p_i and p_j belong to the same instance; and 2) p_i and p_j belong to different instances. By considering this, embedded feature loss L_{EF} is defined by

$$L_{\text{EF}} = \sum_{i}^{N} \sum_{j}^{N} w_{(i,j)} \kappa_{(i,j)}, \tag{2.7}$$

where $w_{(i,j)}$ is a element of $W \in \mathbb{R}^{N \times N}$ which is a weight matrix obtained through element-wise multiplying D_V by D_A, that is,

$$w_{(i,j)} = d_{V(i,j)} s_{A(i,j)}. \tag{2.8}$$

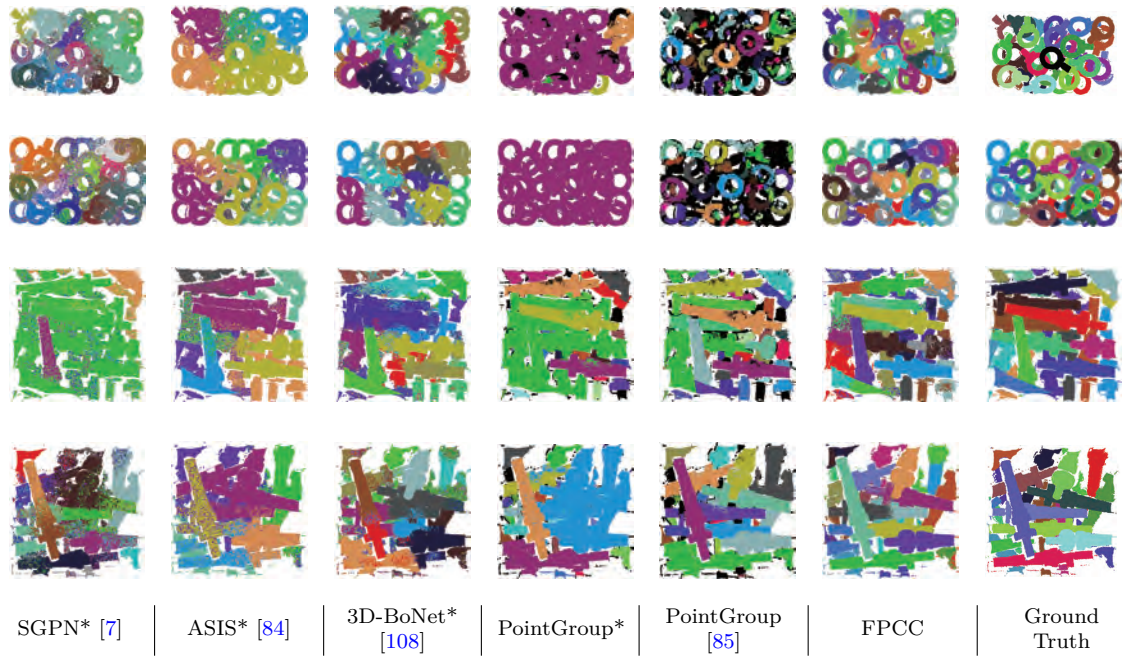


Figure 2.8: Comparison results on IPA. The performance of SGPN and ASIS is poor on bin-picking scene. 3D-BoNet has difficulty in distinguishing some overlapping instances. The performance of FPCC is acceptable even in high clutter environment. *: Feature extractor is DGCNN.

In (2.7), $\kappa_{(i,j)}$ is the loss based on the relationships of point pair and it is defined as:

$$\kappa_{(i,j)} = \begin{cases} \max(0, d_{F(i,j)} - \epsilon_1) & \text{if } p_i \text{ and } p_j \text{ in the same instance} \\ \max(0, \epsilon_2 - d_{F(i,j)}) & \text{otherwise} \end{cases} \tag{2.9}$$

where ϵ_1, ϵ_2 are constants and set to satisfy the condition $0 < \epsilon_1 < \epsilon_2$, because the feature distance of point pairs in different instances should be greater than those belonging to the same instance [7]. We do not need to make the feature distance for point pairs in the same instance close to zero but smaller than the threshold ϵ_1, which is helpful for learning [16].

Center score loss

Smooth $L1$ loss is used as a loss function for the center score branch because of robustness of L1 loss function [4]. The center score loss L_{CS} is defined by

$$L_{\text{CS}} = \frac{1}{N} \sum_{i}^{N} \text{smooth}_{L1}(s_{\text{center}(i)} - \widehat{s}_{\text{center}(i)}), \tag{2.10}$$

where $\widehat{s}_{\text{center}(i)}$ represents the predicted center score and

$$\text{smooth}_{L1}(x) = \begin{cases} 0.5\,|x|^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \tag{2.11}$$

2.3.3 3D Point Cloud Instance Segmentation based on Deep Neural Network

As described in the previous section, two branches of FPCC-Net output the embedded features and the center scores of each point. Non-maximum suppression is performed on all points with center scores to find the centers of each instance. The points with a center score higher than 0.6 are considered as candidates of the center points. The point with the highest center score is selected as a first candidate of the center point, and all the other points located in the sphere with the center being the candidate point and radius $\gamma d_{\max}$ are removed, where $d_{\max}$ is the maximum distance from the geometric center to the farthest point of the object, and γ is the screening factor. This process is repeated until there are no more points left. The detailed processes of selecting the center points are presented in Algorithm 1.

Algorithm 1: Non-maximum suppression algorithm on points; N is the number of points; K is the number of center points.

Input: Threshold for center score θ_{th};
Screening radius $\gamma d_{\max}$;
Set of points $\mathbb{P} = \{p_1, p_2, ..., p_N\}$;
Corresponding predicted center scores of points $\mathbb{S} = \{s_1, s_2, ..., s_N\}$

Output: Center points $\mathbb{C} = \{c_1, c_2, ..., c_K\}$

```
for i = 1 to N do
    if s_i ≤ θth then
        ℙ ← ℙ\{p_i};
        𝕊 ← 𝕊\{s_i};
    end
end
ℂ ← {};
while ℙ ≠ ∅ do
    m* ← arg max_m{s_m | s_m ∈ 𝕊};
    ℂ ← p_m*;
    ℙ ← ℙ\{p_m*};
    𝕊 ← 𝕊\{s_m*};
    for p_i in ℙ do
        if d(p_m*, p_i) ≤ γd_max then
            ℙ ← ℙ\{p_i};
            𝕊 ← 𝕊\{s_i};
        end
    end
end
return ℂ;
```

After the above process, the feature distances between the center points and the other points are computed by

$$d(e_F^{(i)}, e_F^{(k)}) = \|e_F^{(i)} - e_F^{(k)}\|_2, \tag{2.12}$$

where $e_F^{(k)}$ represents the feature of k-th center point selected by Algorithm 1, and $e_F^{(i)}$ represents the feature of i-th point in the remaining points. All points except the center

points are clustered with the nearest center point in terms of the feature distance. Note that, we find the nearest center point c_k of point p_i in the feature embedding space, and then calculate Euclidean distance $d(p_i, c_k)$ between p_i and c_k in 3D space. If $d(p_i, c_k)$ exceeds $d_{\max}$, p_i is regarded as noise, and an instance label will not be assigned to p_i.

It should be emphasized that our clustering method is novel. The conventional clustering method [7, 84] is to downsample the points of the whole scene into multiple batches. The points of each batch are clustered, and then the points of all batches are integrated. In contrast, we perform clustering after all the points have been feed into the network in batches. After the feature and center score of each point are obtained, predicted instances are directly generated based on the feature distance from the center points without any merge or integration steps. In this way, we reduce the calculation time and reduce the accumulated errors due to the merged process.

2.4 Experiment

2.4.1 Dataset

We test five types of industrial objects, as shown in Figure 2.6. Ring screw and gear shaft are from the Fraunhofer IPA Bin-Picking dataset [13] and object A, B and C are from XA Bin-Picking dataset [16]. The details of datasets are as follows:

- Fraunhofer IPA Bin-Picking dataset (IPA) [13]: This is the first public dataset for 6D object pose estimation and instance segmentation for bin-picking that contains enough annotated data for learning-based methods. The dataset consists of both synthetic and real-world scenes. Depth images, 3D point cloud, 6D pose annotation of each object, visibility score, and a segmentation mask for each object are provided in both synthetic and real-world scenes. The dataset contains ten different objects. The training scenes of all objects are synthetic, and only the test scenes of gear shaft and ring screw are real-world data.

- XA Bin-Picking dataset (XA) [16]: Y. Xu and S. Arai et al. have developed a dataset of boundary 3D point cloud containing three types of industrial objects as shown in Figure 2.6 for instance segmentation on bin-picking scene. The training dataset is generated by simulation, while the test dataset is collected from the real-world. There are 500 training scenes and 20 test scenes for each object. We supplement the ground truth of object B and C in the test dataset. Each test scene contains about 60,000 boundary points. The examples of synthetic scenes are presented in Figure 2.7, respectively.

2.4.2 Experimental setting

FPCC-Net is implemented in the TensorFlow framework and trained using the Adam [117] optimizer with initial learning rate of 0.0001, batch size 2 and momentum 0.9. All training and validation are conducted on Nvidia GTX1080 GPU and Intel Core i7 8700K CPU with 32 GB RAM. During the training phase, $\epsilon_1 = 0.5$, $\epsilon_2 = 1$, $\alpha = 3$ and $\beta = 2$ are set. $\gamma = 1$ and $d_{\max} = 0.07, 0.08, 0.6, 1.6, 1.2$ correspond to the size of ring screw, gear shaft, object A, object B and object C, respectively. In each batch in the training process, input points ($N = 4,096$) are randomly sampled from each scene and each point can be sampled only

SD Mask FPCC Ground Truth

Figure 2.9: Comparison results of FPCC and SD Mask on IPA dataset. SD Mask is performed on depth images.

once. Each point is converted to a 6D vector $(\overline{x}, \overline{y}, \overline{z}, n_x, n_y, n_z)$ for inputting FPCC-Net. The sampling is repeated until the remained points of the scene is less than N. The network is trained for 30 epochs. It takes around ten hours to train FPCC-Net for each object.

2.4.3 Performance Evaluation of the Instance Segmentation

FPCC vs 2D instance segmentation SD Mask

Table 2.1: Results of instance segmentation on IPA. The metric is AP(%) with an IoU threshold of 0.5.

Data / Method	IPA	
	Ring Screw	**Gear Shaft**
SD Mask	22.1	21.0
FPCC	63.2	60.9

Figure 2.9 shows the comparison results of instances segmentation with SD Mask, and FPCC. Different predicted instances are shown in different colors. SD Mask misses many instances. In contrast, FPCC is robust to occlusion.

Table 2.1 reports the AP with an IoU threshold of 0.5 on gear shaft and ring screw. Scannet Evaluation [109] is adopted to compute AP. Since the depth images of the scene are not provided by XA, SD Mask cannot be performed on their scenes. In conclusion, FPCC improves AP by about 40 points than SD Mask. We argue that the typical 2D convolution kernel operates on a depth image with local information of each pixel coming

Table 2.2: Results of instance segmentation on industrial objects. The metric is precision(%) and recall(%) with an IoU threshold of 0.5.

Method \ Data	Backbone	Ring Screw		Gear Shaft		Object A		Object B		Object C	
		Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
SGPN	DGCNN	10.92	14.67	15.25	21.65	41.92	28.98	20.07	25.92	24.62	25.39
ASIS	DGCNN	15.54	11.56	20.14	9.46	64.67	28.72	55.43	23.61	67.08	42.06
3D-BoNet	DGCNN	27.88	19.80	26.57	20.11	66.05	50.23	42.22	26.38	45.88	62.40
PointGroup	DGCNN	3.64	1.91	3.65	1.73	6.19	1.54	7.68	1.89	5.37	1.02
PointGroup	Sparse Convolution	52.40	41.22	**58.79**	36.80	**91.88**	46.22	75.22	39.35	**79.83**	41.12
FPCC	DGCNN	**58.43**	**48.74**	54.29	**69.53**	89.71	**67.28**	**80.88**	**50.92**	78.64	**64.28**

from the neighboring pixel and ignoring the actual structure in 3D space. Although the current segmentation approach with input as depth image is faster than that with point cloud as input, it is not as effective and reasonable as the direct 3D instance segmentation approach.

FPCC vs other 3D Instance Segmentation Methods

Figure 2.10 shows results of instances segmentation and center scores predicted by FPCC-Net on the five types of industrial objects. The points near the center have higher score than boundary points. FPCC can distinguish the majority of instances clearly even with heavy occlusion. For ring-shaped objects, i.e., ring screw, the geometric center point is not on the part, thus the reference point used in the clustering is shifted from the center of object. The shifted reference point reduces the performance of segmentation. We consider that our clustering method is not suitable for objects that are mostly empty in the center, e.g., ring.

Figure 2.8 shows the comparison results of instances segmentation with SGPN, ASIS, 3D-BoNet, PointGroup and FPCC. Since the objects in a bin are identical, We eliminate the branches of semantic segmentation in these compared networks and treat the semantic information of all input points as the same. It should be noted that the original SGPN, ASIS and 3D-BoNet use PointNet or PointNet++ as their feature extractors. When we re-evaluate their method in the point cloud without color, we found that the training process could not converge at all. We believe that it is because PointNet and PointNet++ cannot effectively learn local geometric feature. Therefore, we replaced their feature extractor with DGCNN. For PointGroup, we provide two results, one is the original PointGroup, and the other is the PointGroup using DGCNN as backbone. Different predicted instances are shown in different colors. SGPN and ASIS adopt a similar clustering method, which accumulate the error of each potential group in the process of merging and results in predicting multiple instances as one instance in the heavy occlusion. In contrast, FPCC is robust to occlusion because no redundant merging is required. The performance of 3D-BoNet is also poor. This is because the 3D-BoNet trained by synthetic data is difficult to reasonably predict the 3D binding box of each instance on real data. The performance of PointGroup using DGCNN as the backbone is very poor, which shows that the features extracted by DGCNN can not support the clustering algorithm of PointGroup.

Table 2.2 reports the classical metrics precision and recall with an IoU threshold of 0.5 on ring screw, gear shaft, object A, object B and object C. In the case of using DGCNN as the backbone, FPCC achieves the best results and is competitive with the results of the original PointGroup. Original PointGroup uses sparse convolution as its backbone, extracting more discriminative features of the points to support its clustering algorithm. The recall of PointGroup is lower because PointGroup ignores some uncertain

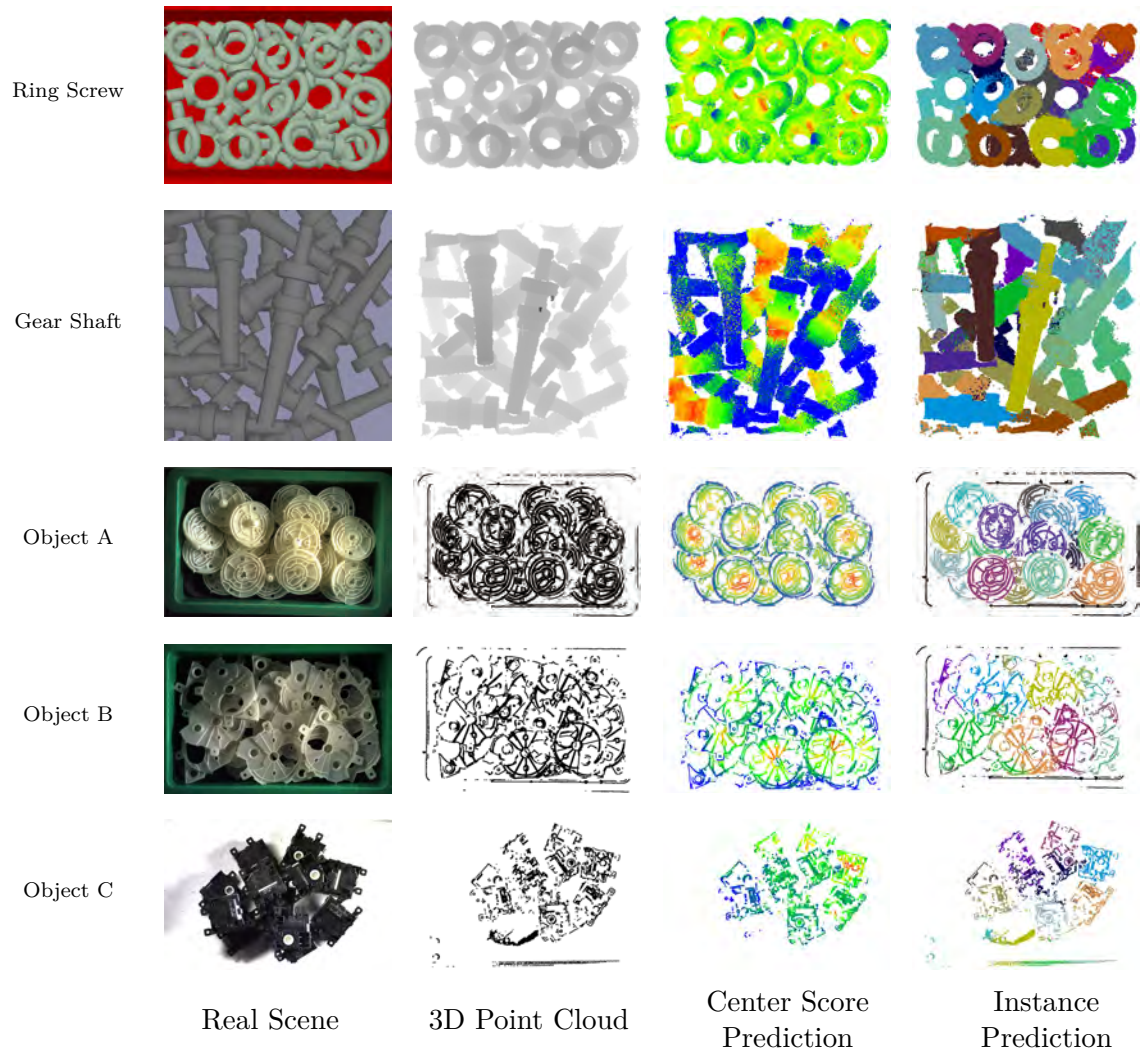


Figure 2.10: Visualization of the results of instance segmentation given by FPCC on IPA [13] (Ring Screw, Gear Shaft) and XA [16] (Object A, B, C). Many objects of the same class are stacked together in the Real Scene. 3D Point Cloud is represented by $(\overline{x}, \overline{y}, \overline{z}, n_x, n_y, n_z)$ and then input into FPCC-Net. Center Score Prediction is the predicted center score and the color bar is the same as one in Figure 2.5. The results for instance segmentation is shown in the last column. Different colors represent different instances.

Table 2.3: Ablation results for non-maximum suppression with different γ on IPA dataset

Data / γ	Ring Screw		Gear Shaft	
	Precision	**Recall**	**Precision**	**Recall**
0.5	27.52	45.21	63.56	38.12
0.8	65.70	54.64	66.35	63.71
1.0	58.43	48.74	54.29	69.53
1.2	10.86	25.66	28.65	12.19

segmentation results based on its confidence scores.

2.4.4 Ablation studies

Ablation on the $d_{\max}$

We use different screening factor γ in non-maximum suppression. The performance is shown in Table 2.3. The larger γ tends to cluster different objects together, while the smaller γ tends to produce over-segmentation.

Ablation on VDM and ASM

Four ablation experiments are conducted on the bin-picking scenes to evaluate the effectiveness of VDM and ASM in FPCC-Net. Precision and recall with an IoU threshold of 0.5 is added to interpret the results. Four groups for this ablation experiments are explained below:

1. VDM and ASM are removed, that is to say, no weights are added to compute the embedded feature loss L_{EF}.
2. Only VDM is used in loss L_{EF}.
3. Only ASM is used in loss L_{EF}.
4. Both VDM and ASM are adopted in loss L_{EF}.

Table 2.4 shows that the performance of the first group is the worst among the four experiments and two weight matrices improve the ability of the network to extract distinctive features of the 3D point cloud. The result indicates that the two weight matrices we designed are reasonable. VDM makes FPCC-Net do not need to care about the feature distance of points with too large Euclidean distance. ASM reduce the network's focus on point pairs whose two points are boundary points.

Processing Time

Table 2.5 reports the average computation time per scene measured on Intel Core i7 8700K CPU and Nvidia GTX1080 GPU. Each bin-picking scene of XA (Object A, B, and C) contains about 20 objects with 60,000 points. It takes around $0.6 \sim 0.8$ [s] to process one scene of XA by FPCC. A scene of IPA contains about $10 \sim 30$ objects with 15,000 points, which could also be processed in about 1.5 [s] by FPCC. PointGroup has faster processing speed with the support of sparse convolution. In summary, FPCC is about 60 times faster

Table 2.4: Results of ablation experiments. The metric is precision and recall with an IoU threshold of 0.5.

	VDM	ASM	**Ring Screw**		**Gear Shaft**	
			Precision	**Recall**	**Precision**	**Recall**
1			52.08	40.54	40.90	23.74
2	✓		54.55	45.86	41.92	28.12
3		✓	54.46	45.69	44.85	48.43
4	✓	✓	**58.43**	**48.74**	**54.29**	**69.53**

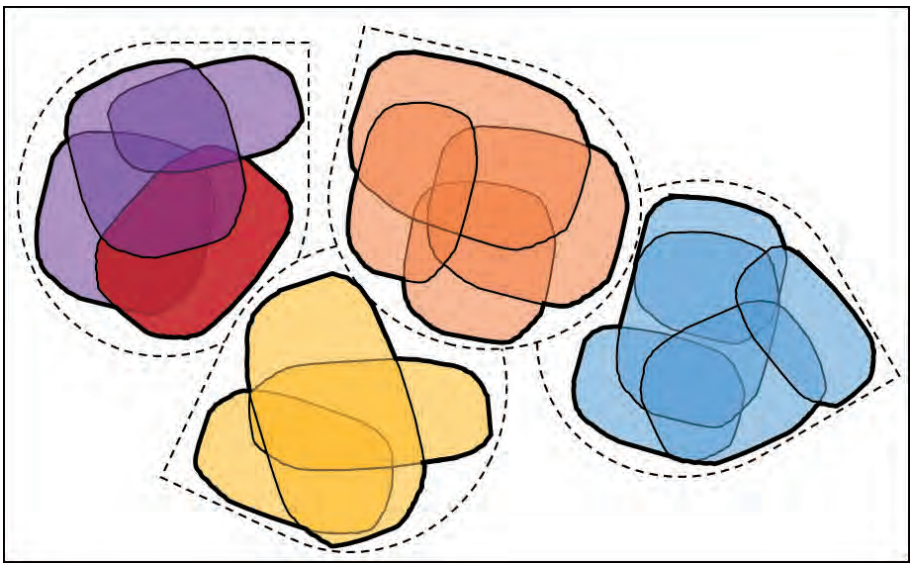


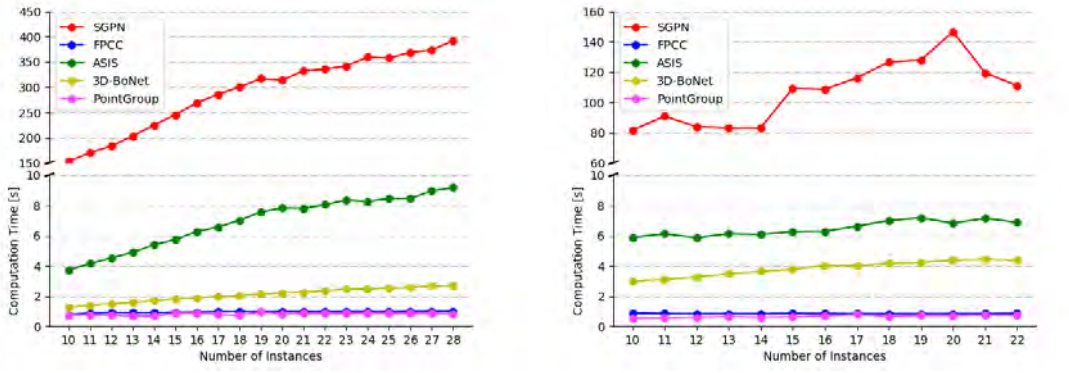
Figure 2.11: Illustration of merging process in SGPN. Dotted lines indicate instance and thin solid lines represent potential groups. Groups with an IoU greater than a threshold are merged and consist a new group represented by the thick solid line. Red group will be merged with purple group, since the IoU between red and purple groups is greater than the threshold.

than SGPN and 2 times faster than ASIS and 3D-BoNet, but slower than PointGroup with sparse convolution. Most of the existing methods employing a clustering strategies similar to SGPN find some reference points to generate potential groups, and then merge each group according to some metrices, e.g., IoU. The computation complexities for clustering processes of SGPN and FPCC are analyzed as follows: Firstly, we assume that there are m instances and n points in the scene ($m \ll n$), and the outputs of clustering algorithms of SGPN and FPCC are correct. The clustering of SGPN is divided into two steps: potential groups generating and groups merging. Firstly, SGPN takes $N_{\text{SGPN}} \gg m$ points with high confidence as reference points of the clustering. Then N_{SGPN} groups are generated based on the feature distance between the reference points and the other points. The computational complexity of groups generating, that is, the number of computation of the feature distances tends towards $\mathcal{O}(nN_{\text{SGPN}})$. Next, two groups with an IoU greater than a threshold, such as 0.5, are merged together, as shown in Figure 2.11. The computation complexity of groups merging for SGPN, that is, the number of computation of IoUs between two potential groups tends towards $\mathcal{O}(mN_{\text{SGPN}})$. In contrast, FPCC does not

Table 2.5: Computation speed comparisons [s/scene]

Method \ Data	Backbone	Ring Screw	Gear Shaft	Object A	Object B	Object C
SGPN	DGCNN	298.60	104.90	53.32	72.83	28.21
ASIS	DGCNN	8.37	7.45	3.62	3.09	1.78
3D-BoNet	DGCNN	3.91	3.80	1.29	0.91	0.80
PointGroup	DGCNN	2.16	1.62	1.26	0.78	1.20
PointGroup	Spase Convolution	0.81	0.64	0.39	0.47	0.36
FPCC	DGCNN	1.78	1.43	0.79	0.55	0.65

generate any potential group. The reference points of FPCC are found by Algorithm 1. Each point in the scene point cloud is directly clustered with the nearest reference point, that is, the center point based on feature distance. Hence the competition complexity of clustering for FPCC tends towards $\mathcal{O}(nm)$. In summary, the computational complexities of groups generating of FPCC is much lower than SGPN, that is $\mathcal{O}(nm) \leq \mathcal{O}(nN_{\text{SGPN}})$, since the number of potential groups N_{SGPN} is much higher than the number of instances m. In addition, the calculation of IoU between groups which is not required for FPCC needs more computational resources than that of feature distances. Thus we can conclude that the computational cost of FPCC is much smaller than that of SGPN theoretically. An example of the relationship of computation time and the number of instances is shown in Figure 2.12. ASIS adopted a clustering method similar to SGPN, and their calculation time increases significantly with the increase of the number of instances. 3D-BoNet also need more calculation time because of increase of bounding boxes. However, the computation time of FPCC and PointGroup are not sensitive to the number of instances.



(a) Computation time on ring

(b) Computation time on gear

Figure 2.12: Average computation time for different number of instances. As the number of instances increases, the computation time of SGPN and ASIS increase significantly.

2.5 Discussion

This chapter proposes a fast and effective 3D point cloud instance segmentation named FPCC for the bin-picking scene, which has multi instances but a single class. FPCC includes FPCC-Net which predicts embedded features and the geometric center score of

each point, and a fast clustering algorithm using the outputs of FPCC-Net. Two hand-designed weight matrices are introduced for improving the performance of FPCC-Net. A novel clustering algorithm is proposed for instance segmentation. Although still slower than typical 2D CNN based on depth images, the results are far superior to the 2D scheme. For multi instances but single class scenes, FPCC achieves better performance than existing methods even without manually labeled data. Besides, we theoretically prove that the computational complexity of FPCC is much lower than SGPN.

This study also has a certain limitation that must be addressed in future work, as follows: 1) FPCC is not particularly suitable for objects whose geometric center is not on the itself. We are considering using other methods to define the reference point of each instance. 2) We use the existing semantic network, i.e., DGCNN, as FPCC's feature extractor. Perhaps the feature extractor does not fit the task of instance segmentation perfectly. The improvement of feature extractor is our future work.

2.6 Conclusion

The chapter developed a framework that enables for fast instance segmentation of point clouds, with the following conclusions:

- the precision and speed of our 3D point cloud instance segmentation for robotic bin-picking scenes reach the current state-of-the-art;
- Experimental results show that it is feasible to train the network with synthetic data;
- The backbone, DGCNN, limits the number of points that the network can process per frame.

Chapter 3

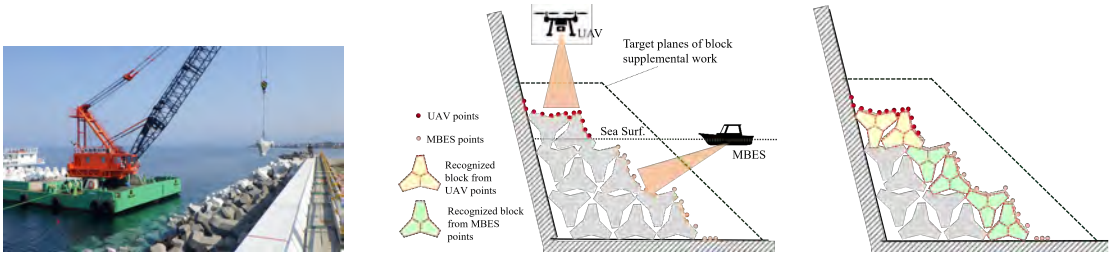
Deep Learning-based Object Instance Segmentation and Pose Estimation from Point Clouds of Stacked Wave-Dissipating Blocks

3.1 Background and Objectives of this Chapter

Breakwaters protect beaches, ports, and harbors from erosion caused by waves. Wave-dissipating blocks made of large concrete slabs are essential components of the armor layer of breakwaters that protect their core from direct wave attacks. However, long-term wave motion and erosion damage the blocks irreversibly, causing them to sink and even break off [21]. Therefore, periodical supplemental work must be conducted on wave-dissipating blocks to maintain breakwaters. Therein, new wave-dissipating blocks are stacked onto the existing ones until their top surfaces exceed the target height, as shown in Figure 3.1(a). Thus, precisely estimating the current block-stacking status is imperative for monitoring long-term block movements in the maintenance and planning of lean supplemental work.

With the recent increased availability of various 3D sensors, such as airborne LiDAR, and UAV-photogrammetry, dense three-dimensional (3D) point clouds of existing offshore object surfaces can easily be measured in a low-cost manner [22]. On the other hand, multibeam echo-sounder (MBES) is enabling to capture a large-scale 3D point cloud of the bottom of water and realizes detailed exploration of undersea objects [118, 119]. Moreover, drone-mounted RGB and multispectral imagery [120], drone-mounted lightweight dual-wavelength LiDAR systems [121] are enabling shallow bathymetric mapping capabilities and undersea object detections. The effectiveness of data integration of bathymetric mapping using single beam echo sounder with ground surface mapping using terrestrial LiDAR and UAV-photogrammetry has also been demonstrated [122].

From the viewpoint of practicality and ease of use among above 3D sensing methods, as shown in Figure 3.1(b), the emergent areas of wave-dissipating blocks can be measured easily by UAV-photogrammetry, while MBES can measure the submerged portion. Suppose that individual block poses can be detected from the measured point clouds of existing block surfaces, as shown in Figure 3.1(c). This allows for the evaluation of more precise overseas and undersea block-wise stacking status required for long-term maintenance and



(a) Block supplemental work (b) 3D measurement of existing block surface (c) Individual block pose detection

Figure 3.1: Supplemental work on wave-dissipating blocks. Periodical repair work aims at stacking new supplemental blocks on top of existing ones until the target height.

a lean supplemental plan.

Accurate monitoring of the poses of individual wave-dissipating blocks offer the following benefits for the administrators:

- 6D poses of individual blocks faithfully reproduce the as-built status of each block to improve the accuracy of estimates of new blocks' quantities and their stacking plan in the supplemental work;
- The as-built status can be grasped block by block after the construction and supplemental works. Thus, the construction results can be recorded and visualized comprehensively compared to recording only the measured point clouds of existing block surfaces;
- By providing the pose and attribute information to each block model, it is possible to check the long-term change in the blocks, such as missing, sinking, and damaged blocks, and implement a more precise and sustainable maintenance activity.

To date, several studies compared the dense 3D point clouds acquired at different times [22–27] to evaluate possible changes in stacked block surfaces within a certain period. However, these studies do not provide sufficient information for individual block movements or block breakage over a long time interval or after severe events [26] that could significantly affect the structural integrity of breakwaters. Due to the technical difficulty, very few studies attempted to detect individual wave-dissipating block poses from 3D dense point clouds. For example, Bueno et al. [21] presented an algorithm for reconstructing cube-shaped wave-dissipating blocks from incomplete point clouds captured by airborne LiDAR. Shen et al. [28] estimated individual brick poses from laser-scanned point clouds that measured a cluttered pile of cuboid bricks. Although the above methods can recognize blocks with simple shapes, it is difficult for them to identify those with complex shapes, like tetrapods or clinger blocks.

To address this issue, a novel deep-learning-based 3D pose detection method of wave-dissipating blocks from as-built point clouds measured by UAV-photogrammetry and MBES with reasonable detection performances is designed and proposed. The method can detect as many blocks as possible all at once from a given scene. The proposed pose detection method enables 6D pose estimation of blocks and block type classification.

To realize our detection method, a category-agnostic instance segmentation network called Fast Point Cloud Clustering v2 (FPCCv2) was designed initially to segment a point

cloud corresponding to a block instance from point clouds measured from large areas by UAV and MBES. FPCCv2 is an instance segmentation network for point clouds extended from our previous segmentation network Fast Point Cloud Clustering (FPCC) [123] with a novel feature extractor. This feature extractor increases each point's receptive field to obtain more discriminative features for the instance segmentation. FPCCv2 is trained on synthetic block-stacking scenes constructed from block CAD models and applied to the instance segmentation of the blocks of real scenes. The 6D pose of the segmented block is estimated using a 3D feature descriptor Point Pair Feature (PPF) [124] and refined by best-fit point cloud alignment, called Iterative Closest Points (ICP) [125]. Finally, a fitness score is used to distinguish each block type in a scene including different types of blocks.

The proposed block detection method is validated on three sites of ports, consisting of different block types. The instance-labeled training dataset used to supervise the network is automatically generated by a physics engine [126] and block CAD models, which avoids laborious manual labeling work on the real scene and secures rich training datasets for our network. Different synthetic point clouds are generated according to the measurement properties of UAV or MBES to bridge the gap between synthetic and real scenes. The combination of PPF and ICP enables the precise estimation of 6D poses of individual blocks in a scene. Moreover, the each block type can be identified from a point cloud scene.

In summary, the original contributions of the proposed method are described as follows:

- The originally proposed convolutional-neural network called FPCCv2 enables rapid segmentation of individual wave-dissipating block instances from a large-scale 3D measured point cloud captured from a scene of stacked blocks. This enables us to estimate 6D poses of multiple blocks at once and improve computational efficiency of the block pose estimation.

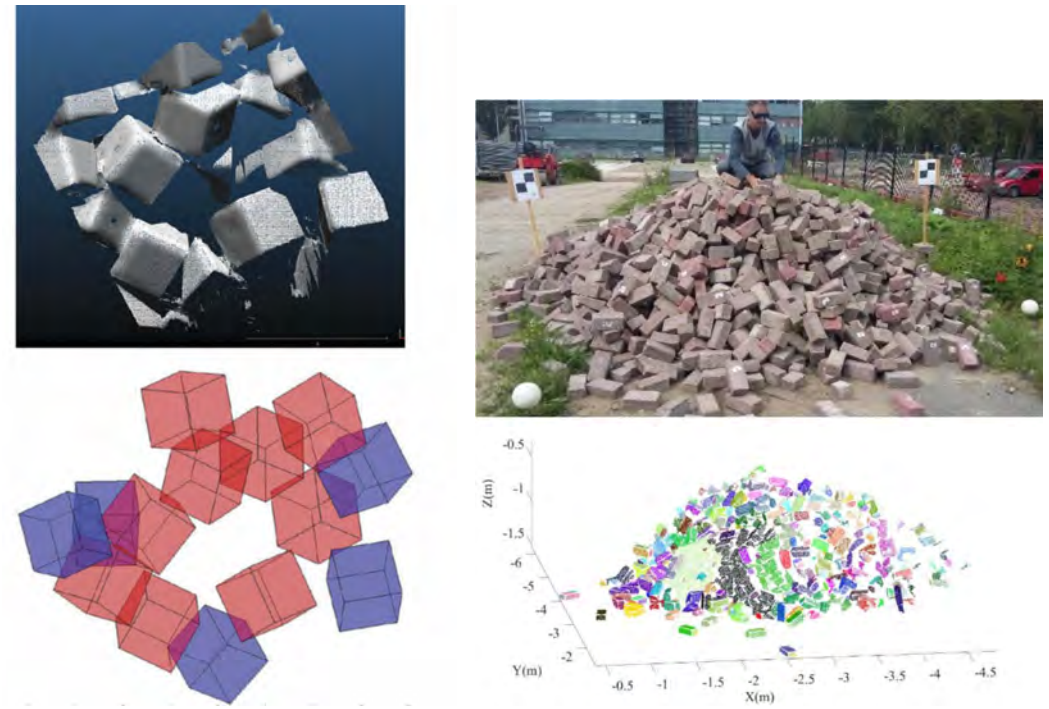
- A physics engine enables synthetic and automatic generation of instance-labeled training datasets for the instance segmentation of blocks. It thus avoids laborious manual labeling work and secures rich training datasets for our convolutional-neural network.

- Synthetic point cloud generation considering the difference in characteristics of measurement using UAV and MBES enhances the performance of instance segmentation.

- The combination of the 3D feature descriptor by PPF and point-to-point registration by ICP enables precise estimation of 6D poses of individual blocks in a scene. Moreover, the difference in the type and size of individual blocks can be identified in a scene. This is useful for the as-built inspection and instance-level monitoring of wave-dissipating blocks.

- The performances of 6D pose estimation of individual wave-dissipating blocks are evaluated both in synthetic scenes and various real construction sites including undersea blocks.

The remainder of this chapter is organized as follows. Section 3.2 reviews related work and clarifies the issues encountered. Section 3.3 introduces an overview of the proposed pose detection method. Section 3.4 describes our convolutional-neural network used for the instance segmentation of measured point clouds. Section 3.5 presents the details of



(a) Detection of cube-shaped blocks [127] (b) Detection of rectangular bricks [28]

Figure 3.2: Detection of the wave-dissipating blocks from large-scale point clouds.

block pose estimation and block type classification. Section 3.6 provides experimental results about detection performances using real and synthesized scenes. Finally, Section 3.8 presents the conclusion and future directions for research.

3.2 Related Work

From a geometric processing viewpoint, this study employs three techniques: (1) 3D monitoring of wave-dissipating blocks in breakwaters, mainly used in civil engineering, (2) instance segmentation on point clouds, and (3) 3D object detection and 6D pose estimation on point clouds, mainly used in computer vision. In this section, the related work on these techniques is introduced, and the challenges faced by these techniques when applied to detect the 3D poses of the wave-dissipating blocks from large-scale point clouds are elucidated as shown in Figure.

3.2.1 3D monitoring of Wave-Dissipating Blocks in Breakwaters

As stated in the introduction, wave-dissipating blocks of breakwaters require periodic monitoring and supplemental work to ensure that they remain in good condition [22, 23, 128–130]. To this end, in recent years, several studies have been reported on the use of photogrammetry and UAVs to monitor the 3D condition of blocks of breakwaters.

Sousa et al. [23] developed a data acquisition system to capture 3D point clouds of breakwaters using UAV-based photogrammetry. Lemos et al. [131] used a terrestrial laser scanner, photogrammetry, and a consumer-grade RGBD sensor to determine the rocking

and displaced wave-dissipating blocks in the breakwater structure. These studies monitor the block surfaces' overall condition by comparing changes in data from different periods. However, they could not assess the state nor estimate the pose of an individual block.

In contrast, methods have gradually been proposed to investigate the condition of individual blocks in breakwaters and estimate their posture from 3D point clouds obtained by laser scanners or photogrammetry. González-Jorge et al. [22] evaluated the measurement accuracy of 3D point clouds obtained by photogrammetry and clarified the detection limit of small translations and rotations on a flat surface of one cube armor, manually segmented from a point cloud. Shen et al. [28] proposed an algorithm to extract individual rectangular bricks from a dense point cloud sampled from an unorganized pile of bricks using a terrestrial laser scanner. Bueno et al. [127] developed a method for the automatic modeling of breakwaters with cube-shaped block armors from a 3D point cloud captured by a terrestrial laser scanner. Unfortunately, the detection methods proposed in [22, 28, 127] are only applicable to a simple block, whose shape is bounded by planes. This cannot be easily extended to the recognition of blocks like tetrapods or accropodes, whose shape is composed of complex surfaces. Moreover, low recall and high computational overhead reduce the practicality of these methods.

Recently, Musumeci et al. [27] used a consumer-grade RBGD sensor to measure the damage in a laboratory-scale model of stacked accropode blocks around a rubble mound breakwater simultaneously above and below the water level. Although they quantified the distribution of rotational and translational shifts between the point clouds captured at different times, they evaluated them only as a directional and positional shift at the centroid of a local point cloud in a small cube, regularly partitioned from the original point cloud. Therefore, their method could not detect individual wave-dissipating block poses from 3D dense point clouds nor quantify the individual block movements or breakage over time.

Compared with the above methods, our method can monitor the wave-dissipating block at instance level within an acceptable computation time. The high recall and low computational consumption broaden the deployment of our approach in real-world scenarios. In addition, our method is robust to various wave-dissipating blocks with complex-shape.

3.2.2 Instance Segmentation on Point Cloud

Detecting point subsets that correspond to an individual wave-dissipating block from an original point cloud can be regarded as a 3D instance segmentation problem. The 3D instance segmentation algorithm aims at assigning a label to each point in an original point cloud, distinguishing different instances of the same class.

Most deep learning-based 3D instance segmentation frameworks recently focused on indoor data and exhibited remarkable performances. For example, some grouping-based methods [7, 16, 67, 84, 110] cluster points in a high-dimensional space based on their similarity of features, while others [85, 86] cluster points in Euclidean space after moving points toward their corresponding instance centers based on the predicted offset vector. These methods [7, 16, 67, 84–86, 110] used publicly-available labeled point cloud datasets [76, 77, 109] as training sets developed for indoor scene segmentation. The object surfaces in these high-quality indoor point cloud scenes [76, 77, 109] are densely sampled with little occlusions and sometimes attached with additional information, such as RGB attributes.

However, the lower sides of actual wave-dissipating blocks stacked on coast sides cannot be measured, and missing portions of the point cloud remain, as only the upper sides of each block are measured. Therefore, the coverage of the measured points of the wave-dissipating blocks is incomplete and considerably different from the one of indoor objects. Thus, the learning-based 3D instance segmentations developed for indoor scenes [7, 16, 67, 84–86, 110] do not necessarily work well for the block instance segmentation of point clouds of stacked wave-dissipating blocks in the natural environment.

Recently, [132] proposed a method based on a deep neural network for instance segmentation of LiDAR point clouds of street-scale outdoor scenes into point cloud instances of pedestrians, roads, cars, bicycles, etc. The authors used not only the labeled point cloud of real scenes, but of simulated scenes as well. In [132], they combined the point clouds measured from real backgrounds with the foreground 3D object models to generate their training data. However, their foreground objects are positioned separately, not jumbled, and not overlapped like wave-dissipating blocks. Moreover, they manually cleaned up the background points to remove moving objects, such as cars. Therefore, it is difficult to directly apply their method of training dataset creation and instance segmentation to our instance segmentation problem of complex overlapping blocks.

There are instance segmentation methods specifically designed according to the complexity of outdoor scenes and according to characteristics of the objects of interest. Walicka et al. [133] designed a method for segmenting individual stone grains on a riverbed from a terrestrial laser-scanned point cloud. After the random forest algorithm separates the grain and the background, a density-based spatial clustering algorithm then segments the individual grains. Luo et al. [134] first employed a neural network to segment tree points from the raw urban point cloud and then classified the tree clusters into single and multiple tree clusters according to the number of detected tree centers. Djuricic et al. [135] developed an automated analysis method to detect and count individual oyster shells placed on a fossil oyster reef from a terrestrial laser-scanned point cloud based on the convex surface segmentation from the point cloud and the openness feature. However, these segmentation algorithms are designed to segment the instances of specific objects in outdoor point cloud scenes, and it is questionable whether they can be applied directly to segment the scenes of wave-dissipating blocks.

In summary, to the best of our knowledge, there are no currently available studies on the detection and recognition of complex-shaped wave-dissipating blocks from large-scale 3D point clouds.

3.2.3 Model-based 3D Object Detection and 6D pose estimation

Given the 3D reference model of a wave-dissipating block to be detected, the problem in our study can be regarded as a model-based 3D object detection and 6d pose estimation problem in point clouds. From a given point cloud scene, a local region on the point cloud, whose geometry is matched to a given reference model shape, must be extracted, and the position and orientation of the model must be identified.

Numerous descriptor-based methods were proposed for 3D object detection and 6D pose estimation in a point cloud scene. So far, they have demonstrated good performances. For example, SHOT [136] is a local feature-based descriptor that establishes a local coordinate system at a feature point and describes the feature point by combining the spatial location information of the neighboring points and the statistical information of geometric

features. The Point Feature Histogram (PFH) [137] uses the statistical distribution of the relationship between the pairs of points in the support area and the estimated surface normal to represent the geometric features. The PFH descriptor is calculated as a histogram of relationships between all point pairs in the neighborhood. These studies show that descriptor-based object recognition methods are highly generalizable and simultaneously perform the detection and 6D pose estimation tasks.

However, pre-processing is often required for scenes containing multiple objects to divide the scene into various regions of interest and then separately match the model and regions of interest [19, 116, 138, 139]. The naive segmentation tends to cause over- and under-segmentation, which seriously deteriorates the matching performance.

Nevertheless, some learning-based 6D pose estimation networks have also been proposed in recent years. PoseCNN [140] is an end-to-end convolutional-neural network that learns the 3D translation and rotation of objects from images. DenseFusion [141] merges the features of images and point clouds and estimates the pose from them. PPR-Net [50] and PPR-Net++ [142] regressed the 6D pose of the object instance to which each point belongs from the point cloud. However, in addition to the difficulty of learning 6D poses from images [143–145], producing a dataset for training also poses a substantial practical problem for a single institution or company. Images of outdoor rubble mound breakwaters can easily change with differences in light, weather, and season. PPR-Net [50] and PPR-net++ [142] estimate the 6D pose of an individual object from a cluttered point cloud scene; however, it is difficult to retrieve the 6D poses of heavily occluded objects.

Collectively, these learning-based methods cannot provide accurate poses that satisfy the monitoring of small displacements that occurred in wave-dissipating blocks, and it is difficult to retrieve the pose of the object with occlusion.

3.3 Overview of the Processing Pipeline

Figure. 3.3 illustrates a processing pipeline of 3D pose detection of individual wave-dissipating blocks from an input point cloud in our study.

- First, the category-agnostic instance segmentation network FPCCv2 segments the input point cloud measured by UAV and MBES into the subsets of points corresponding to individual block instances. FPCCv2 is a kind of deep neural network, which is pre-trained by synthetic point clouds that mimic the point clouds of stacked blocks measured by UAV and MBES, respectively, using the stacked block CAD models and surface point sampling. The detailed algorithms are described in Section 3.4.

- Second, the 6D pose of an individual block is estimated from each segmented point cloud using a conventional descriptor-based 3D object detection algorithm that makes use of PPF [124] and ICP [125]. The detailed algorithm is described in Section 3.5.1.

- Finally, if the scene consists of multiple typed blocks, a fitness score corresponding to each type is calculated for each segmented point cloud to identify the type of detected individual block. The detailed algorithm is shown in Section 3.5.2.

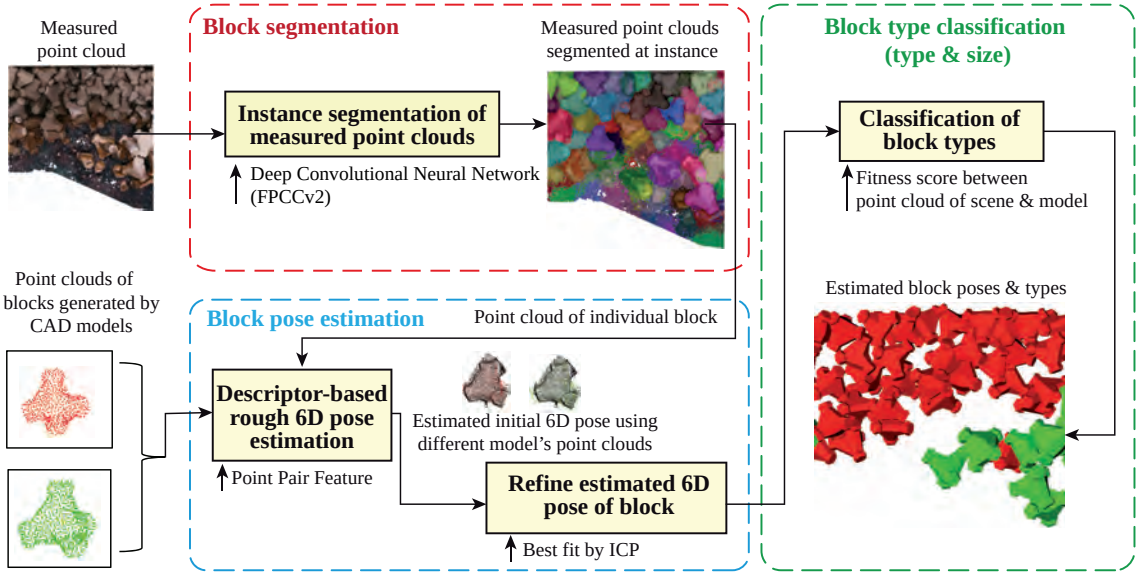


Figure 3.3: Pipeline of block detection. The algorithm consists of three steps for detecting the type and 6D pose of all block instances in the scene. The input is the point cloud of a scene measured by the UAV or MBES. The point cloud of an individual block is output by the convolutional network FPCCv2 (Block segmentation). Next, the 6D poses for multiple models were roughly estimated by PPF and refined by ICP (Block pose estimation). Finally, the block type is determined based on the fitness score (Block type classification).

3.4 3D Instance Segmentation for Wave-Dissipating Blocks based on Deep Neural Network

The first and crucial step of our wave-dissipating block pose detection is an instance segmentation of an original point cloud using a deep neural network. The category-independent instance segmentation network, called FPCCv2, segments the input point clouds measured by UAV and MBES into subsets of points corresponding to individual blocks. FPCCv2 is an extended version of our previous instance segmentation network FPCC [123] with a novel feature extractor. Therefore, this section first revisits FPCC, then explains why we must improve our previous FPCC to address the issue of instance segmentation of wave-dissipating blocks, and how we improved it to solve the issue.

3.4.1 FPCC and its limitation

FPCC was originally developed by Xu et al. [123] as a category-agnostic 3D instance segmentation network for discriminating each part in an industrial bin-picking scene in robotic automation, where the parts have an identical shape and overlap each other. FPCC extracts features of each point, while inferring the centroid of each instance. Subsequently, the remaining points are clustered to the closest centroids in the feature embedding space. It was shown that even FPCC trained by synthetic data performs at an acceptable level on real-world data [123].

As shown in Figure 3.4, FPCC is composed of a point-wise feature extractor and two

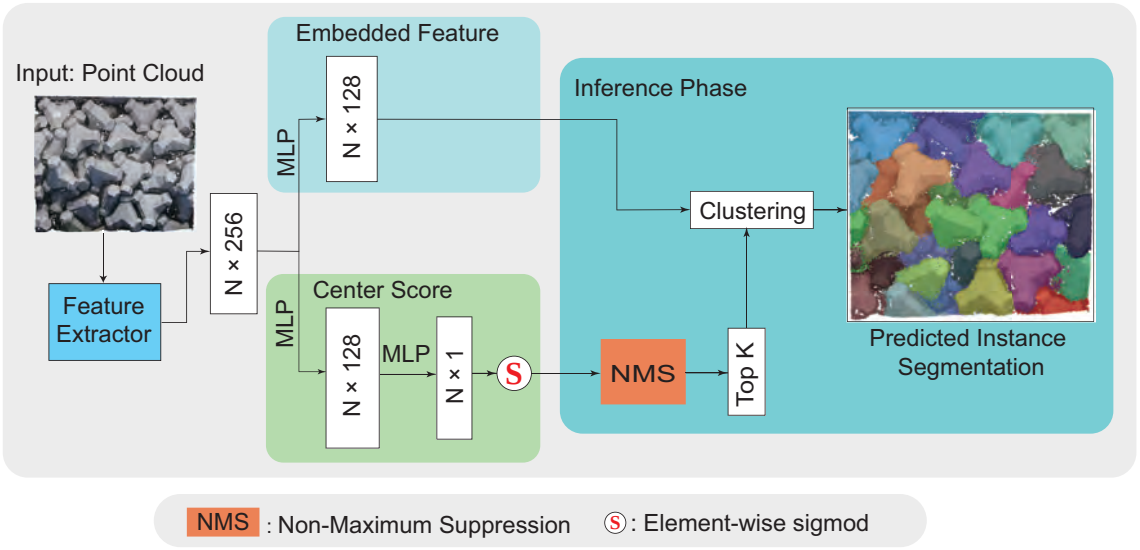


Figure 3.4: Network architecture of FPCC [123]. After the **Center Score** branch predicts the probability that each point is likely to be the centroid of the object, non-maximum suppression is used to select the most likely centroids as reference points for clustering. The reference and remaining points are clustered according to the L2 distance in feature space. Our improvement aimed for FPCCv2 is the feature extractor (Section 3.4.2), where the features of point clouds can be exploited more effectively than the original FPCC.

branches: an embedded feature branch and a center score branch. The point-wise feature extractor has the same semantic segmentation structure as DGCNN [68]. The extracted features are sent to the point-wise embedded feature branch and center score branch, respectively. The instance segmentation can be regarded as a kind of point cloud clustering, and the clustering method of FPCC assumes that the points in the same instance have similar features, while points in the different instances have relatively different features. In contrast, the center score branch predicts the probability of each point to be placed at the centroid. In the prediction phase, non-maximum suppression finds the point with the highest score for each target object as a reference point for clustering. Then, the distances between the remaining points and the reference point were calculated. Finally, the remaining points are clustered into the same instance with the center point by feature distance.

Although FPCC exhibits a promising instance segmentation performance in robotic bin-picking [16, 19, 51, 123], a computational and efficiency issue could arise when applying it to the instance segmentation of large-scale point clouds captured from stacked wave-dissipating blocks in real breakwaters.

FPCC made use of DGCNN [68] as the feature extractor. DGCNN employs the k-nearest neighbor (k-NN) algorithm to construct the graph of the point cloud in three-dimensional space and high-dimensional space, such that the topological structure of the graph of DGCNN is not static, but dynamically updated after each layer of the network. Dynamically updating the graph can heavily increase training and prediction time and limits the number of points that the network can process per frame under the same memory size. To encode the features of large-scale point clouds faster and better, a novel feature extractor is aided to the original FPCC. Details are described in the next section.

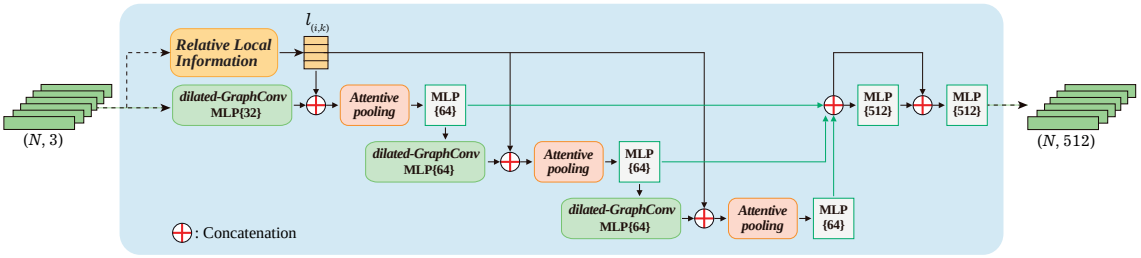


Figure 3.5: Feature extractor of FPCCv2 is built from a sequence of graph convolutions and MLPs. The high-dimensional (e.g., 512) features of each point are fed into the center score branch and the embedded feature branch of the original FPCC network, respectively.

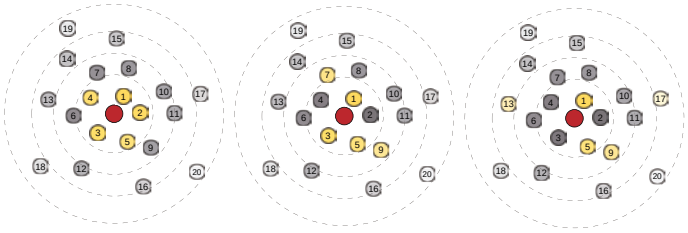


Figure 3.6: Nearest neighbor selection of dilated k-NN [146]. The number in the circle represents the distance from center point (red). k-NN, where $k = 5$, with different dilation rates 1, 2 and 4 (left to right) selecting different points (yellow) as relative points.

3.4.2 FPCCv2 with New Feature Extractor

To solve the issue pointed out in 3.4.1, as shown in Figure 3.5, the feature extractor of FPCCv2 consists of three parts: 1) dilated graph convolutions (GraphConv), 2) relative local information, and 3) attentive pooling.

Dilated Graph Convolution. Our goal was to give each point reasonable and sufficient information within an acceptable computational overhead. Previous studies [146–148] showed that the size of the receptive field is essential to the performance of a network. A larger receptive field can offer neural units more comprehensive and higher dimensional features. However, a too-large receptive field makes it more difficult for the network to learn high-frequency or local features [148]. Stacking convolutional layers or increasing the kernel size of convolution are common approaches to increase the receptive field.

Inspired by previous research [146, 148], the dilated local graph is defined by a dilated K-NN in Euclidean space. The d-dilated K-NN first finds $K \cdot d$ nearest neighbors for a point i, but only selects every d points from the nearest as the neighbor point set $N(i, K, d)$ (see Figure 3.6). Therefore, a few computational overheads are increased to expand the receptive field without increasing the number of model parameters. Three dilated graph convolutions were stacked in our feature extractor. The receptive field sizes with different k and d after each dilated graph convolution are shown in Figure 3.7. Then, the relative feature $r_{(i,k)}$ of a point i to point k in its corresponding d-dilated K-NN neighbor points $N(i, K, d)$ is defined as:

$$r_{(i,k)} = MLP(f_i - f_{(i,k)}), k \in N(i, K, d), \tag{3.1}$$

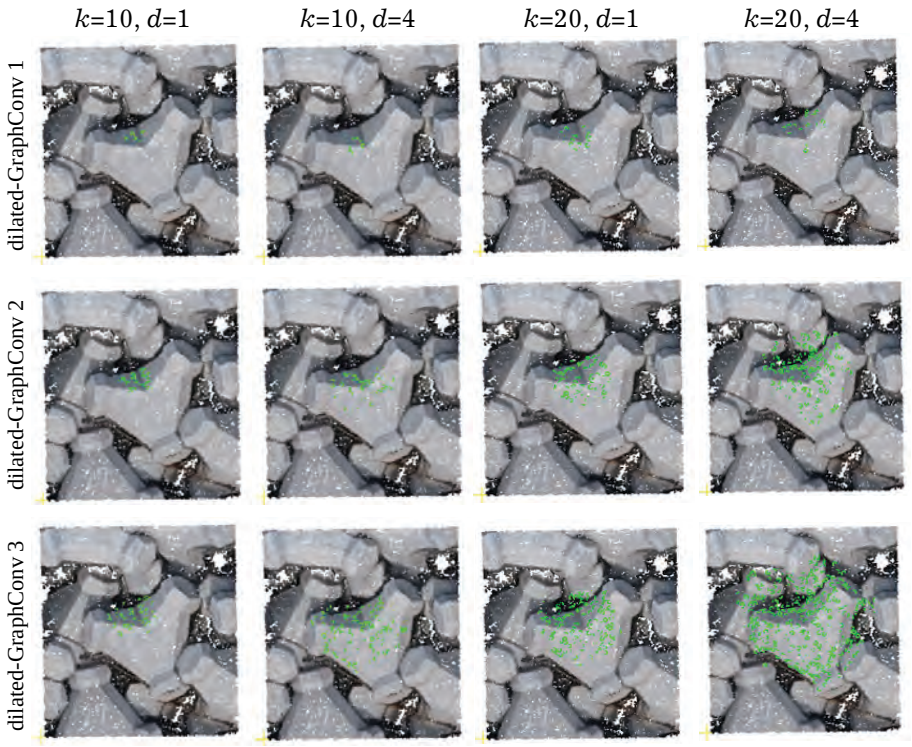


Figure 3.7: Visualization of the receptive field. The receptive field (green point) expands with deeper networks (rows). Increasing nearest neighbors k and dilation rate d can enhance the receptive field with less computational overhead. When $k = 20, d = 4$, the receptive field of the third layer can roughly cover a complete block.

where f_i and $f_{(i,k)}$ are respectively the feature of point i and k for $N(i, K, d)$. In the first graph convolution, f is selected as x-y-z coordinates of the point. MLP represents a multilayer perceptron. Figure 3.8 illustrates the process of the dilated graph convolution.

Relative Local Information. The relative local information of a point i with its d-dilated K neighbor points $N(i, K, d)$ is defined as:

$$l_{(i,k)} = [p_i \oplus p_{(i,k)} \oplus (p_i - p_{(i,k)}) \oplus \|p_i - p_{(i,k)}\|_2], k \in N(i, K, d), \tag{3.2}$$

where p_i and $p_{(i,k)}$ denote three-dimensional coordinates of point i and k for $N(i, K, d)$, $\oplus$ represents the concatenation operation, and $\|\cdot\|_2$ denotes Euclidean distance between the two points. The relative local information of each point is repeatedly introduced into the network to learn local features efficiently.

Attentive Pooling. Given a set of features $F = \{f_1, f_2, ..., f_m, ..., f_M\}$, where $f_m \in \mathbb{R}^{1\times D}$ and M denotes the size of pooling. Attentive pooling aims to integrate F into a single feature $f_{\text{out}} \in \mathbb{R}^{1\times D}$, while achieving permutation invariance of its elements. The existing methods [66,68,149] use simple max/mean pooling to address the permutation invariance, resulting in the loss of crucial information [10]. Influenced by recent studies [10, 150], we employed attentive pooling in aggregating relative features, as shown in Figure 3.9. The attentive pooling operation consisted of the following steps:

1) Association with location information. For each point i, the relative features $r_{(i,k)}$

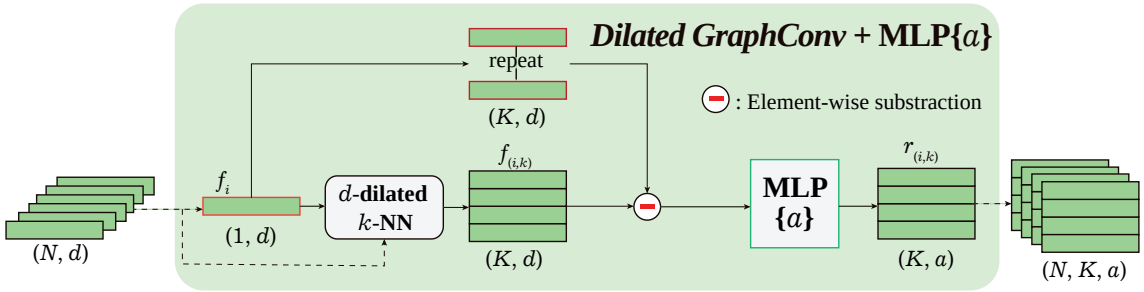


Figure 3.8: Dilated Graph Convolution. N points are input and K relative features of each point are output.

and relative local information $l_{(i,k)}$ of the d-dilated K-NN neighbor points were concatenated to generate a new feature $\tilde{f}_{(i,k)} = r_{(i,k)} \oplus l_{(i,k)}, k \in N(i, K, d)$. The new feature set $\tilde{F}$ is defined as $\tilde{F} = \{\tilde{f}_{(i,k)} \in \mathbb{R}^{1\times D} | k \in N(i, K, d)\}$.

2) The feature set $\tilde{F}$ is fed into a shared MLP to obtain initial attention scores $\mathcal{C} = \{\boldsymbol{c}_{(i,k)} | k \in N(i, K, d)\}$, where $\boldsymbol{c}_{(i,k)}$ has the same dimension as $\tilde{f}_{(i,k)}$, and is defined as:

$$\boldsymbol{c}_{(i,k)} = MLP(\tilde{f}_{(i,k)}). \tag{3.3}$$

3) A softmax normalizes the elements of $\mathcal{C}$ to obtain the attention score $\mathcal{S} = \{\boldsymbol{s}_{(i,k)} | k \in N(i, K, d)\}$. The attention score $\boldsymbol{s}_{(i,k)}$ is defined as:

$$\boldsymbol{s}_{(i,k)} = [s^1_{(i,k)}, s^2_{(i,k)}, ..., s^d_{(i,k)}, ..., s^D_{(i,k)}], \tag{3.4}$$

where

$$s^d_{(i,k)} = \frac{e^{c^d_{(i,k)}}}{\sum_{j=1}^{K} e^{c^d_{(i,j)}}}, \tag{3.5}$$

where $c^d_{(i,k)}$, $c^d_{(i,j)}$ are the d-the elements of $\boldsymbol{c}_{(i,k)}$, $\boldsymbol{c}_{(i,j)}$. The learned attention score can actively discriminate the degree of importance among features around a point. Finally, the weighted summed feature of a point i is given by:

$$f_{i_{\text{out}}} = \sum_{k=1}^{K} (\tilde{f}_{(i,k)} \otimes \boldsymbol{s}_{(i,k)}), \tag{3.6}$$

where $\otimes$ represents element-wise multiplication.

A sequence of Dilated Graph Convolution and Attentive Pooling modules are stacked in the feature extractor. Theoretically, the more modules are stacked, the larger the receptive field of each point will be. However, more modules would inevitably sacrifice the overall computational efficiency, and an excessively large receptive field is unnecessary [10]. The features integrated by the three Attentive Poolings were fed into three MLPs and then concatenated. Finally, these features were fed into two MLPs with a skip connection. Following the structure of the original FPCC, the 512-dimensional features of each point were extracted and fed into the two branches of the FPCC.

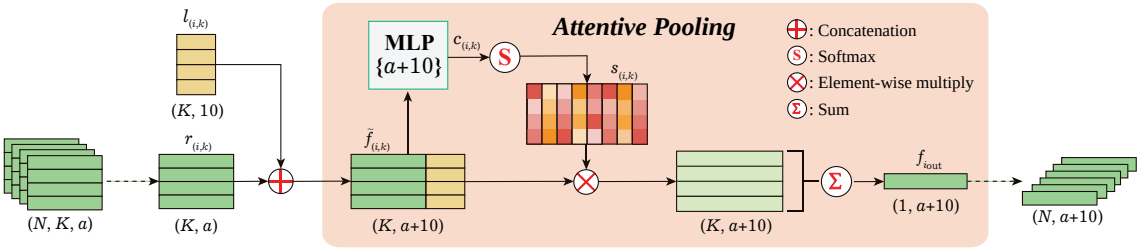


Figure 3.9: Attentive pooling. K a-dimensional relative features of N points are integrated in this module with the permutation invariance

3.4.3 Generation of Synthetic Point Cloud as Training Data for Instance Segmentation using physics engine

As discussed in Section 3.2.2, creating a rich and reliable training dataset is essential for instance segmentation of measured point clouds of wave-dissipating blocks based on deep learning. However, creating this training dataset by manually labeling block instances in large-scale real measured point clouds is labor intensive and practically impossible. To address this issue, FPCCv2 is only trained by the synthetic point cloud data that mimics the stacking poses of wave-dissipating blocks, and evaluated on both real-word data and synthetic data.

Several previous studies [151–153] used synthetic images to train the network through domain randomization. These methods proved successful through sophisticated image rendering. However, the color of blocks in outdoor environments can change due to uncontrollable factors, such as seawater, light and season, etc. Synthetic data faces difficulty to cover these influences. Based on this consideration, RGB information of the point cloud did not be used in both training and prediction. There are at least two advantages to this approach. The first is that the data is easier to synthesize in an acceptable period without manual work. The second is that it makes the network more robust and avoids interference of environmental factors.

The following procedure generates our synthetic point cloud data of the stacked wave-dissipating blocks.

1. A triangular mesh model m^b of a wave-dissipating block is created from the surface tessellation of its 3D CAD model.

2. Subsequently, a point set P^b is densely sampled on every triangle face of the mesh model m^b.

3. A variety of penetration-free stacking poses of piled blocks S_{stack} are generated using a set of triangular mesh model instances $\{m_0^b, m_1^b, \cdots\}$ for the model m^b. Further, a set of sampled points on blocks in stacking poses are calculated for each block as $P_{stack} = \{P_0^b, P_1^b, \cdot\}$ on the model instances $\{m_0^b, m_1^b, \cdot\}$, respectively.

4. A subset of the sample points on the block model in the stacking poses S_{stack} is picked up as $P'_{stack} = \{P'^b_0, P'^b_1, \cdot\}, (P'^b_j \subset P^b_j)$ that are only visible from a given position v_{ms} of the measurement device.

5. For every point $q \in P'^b_j \in P'_{stack}$, Gaussian noise at a certain level is imposed on the coordinates of q, which simulates the possible accidental error induced from the

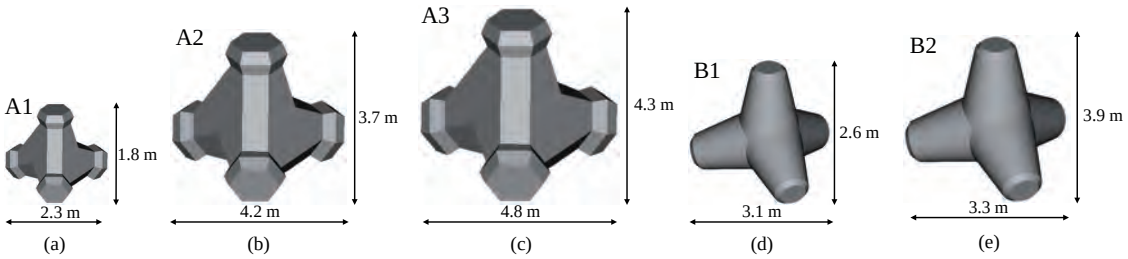


Figure 3.10: Triangular mesh models of wave-dissipating blocks are used in this study. We named these blocks Block A1 (a), Block A2 (b), Block A3 (c), Block B1 (d) and Block B2 (e). They are made of concrete to dissipate the force of waves. Notably, A1, A2, and A3 have the same shape but different sizes, and so do B1 and B2.

measurement device to create the final synthetic point cloud. The standard deviation of the Gaussian noise is estimated from the average distance between between model and real block point cloud.

The point cloud are used to train our instance segmentation network (FPCCv2). Figure 3.10 shows the examples of triangular mesh models of wave-dissipating blocks used for generating the synthetic point cloud data used in this study.

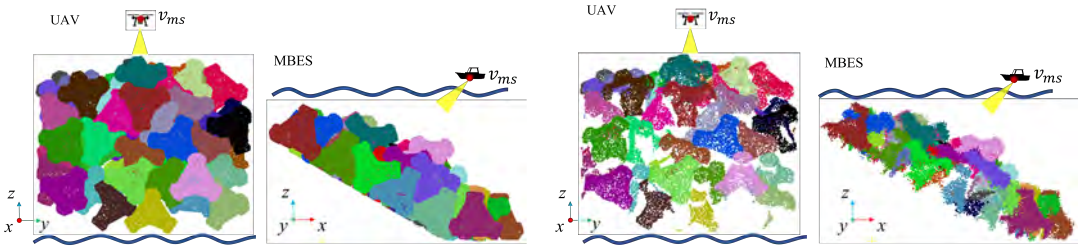
The stacking of blocks is simulated in the Bullet physics engine [126]. As shown in Figure 3.11, the penetration-free block-stacking scene under gravitation can be reconstructed in real time. About 50 identical blocks are stacked in a square with a side length about five times the maximum length of the block in a computer.

Figure 3.11 illustrates the generation procedures of synthetic point cloud data. The position of the measurement devices v_{ms} is defined corresponding to the UAV-mounted camera and sonar of the MBES, as shown in Figure 3.11(a). Figure 3.11(b) illustrates use of the hidden-point removal algorithm [154] to eliminate the invisible sample points on the block model from the original set of sampled points on the blocks P_{stack} to obtain the visible sample points P'_{stack}. Considering the noise levels of UAV and MBES that were found in the preliminary experiment, we added Gaussian noise with a standard deviation of 1 cm to the sampled point cloud P'_{stack} measured from the UAV-mounted camera, and a standard deviation of 3 cm to that from the MBES sonar.

We generated six groups of the synthetic point cloud data corresponding to the UAV and MEBS measurements of the blocks in three ports, each containing 500 training and 100 test point clouds. The generation for one synthetic point cloud took about three minutes in a standard desktop PC.

3.5 Block Pose Estimation and Type Classification

After segmenting the input point cloud into a set of points corresponding to individual block instances, the 6D pose of an individual block is estimated from each segmented point cloud using the PPF descriptor [124] and the best-fit point cloud alignment by ICP [125]. Moreover, the block type is classified based on the pose estimation results. The detailed algorithm is described in this section.



(a) (b)

Figure 3.11: Synthetic point cloud of stacking poses of piled blocks. (a) Original set of sampled points on blocks in stacking poses P_{stack}. Point clouds of different block instances are rendered in different colors. (b) Point cloud P'_{stack} after removing invisible points and adding artificial Gaussian noise that mimics the measurement error.

3.5.1 Block 6D pose estimation using 3D feature descriptor

First, a brief introduction to the PPF [124] descriptors for pose estimation is given. Owing to the powerful instance segmentation of FPCCv2 of the original point clouds, the input original measured point cloud Q has already been partitioned into a set of instance point clouds $Q_{stack} = \{Q_0^b, Q_1^b, \cdots, Q_l^b, \cdots\}$, where Q_l^b denotes the point cloud corresponding to the potential surface of an individual block l. Therefore, the original PPF algorithm is sufficient to estimate the 6D pose of an individual block l from a given point cloud Q_l^b.

PPF-based pose estimation combines a hash table and a voting scheme for matching the point cloud of a 3D object model to the one of 3D scenes to estimate the 6D pose of an object.

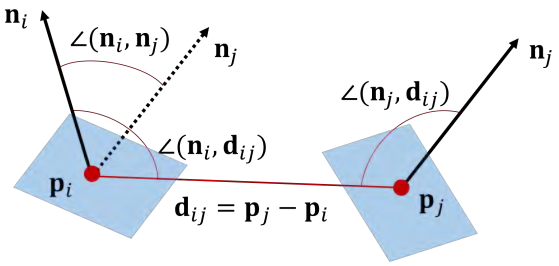


Figure 3.12: Illustration of Point Pair Feature (PPF).

The PPF descriptor $PPF(i, j)$ for a pair of 3D points (i, j) in a point cloud Q_l^b corresponding to the block l is defined as a four-dimensional vector by

$$PPF(i, j) = (\|\mathbf{d}_{ij}\|_2, \angle(\mathbf{n}_i, \mathbf{d}_{ij}), \angle(\mathbf{n}_j, \mathbf{d}_{ij}), \angle(\mathbf{n}_i, \mathbf{n}_j)), \tag{3.7}$$

where $\mathbf{n}_i$ denotes the normal vector at a point $i (\in Q_l^b)$, $\mathbf{d}_{ij} = \mathbf{p}_j - \mathbf{p}_i$, where $\mathbf{p}_i$ denotes a coordinates of point i, and $\angle(\mathbf{a}, \mathbf{b})$ denotes the angle between a vector $\mathbf{a}$ and $\mathbf{b}$, as shown in Figure 3.12. PPF descriptors are generated from every pair of points (i, j) in the point cloud Q_l^b and stored in a hash table, where $PPF(i, j)$ is used for the hash key. Because the PPF descriptor is defined only by a distance and angles between a pair of points, it is invariant to the rotation and translation.

A point cloud corresponding to an instance point cloud Q_l^b in a scene is sampled to approximately the same resolution as the sampled point cloud on a mesh model m^b. After sampling, given a scene point pair (i_s, j_s), the model point pair (i_m, j_m) that shares a similar PPF descriptor value to (i_s, j_s) can be retrieved using the hash table. To match the two point pairs, first, the point pairs (i_s, j_s) and (i_m, j_m) are transformed by the 4×4 homogeneous transformation matrix $T_\mathbf{s}$ and $T_\mathbf{m}$, respectively, such that i_m and i_s move to the origin, and their normals are aligned to the $x - axis$.

Then, the 3×3 rotation matrix $R_x(\alpha)$ that rotates j_m by an angle α around the $x-axis$ is found, such that j_m matches to j_s. These transformations can be defined as Eq. 3.8:

$$[\mathbf{p}_{i_s}, 1]^t = T_\mathbf{s}^{-1} \begin{bmatrix} R_x(\alpha) & 0 \\ \mathbf{0} & 1 \end{bmatrix} T_\mathbf{m} [\mathbf{p}_{i_m}, 1]^t, \tag{3.8}$$

where $\mathbf{p}_{i_s}$ and $\mathbf{p}_{i_m}$ denote the 3D coordinates of the point i_s and i_m, respectively.

Finally, in the voting scheme, for a given reference scene point i_s, PPF is evaluated with all other scene points j_s, after which it is matched with those of all point pairs (i_m, j_m) on the model using the hash table. For each potential match, one vote is cast in a 2D accumulator (i_m, α). After all matches are completed, the candidate poses over the threshold vote number are selected for clustering. The score of a cluster is the number of the candidate poses it contains, and the cluster with the highest score yields the estimate of the block pose. Finally, according to Eq. 3.8, the pose of a block l is estimated as

$$\begin{bmatrix} R_l & \mathbf{t}_l \\ \mathbf{0} & 1 \end{bmatrix} = T_\mathbf{s}^{-1} \begin{bmatrix} R_x(\alpha) & 0 \\ \mathbf{0} & 1 \end{bmatrix} T_\mathbf{m}, \tag{3.9}$$

Where $R_l \in \mathbb{R}^{3\times3}$ denotes the rotation matrix, and $\mathbf{t}_l$ denotes the translation of a block l.

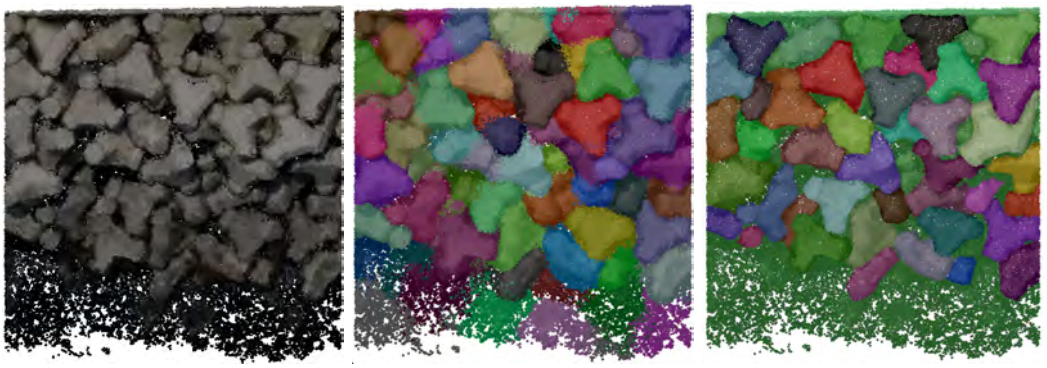
3.5.2 Pose Refinement and Block Type Classification

Some breakwaters are armored with two or more types of blocks, so it is necessary to register the point clouds of the different types of block models with the segmented scene point clouds. First, the poses roughly estimated by PPF is refined by ICP, and subsequently the fitness score is evaluated. The fitness score represents the relative distance between the segmented scene point cloud Q_l^b of a block l and registered model point cloud M_u, where u indicates the type of the block.

The fitness score B is the average minimum Euclidean distance between the point clouds of Q_l^b and M_u, and it is defined as Eq. 3.10:

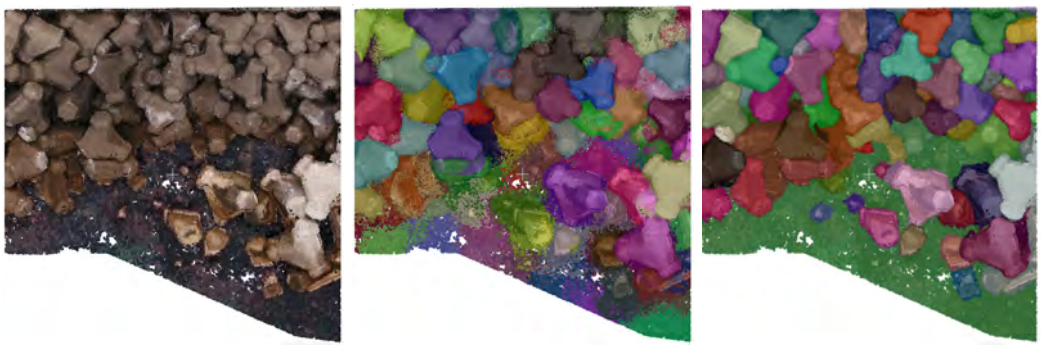
$$B(Q_l^b, M_u) = \frac{1}{|Q_l^b|} \sum_{\mathbf{p}_i \in Q_l^b} \min_{\mathbf{q}_j \in M_u} \|\mathbf{p}_i - \mathbf{q}_j\|_2 , \tag{3.10}$$

where $\mathbf{p}_i$ is the i-th point in the point cloud Q_l^b, $\mathbf{q}_j$ is the j-th point of the model point cloud M_u. The score B reflects how accurately the two point clouds match, where a lower score is better. Among different block types U, $M_u (u \in U)$ with the smallest fitness score B is determined as the most likely one.



(a)

(b)



(c)

Figure 3.13: Instance segmentation results on Sawara port (a), Todohokke port, (b) and Era port (c). The left column is the point cloud scene measured by UAV. The middle column is the predicted result of instance segmentation, and the last column is the ground truth. Different colors represent different blocks.

3.6 Results of Wave-Dissipating Block Detection

3.6.1 Experimental Sites

Three breakwaters at Sawara port, Todohokke Port, and Era Port in Hokkaido prefecture, Japan, were used for evaluation of the proposed block detection algorithm. For these ports, existing block point clouds above the water surface were measured by UAV and a commercial photogrammetry software, and those below the surface were measured by

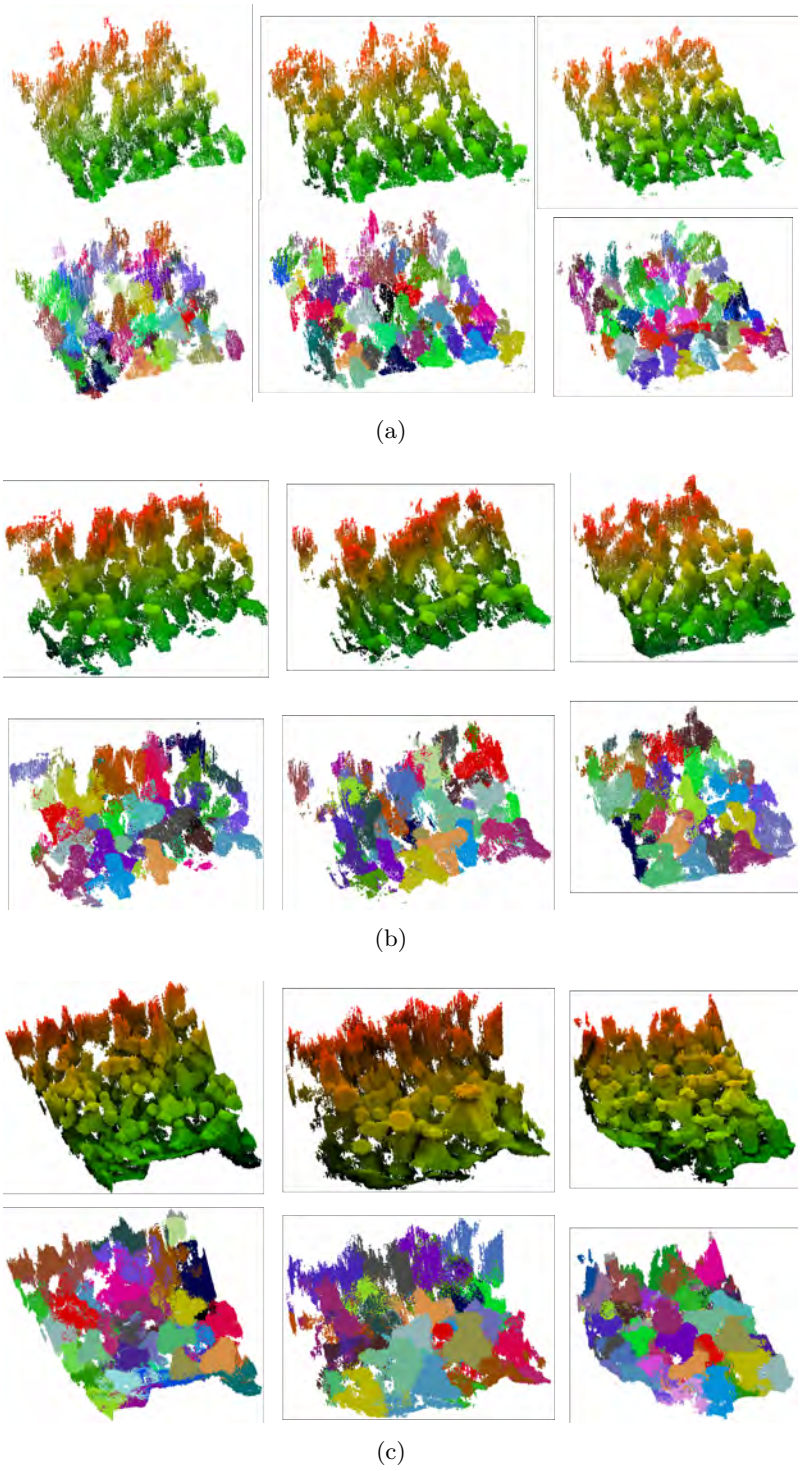


Figure 3.14: Visualization of instance segmentation results on MBES data. The first row of Sawara port (a), Todohokke port (b) and Era port(c) is the original point cloud measured by MBES. The second row of each subfigure is the predicted results of instance segmentation. Different colors represent different blocks.

Table 3.1: Detail of point cloud data of three ports.

Site	**Sawara**		**Todohokke**		**Era**	
Method of measurement	UAV	MBES	UAV	MBES	UAV	MBES
Size of test region [m]	~10 × 10	~10 × 10	~15 × 15	~15 × 15	~25 × 25	~25 × 25
The number of test regions	14	15	20	10	6	7
Point density (pts/m^2)	~600	~260	~370	~250	~580	~800
Block type used	A1	A1	B1	B1+B2	A2+A3	A2+A3
Approx. amount of points in a region	100,000~120,000	30,000~35,000	70,000~100,000	65,000~80,000	400,000~700,000	700,000~1,000,000

MBES. Table 3.1 indicates the details of the data of these sites, and Figure 3.23 shows the top view of the point clouds.

Only one type of block was used for the Sawara port, and two different types of blocks for the Todohokke and Era ports. Because the point cloud of all ports contains more than a million points, the proposed algorithm cannot handle it at once due to the memory limitation. Therefore, we partitioned the original point cloud of the whole site into several regions of tens of meters by tens of meters to test the proposed block detection method as shown in Table 3.1. Depending on the extent of the construction, the length and width of each region is approximately five times the size of the corresponding block, and the areas range approximately 100 to 600 m^2.

3.6.2 Block Instance Segmentation

Experimental Setting of CNN

Our block instance segmentation network FPCCv2 was implemented in the TensorFlow framework and trained using the Adam optimizer with an initial learning rate of 0.0001, batch size of 1, and momentum of 0.9. The network was trained for 60 epochs. It took approximately 30 h to train FPCCv2 for each block-stacking scene. All training and testing are performed on an NVIDIA GeForce RTX 2080 Ti GPU and an Intel Core i9-9900k CPU with 64 GB of memory. The training data is an all synthetic point cloud, as described in Section 3.4.3.

Precision and Recall

The performance of the block instance segmentation by FPCCv2 is evaluated on real UAV data and synthetic MBES data. Blocks of the same type but different sizes were stacked in Todohokke port and Era port. Because FPCCv2 is a category-agnostic clustering algorithm, differences in their size are not classified in this stage, but rather in the block classification stage described in Section 3.5.2.

Figure 3.13 and Figure 3.14 illustrate the results of the block instance segmentation predicted by FPCCv2, and further results are shown in Figure 3.24. Different blocks are rendered by different colors. While there are some burrs at the boundary of neighboring blocks, Figure 3.13 and Figure 3.14 show that the main body of each block could be clearly segmented. However, the segmentation of the area near sea surface is poor, because the original points measured in this area are noisier due to wave splash.

Table 3.2 summarizes the block-wise precision and recall of the instance segmentation with an intersection of union (IoU) threshold of 0.5. The ground truth instance label of the UAV test regions is made manually by referring the orthorectified image of these sites. The MBES test data used for quantitative evaluation are synthetic, because MBES

Table 3.2: Performance of instance segmentation on blocks in three ports. Metrics are precision(%) and recall(%) with an IoU threshold of 0.5. The UAV point cloud was measured from real-world scenes, and ground truth segmentation was performed manually, while the MEBS point cloud was synthesized.

Sawara				**Todohokke**				**Era**			
UAV		MBES		UAV		MBES		UAV		MBES	
Precision	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**
80.05	55.25	90.38	69.70	82.70	74.24	72.72	75.78	88.60	49.47	86.32	68.23

data are noisy, and the boundaries of the objects are blurred, making it difficult to label them manually. The precision on UAV data reached over 80%, while the recall rate ranges between 50 and 76%. The recall on the real UAV data is lower than that on the synthetic MBES data. We attribute this to some extremely small visible part of blocks in the real scene, which are difficult to be retrieved and segmented.

3.6.3 Block Pose Estimation

Synthetic Data Set Creation

Because manually labeling the 6D poses of all blocks included in a real point cloud scene of wave-dissipating blocks is a time-consuming and ambiguous task, the accuracy of block pose estimation is evaluated on six sets of synthetic datasets mimicking the UAV and MBES point clouds from the three ports. Each set contains 100 scenes with pose annotation. Figure 3.15(a) shows examples of the synthetic dataset.

Pose Estimation Accuracy using Synthetic Data Set

A pose of a rigid 3D object is generally represented by a 4×4 matrix $\mathbf{P} = [R, \mathbf{t}; \mathbf{0}, 1]$, where $R \in SO(3)$ is a 3×3 rotation matrix and $\mathbf{t} \in \mathbb{R}^3$ is a 3D translation vector. Because numerous wave-dissipating block shapes exhibit a certain number of geometric symmetries, and these symmetric poses are equivalent in appearance, the accuracy of the block pose estimation must be evaluated by considering these equivalent poses.

Taking the symmetry of a block geometry into account, a set of equivalent symmetric poses of the block $\mathfrak{R}(\overline{\mathbf{P}})$ can be represented by its ground truth 6D pose $\overline{\mathbf{P}} = [\overline{R}, \overline{\mathbf{t}}; \mathbf{0}, 1]$, where $\overline{R} \in SO(3)$ and $\overline{\mathbf{t}} \in \mathbb{R}^3$, as Equation 3.11.

$$\mathfrak{R}(\overline{\mathbf{P}}) \triangleq \{[\overline{R}G, \overline{\mathbf{t}}; \mathbf{0}, 1] | G \in G_s\}, \tag{3.11}$$

where $G_s \subset SE(3)$ is the group of equivalent symmetries that have no effect on the static state of an object [155]. Based on the Equation 3.11, the displacement error E_d and rotation error E_R between an estimated pose $\hat{\mathbf{P}} = [\hat{R}, \hat{\mathbf{t}}; \mathbf{0}, 1]$ and its ground truth pose $\overline{\mathbf{P}}$ are evaluated by

$$E_d(\hat{\mathbf{P}}, \overline{\mathbf{P}}) = \|\hat{\mathbf{t}} - \overline{\mathbf{t}}\|, \tag{3.12}$$

and

$$E_R(\hat{\mathbf{P}}, \overline{\mathbf{P}}) = \min_{G \in G_s} \arccos \frac{\mathbf{trace}(\hat{R}(\overline{R}G)^{\mathrm{T}}) - 1}{2} \tag{3.13}$$

The largest edge size of the bounding box of a block is set to $d_{\max}$; if the displacement error is within $0.1 \times d_{\max}$ and the rotation error is within 5°, the case was counted as a true positive; otherwise, it was considered as a false positive. Herein, performance metrices,

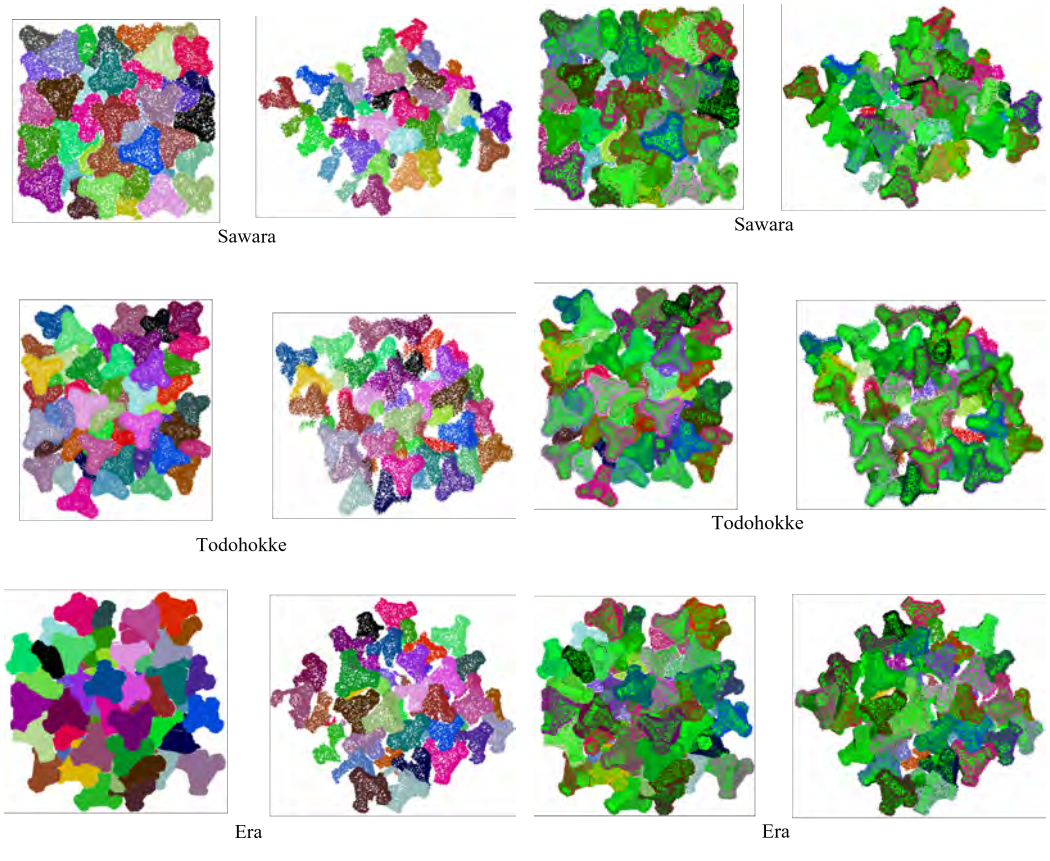


Figure 3.15: (a): Synthetic point clouds for three ports. (b): Pose estimation results on synthetic point clouds. Block models are transformed to the scene using estimated poses.

precision and recall, evaluated the case where a scene contains numerous blocks, and the average precision and recall of all scenes in each port were calculate finally. Only blocks with a lower than 80% occlusion rate are considered of interest to retrieve. Following the work [156], the occlusion rate is defined as:

$$\text{Occlusion rate} = 1 - \frac{\text{visible model surface point in the scene}}{\text{total model surface points}} \tag{3.14}$$

Table 3.3 presents the accuracy of pose estimation for each port, and Table 3.4 offers the displacement and rotation errors. Our method performs well on synthetic data, not only retrieving roughly 80% of blocks, but also with the errors less than 40 mm and 2.4° in the pose estimation. Figure 3.15(b) yields some visualization of the pose estimation results on synthetic data.

Pose Estimation Accuracy using Real Scene Data

It is practically difficult to prepare the ground truth poses of stacked individual blocks in a real scene. Therefore, in this study, the difference between the block surface with the estimated poses and the original point cloud indirectly indicates the accuracy of the block

Table 3.3: Block-wise accuracy of block pose estimation for synthetic scenes. Only blocks with a <80% occlusion rate are considered of interest to retrieve. The metrics are precision(%) and recall(%) with displacement error $\leq 0.1 \times d_{\max}$ and rotation error $\leq 5°$.

Sawara				**Todohokke**				**Era**			
UAV		MBES		UAV		MBES		UAV		MBES	
Precision	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**
99.3	79.4	83.2	85.4	96.1	95.5	86.1	77.9	98.6	97.0	97.9	95.2

Table 3.4: Average of displacement and rotation errors for synthetic scenes. E_d: displacement error(mm); E_R: rotation error(°).

Sawara				**Todohokke**				**Era**			
UAV		MBES		UAV		MBES		UAV		MBES	
E_d	E_R	E_d	E_R	E_d	E_R	E_d	E_R	E_d	E_R	E_d	E_R
30	0.8	31	2.3	23	1.5	31	2.4	29	0.5	39	1.1

pose estimation in the real scenes.

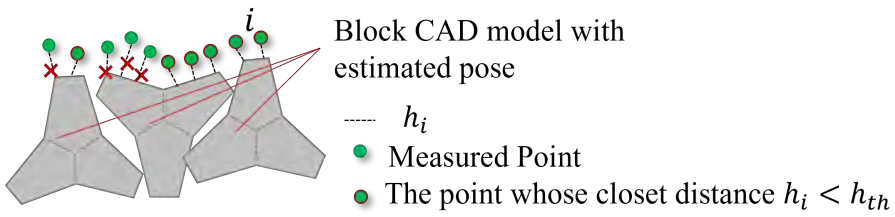


Figure 3.16: Average fitting error for block pose estimation.

To this end, we employed two indirect evaluation metrics: the average fitting error and the matching rate. As shown in Figure 3.16, for every point i in a scene point cloud, the distance $h_i(\in H)$ from i to the closest face on the block CAD model that fits to the point cloud is examined first. Then, the average fitting error E_f is calculated as:

$$E_f = \frac{1}{|H'|} \sum_{h_i \in H'} h_i, \quad H' = \{h_i | h_i \leqslant h_{\text{th}}, h_i \in H\} \tag{3.15}$$

where h_{th} is the threshold distance for the accuracy evaluation.

Another metric, matching rate R, is used to indirectly evaluate the recall of pose estimates, which is defined as:

$$R = \frac{|H^{'}|}{|H|} \tag{3.16}$$

Examples of the average fitting error and matching rate are visualized in Figure 3.21 and 3.22. The average fitting error and matching rate of UAV and MBES data for the three ports are summarized in Table 3.5. The results on real data show that our method can match more than 60% of the points with a small pose estimation error, and almost all visible blocks in a scene can be retrieved. Meanwhile, the measured boundary of the MEBS point cloud decreases the performance of the method due to the excessive noise.

Table 3.5: Accuracy of pose estimation for various blocks in real scenes. E_f: fitting error(mm); R: matching rate(%). h_{th} = 0.1 m for Sawara and Todohokke, h_{th} = 0.2 m for Era.

Sawara				**Todohokke**				**Era**			
UAV		MBES		UAV		MBES		UAV		MBES	
fitting error	**matching rate**	**fitting error**	**matching rate**	**fitting error**	**matching rate**	**fitting error**	**matching rate**	**fitting error**	**matching rate**	**fitting error**	**matching rate**
33	71.32	39	67.28	34	66.17	38	60.54	48	75.12	58	71.52

Table 3.6: Accuracy of block type classification.

Site	**Sawara**		**Todohokke**	
Block Type	B1	B2	A2	A3
Actual block number	77	58	111	70
Miss classified block number	4	5	3	3
Accuracy	94%		96%	

3.6.4 Block Type Classification

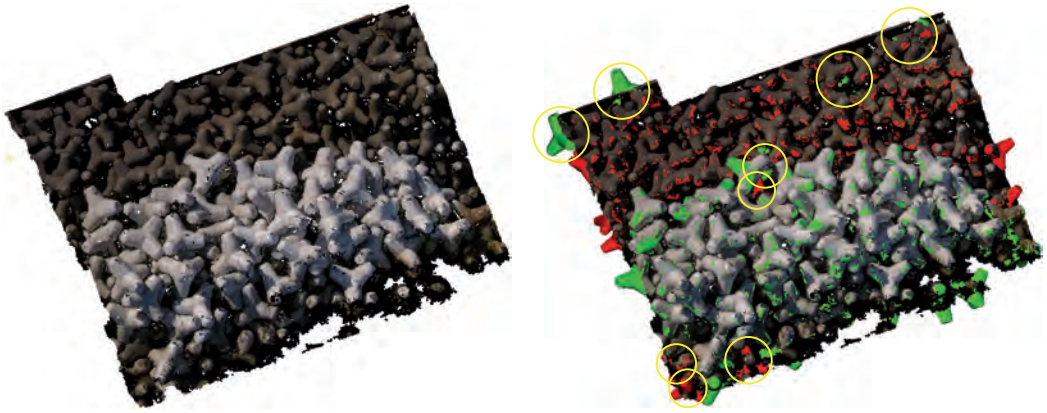
Finally, the performance of our multiple block type classification was evaluated on the UAV point cloud from the Todohokke and Era ports. As described in Section 3.5.2, after estimating the pose of a block, the fitness scores were calculated using different model point clouds of blocks and the corresponding scene point clouds, respectively. The one with the smallest fitness score was regarded as the correct block type. The ground truth of the block type was determined by taking manual size measurement for the points sampled from the original scene point cloud and comparing the size with the standard block size specification disclosed in advance from its manufacturer.

The block type classification performance was indicated in Table 3.6. Figure 3.17 presents the block type classification result for the Todohokke port, where a total of 135 blocks were recognized, of which nine misclassifications are indicated by yellow circles. The block-wise accuracy reached 94%. Figure 3.18 shows the result of block type classification of the Era port, where a total of 181 blocks were recognized, of which six misclassifications are indicated by yellow circles. The accuracy rate reached 96%. As shown in Figure 3.17, the incorrect results occur mainly at the boundaries of the data, where the points of the blocks are incomplete, and in the part near the sea, where the noise is relatively high.

3.6.5 Processing time

This section provides the processing time of each step involved in our method. Table 3.7 presents the processing time for synthetic training data creation and training of our instance segmentation network (FPCCv2). Five hundred artificial point cloud scenes of piled blocks shown in Figure 3.11 that mimic UAV and MBES measurements were used to train FPCCv2 with an epoch of 60. Approximately 30 hours of training were required in each.

Table 3.8 summarizes the computation time of the block detection except for the training. Once learned, FPCCv2 could segment about 100,000 points into the block instances in less than 1 second and 700,000 points in about 4 seconds. The pose estimation time was about 3 minutes per region. Each region of Sawara and Era included 40 blocks on average, and each region of Todohokke contained 30 blocks on average. The time of the



(a) (b)

Figure 3.17: Visualization of results of block type classification on Todohokke port. (a) is the real point cloud scene of Todohokke port measured by UAV, and (b) is the result of type classification after instances segmentation and pose estimation. Block B1 and Block B2 is rendered in red and green, respectively. The yellow circles indicate misclassified blocks.

Table 3.7: Processing time for synthetic training data creation and training of the instance segmentation network.

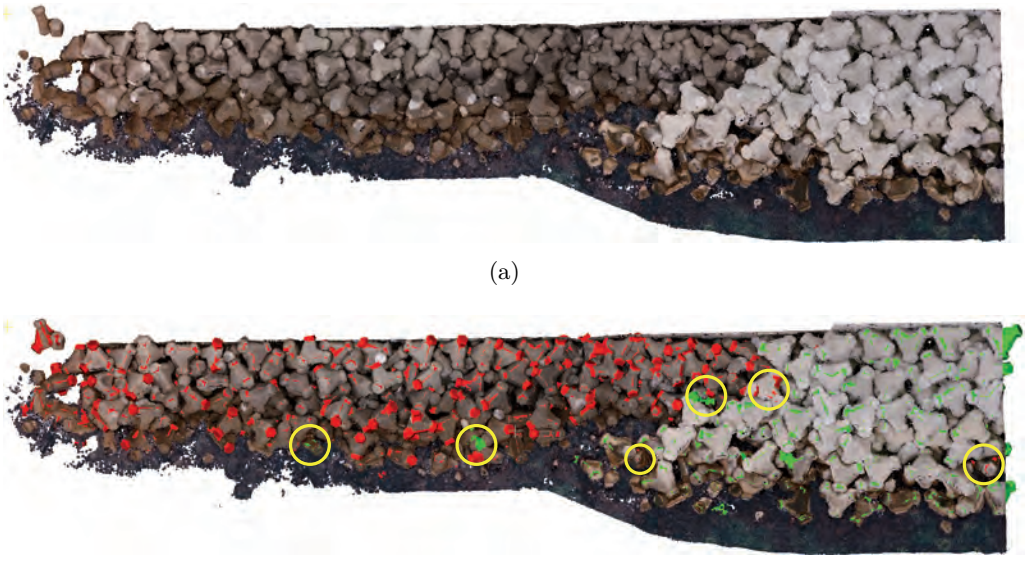
Site	**Sawara**		**Todohokke**		**Era**	
Method of measurement	UAV	MBES	UAV	MBES	UAV	MBES
Size of test region [m]	~10 × 10	~10 × 10	~15 × 15	~15 × 15	~25 × 25	~25 × 25
The number of generated synthetic point cloud scenes	500	500	500	500	500	500
Block type used	A1	A1	B1	B1+B2	A2+A3	A2+A3
The number of blocks / region	50~60	50~60	50~60	50~60	50~60	50~60
Epoch	60	60	60	60	60	60
Time of training [hour]	30	30	30	30	30	30

block classification was about 30 seconds per region for Sawara and Todohokke, and about 2.5 minutes per region for Era.

These results show that the proposed block detection method is fast enough for practical use.

3.7 Need for Instance Segmentation in the Pose Recognition of Wave-Dissipating Blocks

This section provides a complementary explanation of the effectiveness of this pre-processing of instance partitioning. A naive solution is to use the sliding window method instead of instance partitioning, similar to brute-force search. Considering that the size of a block is 4m, we use a window of size 5m× 5m with a given step strip of 2.5m, cut out multiple sub-regions from the original scene as shown in Figure 3.6.5, and then match the blocks in each sub-region with PPF. The experiments is carried out on two regions of Todohokke port as shown in Figure 3.20. Table 3.9 gives the comparison results of using instance segmentation and the sliding window method, respectively. Both the speed and the number of recognized blocks, the method with instance segmentation is far better than the



(a)

(b)

Figure 3.18: Visualization of results of block type classification on Era port. (a) is the real point cloud scene of the Era port measured by UAV, and (b) is the result of type classification after instances segmentation and pose estimation. Block A2 and Block A3 is rendered in red and green, respectively. Yellow circles indicate misclassified blocks.

Table 3.8: Processing time of the block detection.

Site	**Sawara**		**Todohokke**		**Era**	
Method of measurement	UAV	MBES	UAV	MBES	UAV	MBES
Size of test region [m]	~10 × 10	~10 × 10	~15 × 15	~15 × 15	~25 × 25	~25 × 25
The number of test regions	14	15	20	10	6	7
The number of blocks in a region	40~50	35~50	25~35	25~30	30~50	30~40
Point density (pts/m^2)	~600	~260	~370	~250	~580	~800
Block type used	A1	A1	B1	B1+B2	A2+A3	A2+A3
Approx. amount of points in a region	100,000~120,000	30,000~35,000	70,000~100,000	65,000~80,000	400,000~700,000	700,000~1,000,000
Time of Block Instance segmentation [s]/region	0.8	0.4	0.7	0.5	3.6	4.2
Time of Block pose estimation [s]/region	127	72	130	85	175	162
Time of Block type classification [s]/region	32	24	35	26	160	158

method with sliding window.

3.8 Conclusions

A novel deep-learning-based approach to detect individual blocks from large-scale three-dimensional point clouds measured from a pile of wave-dissipating blocks placed overseas and underseas using UAV-photogrammetry and MBES was proposed. The approach consisted of three main steps. First, the instance segmentation using our originally designed deep convolutional-neural network (FPCCv2) partitioned an original point cloud into small subsets of points, each corresponding to an individual block. A physics engine enabled to generate instance-labeled training datasets synthetically and automatically for instance segmentation of blocks, avoiding laborious manual labeling work and secure rich training datasets for our convolutional-neural network. Subsequently, the block-wise 6D pose was estimated using a three-dimensional feature descriptor (PPF), point cloud registra-

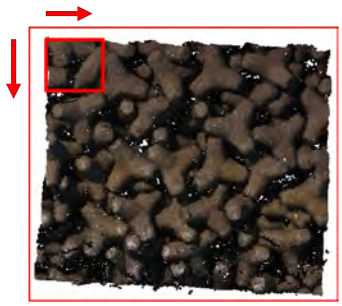


Figure 3.19: A illustration of sliding windows

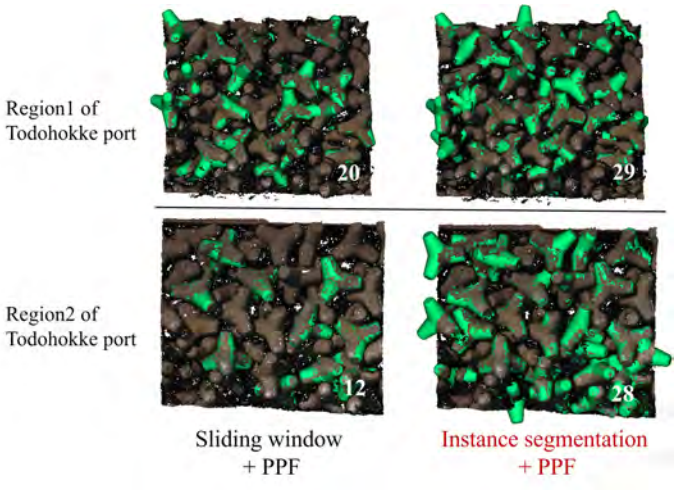


Figure 3.20: The comparison results of sliding windows and proposed instance segmentation. The recognized blocks are shown in the green CAD model.

tion (ICP), and CAD models of blocks. Finally, the type of each segmented block was identified using the model registration results.

The conclusions of this chapter are summered as follow.

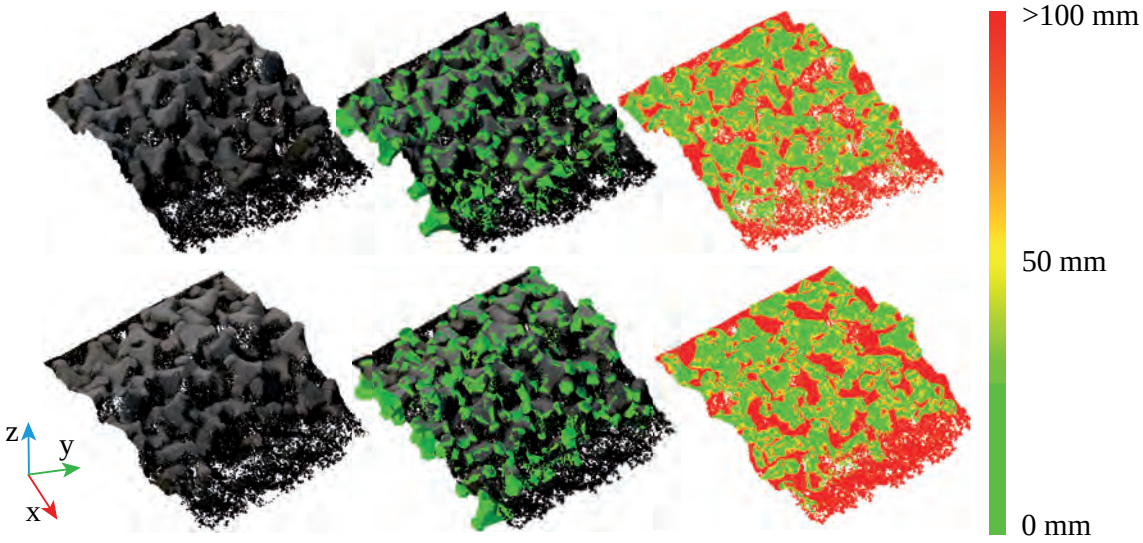
- The results of the instance segmentation on real-world and synthetic point cloud data achieved 70%~90% precision and 50%~76% recall with an intersection of union threshold of 0.5.
- The pose estimation results on synthetic data achieved 83%~95% precision and 77%~95% recall under strict pose criteria.
- The average block-wise displacement error was 30 mm, and the rotation error was less than 2°.
- The pose estimation results on real-world data showed that the fitting error between the reconstructed scene and the scene point cloud ranged between 30 and 50 mm, below 2% of the block size detected.
- The accuracy in the block type classification on real-world point clouds reached approximately 95%.

Table 3.9: Comparison of method with instance segmentation and sliding windows.

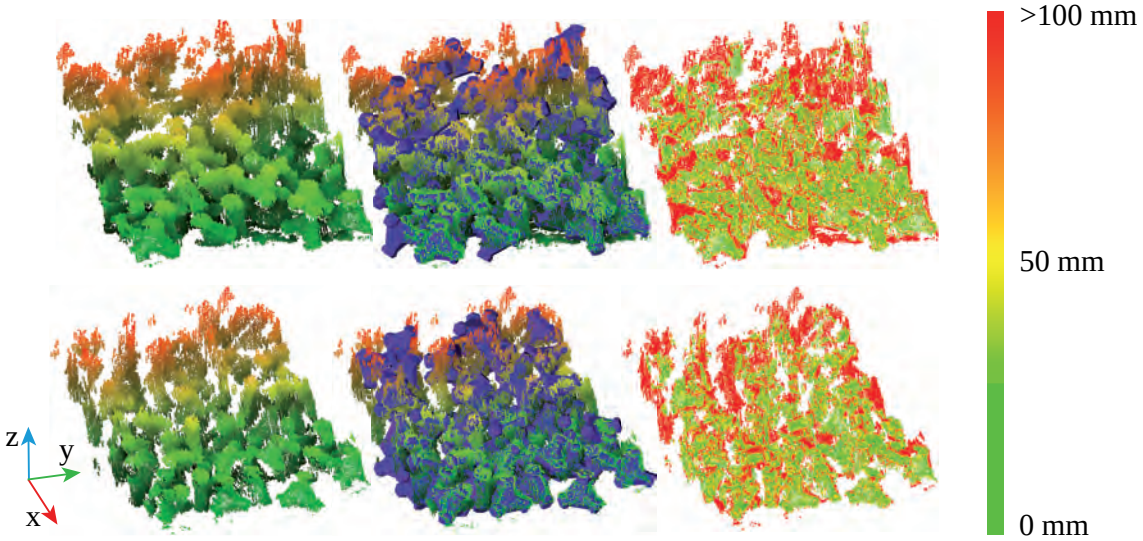
Method	Sliding windows + PPF	Instance segmentation + PPF (Proposed)
Computation time per region [s]	300	60
The number of recognized blocks	32	57

- These block detection performances proved the effectiveness of our approach.

In future studies, we plan to reconstruct the virtual scene of current wave-dissipating blocks in the physics engine based on our detected block poses, and then simulate and generate a more accurate construction process plan of block supplemental work than the conventional one according to practical requirements. This would guide the accuracy, save construction time, and visualize the construction process and results.

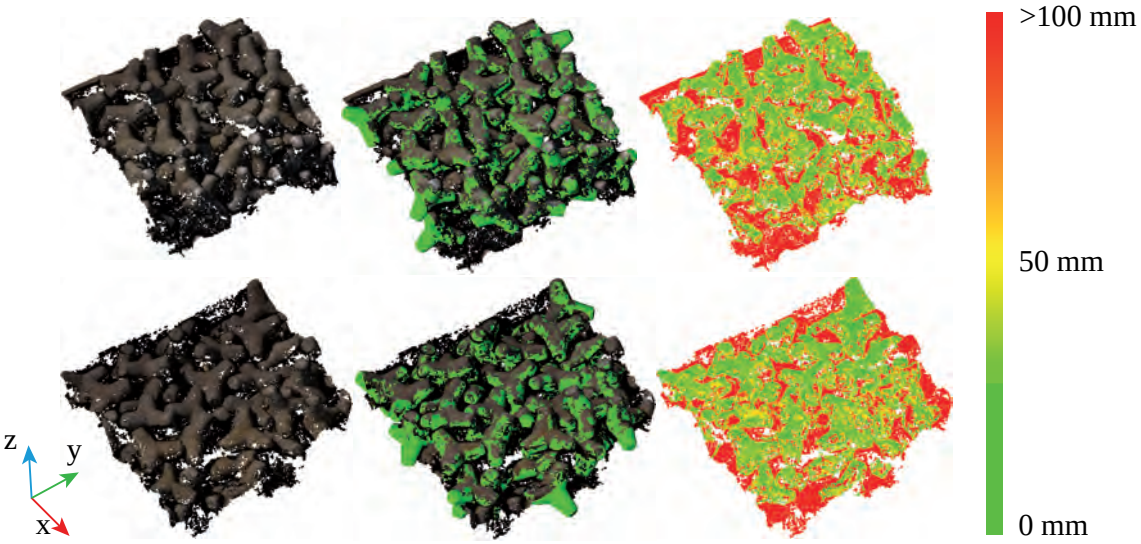


(a)

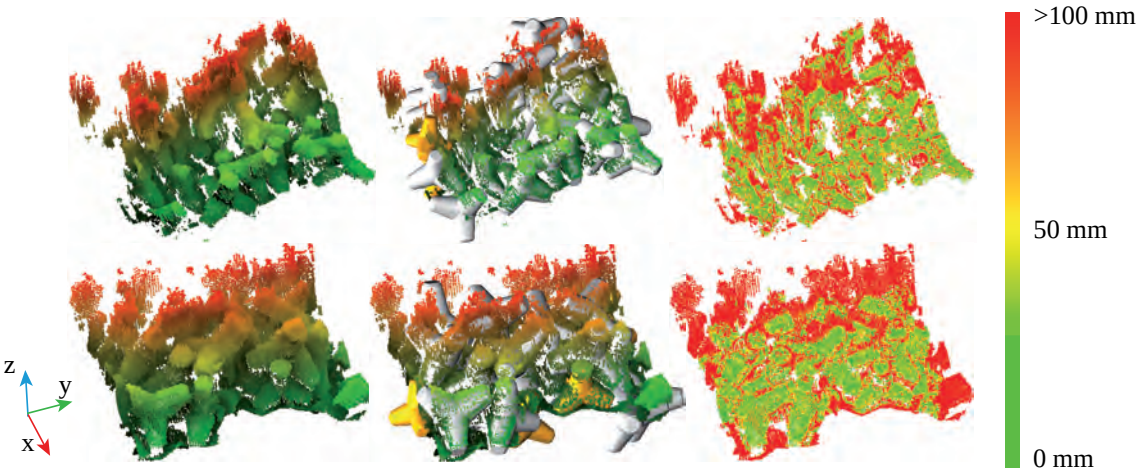


(b)

Figure 3.21: Visualization of pose estimation on UAV data(a) and MBES data(b) of Sawara port. The first column is the measured point cloud. The block models are transformed to the scene space using pose results and displayed in the middle column. The last column is a visualization of the average fitting error of the pose estimates, where the red points indicate that its distance from the nearest block model is larger than 100 mm.

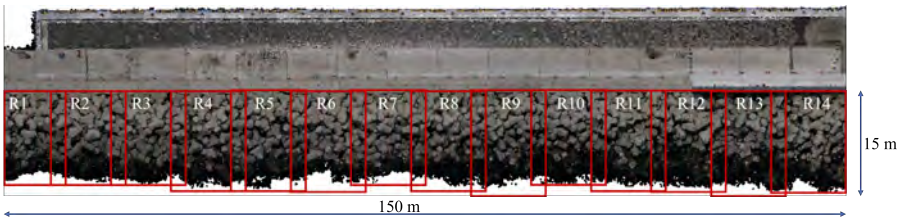


(a)

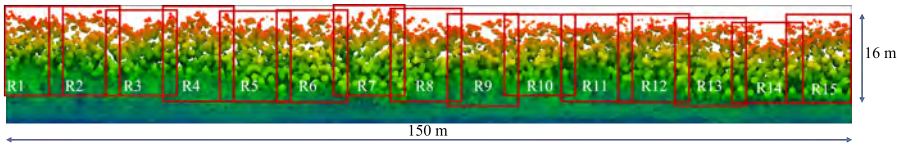


(b)

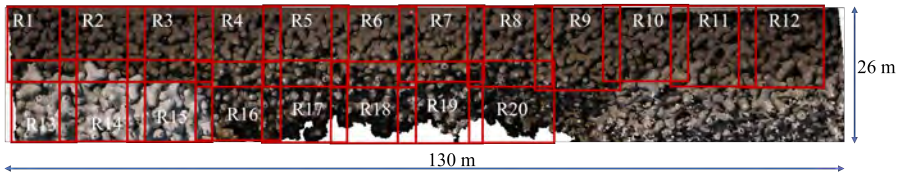
Figure 3.22: Visualization of pose estimation on UAV data(a) and MBES data(b) of Todohokke port. The first column is the measured point cloud. The block models are transformed to the scene space using pose results and displayed in the middle column. The last column is a visualization of the average fitting error of the pose estimates, where red points indicate that its distance from the nearest block model is larger than 100 mm.



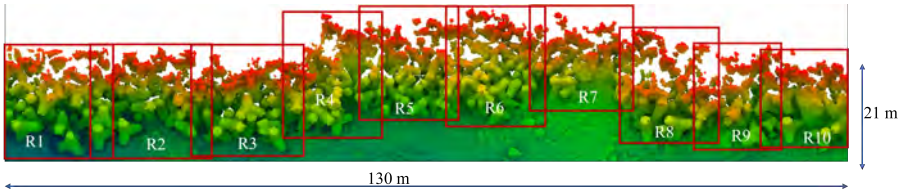
(a) UAV point cloud of Sawara port



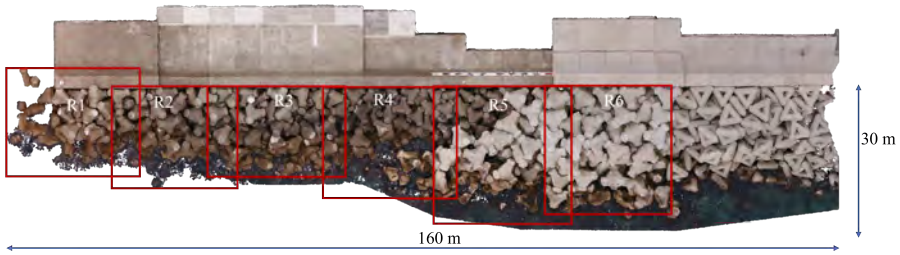
(b) MBES point cloud of Sawara port



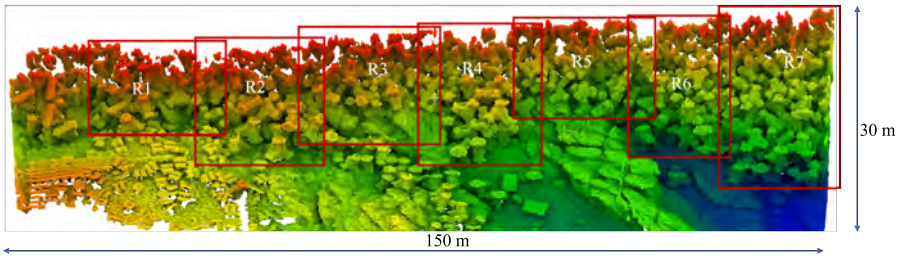
(c) UAV point cloud of Todohokke port



(d) MBES point cloud of Todohokke port

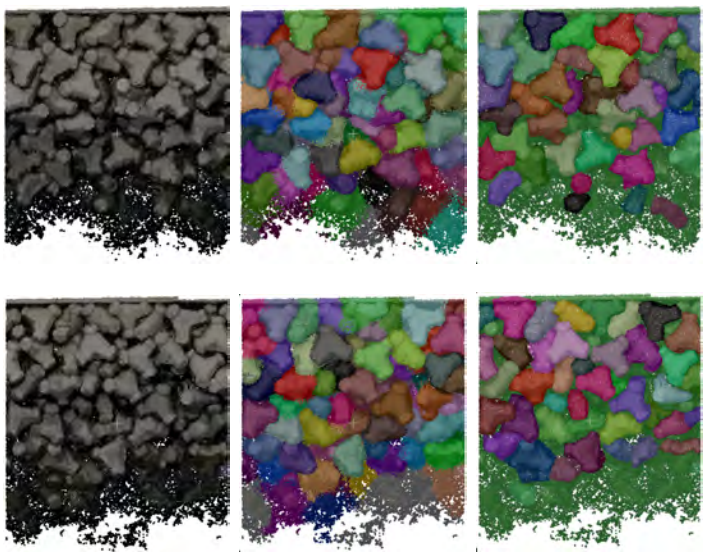


(e) UAV point cloud of Era port

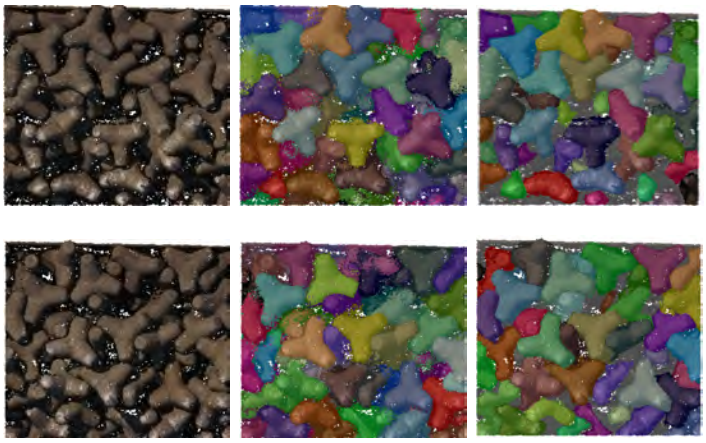


(f) MBES point cloud of Era port

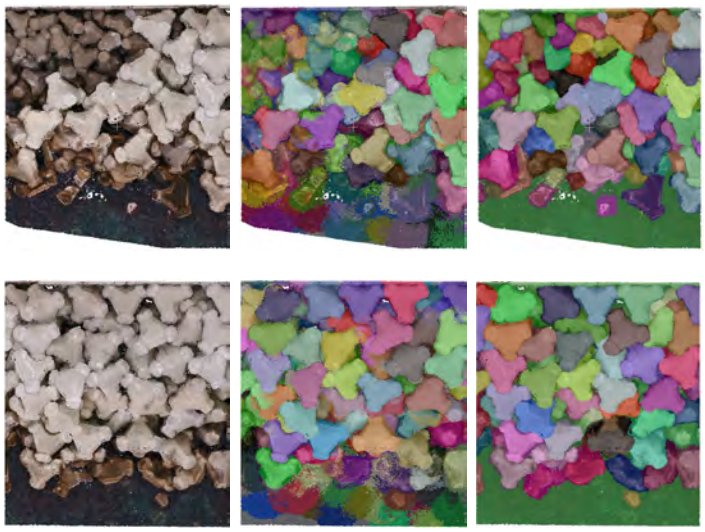
Figure 3.23: Point cloud of three ports.



(b)



(d)



(f)

Figure 3.24: Additional instance segmentation results on Sawara port (a), Todohokke port (b) and Era port(c). The left column is the point cloud scene measured by UAV. The middle column is the predicted results of instance segmentation, and the last column is the ground truth. Different colors represent different blocks.

Chapter 4

Deep Learning-based Object Stacking Operation Plan for Replenishing Wave-Dissipating Blocks

4.1 Background and Objectives of this Chapter

Object stacking operation planning can be regarded as an extension of the packing problem. Object stacking operation planning not only expects to fill as many objects as possible in a given volume or container but also needs to meet the physical aspects of feasibility.

Packing irregular 3D rigid objects into a container with pre-defined dimensions is a well-known Np-hard problem that is faced in practical applications such as the construction industry, 3D printing, and dry stacking. No algorithm can find a globally optimal solution for the problem in polynomial time.

Due to the limitation of computational power, early approaches have approximated irregular objects into some regular shape primitives, such as bounding boxes or bounding cylinders, to reduce the computational requirements for geometric analysis and processing. However, these were inefficient and not current to the application's needs.

Researchers have previously developed methods for packing objects based on mathematical models. However, their mathematical modeling methods suffered from complexion and a large number of vertices, resulting in poor accuracy for typical irregular objects. Furthermore, some studies have emphasized the solution's optimality at the expense of the balance and stability of object stacking, which is particularly important in construction.

As shown in 4.1, this chapter is (1) to develop a method that can plan a stacking pose making more blocks compactly stacked in the same replenishment space, and (2) to verify the matching of the number of blocks replenished from the plan with the actual construction work.

The main contributions of this work are listed below.

- A space detection algorithm is designed for finding where the new blocks need to be inserted.
- A deep learning-based pose prediction method is proposed to make the new blocks

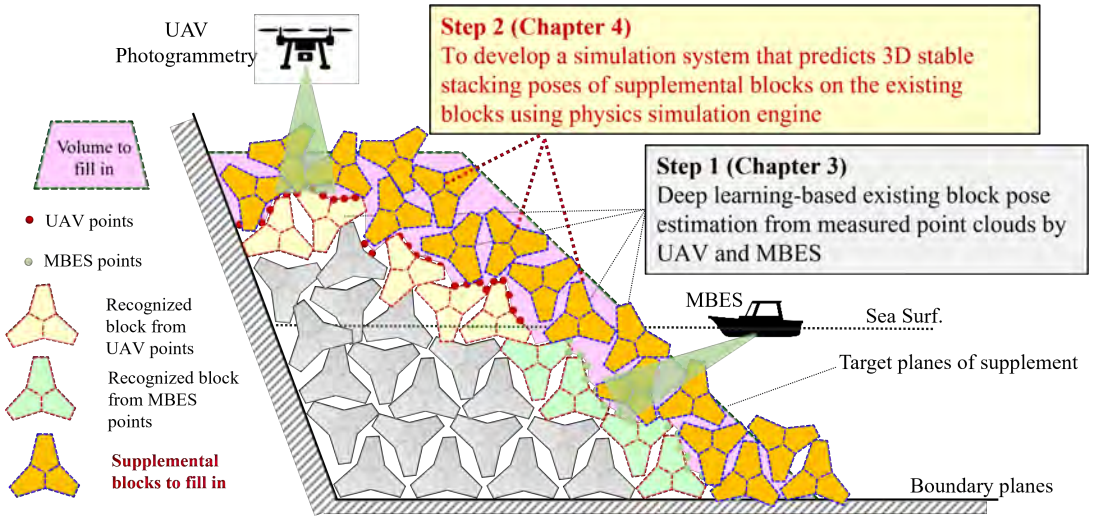


Figure 4.1: The problems to be solved in this chapter.

drop from a fine initial pose so that the breakwater structure is steady and compacted.

- Experiments show that the results of the proposed scheme are reliable

The remainder of this Chapter is organized as follows. Section 4.3 briefly describes the initial processing of the model before it is imported into the physics engine. Section 4.4 shows the overview of the processing pipeline. Section 4.5 4.6 4.7 illustrate three important technical details, space detection, Initial block pose prediction based on deep learning, and stacking simulation, respectively. Experimental analyses are provided in Section 4.8. Section 4.9 discuss the results. Finally, Section 4.10 summaries the Chapter.

4.2 Related Works

Earlier approaches approximated irregular objects as some regular shaped primitives, such as bounding boxes or bounding cylinders, to reduce the computational requirements for geometric analysis and processing [32,33]. However, these were inefficient and do not meet the current application requirements [34].

Some researchers have previously developed mathematical model-based methods for packing objects [35]. However, their mathematical model approach suffers from complexity and a large number of vertices, resulting in poor accuracy for typically irregular objects. In addition, some studies [30,31] emphasized the optimization of solutions while ignoring the balance and stability of object stacking, which is especially important in actual works.

Mitsui et al. [157] developed a real-time simulation system (see Figure 4.2) for the object stacking planning, where the stacking process of each new object is manually controlled. Thus the stacking plan generated by this solution relies heavily on the experience of the controller and cannot be automatically generated.

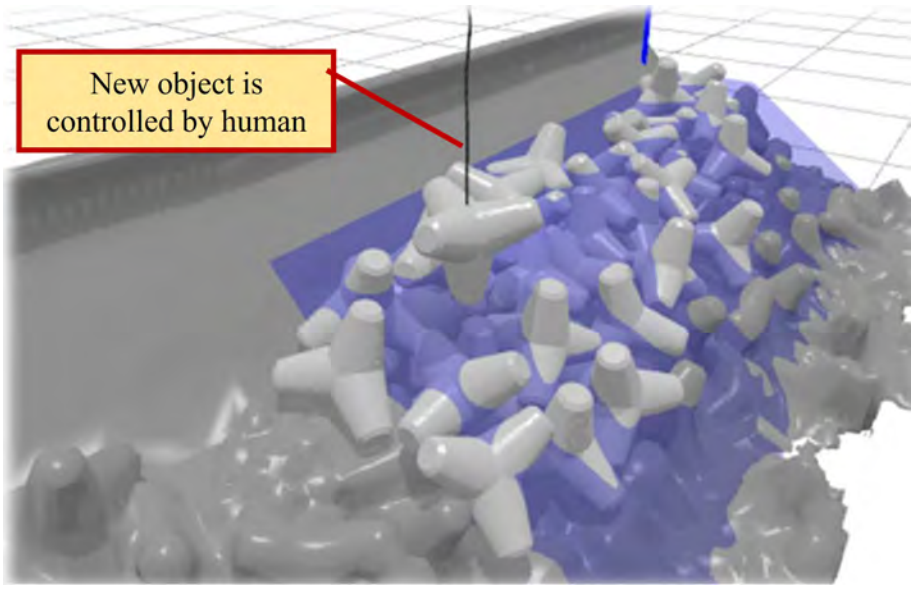


Figure 4.2: The real-time simulation system of object stacking developed by [157].

4.3 Approximate Convex Decomposition

This section briefly explains some basic concepts in the physics engine. To operate complex concave 3D models in physics engines, 3D modeling software developers usually decompose 3D mesh into a set of simple convex shapes, such as ellipsoids, capsules, or convex-hulls.

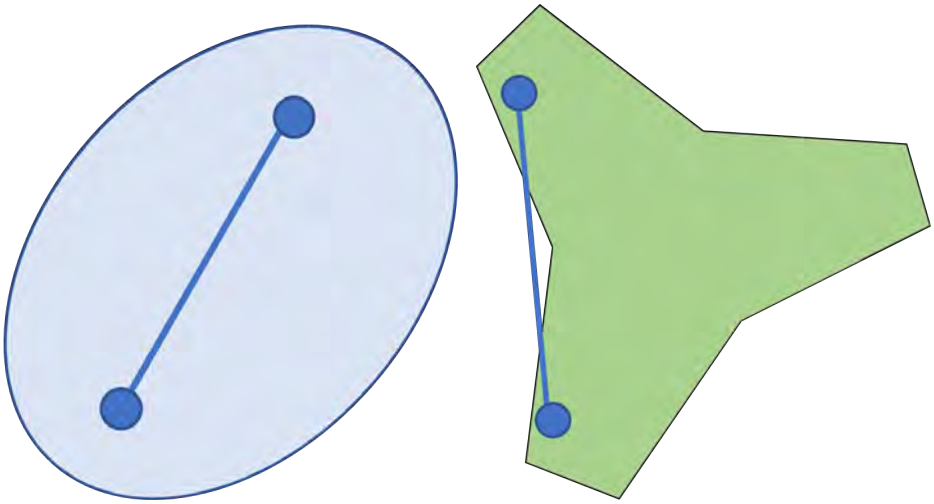
4.3.1 Convex Geometry

A geometry is considered convex if any two points of a line segment within the geometry must fall within the geometry. The contrast is a concave geometry, as shown in Figure 4.3.

4.3.2 Convex Decomposition

Since collision detection in the physics engine is based on convex geometry, it is necessary to approximate the decomposition of a concave geometry into a set of convex geometries. However, how to approximate the decomposition of objects is beyond the scope of this thesis, so in this subsection, we only give a brief description of the HACD [158] and V-HACD [159] techniques recommended by the developer of physics engine Bullet [126].

Approximate Convex Decomposition does not require strict convexity constraints but rather that the decomposed components are approximately convex. They then iteratively decompose and cluster the 3D shapes until the concavity of each decomposed component is within a pre-determined threshold, which measures the difference between a shape and its convex hull. HACD [158] employed a boundary distance as a metric by projecting the grid vertices along the normal direction onto the convex hull surface and then measuring the distance to the edge of the convex hull. Its extended version, V-HACD [159], used the volume difference between the shape and its convex hull as a metric, first voxelizing the input mesh and then greedily disentangling the voxels with axis-aligned cutting planes and voxel-based concave surfaces. Due to its open-source code and good performance in



(a) convex (b) concave

Figure 4.3: The example of convex and concave shape.

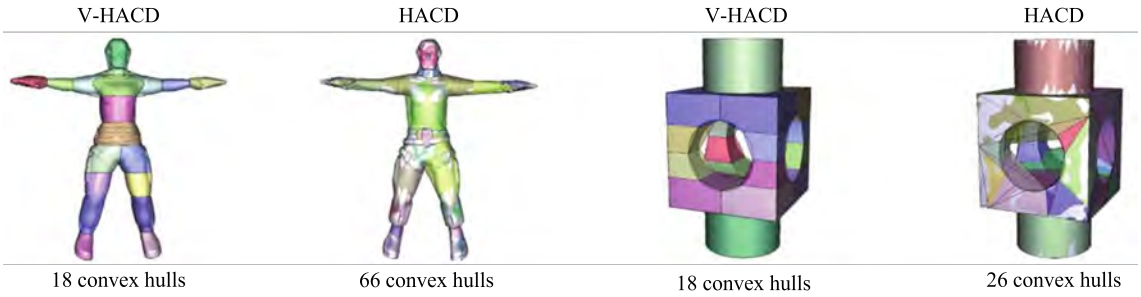


Figure 4.4: Approximate convex decomposition generated by HACD and V-HACD from the work [159].

general, V-HACD is one of the most widely used convex decomposition algorithms today. Figure 4.4 shows some examples of objects approximated by HACD and V-HACD.

4.4 Overview of the Processing Pipeline

We proposed a reliable approximate optimal solution for guiding the actual wave-dissipating block replenishment by simulating the stacking of blocks in the physics engine. Note that the models imported into the physics engine in this Chapter are convex models decomposed by V-HACD [159], if not otherwise specified. The simulation pipeline is outlined in Figure4.5.

The blocks recognized from Chapter3 and the support and boundary planes are imported into the physics engine to reproduce the current scene. The custom-generated target plan is operated in the physics engine for visualization purposes only and does not collide with other objects. In the loop, the point cloud scene of the current blocks is first generated and rasterized for the proposed spatial detection to figure out where a new block needs to be added. Next, a new block is dropped at the detected location. More-

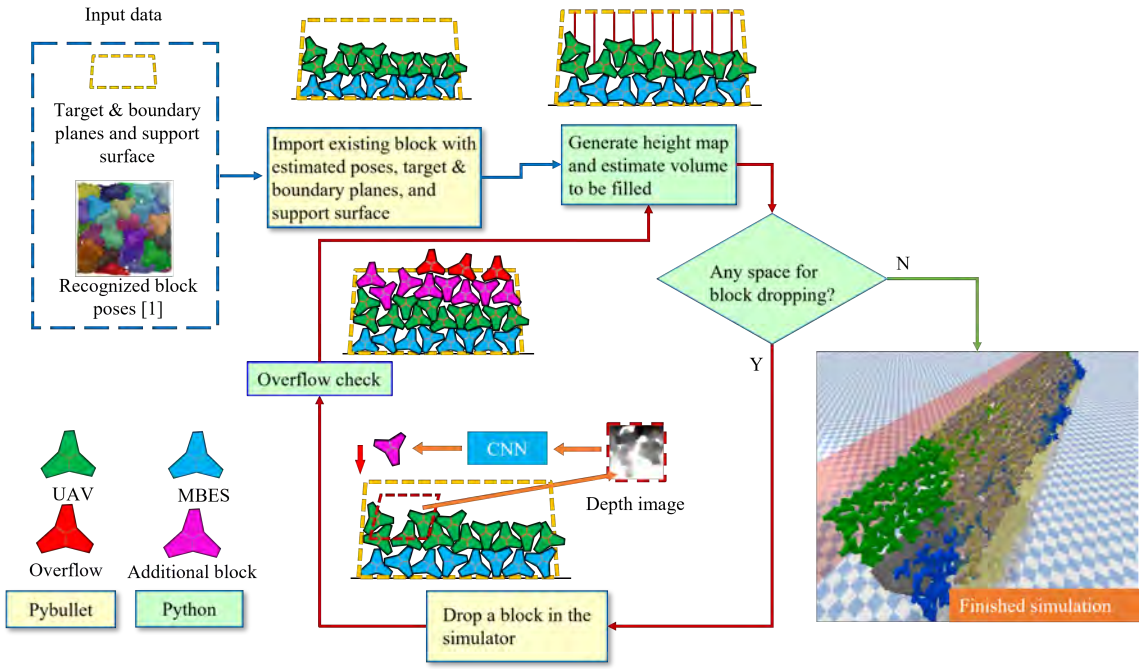


Figure 4.5: Pipeline of object stacking operation plan based on simulation. The simulation consists of four main steps, 1) Initialization: importing the recognized blocks and the target plane, the boundary, and support plans. 2) Space detection: identifying the locations where new blocks need to be added. 3) Adding blocks: the blocks fall into the specified locations with a reasonable attitude predicted by the CNN. 4) Height check: screening the blocks whose height exceeds the target plane. Steps 2), 3), and 4) will be recycled several times until the construction needs are met without inserting more stones.

over, since the initial pose before the block falls has a crucial impact on the results, a deep learning-based estimator is utilized to predict the optimal initialized block pose. Each replenishment round is followed by an automatic checking of whether the block exceeds the predefined target plan. This round will be cycled several times, typically 20 times, until there is no space vacated to replenish a new block, i.e., the desired construction result is achieved.

4.5 Space Detection

Locating the location of the new wave elimination blocks based on the current scene and the intended target shape is the first step in the construction. The scene point cloud is reconstructed according to the current state of all the wave-dissipating blocks.

To evaluate the volume, as shown in Figure4.6 , the XY planes are gridded into a collection of unit cells $C = \{C_k\}$ at intervals Δl in the designated construction area. The z-height of each cell is calculated as:

$$h_k = h_{tk} - \max_z P_{z,k}, \tag{4.1}$$

where h_{tk} is the height of target plane at the center of C_k, $P_{z,k}$ a set of height values of the model points included in C_k. If there is not any point in the cell, h_k is determined by

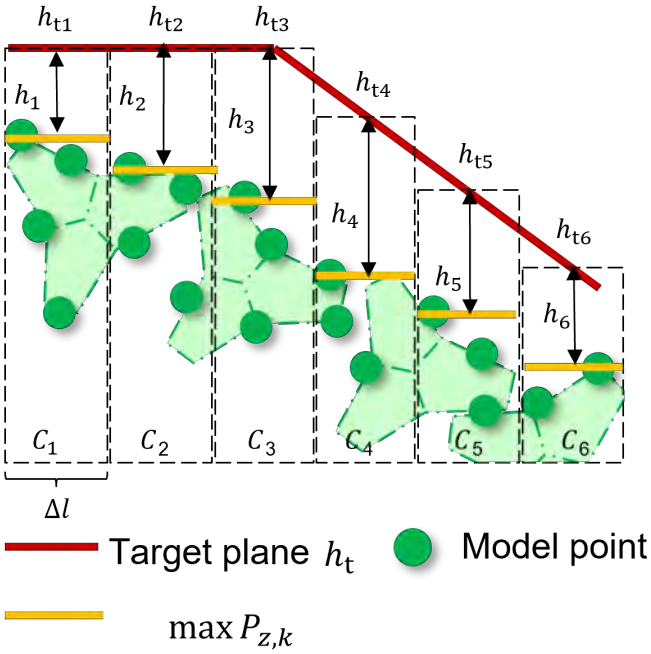


Figure 4.6: Illustration of height map calculation.

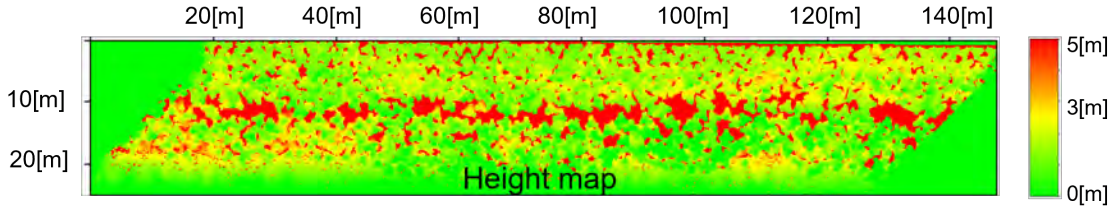


Figure 4.7: Generated height map from a construction area. Δl : 0.1.

the linear interpolation from the heights of surrounding cells. Then the volume V_k of a cell C_k can be estimated as:

$$v_k = (\Delta l)^2 h_k. \tag{4.2}$$

The v_k over the window indirectly represents the insufficient volume to be filled in the window. An example of the height map $\{h_k\}$ is displayed in Figure4.7. A window of size $d_x \times d_y$ slides from the top-left corner of the height map with stride s. Usually $d_x \Delta l$ and $d_y \Delta l$ are close to the size of the block, resulting in the range of the window close to that of the block. We employ the typical non-maximal suppression algorithm2 to filter out the windows that need to be supplemented with blocks, that is, the positions. Figure4.8 shows the processing of NMS on detected windows.

Algorithm 2: Non-maximum suppression algorithm on windows; N is the number of windows; K is the number of detected windows.

Input: $\mathbb{W} = \{w_1, w_2, ..., w_N\}, \mathbb{V} = \{v_1, v_2, ..., v_N\}, v, t_{iou}$;
$\mathbb{W}$ is the list of all windows;
$\mathbb{V}$ is the volume corresponding the windows;
v is the minimum volume;
t_{iou} is the NMS threshold.

Output: Detected windows $\mathbb{D} = \{d_1, d_2, ..., d_K\}$

for $i = 1$ to N **do**
  if $v_i \leq v$ **then**
    $\mathbb{W} \leftarrow \mathbb{W} \backslash \{w_i\}$;
  end
end
$\mathbb{D} \leftarrow \{\}$;
while $\mathbb{P} \neq \emptyset$ **do**
  $m^* \leftarrow \arg\max_m \{v_m \mid v_m \in \mathbb{V}\}$;
  $\mathbb{D} \leftarrow w_{m^*}$;
  $\mathbb{W} \leftarrow \mathbb{W} \backslash \{w_{m^*}\}$;
  for w_i *in* $\mathbb{W}$ **do**
    if $iou(w_{m^*}, w_i) \geq t_{iou}$ **then**
      $\mathbb{W} \leftarrow \mathbb{W} \backslash \{w_i\}$;
      $\mathbb{V} \leftarrow \mathbb{V} \backslash \{v_i\}$;
    end
  end
end
return $\mathbb{W}, \mathbb{V}$;

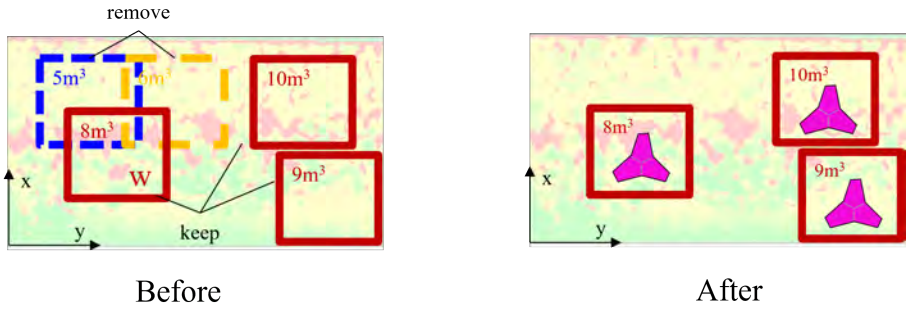


Figure 4.8: Illustration of non-maximum suppression.

4.6 Overflow check

After the dropped blocks are stabilized, we check the height of each fallen block, and if the difference f in height between the centroid of a block and target plane exceeds the threshold, the overflowed block is automatically removed from the stack of new blocks. The setting of f can be adapted freely according to the construction needs. Figure4.9 is

a illustration of overflow check.

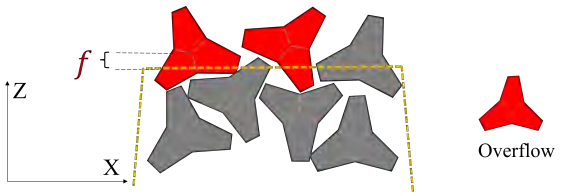


Figure 4.9: Block whose centroid overflows f from the target plane is automatically removed from the stacked blocks.

4.7 Deep Learning-based Initial Block Pose Prediction for stacking

After locating where the blocks needs to be replenished, the next step is to decide in what poses the blocks should be inserted into the insufficient space. Since the initial pose of a block before it drops has a crucial impact on the final stack-up simulation results, this section aims to find an optimal initialized pose of an additional block to fit it tightly to the existing stack of blocks using a physics engine and image-based deep-learning method.

Usually, the 3D irregular packing problem is Np-hard, meaning that it is difficult to find an optimal solution. The computational cost of the exhaustive search selection algorithm is too high to try all possible combinations of subsets. Based on the above considerations, we only use exhaustive search selection in the training phase (e.g., 3000 iterations) to find an approximate optimal solution under a certain space that needs to be filled with blocks for learning. In the simulation process, the learned network can directly predict the initial pose of the block and use it to fill the insufficient space. The experiments in Section 4 and the discussion in Section 4 illustrate the effectiveness of predicting the initialized pose by deep learning.

4.7.1 Overview of the block pose estimation

Our problem can be defined as follows: given an existing block stack as a background, find an optimal initialized pose of a new block so as to best fit to the existing stack. Figure4.10 shows the processing flow of our optimal bock pose estimation. The estimation consists of (1) creation of background block stack scene, (2) finding an optimal block pose by generate-and-test search, (3) training the network for pose estimation, and (4) prediction of the optimal initialized block pose.

4.7.2 Creation of background block stack scene

We used a block CAD model and a physics engine to generate the block stacking scene. The size of the simulated stacking scene is about 220m $\times$ 20m. Then we create a large depth image I_d corresponding to this whole background scene, as shown in Figure4.11.

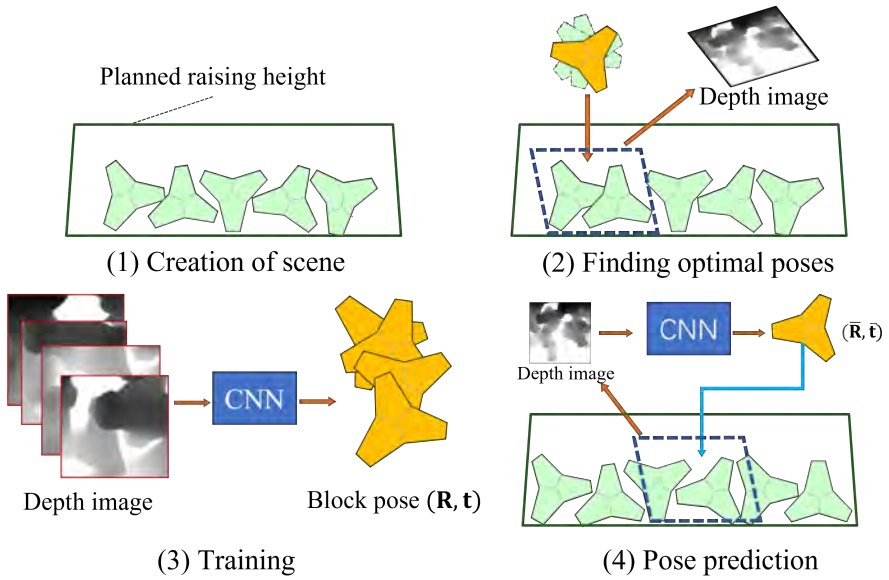


Figure 4.10: The processing flow of our optimal bock pose estimation.

4.7.3 Finding an optimal block pose by generate-and-test

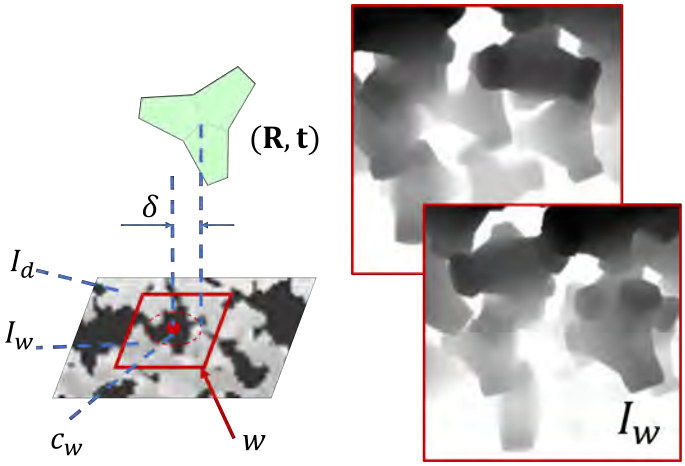
Next, for generating training dataset, we should find a collection of the optimal initialized poses of an additional block so as to best fit to a given existing background block stack scene. To this end, the following generate-and-test search was adopted. First, as shown in Figure 3(a), we place a small 2D window w on the depth image I_d representing the background scene. Then, we set the initial pose (R, t) of a new on w before it drops so that the position t of the block coincide with the centroid c_w of w. To find the best fit pose to the background, we added a uniform random distribution $\sigma \in [-0.5m, 0.5m]$ to each component of t. The initial orientation R is also randomly perturbed. Finally, we drop a new block onto the background scene in the simulator, evaluate the following criteria for optimal block pose, and record the local depth image I_w cut by w.

4.7.4 Criteria for optimal block pose

A reasonable initial block pose needs to satisfy the two conditions.

- **Stability**: There should not be much lateral difference between the initialized pose and final stabilized pose. Therefore, the block's displacement in the horizontal plane should be less than 0.2 m (about 10% of the block size).
- **Compactness**: We want the inserted block to be as close as possible to existing background blocks. So the change in insufficient volume V between a block surface and the target height shown in Figure 4.12 before and after the block insertion should be small enough.

A window w slides at a fixed small interval d on the depth image I_d. At every position of w, local depth image I_w of the background scene is detected as 512×512 pixel image. Examples of I_w are shown in Figure 4.11(b). Then we drop a new block 1000 times at a randomly selected initial pose inside I_w and evaluate the change of the insufficient volume



(a) (b)

Figure 4.11: Block pose generation and samples of the recorded depth images.

V and block's horizontal displacement d for each time. Finally, the pose that gives the smallest change in V and satisfies $d \leq 0.2m$ is taken as the expected optimal block pose, as shown in Figure4.12.

4.7.5 Block Pose Prediction

In our prediction, given an input depth image I, we need to establish the correspondence f between the depth image and a reasonable pose as Equation4.3:

$$(\overline{R}, \overline{t}) = f(I), \tag{4.3}$$

where I is an input depth image, and $\overline{R}$ and $\overline{t}$ are the predicted rotation matrix and translation vector. To implement f as a deep-neural-network, we can use the classical feature extractor [160] or [161] followed by a several fully-connected layers, as show in Figure4.13. To train it, we minimized the loss function,

$$\mathcal{L} = \mathcal{L}_d + \alpha \mathcal{L}_R, \tag{4.4}$$

which combines displacement loss $\mathcal{L}_d$ and rotation loss $\mathcal{L}_R$, and α denotes a balancing constant. As L_d, we used the L2 loss as

$$\mathcal{L}_{\lceil}(t, \overline{t}) = \|t - \overline{t}\|_2, \tag{4.5}$$

where t is the ground truth translation vector. We took $\mathcal{L}_{\mathcal{R}}$ to be the "distance" between different pose defined by:

$$\mathcal{L}_{\mathcal{R}}(R, \overline{R}) = \min_{G \in G_s} \cos^{-1}[\frac{\mathbf{tr}(\overline{R}\,(RG)^T) - 1}{2}], \tag{4.6}$$

where, R is the ground-truth rotation matrix, and G_s is the group of proper symmetries that have no effect on the static state of the object [155]. In our problem, R is encoded by Euler angles. Our network accepts a depth image I_w of size $h \times b \times 1$. After extracting a 2048-dimensional global feature F, the network is divided into two branches that each pass F through a series of fully connected layers to obtain the predicted $\overline{t}$ and $\overline{R}$.

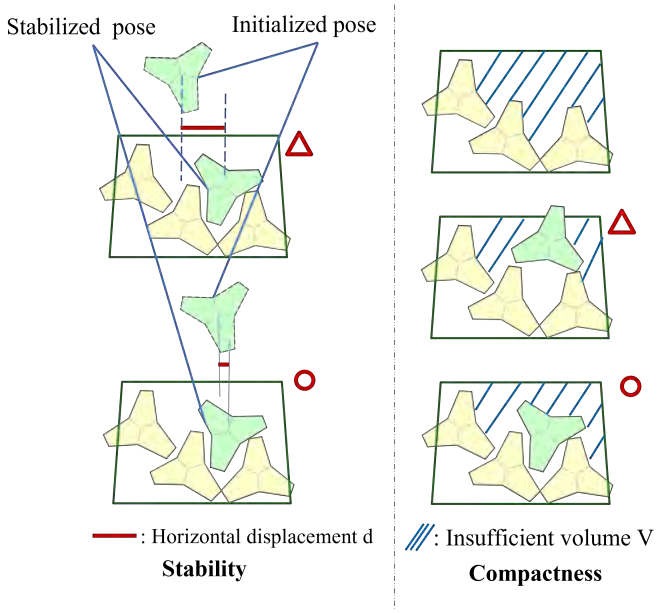


Figure 4.12: Criteria for optimal block pose.

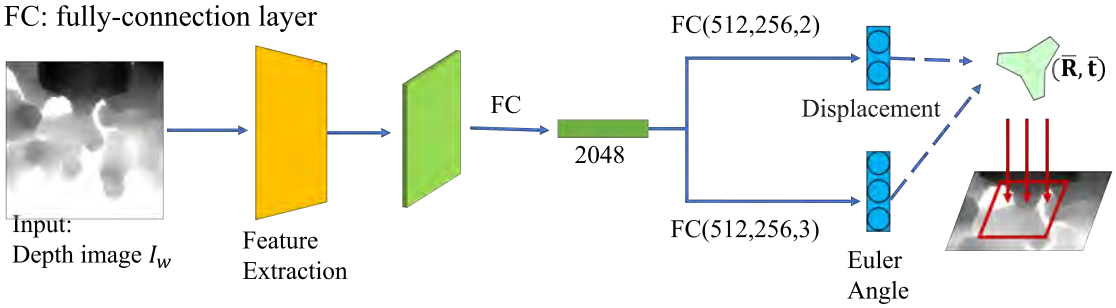


Figure 4.13: Network structure for predicting optimal pose prediction.

4.7.6 Results of optimal pose prediction

We experimented with 800 depth images of 512×512 pixels on the training set and 200 images on the test set. The block type is assumed to be a clinger 6 tons. The displacement and rotation error are estimated by equation 4.5 and 4.6. We use an ADAM optimizer with initial learning rate 0.001, batch size 32. The network is trained for 700 epochs, which took about 24 hours.

The prediction results from the various feature extractors are summarized in Table 4.1. Displacement errors of less than 0.5 m were achieved, but rotational errors were not sufficiently small. Figure 4.14 compares the randomly generated pose and the pose predicted by the network, and it can be seen that the pose generated by the network is more stable and reasonable, while the randomly generated pose causes the block to fall into other positions.

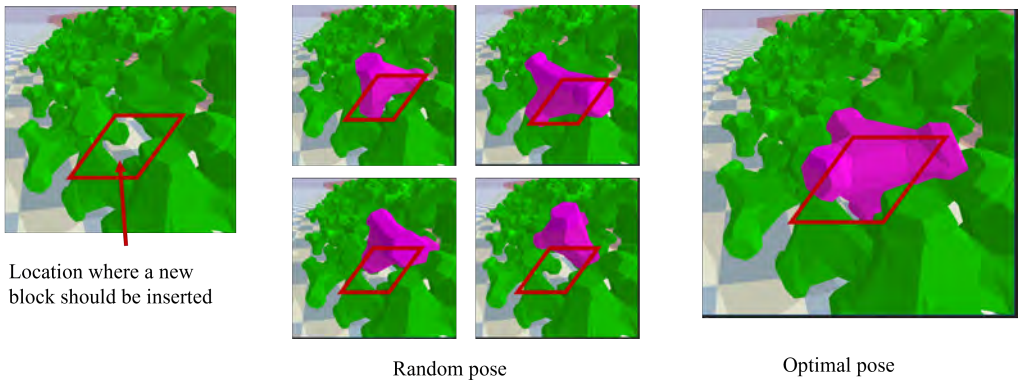


Figure 4.14: Comparison of randomly generated initialized poses and poses predicted by the network.

Table 4.1: Rotation and displacement errors in pose prediction.

Feature extractor	Rotation error(°)	displacement error(m)
Transformer [161]	42.45	0.42
ResNet16 [160]	36.2	0.36
ResNet34 [160]	28.48	0.37
ResNet50 [160]	25.15	0.31

4.8 Experiments

4.8.1 Experimental Sites and Purpose of the Experiments

The breakwater of Sawara port in Hokkaido prefecture, Japan, was used for evaluation of the proposed block detection algorithm, as surface point cloud of the breakwater before and after construction had been recorded respectively. For this port, existing block point clouds above the water surface were measured by UAV and a commercial photogrammetry software, and those below the surface were measured by MBES. Table 4.2 indicates the details of the construction area, and Figure 4.15 shows the top view of the construction areas. Two different types of blocks is shown in Figure 4.16.

Our simulation experiment aims to compare with the real construction results and discuss the accuracy of simulation and the rationality of the current construction plan.

4.8.2 Results of stacking Operation Planning for Replenishing Wave-Dissipating Blocks

In the simulation, we set the vertical gravitational acceleration $9m/s^2$, the mass of block 5t, lateral friction 0.8, spinning friction 0.5, and rolling friction 0.01.

We first imported the existing block poses and target and boundary planes and reconstruct the scene (Figure 4.17 (a)). Then we performed the space checking, as described in Section 4.5, and inserted a couple of new blocks into the detected window (Figure 4.17 (b)) until there are not enough window to insert additional blocks. After the blocks are stabilized, we checked the overflow blocks that exceeded the target plane as shown in Figure 4.17 (c) and eliminated them. The final simulated scene is shown in Figure 4.17 (d). According to the simulation results, the number of new blocks to be supplemented

Table 4.2: Details of the construction area.

Site	Sawara port
Method of measurement	UAV
Construction area [m]	150 × 25
Approx. point density [pts/m^2]	150(MBES) & 900(UAV)
Existing block type	Clinger 5 ton
Supplemental block type	Clinger 6 ton

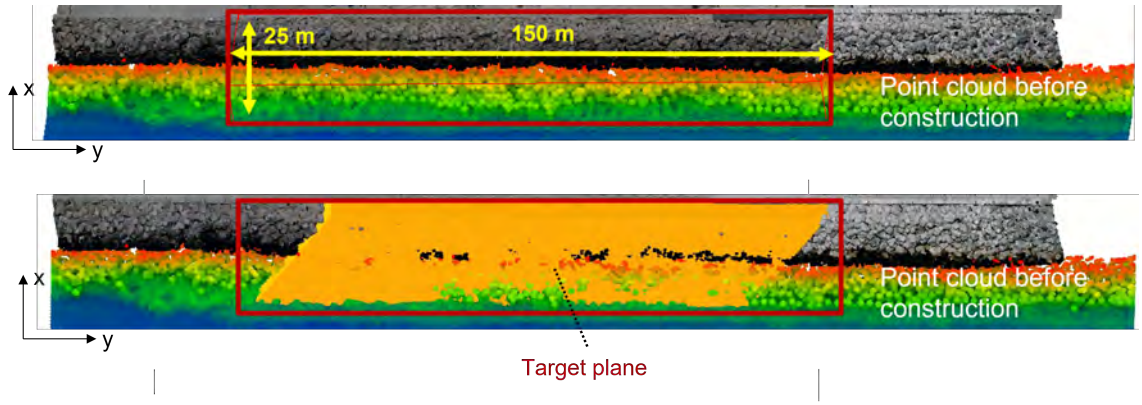


Figure 4.15: The construction area of Sawara port.

the whole construction was estimated as in Table 4.4. As a comparison, the simulation with the randomly generated initialized poses are summarized in Table 4.5. To estimate the fitness between the stacked results and the designed shape, we calculated the distance from the surface point cloud of the stacked CAD model to the designed shape, and the average distances w/ and w/o deep learning are summarized in Table 4.6 and Table 4.7, respectively.

The time taken for each step in the simulation is summarized in Table 4.3, and the total time taken for each simulation is about 1 hour.

4.9 Discussion

This chapter developed a method that can predict the optimal pose of blocks for inserting more blocks in the same replenishment volume. Comparing Tables 4.4 with Table 4.5, the deep learning-based stacking scheme could insert more blocks (around 100) under the same simulation parameters, implying that a more dense block structure of the supplemented blocks was generated. Moreover, comparing the simulation results in Tables 4.6 and 4.7 with the same parameter settings, the deep-learning-based stacking results fit better with the designed target surface, and the average distances were reduced by about 10 mm.

On the other hand, number of supplemented blocks added in the actual maintenance work was 864. As can be seen from Table 4.4, the estimated number based on the deep learning was approximately equal to the actual number, indicating the possibility of generating a strong and solid structure of wave-dissipating block.

We could also find from Figure 4.18 that the measured point cloud of the top surface of the blocks after the actual construction was significantly higher than the design target

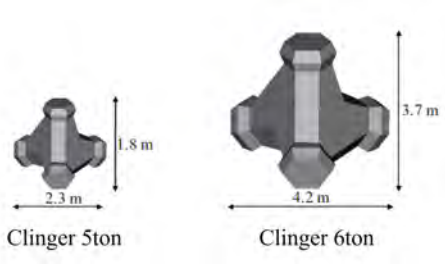
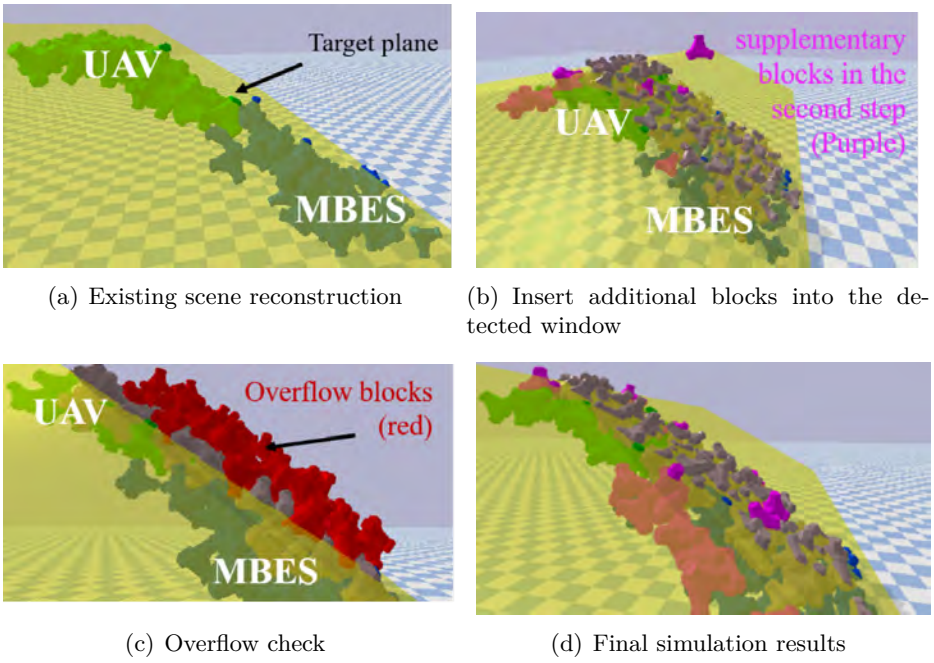


Figure 4.16: The size of Clinger 5 ton and 6 ton.



(a) Existing scene reconstruction

(b) Insert additional blocks into the detected window

(c) Overflow check

(d) Final simulation results

Figure 4.17: An example of the block stack-up simulation process in Sawara port.

height. The results shown that it is possible to plan block replenishment operations more accurately and economically for the design target geometry.

However, the results still need to be verified on other sites, which will be our future work.

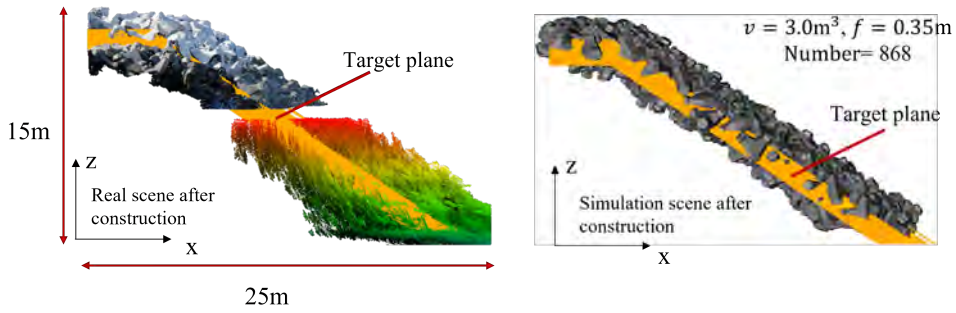
4.10 Conclusion

This chapter used a physics engine to develop a simulation system to automatically generate an optimal operation plan for stacking new blocks on top of already exited wave-dissipating blocks in a stable and compact manner.

- A space detection algorithm was designed to find out the location of the block to be replenished.
- A neural network was employed to predict the stable and compact poses of the supplemental blocks.

Table 4.3: Processing time of simulation.

Processing	Initialization Scene reconstruction)	Generate height map	Space check	Drop	Total time
Time	2 min	1 min × 10	1 min × 10	5 min × 10	1 hour



(a) Real scene after construction (b) Simulation scene after construction

Figure 4.18: Comparison of actual construction results and simulation results.

- We compared the replenishment plans with or without deep learning in a port of Hokkaido.
- It turned out that more blacks could be inserted for the same target replenishment space with deep learning contributing to more robust structures of wave-dissipating blocks.
- The estimated number of blocks was approximately equal to the number of blocks added in the actual work.

According to the construction requirements, we set two parameters, the minimum volume v and the allowable height above the design plane f. However, the simulation accuracy should be also validated at more sites.

Table 4.4: Number of supplemented blocks by simulation w/ deep learning.

The number of blocks	$f = 0m$	$f = 0.1m$	$f = 0.3m$	$f = 0.35m$	$f = 0.5m$
$v = 0.4m^3$	733	781	889	955	1016
$v = 1.0m^3$	698	773	891	925	1017
$v = 2.0m^3$	676	778	894	906	1002
$v = 3.0m^3$	671	739	842	868	898
$v = 4.0m^3$	668	701	761	768	780

Table 4.5: Number of supplemented blocks by simulation w/o deep learning.

The number of blocks	$f = 0m$	$f = 0.1m$	$f = 0.3m$	$f = 0.35m$	$f = 0.5m$
$v = 0.4m^3$	645	683	788	852	903
$v = 1.0m^3$	645	682	792	844	900
$v = 2.0m^3$	645	677	786	823	892
$v = 3.0m^3$	645	686	790	821	884
$v = 4.0m^3$	645	682	756	795	836

Table 4.6: The average distance[m] of the blocks' surface points to the designed shape w/ deep learning.

The number of blocks	$f = 0m$	$f = 0.1m$	$f = 0.3m$	$f = 0.35m$	$f = 0.5m$
$v = 0.4m^3$	0.40	0.40	0.44	0.46	0.50
$v = 1.0m^3$	0.40	0.40	0.43	0.46	0.50
$v = 2.0m^3$	0.41	0.41	0.43	0.46	0.49
$v = 3.0m^3$	0.41	0.42	0.44	0.48	0.50
$v = 4.0m^3$	0.42	0.43	0.44	0.50	0.50

Table 4.7: The average distance[m] of the blocks' surface points to the designed shape w/o deep learning.

The number of blocks	$f = 0m$	$f = 0.1m$	$f = 0.3m$	$f = 0.35m$	$f = 0.5m$
$v = 0.4m^3$	0.42	0.42	0.44	0.46	0.47
$v = 1.0m^3$	0.42	0.43	0.45	0.47	0.48
$v = 2.0m^3$	0.42	0.43	0.45	0.48	0.48
$v = 3.0m^3$	0.42	0.43	0.45	0.48	0.49
$v = 4.0m^3$	0.42	0.43	0.45	0.48	0.48

Chapter 5

Conclusions and Future work

5.1 Conclusions

The research objective of this dissertation is to design and propose an efficient and effective method for stacked object recognition and stacking operation planning combining 3D point cloud representation, deep learning, and a physics engine. To achieve this goal, a framework was developed for fast segmentation of 3D point clouds with an acceptable speed and accuracy. Then, a modified 3D instance segmentation network was applied to the segmentation of the wave-dissipating blocks, and the pose and type of the blocks were estimated from the segmentation results, laying the foundation for simulation. A simulation system was developed for the block supplementation project, which simulated the supplementation process and provided the builder with pre-visualized construction results as well as the stacking plan of blocks.

The conclusions of the thesis are summarized as follows:

(1) In Chapter 2, a fast 3D instance segmentation network, FPCC, was been proposed for robotic bin-picking scene. The network joined a novel clustering algorithm to segment individual object from a point cloud of stacked objects. Experiments shown that FPCC trained by synthetic data demonstrates excellent performance on real-world data compared with existing methods. FPCC achieved 55% precision on IPA Bin-Picking dataset and 80% precision on XA Bin-Picking dataset. FPCC significantly promoted the application of deep learning in industrial production and make robots more intelligent.

(2) In Chapter 3, a novel feature extractor was added into FPCC instead of DGCNN. By stacking graph convolution layers and constructing local information of the point cloud, the perceptual field of each point was expanded at a small cost of the computational overhead. The training data samples were all provided by our developed simulation system based on Pybullet, so that the network could be trained without any manually labeled data. Experiments shown that our method could retrieve 70% 95% blocks in a point cloud scene.

(3) In Chapter 4, a novel method for the planning of replenishment of wave-dissipating blocks was proposed. The method predicts the stable and compact stacking poses of additional blocks that minimizes the height of the center at the time of stacking by combining deep learning and a physics engine. The proposed method has been

applied to the case study of a port in Hokkaido, Japan. The replenishment plans with and without deep learning were compared in this Chapter. Experiment results show that the proposed method could generate a more stable structure of wave-dissipating blocks because more blocks can be inserted densely in the same target replenishment space. In addition, the estimated number of blocks is comparable to the number of blocks added in the actual work.

In summary, the object stacking system developed based on our recognized results could provide valuable guidance for actual construction. s.

5.2 Future work

In order to enhance or enrich the current studies, future researches are going to be carried out in the following aspects:

(1) The proposed 3D instance segmentation scheme can be applied in other similar stacked objects scenes. Our method will significantly promote the application of deep learning in industrial production and make robots more intelligent.

(2) The strategy of how to insert a new block need to be complete and be more flexible.

(3) After reconstructing the scene in a physical engine based on the recognition result, we can employ a reinforcement learning network then randomly inserts blocks in the construction area, and experienced construction workers act as supervisors to evaluate each insertion and give feedback to the network.

References

[1] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., Imagenet large scale visual recognition challenge, International journal of computer vision 115 (3) (2015) 211–252.

[2] Zhirong Wu, S. Song, A. Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, J. Xiao, 3d shapenets: A deep representation for volumetric shapes, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1912–1920. doi:10.1109/CVPR.2015.7298801.

[3] D. Liu, S. Arai, Z. Feng, J. Miao, Y. Xu, J. Kinugawa, K. Kosuge, 2d object localization based point pair feature for pose estimation, in: 2018 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2018, pp. 1119–1124. doi:10.1109/ROBIO.2018.8665097.

[4] R. Girshick, Fast r-cnn, in: 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1440–1448. doi:10.1109/ICCV.2015.169.

[5] E. Shelhamer, J. Long, T. Darrell, Fully convolutional networks for semantic segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (4) (2017) 640–651. doi:10.1109/TPAMI.2016.2572683.

[6] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, in: 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2980–2988. doi:10.1109/ICCV.2017.322.

[7] W. Wang, R. Yu, Q. Huang, U. Neumann, Sgpn: Similarity group proposal network for 3d point cloud instance segmentation, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 2569–2578. doi:10.1109/CVPR.2018.00272.

[8] M. Salim, E. Wåhlin, K. Dembrower, E. Azavedo, T. Foukakis, Y. Liu, K. Smith, M. Eklund, F. Strand, External evaluation of 3 commercial artificial intelligence algorithms for independent assessment of screening mammograms, JAMA oncology 6 (10) (2020) 1581–1588.

[9] Y. Chen, Y. Li, X. Zhang, J. Sun, J. Jia, Focal sparse convolutional networks for 3d object detection, in: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 5418–5427. doi:10.1109/CVPR52688.2022.00535.

[10] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, A. Markham, Randlanet: Efficient semantic segmentation of large-scale point clouds, in: 2020 IEEE/CVF

Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 11105–11114. doi:10.1109/CVPR42600.2020.01112.

[11] V. Casser, S. Pirk, R. Mahjourian, A. Angelova, Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 33, 2019, pp. 8001–8008.

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

[13] K. Kleeberger, C. Landgraf, M. F. Huber, Large-scale 6d object pose estimation dataset for industrial bin-picking, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019, pp. 2573–2578. doi:10.1109/IROS40897.2019.8967594.

[14] G. Waltner, M. Schwarz, S. Ladstätter, A. Weber, P. Luley, M. Lindschinger, I. Schmid, W. Scheitz, H. Bischof, L. Paletta, Personalized dietary self-management using mobile vision-based assistance, in: S. Battiato, G. M. Farinella, M. Leo, G. Gallo (Eds.), New Trends in Image Analysis and Processing – ICIAP 2017, Springer International Publishing, Cham, 2017, pp. 385–393.

[15] M. Grard, E. Dellandréa, L. Chen, Deep multicameral decoding for localizing unoccluded object instances from a single rgb image, International Journal of Computer Vision 128 (5) (2020) 1331–1359.

[16] Y. Xu, S. Arai, F. Tokuda, K. Kosuge, A convolutional neural network for point cloud instance segmentation in cluttered scene trained by synthetic data without color, IEEE Access 8 (2020) 70262–70269. doi:10.1109/ACCESS.2020.2978506.

[17] P. Kim, J. Chen, Y. K. Cho, Slam-driven robotic mapping and registration of 3d point clouds, Automation in Construction 89 (2018) 38–48. doi:https://doi.org/10.1016/j.autcon.2018.01.009.
URL https://www.sciencedirect.com/science/article/pii/S0926580517303990

[18] G. Du, K. Wang, S. Lian, K. Zhao, Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review, Artificial Intelligence Review 54 (3) (2021) 1677–1734.

[19] D. Liu, S. Arai, J. Miao, J. Kinugawa, Z. Wang, K. Kosuge, Point pair feature-based pose estimation with multiple edge appearance models (ppf-meam) for robotic bin picking, Sensors 18 (8) (2018) 2719.

[20] J. Shi, G. S. Koonjul, Real-time grasping planning for robotic bin-picking and kitting applications, IEEE Transactions on Automation Science and Engineering 14 (2) (2017) 809–819. doi:10.1109/TASE.2017.2671434.

[21] M. Bueno, L. Díaz-Vilariño, J. Martínez-Sánchez, H. González Jorge, P. Arias, 3d reconstruction of cubic armoured rubble mound breakwaters from incomplete lidar data, International Journal of Remote Sensing 36 (21) (2015) 5485–5503. doi:10.1080/01431161.2015.1093191.

[22] H. González-Jorge, I. Puente, D. Roca, J. Martínez-Sánchez, B. Conde, P. Arias, Uav photogrammetry application to the monitoring of rubble mound breakwaters, Journal of Performance of Constructed Facilities 30 (1) (2016) 04014194. doi:10.1061/(ASCE)CF.1943-5509.0000702.

[23] P. J. Sousa, A. Cachaço, F. Barros, P. J. Tavares, P. M. Moreira, R. Capitão, M. G. Neves, E. Franco, Structural monitoring of a breakwater using uavs and photogrammetry, Procedia Structural Integrity 37 (2022) 167–172, iCSI 2021 The 4th International Conference on Structural Integrity. doi:https://doi.org/10.1016/j.prostr.2022.01.073.

[24] R. Lemos, M. A. Loja, J. Rodrigues, J. A. Rodrigues, Photogrammetric analysis of rubble mound breakwaters scale model tests, AIMS Environmental Science 3 (3) (2016) 541–559. doi:10.3934/environsci.2016.3.541.

[25] I. Puente, J. Sande, H. González-Jorge, E. Peña-González, E. Maciñeira, J. Martínez-Sánchez, P. Arias, Novel image analysis approach to the terrestrial lidar monitoring of damage in rubble mound breakwaters, Ocean Engineering 91 (2014) 273–280. doi:https://doi.org/10.1016/j.oceaneng.2014.09.011.
URL https://www.sciencedirect.com/science/article/pii/S0029801814003278

[26] D. Gonçalves, G. Gonçalves, J. A. Pérez-Alvávez, U. Andriolo, On the 3d reconstruction of coastal structures by unmanned aerial systems with onboard global navigation satellite system and real-time kinematics and terrestrial laser scanning, Remote Sensing 14 (6) (2022). doi:10.3390/rs14061485.
URL https://www.mdpi.com/2072-4292/14/6/1485

[27] R. E. Musumeci, D. Moltisanti, E. Foti, S. Battiato, G. M. Farinella, 3-d monitoring of rubble mound breakwater damages, Measurement 117 (2018) 347–364. doi:https://doi.org/10.1016/j.measurement.2017.12.020.
URL https://www.sciencedirect.com/science/article/pii/S0263224117307911

[28] Y. Shen, R. Lindenbergh, J. Wang, V. G. Ferreira, Extracting individual bricks from a laser scan point cloud of an unorganized pile of bricks, Remote Sensing 10 (11) (2018). doi:10.3390/rs10111709.
URL https://www.mdpi.com/2072-4292/10/11/1709

[29] I. Puente, R. Lindenbergh, H. González-Jorge, P. Arias, Terrestrial laser scanning for geometry extraction and change monitoring of rubble mound breakwaters, ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences II-5 (2014) 289–295. doi:10.5194/isprsannals-II-5-289-2014.
URL https://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/II-5/289/2014/

[30] Y. Ma, Z. Chen, W. Hu, W. Wang, Packing irregular objects in 3d space via hybrid optimization, Computer Graphics Forum 37 (5) (2018) 49–59. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13490, doi:https://doi.org/10.1111/cgf.13490.
URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13490

[31] Y. Zhao, C. Rausch, C. Haas, Optimizing 3d irregular object packing from 3d scans using metaheuristics, Advanced Engineering Informatics 47 (2021) 101234. doi:https://doi.org/10.1016/j.aei.2020.101234.
URL https://www.sciencedirect.com/science/article/pii/S1474034620302032

[32] X. Zhang, B. Zhou, Y. Zeng, P. Gu, Model layout optimization for solid ground curing rapid prototyping processes, Robotics and Computer-Integrated Manufacturing 18 (1) (2002) 41–51. doi:https://doi.org/10.1016/S0736-5845(01)00022-9.
URL https://www.sciencedirect.com/science/article/pii/S0736584501000229

[33] S.-M. Hur, K.-H. Choi, S.-H. Lee, P.-K. Chang, Determination of fabricating orientation and packing in sls process, Journal of Materials Processing Technology 112 (2) (2001) 236–243. doi:https://doi.org/10.1016/S0924-0136(01)00581-7.
URL https://www.sciencedirect.com/science/article/pii/S0924013601005817

[34] J. Cagan, K. Shimada, S. Yin, A survey of computational approaches to three-dimensional layout problems, Computer-Aided Design 34 (8) (2002) 597–611. doi:https://doi.org/10.1016/S0010-4485(01)00109-9.
URL https://www.sciencedirect.com/science/article/pii/S0010448501001099

[35] T. Romanova, J. Bennell, Y. Stoyan, A. Pankratov, Packing of concave polyhedra with continuous rotations using nonlinear optimisation, European Journal of Operational Research 268 (1) (2018) 37–53. doi:https://doi.org/10.1016/j.ejor.2018.01.025.
URL https://www.sciencedirect.com/science/article/pii/S0377221718300468

[36] B. Tippetts, D. J. Lee, K. Lillywhite, J. Archibald, Review of stereo vision algorithms and their suitability for resource-limited systems, Journal of Real-Time Image Processing 11 (1) (2016) 5–25.

[37] S. Arai, Y. Iwatani, K. Hashimoto, Fast sensor scheduling with communication costs for sensor networks, in: Proceedings of the 2010 American Control Conference, 2010, pp. 295–300.

[38] S. Arai, Y. Iwatani, K. Hashimoto, Fast sensor scheduling for spatially distributed sensors, IEEE Transactions on Automatic Control 56 (8) (2011) 1900–1905. doi:10.1109/TAC.2011.2141450.

[39] X. Lin, J. R. Casas, M. Pardàs, Temporally coherent 3d point cloud video segmentation in generic scenes, IEEE Transactions on Image Processing 27 (6) (2018) 3087–3099. doi:10.1109/TIP.2018.2811541.

[40] S. Zhang, D. V. D. Weide, J. Oliver, Superfast phase-shifting method for 3-d shape measurement, Opt. Express 18 (9) (2010) 9684–9689. doi:10.1364/OE.18.009684.

[41] H. B. Wu, Y. Chen, M. Y. Wu, C. R. Guan, X. Y. Yu, 3d measurement technology by structured light using stripe-edge-based gray code, Journal of Physics: Conference Series 48 (2006) 537–541. doi:10.1088/1742-6596/48/1/101.

[42] N. Chiba, S. Arai, K. Hashimoto, Feedback projection for 3d measurements under complex lighting conditions, in: 2017 American Control Conference (ACC), 2017, pp. 4649–4656. doi:10.23919/ACC.2017.7963673.

[43] S. Yu, J. Zhang, X. Yu, X. Sun, H. Wu, X. Liu, 3d measurement using combined gray code and dual-frequency phase-shifting approach, Optics Communications 413 (2018) 283 – 290. doi:https://doi.org/10.1016/j.optcom.2017.12.071.

[44] W. Lu, Y. Zhou, G. Wan, S. Hou, S. Song, L3-net: Towards learning based lidar localization for autonomous driving, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 6382–6391. doi:10.1109/CVPR.2019.00655.

[45] S. Arai, A. L. Pettersson, K. Hashimoto, Fast prediction of a worker's reaching motion without a skeleton model (f-premo), IEEE Access 8 (2020) 90340–90350.

[46] J. Kinugawa, A. Kanazawa, S. Arai, K. Kosuge, Adaptive task scheduling for an assembly task coworker robot based on incremental learning of human's motion patterns, IEEE Robotics and Automation Letters 2 (2) (2017) 856–863. doi:10.1109/LRA.2017.2655565.

[47] K.-B. Park, M. Kim, S. H. Choi, J. Y. Lee, Deep learning-based smart task assistance in wearable augmented reality, Robotics and Computer-Integrated Manufacturing 63 (2020) 101887. doi:https://doi.org/10.1016/j.rcim.2019.101887.

[48] C. Zhuang, Z. Wang, H. Zhao, H. Ding, Semantic part segmentation method based 3d object pose estimation with rgb-d images for bin-picking, Robotics and Computer-Integrated Manufacturing 68 (2021) 102086. doi:https://doi.org/10.1016/j.rcim.2020.102086.

[49] D. Liu, S. Arai, F. Tokuda, Y. Xu, J. Kinugawa, K. Kosuge, Deep-learning based robust edge detection for point pair feature-based pose estimation with multiple edge appearance models, in: 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2019, pp. 2920–2925. doi:10.1109/ROBIO49542.2019.8961752.

[50] Z. Dong, S. Liu, T. Zhou, H. Cheng, L. Zeng, X. Yu, H. Liu, Ppr-net:point-wise pose regression network for instance segmentation and 6d pose estimation in bin-picking scenarios, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019, pp. 1773–1780. doi:10.1109/IROS40897.2019.8967895.

[51] D. Liu, S. Arai, Y. Xu, F. Tokuda, K. Kosuge, 6d pose estimation of occlusion-free objects for robotic bin-picking using ppf-meam with 2d images (occlusion-free ppf-meam), IEEE Access (2021) 1–1doi:10.1109/ACCESS.2021.3068467.

[52] D. Li, H. Wang, N. Liu, X. Wang, J. Xu, 3d object recognition and pose estimation from point cloud using stably observed point pair feature, IEEE Access 8 (2020) 44335–44345. doi:10.1109/ACCESS.2020.2978255.

[53] X. Liang, H. Cheng, Rgb-d camera based 3d object pose estimation and grasping, in: 2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER), 2019, pp. 1279–1284. doi:10.1109/CYBER46603.2019.9066550.

[54] L. Yang, Y. Liu, J. Peng, Z. Liang, A novel system for off-line 3d seam extraction and path planning based on point cloud segmentation for arc welding robot, Robotics and Computer-Integrated Manufacturing 64 (2020) 101929. doi:https://doi.org/10.1016/j.rcim.2019.101929.

[55] G. Georgakis, S. Karanam, Z. Wu, J. Ernst, J. Košecká, End-to-end learning of keypoint detector and descriptor for pose invariant 3d matching, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 1965–1973. doi:10.1109/CVPR.2018.00210.

[56] R. Li, X. Li, C. Fu, D. Cohen-Or, P. Heng, Pu-gan: A point cloud upsampling adversarial network, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 7202–7211. doi:10.1109/ICCV.2019.00730.

[57] S. Arai, N. Fukuchi, K. Hashimoto, Fast detection algorithm for 3d keypoints (fada-3k), IEEE Access 8 (2020) 189556–189564. doi:10.1109/ACCESS.2020.3025534.

[58] F. G. Zanjani, A. Pourtaherian, S. Zinger, D. A. Moin, F. Claessen, T. Cherici, S. Parinussa, P. H. de With, Mask-mcnet: Tooth instance segmentation in 3d point clouds of intra-oral scans, Neurocomputing 453 (2021) 286–298. doi:https://doi.org/10.1016/j.neucom.2020.06.145.
URL https://www.sciencedirect.com/science/article/pii/S0925231221001041

[59] K. Wada, K. Okada, M. Inaba, Joint learning of instance and semantic segmentation for robotic pick-and-place with heavy occlusions in clutter, in: 2019 International Conference on Robotics and Automation (ICRA), 2019, pp. 9558–9564. doi:10.1109/ICRA.2019.8793783.

[60] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, N. Fazeli, F. Alet, N. C. Dafle, R. Holladay, I. Morena, P. Qu Nair, D. Green, I. Taylor, W. Liu, T. Funkhouser, A. Rodriguez, Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), 2018, pp. 3750–3757. doi:10.1109/ICRA.2018.8461044.

[61] A. Zeng, K.-T. Yu, S. Song, D. Suo, E. Walker, A. Rodriguez, J. Xiao, Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge,

in: 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 1386–1383. doi:10.1109/ICRA.2017.7989165.

[62] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft coco: Common objects in context, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), Computer Vision – ECCV 2014, Springer International Publishing, Cham, 2014, pp. 740–755.

[63] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, A. Torralba, Places: A 10 million image database for scene recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 40 (6) (2018) 1452–1464. doi:10.1109/TPAMI.2017.2723009.

[64] R. Q. Charles, H. Su, M. Kaichun, L. J. Guibas, Pointnet: Deep learning on point sets for 3d classification and segmentation, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 77–85. doi:10.1109/CVPR.2017.16.

[65] D. Maturana, S. Scherer, Voxnet: A 3d convolutional neural network for real-time object recognition, in: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015, pp. 922–928. doi:10.1109/IROS.2015.7353481.

[66] C. R. Qi, L. Yi, H. Su, L. J. Guibas, Pointnet++: Deep hierarchical feature learning on point sets in a metric space, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, Curran Associates Inc., Red Hook, NY, USA, 2017, p. 5105–5114.

[67] Q. Pham, T. Nguyen, B. Hua, G. Roig, S. Yeung, Jsis3d: Joint semantic-instance segmentation of 3d point clouds with multi-task pointwise networks and multi-value conditional random fields, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 8819–8828. doi:10.1109/CVPR.2019.00903.

[68] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, J. M. Solomon, Dynamic graph cnn for learning on point clouds, ACM Trans. Graph. 38 (5) (Oct. 2019). doi:10.1145/3326362.

[69] L. Wang, Y. Huang, Y. Hou, S. Zhang, J. Shan, Graph attention convolution for point cloud semantic segmentation, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 10288–10297. doi:10.1109/CVPR.2019.01054.

[70] H. Zhao, L. Jiang, C. Fu, J. Jia, Pointweb: Enhancing local neighborhood features for point cloud processing, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 5560–5568. doi:10.1109/CVPR.2019.00571.

[71] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, B. Chen, Pointcnn: Convolution on x-transformed points, in: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), Advances in Neural Information Processing Systems, Vol. 31, Curran Associates, Inc., 2018, pp. 820–830.

[72] Q. Huang, W. Wang, U. Neumann, Recurrent slice networks for 3d segmentation of point clouds, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 2626–2635. doi:10.1109/CVPR.2018.00278.

[73] Z. Xie, J. Chen, B. Peng, Point clouds learning with attention-based graph convolution networks, Neurocomputing 402 (2020) 245–255. doi:https://doi.org/10.1016/j.neucom.2020.03.086.
URL https://www.sciencedirect.com/science/article/pii/S0925231220304732

[74] Y. Cui, X. Liu, H. Liu, J. Zhang, A. Zare, B. Fan, Geometric attentional dynamic graph convolutional neural networks for point cloud analysis, Neurocomputing 432 (2021) 300–310. doi:https://doi.org/10.1016/j.neucom.2020.12.067.
URL https://www.sciencedirect.com/science/article/pii/S0925231220319676

[75] R. Li, Y. Zhang, D. Niu, G. Yang, N. Zafar, C. Zhang, X. Zhao, Pointvgg: Graph convolutional network with progressive aggregating features on point clouds, Neurocomputing 429 (2021) 187–198. doi:https://doi.org/10.1016/j.neucom.2020.10.086.
URL https://www.sciencedirect.com/science/article/pii/S0925231220316994

[76] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, S. Savarese, 3d semantic parsing of large-scale indoor spaces, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1534–1543. doi:10.1109/CVPR.2016.170.

[77] B. Hua, Q. Pham, D. T. Nguyen, M. Tran, L. Yu, S. Yeung, Scenenn: A scene meshes dataset with annotations, in: 2016 Fourth International Conference on 3D Vision (3DV), 2016, pp. 92–101. doi:10.1109/3DV.2016.18.

[78] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the kitti vision benchmark suite, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 3354–3361. doi:10.1109/CVPR.2012.6248074.

[79] X. Huang, P. Wang, X. Cheng, D. Zhou, Q. Geng, R. Yang, The apolloscape open dataset for autonomous driving and its application, arXiv preprint arXiv:1803.06184 (2018).

[80] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, S. Birchfield, Training deep networks with synthetic data: Bridging the reality gap by domain randomization, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2018, pp. 1082–10828. doi:10.1109/CVPRW.2018.00143.

[81] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, N. Navab, Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again, in: 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 1530–1538. doi:10.1109/ICCV.2017.169.

[82] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, R. Triebel, Implicit 3d orientation learning for 6d object detection from rgb images, in: V. Ferrari, M. Hebert, C. Sminchisescu, Y. Weiss (Eds.), Computer Vision – ECCV 2018, Springer International Publishing, Cham, 2018, pp. 712–729.

[83] M. Danielczuk, M. Matl, S. Gupta, A. Li, A. Lee, J. Mahler, K. Goldberg, Segmenting unknown 3d objects from real depth images using mask r-cnn trained on synthetic data, in: 2019 International Conference on Robotics and Automation (ICRA), 2019, pp. 7283–7290. doi:10.1109/ICRA.2019.8793744.

[84] X. Wang, S. Liu, X. Shen, C. Shen, J. Jia, Associatively segmenting instances and semantics in point clouds, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 4091–4100. doi:10.1109/CVPR.2019.00422.

[85] L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, J. Jia, Pointgroup: Dual-set point grouping for 3d instance segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 4866–4875. doi:10.1109/CVPR42600.2020.00492.

[86] S. Chen, J. Fang, Q. Zhang, W. Liu, X. Wang, Hierarchical aggregation for 3d instance segmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 15467–15476.

[87] L. Han, T. Zheng, L. Xu, L. Fang, Occuseg: Occupancy-aware 3d instance segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 2937–2946. doi:10.1109/CVPR42600.2020.00301.

[88] F. Engelmann, M. Bokeloh, A. Fathi, B. Leibe, M. Nießner, 3d-mpa: Multi-proposal aggregation for 3d semantic instance segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 9028–9037. doi:10.1109/CVPR42600.2020.00905.

[89] X. Li, Y. Li, C. Shen, A. Dick, A. V. D. Hengel, Contextual hypergraph modeling for salient object detection, in: 2013 IEEE International Conference on Computer Vision, 2013, pp. 3328–3335. doi:10.1109/ICCV.2013.413.

[90] B. Douillard, J. Underwood, N. Kuntz, V. Vlaskine, A. Quadros, P. Morton, A. Frenkel, On the segmentation of 3d lidar point clouds, in: 2011 IEEE International Conference on Robotics and Automation, 2011, pp. 2798–2805. doi:10.1109/ICRA.2011.5979818.

[91] M. Johnson-Roberson, J. Bohg, M. Björkman, D. Kragic, Attention-based active 3d point cloud segmentation, in: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010, pp. 1165–1170. doi:10.1109/IROS.2010.5649872.

[92] A. Kirillov, K. He, R. Girshick, C. Rother, P. Dollár, Panoptic segmentation, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 9396–9405. doi:10.1109/CVPR.2019.00963.

[93] A. Kirillov, R. Girshick, K. He, P. Dollár, Panoptic feature pyramid networks, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 6392–6401. doi:10.1109/CVPR.2019.00656.

[94] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (6) (2017) 1137–1149. doi:10.1109/TPAMI.2016.2577031.

[95] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature pyramid networks for object detection, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 936–944. doi:10.1109/CVPR.2017.106.

[96] S. Liu, L. Qi, H. Qin, J. Shi, J. Jia, Path aggregation network for instance segmentation, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 8759–8768. doi:10.1109/CVPR.2018.00913.

[97] L.-C. Chen, A. Hermans, G. Papandreou, F. Schroff, P. Wang, H. Adam, Masklab: Instance segmentation by refining object detection with semantic and direction features, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 4013–4022. doi:10.1109/CVPR.2018.00422.

[98] X. Chen, R. Girshick, K. He, P. Dollar, Tensormask: A foundation for dense object segmentation, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 2061–2069. doi:10.1109/ICCV.2019.00215.

[99] D. Bolya, C. Zhou, F. Xiao, Y. J. Lee, Yolact: Real-time instance segmentation, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 9156–9165. doi:10.1109/ICCV.2019.00925.

[100] H. Chen, K. Sun, Z. Tian, C. Shen, Y. Huang, Y. Yan, Blendmask: Top-down meets bottom-up for instance segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 8570–8578. doi:10.1109/CVPR42600.2020.00860.

[101] Y. Lee, J. Park, Centermask: Real-time anchor-free instance segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 13903–13912. doi:10.1109/CVPR42600.2020.01392.

[102] E. Xie, P. Sun, X. Song, W. Wang, X. Liu, D. Liang, C. Shen, P. Luo, Polarmask: Single shot instance segmentation with polar representation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 12190–12199. doi:10.1109/CVPR42600.2020.01221.

[103] B. De Brabandere, D. Neven, L. Van Gool, Semantic instance segmentation with a discriminative loss function, arXiv preprint arXiv:1708.02551 (2017).

[104] M. Bai, R. Urtasun, Deep watershed transform for instance segmentation, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2858–2866. doi:10.1109/CVPR.2017.305.

[105] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, The cityscapes dataset for semantic urban scene understanding, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 3213–3223. doi:10.1109/CVPR.2016.350.

[106] J. Hou, A. Dai, M. Nießner, 3d-sis: 3d semantic instance segmentation of rgb-d scans, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 4416–4425. doi:10.1109/CVPR.2019.00455.

[107] L. Yi, W. Zhao, H. Wang, M. Sung, L. J. Guibas, Gspn: Generative shape proposal network for 3d instance segmentation in point cloud, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 3942–3951. doi:10.1109/CVPR.2019.00407.

[108] B. Yang, J. Wang, R. Clark, Q. Hu, S. Wang, A. Markham, N. Trigoni, Learning object bounding boxes for 3d instance segmentation on point clouds, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems, Vol. 32, Curran Associates, Inc., 2019, pp. 6740–6749.

[109] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, M. Nießner, Scannet: Richly-annotated 3d reconstructions of indoor scenes, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2432–2443. doi:10.1109/CVPR.2017.261.

[110] J. Lahoud, B. Ghanem, M. R. Oswald, M. Pollefeys, 3d instance segmentation via multi-task metric learning, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 9255–9265. doi:10.1109/ICCV.2019.00935.

[111] B. Zhang, P. Wonka, Point cloud instance segmentation using probabilistic embeddings, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 8879–8888. doi:10.1109/CVPR46437.2021.00877.

[112] M. Schwarz, A. Milan, A. S. Periyasamy, S. Behnke, Rgb-d object detection and semantic segmentation for autonomous manipulation in clutter, The International Journal of Robotics Research 37 (4-5) (2018) 437–451. arXiv:https://doi.org/10.1177/0278364917713117, doi:10.1177/0278364917713117.
URL https://doi.org/10.1177/0278364917713117

[113] K. Wada, K. Okada, M. Inaba, Joint learning of instance and semantic segmentation for robotic pick-and-place with heavy occlusions in clutter, in: 2019 International Conference on Robotics and Automation (ICRA), 2019, pp. 9558–9564. doi:10.1109/ICRA.2019.8793783.

[114] C. Zhuang, Z. Wang, H. Zhao, H. Ding, Semantic part segmentation method based 3d object pose estimation with rgb-d images for bin-picking, Robotics and Computer-Integrated Manufacturing 68 (2021) 102086. doi:https://doi.org/10.1016/j.rcim.2020.102086.
URL https://www.sciencedirect.com/science/article/pii/S0736584520302969

[115] T.-T. Le, C.-Y. Lin, Bin-picking for planar objects based on a deep learning network: A case study of usb packs, Sensors 19 (16) (2019). doi:10.3390/s19163602.
URL https://www.mdpi.com/1424-8220/19/16/3602

[116] W. Abbeloos, T. Goedemé, Point pair feature based object detection for random bin picking, in: 2016 13th Conference on Computer and Robot Vision (CRV), 2016, pp. 432–439. doi:10.1109/CRV.2016.59.

[117] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

[118] A. Stateczny, W. Błaszczak-Bąk, A. Sobieraj-Żłobińska, W. Motyl, M. Wisniewska, Methodology for processing of 3d multibeam sonar big data for comparative navigation, Remote Sensing 11 (19) (2019). doi:10.3390/rs11192245.
URL https://www.mdpi.com/2072-4292/11/19/2245

[119] M. Kulawiak, Z. Lubniewski, Processing of lidar and multibeam sonar point cloud data for 3d surface and object shape reconstruction, in: 2016 Baltic Geodetic Congress (BGC Geomatics), 2016, pp. 187–190. doi:10.1109/BGC.Geomatics.2016.41.

[120] E. Alevizos, D. Oikonomou, A. V. Argyriou, D. D. Alexakis, Fusion of drone-based rgb and multi-spectral imagery for shallow water bathymetry inversion, Remote Sensing 14 (5) (2022). doi:10.3390/rs14051127.
URL https://www.mdpi.com/2072-4292/14/5/1127

[121] D. Wang, S. Xing, Y. He, J. Yu, Q. Xu, P. Li, Evaluation of a new lightweight uav-borne topo-bathymetric lidar for shallow water bathymetry and object detection, Sensors 22 (4) (2022). doi:10.3390/s22041379.
URL https://www.mdpi.com/1424-8220/22/4/1379

[122] P. S. Dąbrowski, C. Specht, M. Specht, P. Burdziakowski, A. Makar, O. Lewicka, Integration of multi-source geospatial data from gnss receivers, terrestrial laser scanners, and unmanned aerial vehicles, Canadian Journal of Remote Sensing 47 (4) (2021) 621–634. doi:10.1080/07038992.2021.1922879.

[123] Y. Xu, S. Arai, D. Liu, F. Lin, K. Kosuge, Fpcc: Fast point cloud clustering-based instance segmentation for industrial bin-picking, Neurocomputing 494 (2022) 255–268. doi:https://doi.org/10.1016/j.neucom.2022.04.023.
URL https://www.sciencedirect.com/science/article/pii/S0925231222003915

[124] B. Drost, M. Ulrich, N. Navab, S. Ilic, Model globally, match locally: Efficient and robust 3d object recognition, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2010, pp. 998–1005. doi:10.1109/CVPR.2010.5540108.

[125] P. J. Besl, N. D. McKay, Method for registration of 3-D shapes, in: P. S. Schenker (Ed.), Sensor Fusion IV: Control Paradigms and Data Structures, Vol. 1611, International Society for Optics and Photonics, SPIE, 1992, pp. 586 – 606. doi:10.1117/12.57955.
URL https://doi.org/10.1117/12.57955

[126] E. Coumans, Y. Bai, Pybullet, a python module for physics simulation for games, robotics and machine learning, http://pybullet.org (2016–2021).

[127] M. Bueno, L. Díaz-Vilariño, H. González-Jorge, J. Martínez-Sánchez, P. Arias, Automatic modelling of rubble mound breakwaters from lidar data, The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 40 (3) (2015) 9.

[128] K. Tulsi, D. Phelp, Monitoring and maintenance of breakwaters which protect port entrances (2009).

[129] Á. Campos, C. Castillo, R. Molina-Sanchez, Damage in rubble mound breakwaters. part i: Historical review of damage models, Journal of Marine Science and Engineering 8 (5) (2020). doi:10.3390/jmse8050317.
URL https://www.mdpi.com/2077-1312/8/5/317

[130] Á. Campos, R. Molina-Sanchez, C. Castillo, Damage in rubble mound breakwaters. part ii: Review of the definition, parameterization, and measurement of damage, Journal of Marine Science and Engineering 8 (5) (2020). doi:10.3390/jmse8050306.
URL https://www.mdpi.com/2077-1312/8/5/306

[131] R. Lemos, M. T. Reis, C. J. Fortes, E. Peña, J. Sande, A. Figuero, A. Alvarellos, E. Laino, J. Santos, N. B. Kerpen, Measuring armour layer damage in rubble-mound breakwaters under oblique wave incidence, 2019. doi:10.18451/978-3-939230-64-9_030.

[132] F. Zhang, C. Guan, J. Fang, S. Bai, R. Yang, P. H. Torr, V. Prisacariu, Instance segmentation of lidar point clouds, in: 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 9448–9455. doi:10.1109/ICRA40945.2020.9196622.

[133] A. Walicka, N. Pfeifer, Automatic segmentation of individual grains from a terrestrial laser scanning point cloud of a mountain river bed, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 15 (2022) 1389–1410. doi:10.1109/JSTARS.2022.3141892.

[134] H. Luo, K. Khoshelham, C. Chen, H. He, Individual tree extraction from urban mobile laser scanning point clouds using deep pointwise direction embedding, ISPRS Journal of Photogrammetry and Remote Sensing 175 (2021) 326–339. doi:https://doi.org/10.1016/j.isprsjprs.2021.03.002.
URL https://www.sciencedirect.com/science/article/pii/S0924271621000654

[135] A. Djuricic, P. Dorninger, C. Nothegger, M. Harzhauser, B. Székely, S. Rasztovits, O. Mandic, G. Molnár, N. Pfeifer, High-resolution 3D surface modeling of a fossil oyster reef, Geosphere 12 (5) (2016) 1457–1477. doi:10.1130/GES01282.1.
URL https://doi.org/10.1130/GES01282.1

[136] S. Salti, F. Tombari, L. Di Stefano, Shot: Unique signatures of histograms for surface and texture description, Computer Vision and Image Understanding 125 (2014) 251–264. doi:https://doi.org/10.1016/j.cviu.2014.04.011.

URL https://www.sciencedirect.com/science/article/pii/S1077314214000988

[137] R. B. Rusu, N. Blodow, Z. C. Marton, M. Beetz, Aligning point cloud views using persistent feature histograms, in: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008, pp. 3384–3391. doi:10.1109/IROS.2008.4650967.

[138] S. Hinterstoisser, V. Lepetit, N. Rajkumar, K. Konolige, Going further with point pair features, in: B. Leibe, J. Matas, N. Sebe, M. Welling (Eds.), Computer Vision – ECCV 2016, Springer International Publishing, Cham, 2016, pp. 834–848.

[139] T. Birdal, S. Ilic, Point pair features based object detection and pose estimation revisited, in: 2015 International Conference on 3D Vision, 2015, pp. 527–535. doi:10.1109/3DV.2015.65.

[140] Y. Xiang, T. Schmidt, V. Narayanan, D. Fox, Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes, 2018.

[141] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, S. Savarese, Densefusion: 6d object pose estimation by iterative dense fusion, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 3338–3347. doi:10.1109/CVPR.2019.00346.

[142] L. Zeng, W. J. Lv, Z. K. Dong, Y. J. Liu, Ppr-net++: Accurate 6-d pose estimation in stacked scenarios, IEEE Transactions on Automation Science and Engineering (2021) 1–13doi:10.1109/TASE.2021.3108800.

[143] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, D. Fox, Poserbpf: A rao–blackwellized particle filter for 6-d object pose tracking, IEEE Transactions on Robotics 37 (5) (2021) 1328–1342. doi:10.1109/TRO.2021.3056043.

[144] X. Deng, Y. Xiang, A. Mousavian, C. Eppner, T. Bretl, D. Fox, Self-supervised 6d object pose estimation for robot manipulation, in: 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 3665–3671. doi:10.1109/ICRA40945.2020.9196714.

[145] Y. Yin, Y. Cai, H. Wang, B. Chen, Fishermatch: Semi-supervised rotation regression via entropy-based filtering, CVPR (2022).

[146] G. Li, M. Mueller, G. Qian, I. C. Delgadillo Perez, A. Abualshour, A. K. Thabet, B. Ghanem, Deepgcns: Making gcns go as deep as cnns, IEEE Transactions on Pattern Analysis and Machine Intelligence (2021) 1–1doi:10.1109/TPAMI.2021.3074057.

[147] W. Luo, Y. Li, R. Urtasun, R. Zemel, Understanding the effective receptive field in deep convolutional neural networks, in: D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems, Vol. 29, Curran Associates, Inc., 2016.

[148] F. Engelmann, T. Kontogianni, B. Leibe, Dilated point convolutions: On the receptive field size of point convolutions on 3d point clouds, in: 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 9463–9469. doi:10.1109/ICRA40945.2020.9197503.

[149] R. Li, S. Wang, F. Zhu, J. Huang, Adaptive graph convolutional neural networks, Proceedings of the AAAI Conference on Artificial Intelligence 32 (1) (2018).
URL https://ojs.aaai.org/index.php/AAAI/article/view/11691

[150] B. Yang, S. Wang, A. Markham, N. Trigoni, Robust attentional aggregation of deep feature sets for multi-view 3d reconstruction, International Journal of Computer Vision 128 (1) (2020) 53–73. doi:https://doi.org/10.1007/s11263-019-01217-w.

[151] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, P. Abbeel, Domain randomization for transferring deep neural networks from simulation to the real world, in: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017, pp. 23–30. doi:10.1109/IROS.2017.8202133.

[152] B. Planche, Z. Wu, K. Ma, S. Sun, S. Kluckner, O. Lehmann, T. Chen, A. Hutter, S. Zakharov, H. Kosch, J. Ernst, Depthsynth: Real-time realistic synthetic data generation from cad models for 2.5d recognition, in: 2017 International Conference on 3D Vision (3DV), 2017, pp. 1–10. doi:10.1109/3DV.2017.00011.

[153] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, S. Birchfield, Training deep networks with synthetic data: Bridging the reality gap by domain randomization, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2018, pp. 1082–10828. doi:10.1109/CVPRW.2018.00143.

[154] S. Katz, A. Tal, R. Basri, Direct visibility of point sets, in: ACM SIGGRAPH 2007 Papers, SIGGRAPH '07, Association for Computing Machinery, New York, NY, USA, 2007, p. 24–es. doi:10.1145/1275808.1276407.
URL https://doi.org/10.1145/1275808.1276407

[155] R. Brégier, F. Devernay, L. Leyrit, J. L. Crowley, Defining the pose of any 3d rigid object and an associated distance, International Journal of Computer Vision 126 (6) (2018) 571–596.

[156] A. Johnson, M. Hebert, Using spin images for efficient object recognition in cluttered 3d scenes, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (5) (1999) 433–449. doi:10.1109/34.765655.

[157] J. MITSUI, S. ichi KUBOTA, M. HASHIDA, S. NOBORU, Development of a real-time placement simulation method for wave-dissipating blocks, Journal of Japan Society of Civil Engineers, Ser. B2 (Coastal Engineering) 78 (2) (2022) 685–690.

[158] K. Mamou, F. Ghorbel, A simple and efficient approach for 3d mesh approximate convex decomposition, in: 2009 16th IEEE International Conference on Image Processing (ICIP), 2009, pp. 3501–3504. doi:10.1109/ICIP.2009.5414068.

[159] K. Mamou, E. Lengyel, A. Peters, Volumetric hierarchical approximate convex decomposition, in: Game Engine Gems 3, AK Peters, 2016, pp. 141–158.

[160] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[161] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, A. Ku, D. Tran, Image transformer, in: J. Dy, A. Krause (Eds.), Proceedings of the 35th International Conference on Machine Learning, Vol. 80 of Proceedings of Machine Learning Research, PMLR, 2018, pp. 4055–4064.
URL https://proceedings.mlr.press/v80/parmar18a.html

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisor, Professor Satoshi Kanai, an amiable and caring scholar, whose immense knowledge, impressive friendliness and patience make me successfully complete this dissertation. His conscientious academic spirit and open-minded personality inspire me both in academic study and daily life. Without his supervision, I could not make my accomplishments possible. I am so proud of being his student.

Also, I would like to express my sincere gratitude to my advisor, Associate Professor Hiroaki Date who have shared many good ideas with me. His insightful guidance makes my doctoral course meaningful.

Special thanks go to Professor Masahiko Onosato and Professor Atsushi Konno for their valuable advice on this dissertation.

I want to thank Mr. Sano of Alpha Hydraulic Engineering Consultants Co.,Ltd. for providing such an interesting topic for my research and a lot of data. I am appreciated to have the opportunity to be able to work on a practical problem.

I want to thank all members and former members in Digital Geometry Processing Lab, for their continuous support in these three years. I'm very lucky to be your junior. Especially thanks to secretary Ms.Tuji and Ms.Miyao for solving my problems in university.

Thanks to Hokkaido University DX Doctoral Fellowship Program for granting me financial support, research support, and career path support so that I can focus on my research without worrying about living expenses. I wish I could repay this kindness to Hokkaido University in the future.

I have learned a valuable lesson on conducting research from Associate Professor Shogo Arai, whose academic sophistication deeply impressed me.

Last but not least, I would like to thank my parents for their encouragement and support. Thanks to my little sister, you have been a source of courage whenever I faced difficulties head-on. Looking forward to your growth.

Yajun, Xu

2022, Japan

Sapporo, Hokkaido, Japan

Appendix A

Publication list

Journal:

[1] Y, Xu, S. Kanai, H. Date, T. Sano, "Deep-learning-based three-dimensional detection of individual wave-dissipating blocks from as-built point clouds measured by UAV-photogrammetry and multibeam echo-sounder," Remote Sensing, vol.14, no.21, p. 5575, 2022.

[2] Y. Xu, S. Arai, H. Lin, F. Tokuda and K. Kosuge: "FPCC: Fast Point Cloud Clustering-based Instance Segmentation for Industrial Bin-picking," Neurocomputing, vol.494, pp. 255–268, 2022.

International Conference Proceedings:

None.

Domestic Conferences Proceedings:

[3] Y, Xu, S. Kanai, H. Date, T. Sano and T. Teranishi, "Deep learning-based optimal pose estimation of wave-dissipating blocks for regular supplementary works," Proceedings of 2022 Annual Conference of Japan Society for Precision Engineering Hokkaido Branch, pp. 45-46, 2022.

[4] Y, Xu, S. Kanai, H. Date, T. Sano and T. Teranishi, "Prediction of 3D stacking poses of supplemental wave-dissipating blocks based on existing block poses and physics engine," Proceedings of 2022 Spring Conference of Japan Society for Precision Engineering, pp. 156-157, 2022.

[5] Y, Xu, S. Kanai, H. Date, T. Sano and T. Teranishi, " Prediction of the number of wave-dissipating blocks for repair work based on the identification and pose estimation of existing blocks," Proceedings of 2021 Autumn Conference of Japan Society for Precision Engineering, pp. 538-539, 2021.

[6] Y, Xu, S. Kanai, H. Date, T. Sano and T. Teranishi, " Recognition of wave-dissipating blocks with multiple types from 3D large-scale point clouds," Proceedings of 2021 Spring Conference of Japan Society for Precision Engineering, pp. 53-54, 2021.

[7] Y, Xu, S. Kanai, H. Date, T. Sano and T. Teranishi, "Recognition of wave-dissipating blocks from large-scale measured point cloud using deep learning," Proceedings of 2020 Annual Conference of Japan Society for Precision Engineering Hokkaido Branch, pp. 27-28, 2020.

Appendix B

Pipeline of the dissertation