

# Carnegie Mellon University

CARNEGIE INSTITUTE OF TECHNOLOGY

## THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF Doctor of Philosophy

TITLE Constructing a Complete and Accurate As-Built BIM  
Based on an As-Designed BIM and Progressive Laser Scans

PRESENTED BY Te Gao

ACCEPTED BY THE DEPARTMENTS OF

Civil and Environmental Engineering

Burcu Akinci February 7, 2014  
CO-ADVISOR, MAJOR PROFESSOR DATE

Semiha Ergan February 7, 2014  
CO-ADVISOR, MAJOR PROFESSOR DATE

James H. Garrett February 7, 2014  
CO-ADVISOR, MAJOR PROFESSOR DATE

David A. Dzombak February 10, 2014  
DEPARTMENT HEAD DATE

APPROVED BY THE COLLEGE COUNCIL

Vijayakumar Bhagavatula February 19, 2014  
DEAN DATE

**Constructing a Complete and Accurate As-built BIM based on an As-  
Designed BIM and Progressive Laser Scans**

Submitted in partial fulfillment of the requirements for  
the degree of  
Doctor of Philosophy  
in  
Department of Civil & Environmental Engineering

Te Gao

B.S., Civil Engineering, Zhejiang University

M.S., Civil and Environmental Engineering, Carnegie Mellon University

Carnegie Mellon University

Pittsburgh, Pennsylvania 15213

January, 2014

© Copyright by Te Gao 2014

All Rights Reserved

## **ACKNOWLEDGEMENT**

I would like to express my deepest gratitude to my academic advisors, Dr. Burcu Akinci, Dr. James H. Garrett, Jr. and Dr. Semiha Ergan for their continuous help and support at all stages of my research. First, I would like to thank Dr. Burcu Akinci, who taught me how to work towards a research question from scratch and showed me the beauty and pleasure of doing research. I benefit a lot from her scientific advice and knowledge. Second, I would like to thank Dr. James Garrett for his guidance, patience, and insightful suggestions. I also want to express my respect to him for being my role model as a research and mentor with integrity, creative thinking and enthusiasm. Third, I am thankful for all the time, creative idea and efforts Dr. Semiha Ergan provide for this thesis and her mentorship and encouragements.

My sincerest appreciation goes to my Ph.D. committee members, Dr. Daniel Huber and Dr. Matteo Pozzi for devoting their time and expertise in this thesis. I am grateful for their constructive feedback and insightful comments. I would also like to thank Dr. Lucio Soibelman for his help and guidance at the early stage of my research.

I am grateful for working with the former and current Mosaic team members. This team has been a source of friendships as well as constructive criticisms and collaboration. In addition, I would like to acknowledge the financial support from Glodon Software Company, China and National Institute of Standards and Technology (NITS), which make my PhD work possible.



Lastly, I would like to thank my parents and my younger sister for all their love, support and encouragement. And most importantly, I would like to thank my fiancée, Xiaofei Wang, who is also my lifetime best friend. She always has faith in me and has supported me during my good and bad times. I dedicate this thesis to my family, my beloved fiancée for their constant support and unconditional love.

## **ABSTRACT**

In the current practice, changes that occurred in the construction or renovation phase are not frequently captured and updated to as-built documentations. As a semantically rich digital representation, a building information model (BIM) can be used as an information repository to store and deliver as-built information. Since a building might not be constructed exactly as its design specifies, discrepancies might exist between BIMs created based on design information and actual building conditions. It is important to keep information stored in a BIM accurate and complete so that it serves as a reliable data source throughout the service life of a facility.

Point cloud data captured by laser scans is capable of providing accurate 3D depiction of building conditions. Hence, point cloud data can be used as the reference to update the information contained in an as-designed BIM. There are two main challenges associated with the update process. First, in order to recognize the differences between a point cloud and an as-designed BIM, segments captured by the point cloud need to be matched to building components modeled in the BIM. However, a building component might have different shapes, dimensions, and locations in a point cloud as compared to how it is modeled in an as-designed BIM. The discrepancies between a point cloud and an as-designed BIM increase the complexity of matching the two data sets together.

Second, occlusions and construction progress prevent a laser scan performed at a single point in time from capturing a complete set of geometric information associated with a building. While it is possible to retrieve more geometric

information by combining point clouds captured at different points in time, the large file size of a combined point cloud make it difficult to process and store. Hence, there is a need for an approach to achieve the balance between the richness of the information provided by a combined point cloud and the file size of it.

In order to address the above two challenges, this research aims to develop a framework that supports the update process by matching information from a point cloud to a BIM and leveraging point clouds captured at different times to provide a more complete and accurate reference. To develop such a framework, this research developed the following research objectives:

- (1) Identifying a general set of features that can be applied to recognize correspondences between a point cloud and a BIM.
- (2) Developing matching approaches that match point cloud segments to BIM components based on different types of features.
- (3) Conducting an experimental analysis to evaluate the performances of the developed matching approaches and effectiveness of the features used by these approaches.
- (4) Developing an approach that selects and combines point clouds captured at different times in order to reduce the file size of combined point cloud while maintaining the completeness of geometric information.

## Table of Content

<b>CHAPTER 1. INTRODUCTION.....</b>	<b>2</b>
1.1 Vision.....	6
1.2 Research Questions .....	9
1.3 Scope and High Level Assumptions .....	12
1.4 Dissertation Organization .....	13
<b>CHAPTER 2. IDENTIFICATION OF FEATURES TO BE USED FOR MATCHING POINT CLOUDS TO BIMs .....</b>	<b>15</b>
2.1 Background research studies .....	18
2.1.1. Research studies in relation to point cloud processing .....	18
2.1.2. Product representation and data exchange standards for building information models.....	20
2.1.3. Point cloud and 3D object recognition approaches.....	24
2.2 Research Method And Findings: Features To Be Used For Matching Point Clouds To BIMs .....	30
2.2.1. Features of point clouds that can be applied for the matching process .....	32
2.2.2. Features of objects represented in BIMs.....	35
2.3 Validation .....	39
2.4 Conclusions .....	42
<b>CHAPTER 3. EVALUATION OF DIFFERENT FEATURES FOR MATCHING POINT CLOUDS TO BUILDING INFORMATION MODELS .....</b>	<b>44</b>
3.1 Background Research Studies .....	46
3.2 Feature Based Matching Approaches .....	50
3.2.1 2D overlap area matching approach .....	51
3.2.2 Spatial relationship based graph-matching approach .....	54
3.2.3. Distribution-based 3D shape matching approach .....	55
3.2.4 Distribution-based 2D shape matching approach .....	59
3.3 Experiment Setup .....	60
3.3.1 Experiment setting.....	60
3.3.2. Tuning the matching approaches.....	63
3.4 Experiment Results .....	67
3.5 Combination Of Different Features To Improve Matching Results .....	73

3.5.1. Re-aligning point cloud and BIM based on 3D shape feature.....	73
3.5.2. Filtering the matching results generated by the 2D overlap area by the result of matching approach using 3D shape features.....	74
<b>3.6 Validation .....</b>	<b>76</b>
<b>3.7 Conclusions .....</b>	<b>78</b>
<b>CHAPTER 4. AN APPROACH FOR LEVERAGING PROGRESSIVE LASER SCANS.....</b>	<b>79</b>
<b>4.1 Related Research Studies .....</b>	<b>80</b>
4.1.1. Evaluation of the quality and quantity of information captured by a point cloud data .....	80
4.1.2. Comparing two point clouds and identifying the differences.....	83
4.1.3. Next-best-view planning problem .....	84
<b>4.2 Motivating Case Study: an Analysis of Completeness of Geometric Information in Progressively Captured Point Clouds .....</b>	<b>85</b>
<b>4.3. An Approach to Evaluate Information Contained in Progressively Captured Point Clouds and Select Point Clouds to Be Combined .....</b>	<b>93</b>
4.3.1. Content assessment module for quantifying the information contained in a point cloud .....	93
4.3.2. Content improvement module: Assessing the information gain by combining additional point clouds.....	100
<b>4.4. Validation .....</b>	<b>103</b>
<b>4.5 Conclusion .....</b>	<b>107</b>
<b>CHAPTER 5 CONCLUSION AND FUTURE WORK.....</b>	<b>109</b>
<b>5.1 Contributions .....</b>	<b>109</b>
<b>5.2 Practical Implication .....</b>	<b>113</b>
<b>5.3 Future Research Directions .....</b>	<b>115</b>
<b>Appendix A: Matching Results for the Four Matching Approaches .....</b>	<b>118</b>
<b>Appendix B. Codes for implementing the four matching approaches .....</b>	<b>129</b>
<b>Appendix C. Codes for the discrepancy simulation algorithm .....</b>	<b>143</b>
<b>REFERENCES.....</b>	<b>155</b>

## List of Figures

Figure 1. Envisioned framework for updating a BIM based on point cloud data .....	8
Figure 2. Shape discrepancies between ductworks modeled in the BIM and the same ductworks shown in the point cloud.....	16
Figure 3. Different components with similar shapes and dimensions .....	17
Figure 4. Distribution of reviewed papers in different fields of applications.....	32
Figure 5 (b). Count the number of research studies associated with each BIM related feature .....	42
Figure 6. Examples of different types of discrepancies.....	45
Figure 7. Classifications of features required by the four matching approaches .....	51
Figure 8. 2D projection of the registered point cloud and BIM .....	53
Figure 9. Graph constructed from the point cloud and the BIM .....	55
Figure 10. A flowchart depicting the distribution-based 3D shape matching approach .....	58
Figure 11 As-designed BIM and the corresponding point cloud.....	61
Figure 12. Example set of models with different discrepancies for the experiment...	63
Figure 13. Tradeoff of the precision and recall when threshold is changing.....	65
Figure 14. Visualization of a matching matrix .....	68
Figure 15. Mapping results for the four mapping approach in a case where there is a slight (2”) difference in the length of a component between a point cloud and a BIM.....	69
Figure 16. Matches identified by the 2D overlap area matching approach for one example case.....	70
Figure 17. The matches identified by the 2D and 3D shape matching approaches or one example case .....	71
Figure 18. The process of 3D shape plus 2D overlap area matching approach.....	74
Figure 19. Validation data set includes different ductworks and different types of discrepancies applied to all three sets of ductworks .....	77
Figure 20. Project site layout and the scan locations .....	86
Figure 21. The testbed for the motivation case study: ductworks captured by the point cloud data .....	87
Figure 22. Targets shown in a point cloud .....	88
Figure 23 Status of geometric properties captured by individual point clouds (P1, P2...P6) and the combination of them .....	89
Figure 24. Screenshots of the point clouds captured at different times.....	91
Figure 25. The flowchart of content assessment module.....	94
Figure 26(b) Ductworks captured by a point cloud and the coverage image, where the red cells indicate that the cells contain the points associated with the target surface. ....	97
Figure 27 (a). Processing time and point cloud density vs. grid size .....	98
Figure 28 Point cloud data captured at different points in time and their occlusion images for duct segments.....	100
Figure 29. Flowchart of the content improvement module .....	101
Figure 30. Dissimilarity ratio by adding P2 to P1.....	102

Figure 31. The four different building components targeted by the validation ..... 105

## List of Tables

Table 1. Previous research studies on point cloud and 3D object recognition .....	27
Table 2. Features of point clouds to support mapping point clouds to BIMs .....	33
Table 3. A list of features associated with BIMs for matching point clouds to BIMs	36
Table 4. The definitions and optimal values for the thresholds .....	67
Table 5. Results of matching process for the four matching approaches for testing and training data set .....	72
Table 6. The precision and recall comparison for 2D overlap & 3D shape approaches with the combination of the two approaches .....	76
Table 7. Matching results for the validation dataset .....	77
Table 8 The resulting subset of point clouds to be registered following the three approaches.....	105
Table 9. Comparison of the developed approach (approach 3) with respect to other baseline approaches.....	107



## CHAPTER 1. INTRODUCTION

For every construction or renovation project, it is expected that the general contractor of the project will hand over the as-built documentations to the owner or the facility management team at the end of the project. In the current practice, the as-built information is provided in paper-based documents, such as drawings and equipment lists, which are hard to update and maintain during the service life of a facility (Dickinson et al. 2009; East and Brodt 2007; Fallon and Palmer 2007; Pettee 2005). One of the main challenges for the handover process is to ensure that captured building information is complete and reflects as-built conditions (Dickinson et al. 2009; East and Brodt 2007; Pettee 2005). A report published by NIST in 2004 pointed out that \$4.8 billion was spent every year just to verify that the captured building information describes accurate building conditions (NIST 2004).

As a digital representation that captures and exchanges building information between different stakeholders, a building information model (BIM) can be used as an information repository to store and deliver as-built information. Currently, BIMs are usually created based on design information in the design or construction phases (called as as-designed BIMs). Such as-designed BIMs might not be able to reflect the as-built conditions of a building, since a building does not always get constructed exactly as it is specified at the design phase. Changes and updates are likely to happen during the construction phase (Tang et al. 2010, Goedert et al. 2008(O'Brien 1998; Rowland Jr 1981; Terwiesch and Loch 1999). Extensive surveys are needed to capture as-built building conditions so as to remove differences between an as-

designed BIM and the as-built conditions that need to be delivered (Dickinson et al. 2009; Tang et al. 2010).

It is important to keep the information captured in a BIM complete (i.e., all building components and their associated geometric information are captured in the BIM) and accurate (i.e., geometric information contained in the BIM reflects the as-built conditions). Laser scanning technology is able to capture accurate geometric data in the form of a point cloud and depict as-built spatial condition of buildings. Hence, point clouds captured by laser scans can be used as a reference to update an as-designed BIM into an as-built BIM (i.e., a BIM that captures the as-built condition of a building). However, there are two main challenges associated with the process of updating an as-designed BIM using point cloud data.

***(1) Matching segments in point clouds to BIM components under discrepancy conditions***

Unlike a BIM, which contains both geometric and semantic information, a point cloud is a collection of 3D points without any semantic information (e.g., which building component that a point belongs to). Therefore, to support the updating process, segments captured by a point cloud need to be matched to components modeled in an as-designed BIM in order to compare the information contained in the as-designed BIM with the as-built geometric data captured by the point cloud.

Typically, matches between a point cloud and a BIM can be identified based on the features associated with segments extracted from the point cloud and components modeled in the BIM. For example, if segment  $P_i$  of a point cloud is located at the

same position where component  $B_j$  is modeled in a BIM, it is possible that  $P_i$  and  $B_j$  are referring to the same component and should be matched together.

However, due to changes that occur during the construction phase, a building component might have different shapes, dimensions, and be located at different positions in a point cloud as compared to how it is modeled in an as-designed BIM. These discrepancies could impact the matching process and generate incorrect matches between a point cloud and an as-designed BIM.

Some features of point clouds and BIMs might remain similar under a given discrepancy condition, and hence these features can be used to correctly match point cloud segments to BIM components. Currently, there is a limited understanding of what features could be used to match a point cloud to a BIM and how these features would perform under different discrepancies. Therefore, this is a need to identify a general set of features that can be used to support the matching process and evaluate their performances in terms of how well those features are able to identify correct matches between a point cloud and a BIM.

## ***(2) Leveraging point clouds captured at different times to provide accurate and complete geometric information***

Point cloud data collected at a single point in time typically is not capable of providing all the geometric information required for updating an as-designed BIM due to the following reasons: (a) a construction scene is periodically occluded by temporary work, machinery, laborers, materials, (b) permanent building components might be hidden behind the finished surfaces (e.g., ductwork that are hidden behind

ceiling tiles) and (c) some building components are not installed/constructed at the time of scanning.

Scanning a building at discrete points in time during its construction process (called as progressive scanning in this thesis) and combining these scans together could provide more geometric information for updating a BIM. However, combining point clouds together could result in a file with huge size that is difficult to process and store, given that each point cloud often contains millions of data points and its size could range between a few hundred megabytes to a few gigabytes depending on the scanning range and resolution (Bentley 2012; Song and Feng 2009). There is a trade-off between retrieving more geometric information by combining all of captured point clouds and reducing the file size by only combining a subset of point clouds. To address this challenge, there is a need for an approach that evaluates geometric information contained in a point cloud data and supports the selection of point clouds.

## 1.1 Vision

To address the two challenges stated in the previous section, I have envisioned a framework that matches point clouds to components modeled in an as-designed BIM and leverage geometric information contained in the progressively captured point clouds to support the BIM update. The envisioned framework is shown in Figure 1.

The framework is composed of three main processes:

### *(1) Matching point cloud segments to BIM components*

This process matches the segments from a point cloud to components modeled in an as-designed BIM. Based on the identified matches, construction professionals are able to identify the differences between an as-designed BIM and the as-built information captured by point clouds so as to update the model accordingly. In this research, the following research tasks have been accomplished in order to support this process: (a) identifying a general set of features that can be used to match point cloud segments to BIM components, (b) developing matching approaches that reason with and combine different types of features to identify matches between point cloud segments and BIM components, and (c) evaluating the matching results that are identified by different features and under various discrepancy conditions. These three research tasks are discussed in details in Chapter 2 and 3.

### *(2) Leveraging point clouds captured at different points in time*

This process evaluates the geometric information contained in a set of progressively captured point clouds and assesses the information gained by combining two point

clouds together. Based on unique data sets that each point cloud brings, this process combines point clouds collected at different points in time in order to ensure the completeness of the captured geometric information while reducing the file size that caused by merging point clouds with repetitive information. In this research, the following research tasks have been accomplished in order to implement this process: (a) comparing the completeness of geometric information contained in progressively captured point clouds with point clouds captured at a single point in time, and (b) developing an approach that evaluates the geometric information of a target building component captured by point cloud data and assesses additional information gained by point cloud combination. These two research tasks are discussed in details in Chapter 4.

### ***(3) Updating BIM to represent as-built condition:***

According to the matches identified between point cloud segments and BIM components, this process identifies the discrepancies between the two data sets and updates the BIM to represent the as-built conditions. The output of this process is an up-to-date as-built BIM, which contain both visible (e.g., wall, ceiling, floor that are shown in the point cloud) and invisible (e.g., air duct, water pipe that are hidden behind the finished surfaces) components with accurate 3D geometries. The as-built BIM can be then delivered to building owners or facility management teams as a part of the as-built handover documents. In this thesis, I mainly focus on the first and second process in support of this third process.

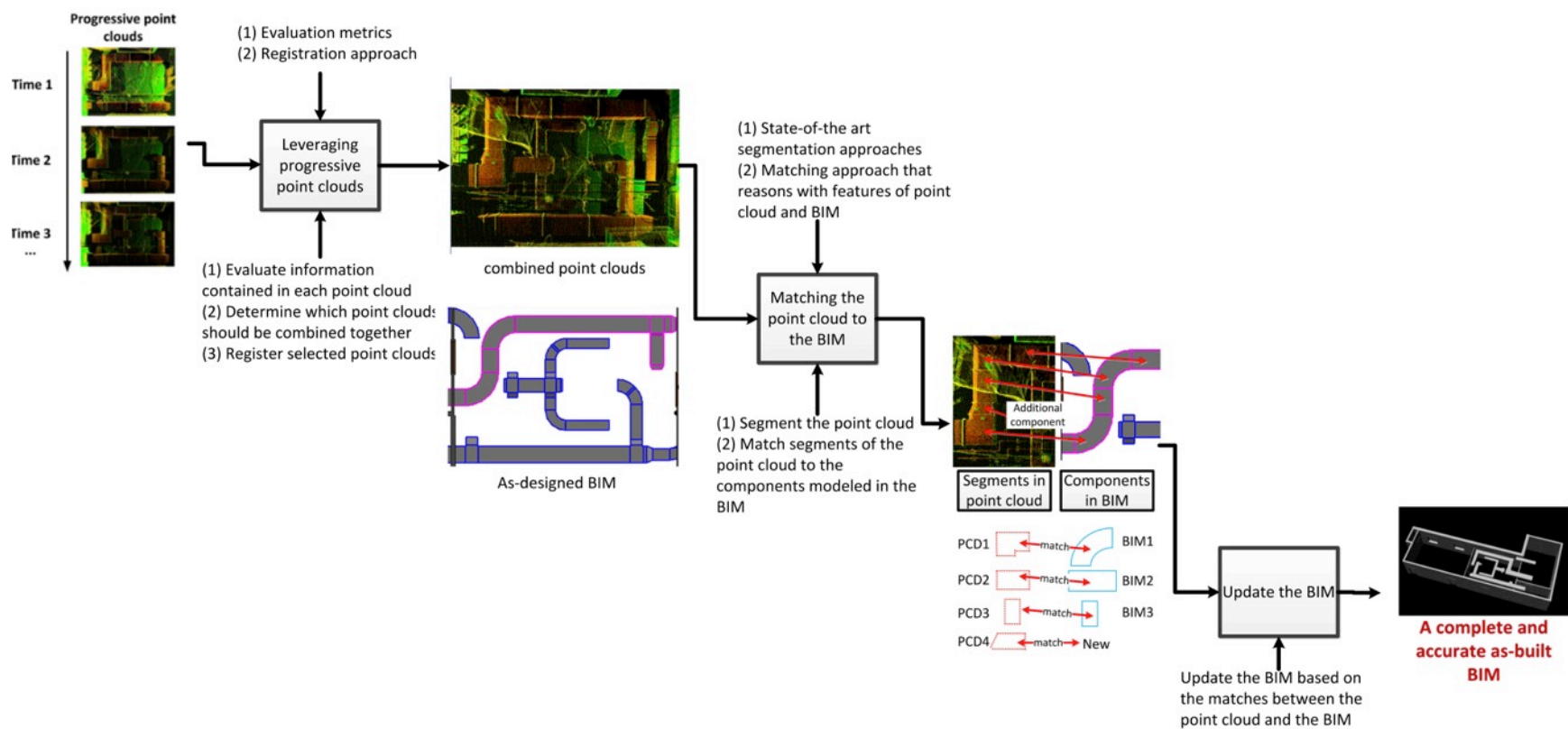


Figure 1. Envisioned framework for updating a BIM based on point cloud data

## **1.2 Research Questions**

This section discusses three research questions aiming to address the challenges associated with the process of updating an as-designed BIM based on point cloud data.

### **Research Question 1: What features can be used to match point clouds to BIMs?**

It is challenging to match a point cloud to a BIM when there are discrepancies between the real-world components and the components modeled in the BIM. A general set of features can potentially help in identifying correspondences between these two data sets, and thus serve as the foundation for matching approaches.

In addressing this research question, I reviewed previous research studies in relation to: (1) 3D shape matching and retrieval, (2) product modeling and BIM standard, (3) building model reconstruction based on point cloud data, (4) object detection and recognition in construction sites for progress monitoring and equipment control, (5) 3D scene annotation for urban modeling, and (6) indoor point cloud recognition for mobile robot navigation. Building on these research studies, I identified a general set of features that can be used to match point clouds to BIMs, and I classified these features into different groups. The coverage of the identified features is validated to ensure that the feature list identified in this research covers the majority of the features that can be potentially used for the matching process.



**Research Question 2: How well, in terms of precision and recall, do different features perform in identifying matches between a segmented point cloud and a BIM under different discrepancies?**

Matches between a point cloud and a BIM can be identified based on features shared by these two data sets. However, there is a limited understanding of how different features perform in the matching process under different discrepancy conditions. Hence, there is a need to evaluate the performance of different features in terms of how well they are able to identify correct matches between a point cloud and a BIM.

Building on the features identified in research question 1, I developed a set of matching approaches that combine and reason with different types of features. I then evaluated the performance of the features based on the precision and recall of the matching results.

**Research Question 3: How well do the progressive point clouds improve the accuracy and completeness of an as-built BIM?**

Combining point clouds that are progressively captured in construction could provide more geometric information for the BIM update. However, the combination of point clouds could also result in a dataset with large file sizes. However, as discussed previously, there is a trade-off between having a completely registered point clouds to provide more geometric information and having a registered point cloud with less number of point clouds to reduce the file size and processing time.

In relation to this research question, I developed an approach to evaluate the information contained in a point cloud so as to support the decision about which point clouds should be combined together in order to increase the completeness of geometric information with less file size. The developed approach consists of two modules: (a) content assessment module, which quantifies the geometric information contained in each point cloud for updating target building components in a BIM, and (b) content improvement module, which calculates the information gained by adding new point cloud to the initial data set. Two metrics are used in this approach: (a) coverage ratio, which is used to evaluate the information captured by a point cloud, and (b) dissimilarity ratio, which assesses the additional information generated by adding a point cloud to the initial data set. The effectiveness of this approach is validated using point clouds captured during a construction project.

### **1.3 Scope and High Level Assumptions**

This section describes the scope and high-level assumptions for my PhD research.

#### **1.3.1. Assumptions**

The point cloud data is segmented by the state-to-art segmentation approaches and the geometric primitives (e.g., lines, planes, cylinder, cube, etc.) are fitted to the point clouds.

#### **1.3.2. Scope**

- a. This research focuses on updating an as-designed BIM to improve the accuracy and completeness of its geometric information. The semantic information (e.g., object ID, type, material) contained in an as-designed BIM is not the focus on this research study.
- b. This research focuses on matching point clouds to BIMs and leveraging point clouds captured during construction to provide more geometric information. The actual modification and update of as-designed BIMs are not within the scope of this research.

## **1.4 Dissertation Organization**

This dissertation consists of five chapters including Introduction and Conclusions. Chapter 1 provides an overview of the research problems and vision, and describes the three research questions addressed in this thesis. Chapter 5 highlights the research contributions, practical implications and future research directions. The remaining three chapters focus on three specific research contributions.

### **Chapter 2: Identification of Features to Be Used for Matching Point Clouds to BIMs**

Chapter 2 presents the findings of the research study conducted to identify a general set of features associated with point cloud data and BIM, which can be potentially used to support the matching between these two data sets.

### **Chapter 3: Evaluation of Features for Matching Point Clouds to Building Information Models**

Chapter 3 introduces six feature-based matching approaches developed based on prior work to match a point cloud to a BIM. It then presents the results of a set of experiments conducted to evaluate the performance of different features in matching point clouds to BIMs under various discrepancy conditions. In addition, this chapter provides a discussion on how the combination of different features could improve the matching results.

## **Chapter 4: An Approach for Leveraging Progressive Laser Scans to Improve Completeness**

Chapter 4 describes an experimental analysis conducted to evaluate the completeness of geometric information contained in point clouds captured at a single point in time versus the combined data set that is composed of point clouds captured at different times. It then introduces an approach developed to select and combine progressively captured point cloud so as to provide more complete geometric information with less file size.

## **CHAPTER 2. IDENTIFICATION OF FEATURES TO BE USED FOR MATCHING POINT CLOUDS TO BIMS**

Matching a point clouds to a BIM is an important step in the updating process. This step associates the segments captured by the point cloud to components modeled in the BIM, so that the difference between the design information stored in the BIM and the as-built conditions captured by the point cloud can be recognized. One way of identifying the matches between a point cloud and a BIM is to reason with features associated with these two data sets. In this thesis, I defined a feature as a geometric primitive or a geometric property that have distinct characteristics. Features can be used to identify the correspondences between a point cloud and a BIM. For example, if segment  $P_i$  of a point cloud is located at the same location where component  $B_j$  is modeled in a BIM, then it is possible that  $P_i$  and  $B_j$  correspond to each other. However, there are two challenges associated with matching a point cloud to a BIM.

First, building components might have different shapes, dimensions, orientations and positions in a point cloud as compared to how they are modeled in an as-designed BIM. These discrepancies could impact the performance of matching approaches and generate incorrect matches between a point cloud and a BIM. For instance, as shown in Figure 1, the highlighted air ducts were modeled as L-shape rectangular elbows in the as-designed BIM, whereas they were manufactured as curve-shaped elbows, shape of which was captured within the point cloud data. Due to such shape discrepancies, matching approaches based solely on shape related features (e.g.,

dimension, volume, surface area) would not be able to identify the correct matches of the highlighted air ducts.

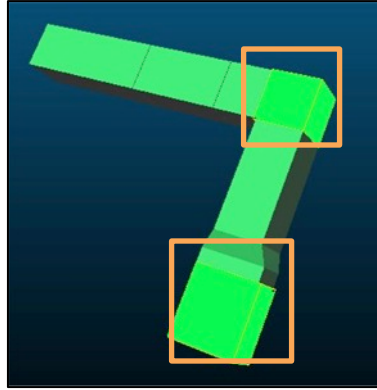


Figure 2(a) Ductworks modeled in the BIM

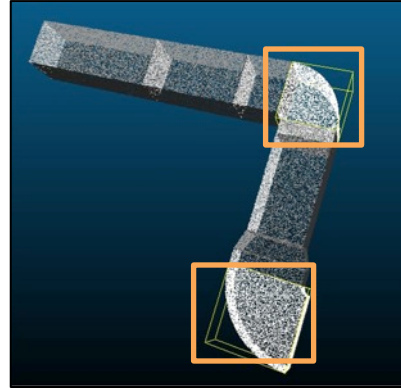


Figure 2 (b) Ductworks captured by the point cloud

Figure 1. Shape discrepancies between ductworks modeled in the BIM and the same ductworks shown in the point cloud

Second, in a construction project some building components might have the same shape, orientation and dimension (e.g., a collection of columns, beams, and duct segments). For example, as seen in Figure 2, air ducts A and B were manufactured in the same shape and size, but were installed at two different locations. Matching approaches that are based solely on shape-related features would not be able to distinguish air duct A from air duct B, even though they are two different building components. Hence, it is important to identify features that are able to distinguish building components with the same 3D geometry so as to eliminate mismatches.

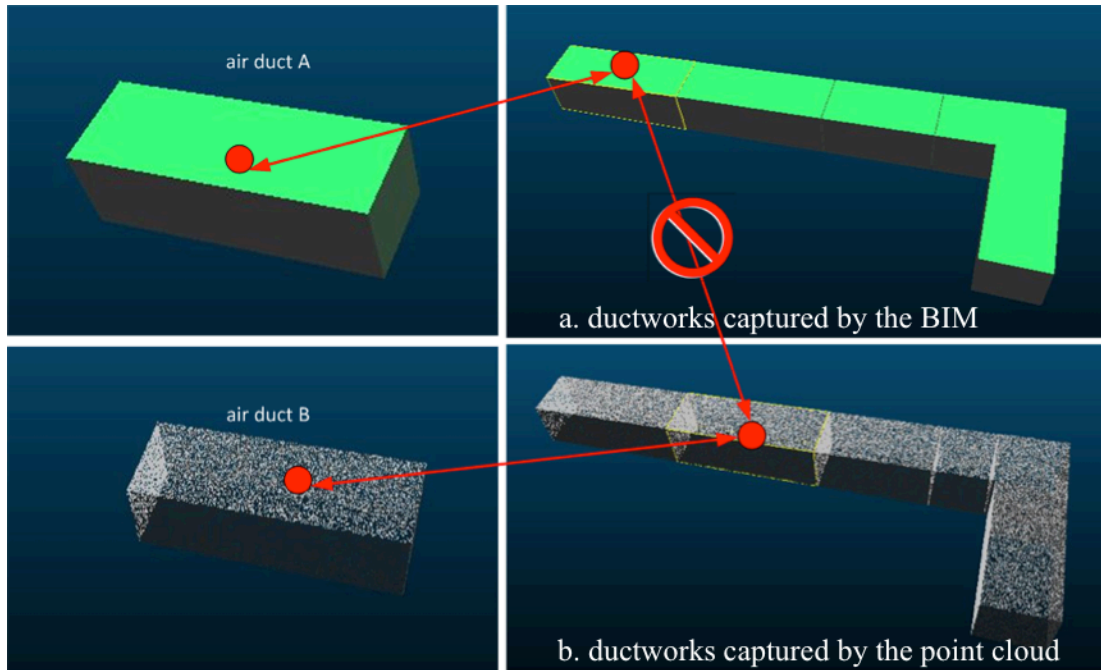


Figure 2. Different components with similar shapes and dimensions

Some feature of point clouds and BIMs might remain constant under a given discrepancy situation. These features have the potential to be used in the matching approaches and reduce the impacts of the given discrepancy situation. For example, the highlighted air ducts shown in Figure 1 have different shapes in the point cloud and the BIM due to the shape discrepancies however they are located at the same position in the corresponding data sets. In this case, a matching approach based on location features might be robust to the shape discrepancies and is able to identify the correct matches for the highlighted air ducts. Development of a robust matching approach would require a comprehensive understanding of the features of point clouds and BIMs. This chapter provides the findings of a research conducted to identify features that can be potentially applied in the matching process. The identified features could serve as a foundation to develop matching approaches that



utilize and combine different feature to correctly match point cloud segments to components in a BIM under the presence of discrepancies.

## **2.1 Background research studies**

An extensive literature review has been done in multiple domains, such as computer vision, robotic, and civil engineering, where point clouds and laser scanners are used. The research studies that were investigated in this research can be grouped into three categories: (1) point cloud processing (e.g., point cloud segmentation, feature extraction), (2) data standards for building information models and product modeling approaches, and (3) point cloud and 3D object recognition approaches. These studies were examined in details in order to generate a synthesized list of features that can be used to match a point cloud to a BIM. An overview of research studies that were examined is provided below.

### **2.1.1. Research studies in relation to point cloud processing**

Features associated with point clouds have been used to support various point cloud processing tasks, including point cloud segmentation, point cloud feature extraction, and surface reconstruction from point clouds. A synthesis of the findings from these studies suggests that the features of point clouds can be grouped under three categories: *local features*, *semi-local features* and *global features*.

*Local features* are referred to as the features associated with each individual point in a point cloud. Such features include 3D coordinates, surface normal, point intensity and color of a point (Schnabel et al. 2007; Tang et al. 2010; Tangelder and Veltkamp

2004). *Semi-local features* are referred to as the features calculated from points within a region (Huber et al. 2004; Johnson and Hebert 1999; Tang et al. 2010). Examples of semi-local features are spin-image, point density, and the maximum height of the points within a neighborhood (Johnson and Hebert 1999, Huber et al. 2004). *Global features* are referred to as the features calculated by taking the entire point cloud into consideration. Examples of global features include histogram of pairwise distance of random points and the distribution of point density (Körtgen et al. 2003; Mahmoudi and Sapiro 2009; Tangelder and Veltkamp 2004). The *semi-local* and *global features* can be used as descriptors to describe the shape of objects captured by a point cloud (Frome et al. 2004; Huber et al. 2004; Tang et al. 2010; Tangelder and Veltkamp 2004). The *local features* are less sensitive to noise and clutters shown in the scene, but they have lower discriminative power when comparing to *global features* (Tangelder and Veltkamp 2004). The *semi-local features* are evaluated as having the similar discriminative power as compared to *global features*, and they are robust to noises and clutters, which is similar to *local features* (Johnson 1997).

A point cloud is a collection of points and contains no semantic information. Using segmentation or clustering approaches, points can be grouped and assigned into higher level geometric primitives, such as planer patches, spheres, cylinders, cones and torus (Huang and Brenner 2011; Rabbani et al. 2006; Roggero 2002; Schnabel et al. 2007; Schnabel et al. 2008). Random Sampling Consensus Algorithm (RANSAC) fits a parameterized mathematical model to a point cloud by iteratively selecting the model parameters that best fit to the data set (Fischler and Bolles 1981). RANSAC approach is capable of extracting geometric primitives, such as planes, spheres,

cylinders from a point cloud (Schnabel et al. 2007; Schnabel et al. 2008). The geometric primitives fitted to a point cloud form a concise and abstract representation for a point cloud. A large amount of research studies have used geometric primitives fitted to a point cloud as important features to support different point cloud processing activities, such as model reconstruction and object recognition from point clouds (Duan et al. 2010; Golovinskiy et al. 2009; Huang and Brenner 2011).

The prior research studies in relation to point cloud processing provide a point of departure to identify a general list of features associated with point clouds. This feature list will be further evaluated to identify what features can be potentially used to match a point cloud to a BIM.

### **2.1.2. Product representation and data exchange standards for building information models**

A BIM is a digital representation of a facility's life cycle information and contains both physical and functional information of that facility. Since a point cloud only contains geometric information, most of non-geometric information stored in BIMs, such as functionality (e.g., usage, air flow rate for ductworks), material and cost cannot help in establishing the matches between point clouds and BIMs. Therefore, in this research, non-geometric information of a BIM was not considered. In this research, I mainly focus on shape (i.e., features related to the size and shape of a building component) and spatial (i.e., features related to the location of a building components) related features associated with 3D objects modeled in a BIM.

3D geometry of building components are represented in a BIM using different representation techniques (BuildingSMART 2012; buildingSMART 2012; Eastman 1999). Boundary representation (B-rep for short) represents a surface model as a collection of faces, which are connected by edges and vertices (Campbell and Flynn 2001; Zhang and Lu 2004). Features defined by B-rep include edges, vertices, faces and their connectivity. Constructive solid geometry (CSG for short) represents a solid model by overlapping geometric primitives (e.g., sphere, cuboids, cylinders) together using Boolean operators. Boolean operators include union, intersection and difference (Eastman 1999; Requicha 1980; Tang et al. 2010). The features defined in CSG are geometric primitives, such as sphere, cuboids, cylinders and blocks. Sweep representation is another presentation technique that forms a 3D shape by sweeping a geometric primitive along a path (i.e., a principle axis) (Eastman 1999; Requicha 1980). The features defined by sweep representation are the geometric primitives (e.g., planer surfaces, cylinder, etc.) and the path, along which a geometric primitive is swept to form a 3D shape. Additional features can be derived from the 3D shapes of building components represented in a BIM. Such features include volume, area, geometric ratios (e.g., the number of facets, surface area to volume, etc.), and dimensions of a bounding box fitted to an object or a group of objects (Cybenko et al. 1997; Iyer et al. 2005; Tangelder and Veltkamp 2004; Zhang and Chen 2001).

Extensive literature review within the AEC/FM domain suggests that spatial information representation in a BIM can be divided into three categories (Borrmann and Beetz 2010; Borrmann and Rank 2009; Clementini and Di Felice 1996; Schneider and Behr 2006). These three categories of features are: (a) *topology related*

*features*, which define the relative position and connectivity between an object with respect to a reference (e.g., disjoint, touch, equals, covers, overlap), (b) *orientation related features*, which define the relative orientation of an object with respect to a reference (e.g., parallel, orthogonal) and (c) *distance related features*, which define the relative distance from an object to a reference.

Various data standards have been developed to represent construction project data and facility information. Examples of those data standards include Industry Foundation Classes (IFC), Construction Operation Building Information Exchange (COBIE), and ISO 15926 (i.e., standard for project data integration, sharing, and exchange between systems. Among those data standards, IFC standard is the most well known open data model specification standard that enables sharing and exchanging of information contained in BIMs among different systems (buildingSMART 2012). The IFC4 standard defines more than 700 classes to describe the information, such as dimension, location, material, functionality and spatial relationships associated with building components (buildingSMART 2012). In this research, I used IFC standard as the reference to identify features associated with BIM, for two reasons: (1) as compared to COBIE, IFC standard contains more classes and attributes in relation to the 3D geometry of a building component, and (2) ISO 15926 targets on the heavy construction in oil and gas industry whereas IFC standard targets on buildings, which is well aligned with the scope of this research.

In the IFC standard, the *IfcProductRepresentation* class defines both shape and spatial representation of a building component. An instance of the *IfcProductRepresentation*

class is composed of one or more instances defined by the *IfcShapeModel* class. The *IfcShapeModel* class defines the concept of a geometrical and/or topological representation for 3D shape of a building object. The *IfcShapeModel* class has two subclasses: (a) *IfcShapeRepresentation* class, which represents the geometric features of a product; and (b) *IfcTopologyRepresentation* class, which represents the connectivity of a product or components within the product. Various subtypes are defined for the *IfcShapeRepresentation* class, which include curve2D (i.e., two dimensional curves), geometricSet (e.g., points, curves and surfaces), surfaceModel (i.e., face based or shell based surface models) and solidModel. Similarly, the subtypes of *IfcTopologyRepresentation* class include vertex (i.e., a corner where two or more than two lines intersect), edge (i.e., a line segment that connects two faces), and face (i.e., an surface of a 3D shape). In addition, *IfcAxis2Placement3D* class explicitly defines the location and orientation of an object modeled in a BIM in the 3D space.

Additional features associated with a building information model have been defined from the computer-aided design, cost estimation and construction management perspectives. Within the computer-aided architectural or mechanical design domain, features are defined as generic shapes that carry engineering or design configurations (Anderson and Chang 1990; Salomons et al. 1993; Van Leeuwen and Wagter 1997). Examples of such features include form features that represent the form, shape and topology of design entities, and physical features that describe the physical qualities of design entities (Anderson and Chang 1990; Salomons et al. 1993). In order to support cost estimation and construction management, features are defined to describe the design information that is important for cost estimators or construction

manager. Examples of these features include: component feature, which is referring to the components existed in building product model and intersection features, which describe the intersection of two features, such as openings and turns (Nepal et al. 2009; Staub-French et al. 2003). The features defined by these research studies contain both semantic and geometric information. However, since these features are mainly designed to support applications such as computer-aided design, cost estimation and construction management, they cannot be directly used to support the matching between point clouds and BIMs.

The research studies in relation to product representation and BIM standards provide a point of departure for identifying a general set of features associated with BIMs. These features will be further evaluated to identify which features can potentially be used to match a point cloud to a BIM.

### **2.1.3. Point cloud and 3D object recognition approaches**

A large amount of approaches within the computer vision, remote sensing and robotics domains have been developed to recognize objects from point cloud data or classify 3D objects contained in a 3D model into different classes (Bustos et al. 2007; Frome et al. 2004; Golovinskiy et al. 2009; Pu and Vosselman 2006). These approaches have been used to support various applications including, but not limited to: (a) 3D shape matching and retrieval (Belongie et al. 2002; Körtgen et al. 2003), (b) building model reconstruction from point cloud data (Adan and Huber 2011; Brenner 2005), (c) 3D scene annotation for urban modeling (Douillard et al. 2011; Golovinskiy and Funkhouser 2009), (d) indoor point cloud recognition for mobile

robot navigation (Rusu et al. 2008; Koppula et al. 2011), and (e) construction site object detection for progress monitoring and equipment control (Bosche and Haas 2008; Kurtan et al. 2011). Although not all of these approaches are directly targeting on the problem of matching point clouds to BIMs, they certainly provide some useful insights in terms of: (a) what features can be used to describe objects contained in a point cloud or objects modeled in a BIM, (b) how different features of point clouds and BIMs help for the object recognition, and (c) what features are shared by point clouds and BIMs so that they can be applied to match the two data sets together.

Based on the characteristics of features used by the object recognition approaches, I divided these approaches into two different groups: (a) object recognition approaches based on spatial information, and (b) object recognition approaches based on shape representation. Brief descriptions and a synthesis of these studies are provided below and the findings of the literature review on these studies are summarized in Table 1.

#### ***(a) Object recognition approaches based on spatial information***

Object recognition approaches based on spatial information classify objects extracted from point cloud data into different classes. The classification is made based on spatial related information gained from heuristics or design models (e.g., BIM, floor plan, 3D design model), which are served as references reflecting the nature of collected point clouds. The spatial related features include relative distance, angles, positions and connectives between objects captured by a point cloud or components in a design model.



The approaches that reason with relative locations and distances between 3D objects can be called as spatial proximity-based object recognition approaches. Within this group, Bosche (2008 and 2009) assumed that the BIM is exactly the same as the point cloud. Hence, points within a point cloud can be matched to the building components modeled in a BIM according to the relative location between the points to the objects modeled in the BIM. This approach matches the components from the design model to the point cloud, which provides the capability of automatically recognizing 3D design model objects from a point cloud. Golovinskiy et al. (2009) focused on urban environment modeling and recognized objects (e.g., cars, traffic lights) captured by a point cloud based on their relative location to the reference objects (e.g., cars are more likely to be parked along a street, etc.). Koppula et al. (2011) used the relative distance and angles between different segments extracted from a point cloud to improve the performance of object recognition from the 3D point cloud data. Features used in these approaches are summarized in Table 1.

Table 1. Previous research studies on point cloud and 3D object recognition

Approach	Features of point clouds and 3D models applied for object recognition	References
<b>Object recognition approaches using spatial information</b>		
Spatial relationship-based object recognition	(1) Relative angles, orientations and locations between segments (e.g., planes, surfaces) extracted from a point cloud: orthogonal, parallel, adjacent, coplanar, above and below; (2) connectivity between different segments	(Adan and Huber 2011; Duan et al. 2010; Eich et al. 2010; Elberink and Vosselman 2009; Huber et al. 2011; Koppula et al. 2011; Nüchter and Hertzberg 2008; Nuchter et al. 2003; Pangercic et al. 2012; Pu and Vosselman 2009; Schnabel et al. 2008; Verma et al. 2006; Xiong et al. 2013)
Spatial proximity-based object recognition	(1) Spatial features associated with segments or point clusters extracted from a point cloud: position of centroid, surface normal and (2) relative distance between 3D points to 3D objects modeled in the BIM	(Anand et al. 2013; Bosché 2009; Bosche and Haas 2008; Bosche and Haas 2008; Golovinskiy and Funkhouser 2009; Golovinskiy et al. 2009; Koppula et al. 2011; Pu and Vosselman 2006; Pu and Vosselman 2009; Rusu et al. 2008; Rusu et al. 2009; Son and Kim 2010; Triebel et al. 2006)
<b>Object recognition approaches using shape information</b>		
Regional feature-based object recognition	Features associated with a point or points within a neighborhood region: (1) spin-image of a point, a descriptive image that encodes the properties of the surface fitted to the point cluster in an object-centered coordinates; (2) principal component analysis (PCA) feature: the magnitude of the linearness, surfaceness, and scatterness of a cluster of points	(Anguelov et al. 2005; Chen and Bhanu 2007; Chua and Jarvis 1997; Douillard et al. 2010; Frome et al. 2004; Johnson and Hebert 1999; Lai and Fox 2010)
Entity feature-based object recognition approaches	(1) Geometric features associated with segments fitted to a point cloud: orientation (i.e., surface normal and direction of principle axis), area, volume, point density, and height, (2) geometric feature associated with bounding boxes fitted to point clouds: size, height, width, and fractional occupancy (i.e., the ratio of the volume of the object to the volume of the box)	(Gilsinn et al. 2005; Golovinskiy et al. 2009; Mozos et al. 2005; Nüchter and Hertzberg 2008; Okorn et al. 2010; Paquet et al. 2000; Pu and Vosselman 2006; Pu and Vosselman 2009; Rusu et al. 2009; Teizer et al. 2007; Truong et al. 2012)
Distribution feature-based object recognition	(1) Shape distribution: probability distribution of the pairwise distances/angles between random points; (2) shape context: the relative distribution of points relative to all other points on the shape	(Belongie et al. 2001; Belongie et al. 2002; Ip et al. 2002; Körtgen et al. 2003; Mahmoudi and Sapiro 2009; Osada et al. 2001; Osada et al. 2002; Valero et al. 2011)
View-based object recognition	Features associated with 2D slices obtained by cutting 3D objects or projecting 3D objects in a 2D plane	(Chen et al. 2003; Funkhouser and Kazhdan 2004; Funkhouser et al. 2003; Jiantao et al. 2004; Nüchter and Hertzberg 2008)

Approaches that reason with the relative orientation and connectivity between 3D objects can be called as spatial relationship-based object recognition approaches. Examples of the spatial relationships include “two walls are parallel to each other”, “the floor surface is bounded by walls”, and so on (Cantzler et al. 2002; Duan et al. 2009; Golovinskiy et al. 2009; Nüchter and Hertzberg 2008; Xiong and Huber 2010). For instance, Xiong and Huber (2010) developed a method to classify planar patches extracted from a point cloud into walls, floors, ceilings and clutters by reasoning with the relationships about the extracted patches, such as orthogonal, parallel, adjacent, and coplanar. Cantzler et al. (2002) and Fisher (2003) developed a semantic net representation to encode spatial relationships between different building components. In the semantic net representation, each node represents a specific type of building components and each edge represents a spatial relationship between adjacent building components. This semantic net is then used to facilitate the object recognition task.

***(b) Object recognition approaches based on shape information***

There is a sizable amount of research studies that focus on shape matching and retrieval. They identify the correspondences between two shapes so as to calculate the similarities between two shapes. Some of these approaches have been extended to support the object recognition task. With shape similarities being calculated, algorithms are able to recognize objects from a database or cluttered 3D images, which are similar to the objects shown in the training data or directly inputted by users.

Object recognition approaches based on shape information first define a set of descriptors to depict the shapes of objects shown in point clouds or 3D models. Second, they calculate the descriptors for both training and testing data (the training and testing data is a set of 3D objects) and define a similarity metric to measure similarities between two different descriptors. Finally, objects captured in the testing data are matched to the objects in the training data that have the closest descriptors (Frome et al. 2004; Huber et al. 2004; Tang et al. 2010; Tangelder and Veltkamp 2004).

The descriptors are usually defined based on shape-related features, and are used to identify the correspondences between 3D shapes captured by two different point clouds or 3D models. For instance, a shape descriptor, named *shape context*, was developed for measuring similarity between different shapes (Belongie et al. 2001, 2002). The shape context is calculated by taking a set of points sampled from the contours of an object, and measuring the distribution of distances from each point to all the other points on the contours. Osada et al. (2002) created a descriptor named shape distribution to describe a 3D shape. The approach represents the descriptor of a 3D shape as a probability distribution of properties extracted from a 3D shape. Examples of such properties include the distribution of distances between two random points located on 3D surfaces, and angles between two random points picked on 3D surfaces (Ip et al. 2002; Mahmoudi and Sapiro 2009; Osada et al. 2002).

Depending on the characteristics of features used in object recognition approaches based on shape information, these approaches can be further divided into four

groups: (1) regional feature-based object recognition approaches, which use the shape related features derived from points and a group of points in a region; (2) entity feature-based object recognition approaches, which fit geometric primitives to the point cloud data and reason with the features derived from these geometric primitives; (3) distribution feature-based object recognition approaches, which describe a 3D shape using a probability distribution of geometric properties, and (4) view-based object recognition approaches, which utilize 2D projections of a 3D model or a point cloud as features. The list of features utilized in the approaches under these four categories can be found in Table 1.

The research studies in relation to 3D point cloud object recognition approaches provide a point of departure for a list of features that can be applied to distinguish objects shown in a point cloud or in a BIM. The same features shared by the point cloud and the BIM have the potential to be used to recognize the correspondences and identify the matches between the two data sets.

## **2.2 Research Method And Findings: Features To Be Used For Matching Point Clouds To BIMs**

As detailed in section 2.1, a general set of features associated with point clouds and BIMs were identified by investigating research studies in relation to point cloud processing, product representation, and data standards for building information models. I then further investigate point cloud and 3D object recognition approaches in order to identify the features that can be potentially used for matching the two data sets together. In total, 80 papers were reviewed in the following fields: (1) 3D

shape matching and retrieval, (2) product modeling and BIM standard, (3) building model reconstruction based on point cloud data, (4) object detection and recognition in construction sites for progress monitoring and equipment control, (5) 3D scene annotation for urban modeling, and (6) indoor point cloud recognition for mobile robot navigation. The field of 3D shape matching and retrieval provides the general knowledge on what features can be used to describe 3D shapes. The product modeling and BIM standard shows how the shape and spatial related features are defined in a BIM, and provide general categorizations to classify the BIM-related features. The rest of the fields of applications have one thing in common, even though they target different applications; they all try to detect and recognize components from point cloud either based on design information or the knowledge about building configurations. Hence, these fields of applications provide a set of features that are shared by point clouds and design/designed-based model, which can be applied to support identifying matches between point clouds and BIMs. The distribution of papers within these fields is shown in Figure 3. The features identified in this research studies are provided in the following two sections: features associated with point clouds and features associated with components modeled in BIMs.

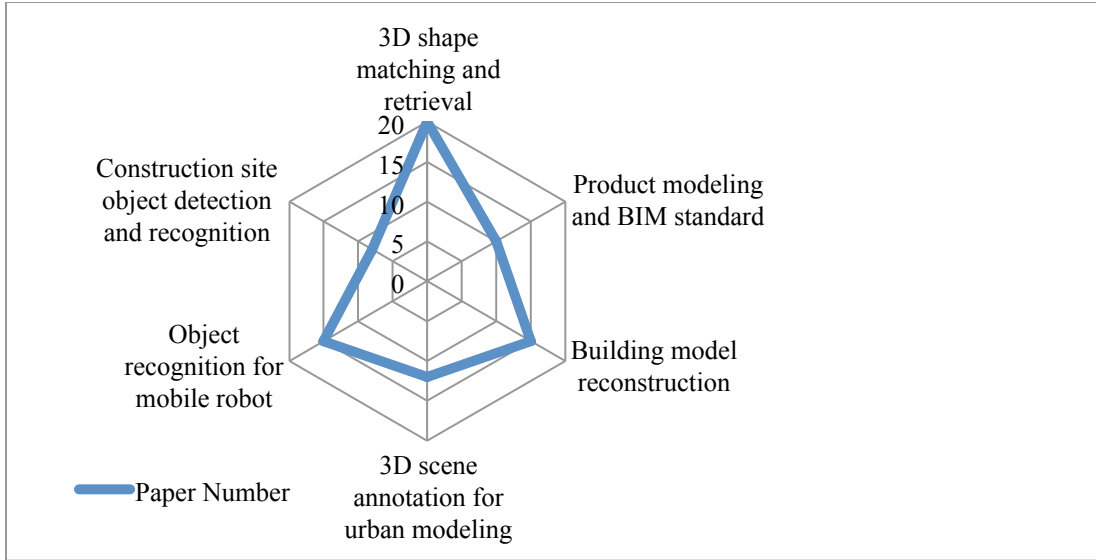


Figure 3. Distribution of reviewed papers in different fields of applications

### 2.2.1. Features of point clouds that can be applied for the matching process

In total, I have identified 29 features of point clouds that can be applied for the matching process. These features are grouped under three categories: (a) *point level features*, which refer to the features associated with each individual point or a cluster of points. Examples of *point level features* include point position, surface normal and point intensity; (b) *entity level features*, which are the features associated with geometric primitives fitted to a point cloud. Examples of entity level features include planer patches fitted to a point cloud and their orientations; and (c) *global features*, which refer to the features that are derived from a set of geometric primitives or segments. A feature can either be a *basic feature*, which refers to primitives with distinct geometric or topological patterns or a *derived feature*, which refers to a feature that is derived from a basic feature. For instance, a point is a basic feature, from which a point coordinate and a surface normal can be derived. Table 2 shows the

features of point clouds that can be used to support the matching between point clouds and BIMs.

Table 2. Features of point clouds to support mapping point clouds to BIMs

Category	Basic Features	Derived Features	Definition	Reference
Point level features	Point	Point position	3D coordinates (x, y, z) of a point	(Adan and Huber 2011; Adan et al. 2011; Anand et al. 2013; Bosché 2009; Bosche and Haas 2008; Bosche and Haas 2008; Choi et al. 2009; Gong and Caldas 2008; Ip et al. 2002; Rusu et al. 2009; Rusu et al. 2010; Rusu et al. 2008; Rusu et al. 2009; Xiong et al. 2013)
		Surface normal	The vector that is perpendicular to a tangent plane fitted to a point	(Adan and Huber 2011; Adan et al. 2011; Ben Hmida et al. 2011; Cantzler et al. 2002)
		Color	RGB value of the point	(Koppula et al. 2011; Pangercic et al. 2012; Posner et al. 2009; Posner et al. 2008)
		Relative distance	Distance between two selected points	(Belongie et al. 2001; Belongie et al. 2002; Cantzler et al. 2002; Iyer et al. 2005)
		Relative angle	Angle between two selected points	(Ip et al. 2002)
		Gradient	The slope of a line connecting two selected points	(Nuchter et al. 2003) (Nüchter and Hertzberg 2008)
Entity level features	Cluster of points segmented from a point cloud	Spin-image	A descriptive image that encodes the properties of a surface fitted to a selected point cluster in an object-centered coordinates	(Bimbo and Pala 2006; Douillard et al. 2011; Frome et al. 2004; Johnson 1997; Johnson and Hebert 1999; Lai and Fox 2010)
		Maximum height difference	The maximum height difference of the points within a given point cluster	(Golovinskiy and Funkhouser 2009; Golovinskiy et al. 2009; Nüchter and Hertzberg 2008; Okorn et al. 2010)
		Standard deviation of the heights	The standard deviation for the z coordinates of points within a selected point cluster in a given coordinate system	(Golovinskiy and Funkhouser 2009; Golovinskiy et al. 2009)
		PCA feature	The magnitude of the linearness (the likelihood of points locating in a line), surfaceness (the likelihood of points locating at a surface), and scatterness (the likelihood of points being scattered)	(Douillard et al. 2010; Koppula et al. 2011)
		Point density	Number of points per square feet within a point cluster	(Douillard et al. 2010; Koppula et al. 2011; Okorn et al. 2010)
		Shape distribution	The distribution of pairwise distances/angles between random points	(Ip et al. 2002; Mahmoudi and Sapiro 2009; Osada et al. 2001; Osada et al. 2002)
		Shape	The distribution of a point	(Belongie et al. 2001; Belongie



		context	relative to all other points in a given point cluster	et al. 2002; Körtgen et al. 2003)
	Geometric primitives fitted to a point cloud	Position	The position of the centroid of a geometric primitive or a reference point on a given geometric primitive	(Anand et al. 2013; Eich et al. 2010; Koppula et al. 2011; Posner et al. 2009; Pu and Vosselman 2006; Pu and Vosselman 2009)
		Area	The area of a surface fitted to a point cloud	(Eich et al. 2010; Nüchter and Hertzberg 2008; Xiong and Huber 2010)
		Volume	The volume of geometric primitives (e.g., sphere, cylinder, cone) fitted to a point cloud	(Golovinskiy and Funkhouser 2009; Golovinskiy et al. 2009; Schnabel et al. 2008)
		Principle direction	The orientation of the principle axis derived from a geometric primitive	(Koppula et al. 2011)
		Surface normal	The orientation of the vector perpendicular to the surface of a geometric primitive	(Adan and Huber 2011; Adan et al. 2011; Anand et al. 2013; Ben Hmida et al. 2011; Cantzler et al. 2002; Hofman and Jarvis 2000; Huber et al. 2011; Huber et al. 2011; Huber et al. 2010; Koppula et al. 2011; Nüchter and Hertzberg 2008; Posner et al. 2009; Posner et al. 2008; Pu and Vosselman 2006; Pu and Vosselman 2009; Xiong et al. 2013)
		Dimension	Length, width, and height of geometric primitives	Xiong et al. 2013
		2D contours	2D boundary of a geometric primitive fitted to a point cloud	(Elberink and Vosselman 2009; Jiantao et al. 2004; Okorn et al. 2010; Rusu et al. 2010; Rusu et al. 2008)
		2D projection	The projection of a geometric primitive on a given plane	(Elberink and Vosselman 2009; Okorn et al. 2010)
	Bounding box	Position	The position of the centroid or reference point in a bounding box	(Anand et al. 2013)
		Volume	The volume of the bounding box	(Paquet et al. 2000)
		Dimension	Length, width and height of the bounding box	(Paquet et al. 2000)
		Fractional occupancy	The ratio of the volume of an object to the volume of the corresponding bounding box	(Anand et al. 2013; Truong et al. 2012)
Global feature	Cluster of points or Geometric primitives	Relative distance	The relative distance between two clusters of points or two geometric primitives	(Anand et al. 2013; Elberink and Vosselman 2009; Golovinskiy and Funkhouser 2009; Golovinskiy et al. 2009; Koppula et al. 2011; Rusu et al. 2009; Rusu et al. 2010; Rusu et al. 2008; Rusu et al. 2009)
		Relative angles	The relative angles and orientation between two clusters of points or two geometric primitives	(Anand et al. 2013; Cantzler et al. 2002; Elberink and Vosselman 2009; Golovinskiy and Funkhouser 2009;

				Golovinskiy et al. 2009; Koppula et al. 2011; Rusu et al. 2009; Rusu et al. 2010; Rusu et al. 2008; Rusu et al. 2009)
		Spatial relationship	How an object is located in relation to other objects (e.g., orthogonal, parallel, adjacent, coplanar, disjoint, touch, equals, covers, overlap)	(Adan and Huber 2011; Adan et al. 2011; Cantzler et al. 2002; Hofman and Jarvis 2000; Huber et al. 2011; Huber et al. 2011; Huber et al. 2010; Nüchter et al. 2003; Verma et al. 2006)
		Topological relationship	The connectivity between cluster of points or geometric primitives fitted to a point cloud	(Adan and Huber 2011; Hofman and Jarvis 2000; Nüchter et al. 2003; Pu and Vosselman 2006; Pu and Vosselman 2009; Schnabel et al. 2008; Triebel et al. 2006)

### 2.2.2. Features of objects represented in BIMs

The features of BIMs that can be used for the matching process are grouped under two categories: (a) *shape features*, which are features related to the geometry, shape and size of an object modeled in a BIM, and (b) *spatial features*, which are features in relation to the position and orientation of an object in the BIM. In addition to the categorization given above, the features of BIMs can be also categorized as: (a) *basic features*, which refer to the geometric elements that form the 3D representation of a BIM and (b) *derived features*, which are the features derived from the basic elements. Table 3 provides a list of features associated with 3D objects modeled in BIMs.

Table 3. A list of features associated with BIMs for matching point clouds to BIMs

Category	Basic Features	Derived Features	Definition	Reference
Shape features	Geometric set: vertex, edge and face	Position	The position of a vertex or position of the referencing points on an edge or a face	(Brenner 2005; BuildingSMART 2012)
		Dimension	The length, width and height of an edge or a face	(BuildingSMART 2012; Pu and Vosselman 2006; Pu and Vosselman 2009)
		Surface normal	A vector that is perpendicular to the tangent plane fitted at a vertex	((Biasotti et al. 2003; Bosché 2009; Bosche and Haas 2008; Bosche and Haas 2008; Hilaga et al. 2001; Pu and Vosselman 2006; Pu and Vosselman 2009; Xiong et al. 2013)
		Shape context,	The distances between the reference point and the points selected on edges or surfaces	(Belongie et al. 2001; Belongie et al. 2002; Bimbo and Pala 2006; Iyer et al. 2005)
		Shape distribution	The distribution of pairwise distances/angles between points subsampled from a face.	(Bimbo and Pala 2006; Ip et al. 2002; Iyer et al. 2005; Osada et al. 2001; Osada et al. 2002)
	Surface model and solid model	Position	The position of the referencing point of a surface model or a solid model	(Bosché 2009; Bosche and Haas 2008; Bosche and Haas 2008; BuildingSMART 2012)
		Orientation	Orientation of the principle axis extracted from a 3D model	(BuildingSMART 2012)
		Volume	The volume of a solid model	(Bimbo and Pala 2006; Liu et al. 2007)
		Dimension	Example of which includes length, width, height, and depth	(BuildingSMART 2012; Gong and Caldas 2008)
		The number of faces	The number of faces belonging to a surface model	(Bimbo and Pala 2006)
		Area to volume ratio	The ratio of the surface area of a model over the volume of the model	(Bimbo and Pala 2006)
		2D projection	2D shape obtained by projecting a 3D model into a 2D plane	(Bimbo and Pala 2006; Bustos et al. 2007; Chen et al. 2003; Funkhouser and Kazhdan 2004; Funkhouser et al. 2003; Jiantao et al. 2004; Min et al. 2002; Vosselman and Dijkman 2001)
		Cross section	The surface obtained by making a straight cut through a surface model	(Bustos et al. 2007; Funkhouser and Kazhdan 2004; Funkhouser et al. 2003; Jiantao et al. 2004; Min et al. 2002)
	Bounding box fitted to the model	Dimension	Length, width, height of the bounding box	(Novotni and Klein 2003; Paquet et al. 2000; Truong et al. 2012)
		Volume	The volume of the bounding box	(Novotni and Klein 2003; Truong et al. 2012)
		Fractional occupancy	The ratio of the volume of an object to the volume of the corresponding bounding box	(Novotni and Klein 2003; Truong et al. 2012)

		Position	The position of the centroid or reference point in a bounding box	(Truong et al. 2012)
Spatial features	Geometric set, surface model, solid model	Relative distance	The distances from the centroid of a 3D object to the centroid of the others	(Anand et al. 2013; Biasotti et al. 2003; Borrmann and Beetz 2010; Borrmann and Rank 2009; BuildingSMART 2012; Chao et al. 2011; Eich et al. 2010; Koppula et al. 2011)
		Relative angle	The angle between surface normal fitted at the centroid or reference points of 3D objects	(Anand et al. 2013; Biasotti et al. 2003; Borrmann and Beetz 2010; Borrmann and Rank 2009; BuildingSMART 2012; Chao et al. 2011)
		Spatial relationships	Defines how an object is located in the space in relation to the other objects (orthogonal, parallel, adjacent, coplanar, disjoint, touch, equals, covers, overlap)	(Adan and Huber 2011; Adan et al. 2011; Beetz et al. 2006; Borrmann and Beetz 2010; Borrmann and Rank 2009; BuildingSMART 2012; Chao et al. 2011; Cruz 2007; Duan et al. 2009; Duan et al. 2010; Eich et al. 2010; Pangercic et al. 2012)
		Topological relationships	The connectivity and aggregation relationships between 3D objects.	(Beetz et al. 2006; Borrmann and Beetz 2010; Borrmann and Rank 2009; BuildingSMART 2012; Duan et al. 2009; Duan et al. 2010; Eich et al. 2010; Iyer et al. 2005; Pangercic et al. 2012; Schnabel et al. 2008)

As shown in Table 2 and Table 3, the identified features of point clouds and BIMs are either shape-related or spatial-related. Shape-related features describe the geometry, shape and size of objects captured by a BIM or a point cloud. Spatial-related features describe the position and orientation of objects captured by a BIM or a point cloud and their relative position, distance, angles and relationships with other objects in the scene. Shape-related features are sensitive to shape discrepancies, and might not be able to identify the correct matches between a point cloud and a BIM when shape discrepancies exist. On the contrary, spatial-related features are more robust to the shape discrepancies, but are more likely to be impacted by location discrepancies.

Spatial-related features are most frequently used in the applications of building model reconstruction, 3D scene annotation for urban modeling, object recognition for mobile robot navigation, and construction site object recognition. These applications use the spatial-related information (e.g., connectivity, relative location and angles between different objects) derived from the 3D scene for object recognition. The limitation of the spatial-related features is their lack of discriminative power to distinguish objects with the similar spatial patterns. The shape-related features are frequently used in the applications of 3D shape matching and retrieval. The shape-related features describe 3D shapes captured by a point cloud or modeled in a BIM. Among the shape-related features, the shape distribution measures the global geometric properties of an object in the 3D space and simplifies problem of comparing 3D shapes to the problem of comparing probability distributions.

Point-level features are the most fundamental features associated with a point cloud. The position of a point and the surface normal derived from the point are the most frequently used features in the review research studies. It is because that these two features provide the basis to derive other features, such as point density, orientation of a point cluster, relative angles between different segments extracted from a point cloud and so on. However, point-level features themselves do not have enough discriminative power to describe an entire object captured by a point cloud. On the contrary, objects are modeled in a BIM at the component level. As a result, geometric information formatted in a point cloud is at a different level of detail as compared to the information represented in a BIM. In order to remove the

differences of levels of detail between point clouds and BIMs, points need to be aggregated into higher-level entities, such as clusters of points, segments and geometric primitives, using point cloud segmentation and clustering techniques. While points can be aggregated into higher-level entities, a building component modeled in the BIM can also be decomposed into different geometric primitives (e.g., faces, vertices and edges) or different 2D views (e.g., cross section, principle axis and 2D projection). By sampling the BIM surfaces, BIM components can be decomposed into a collection of points. The matches between a point cloud and the points sampled from the BIM surfaces can be made at the point-level. Features generated through aggregation of point clouds or decomposition of BIMs allows comparison between the two data sets is made at the same level of detail.

### **2.3 Validation**

In this chapter, I validated the coverage of the identified features in terms of whether the feature list identified in this research covers the majority of features that can be used to identify matches between point clouds and BIMs. I have reviewed research studies from different domains, such as computer vision, robotic, construction, and remote sensing, in order to ensure that the features identified in this chapter are not biased or limited to a specific field.

I then applied cross-validation technique to evaluate the coverage of the features identified in this chapter. Cross-validation is a technique to estimate the generality of a statistical model by dividing the data into two sets: training set and testing set (Kohavi 1995; Refaeilzadeh et al. 2009). Cross-validation validates whether the

performance of a classifier or a prediction model that learned from a given training set is generalized to the testing set. In this chapter, I extended the cross-validation technique to validate the coverage of the identified features and prove that the features identified in this chapter are not biased to one specific field of research studies. I randomly divided the 80 papers reviewed in this research into two sets: training set and testing set. The training set included 60 papers, and the remaining papers were used for the testing purpose. I then evaluated the distribution of the features within the testing and training set and counted the number of unique features that are not covered by the training set but are provided by the testing set. The results are provided in Figure 4.

In Figure 4, the numbers under training and testing columns represent the total number of papers that mention a given feature in the training and testing sets, respectively. As seen in Figure 4, features identified from the testing and training sets show a similar distribution among different the reviewed papers. In other words, if a feature was widely referred in the papers that were within the training set, it was also widely referred in the papers that were in the testing set. For example, within the point cloud related research studies, “point position” and “surface normal of geometric primitives fitted to a point cloud” have been identified as the two most frequently used features in the reviewed research studies. “Spatial” and “topological” relationships have been widely referred both in the training and the testing sets. It was observed that only one additional feature – ‘standard deviation of heights for points within a point cluster’ was identified in the testing set but not shown in the training set. Given the total number of features identified in this research (50 features

in both domains), having only one such feature showed that this research has covered the majority of features that can be used to support the matching between point clouds and BIMs from diverse research studies.

Point Cloud				
Category	Basic Features	Derived Features	Training	Testing
Point level features	Point	Point position	10	4
		Surface normal	3	1
		Color	4	0
		Relative distance between two points	8	0
		Relative angle between two points	4	0
		Gradient between nearby points	2	0
Entity level features	Cluster of points	Spin-image	4	2
		Maximum height difference	2	2
		Standard deviation of heights	0	2
		PCA Feature	2	0
		Point density	3	0
		Shape distribution	2	2
		Shape context	3	0
	Geometric primitives: edge, face, sphere, cylinder, cone and torus	Position	4	2
		Area	3	0
		Volume	3	0
		Principle direction	1	1
		Surface normal	11	5
		Dimension	4	0
		2D contours	4	1
		2D projection	1	1
	Bounding box	Position	1	0
		Volume	1	0
		Dimension	1	0
		Factional occupancy	2	0
Global feature	Cluster of points or Geometric primitives	Relative distance between two clusters of points or geometric primitives.	5	3
		Relative angles between two point clusters or geometric primitives.	5	3
		Spatial relationships	7	2
		Topological relationships	6	1

Figure 4 (a). Count the number of research studies associated with each point cloud related features



BIM				
Category	Basic Features	Derived Features	Training	Testing
Shape features	Geometric set: vertex, edge and face	Position	8	2
		Dimension	2	1
		Surface normal	4	4
		Shape context	2	2
		Shape distribution	3	2
	Surface model and solid model	Position	3	2
		Orientation	1	0
		Volume	2	0
		Dimension	2	0
		The number of faces	1	0
		Surface area to volume ratio	1	0
		2D projection	4	4
		Cross section	3	2
	Bounding box fitted to the model	Dimension	2	1
		Volume.	1	1
		Fractional occupancy	1	1
		Position of the boundi	1	0
Spatial features	Geometric set, surface model, solid model	Relative distanc between 3D objects	5	3
		Relative angle between 3D objects	4	2
		Spatial relationships	9	3
		Topological relationships	8	2

Figure 4 (b). Count the number of research studies associated with each BIM related feature

## 2.4 Conclusions

In this research, I built on the previous research studies in relation to point cloud feature extraction, product representation, data standards for building information models and point cloud and 3D object recognition, and identified a list of features that can be applied to match point clouds to BIMs. I identified 21 features associated with BIMs and 29 features of point clouds. The identified feature could be served as

the foundation to develop matching algorithms that match point clouds to BIMs under different discrepancy conditions. However, there is still a limit understanding on how these features would perform under different discrepancy conditions. Hence, in the next Chapter, I evaluate the performance of the features for matching point clouds to BIMs.

### CHAPTER 3. EVALUATION OF DIFFERENT FEATURES FOR MATCHING POINT CLOUDS TO BUILDING INFORMATION MODELS

The matches between segments extracted from point cloud data and components modeled in BIMs integrate accurate geometric information provided by point cloud data with semantic information contained in BIMs, which supports the identification of possible deviations between the two data sets. As discussed in Chapter 1, the actual shape and location of a component can be different from the design, due to changes occurred in the construction and the different levels of detail that a component modeled in a BIM versus how it is constructed on site.

There can be different types of discrepancies between point cloud data and an as-designed BIM, as seen in Figure 5: (a) *Shape discrepancy*: a component's shape is different than the shape specified in the as-designed BIM; (b) *Location discrepancy*: a component's location is different than the location specified in the as-designed BIM; (c) *Dimension discrepancy*: a component's dimensions are different than what is specified in the as-designed BIM; (d) *Content discrepancy*: a component modeled in the as-designed BIM does not exist in point cloud, and vice versa; and (e) *Composition discrepancy*: multiple components are used to build a single component that is modeled in the as-designed BIM or vice versa.

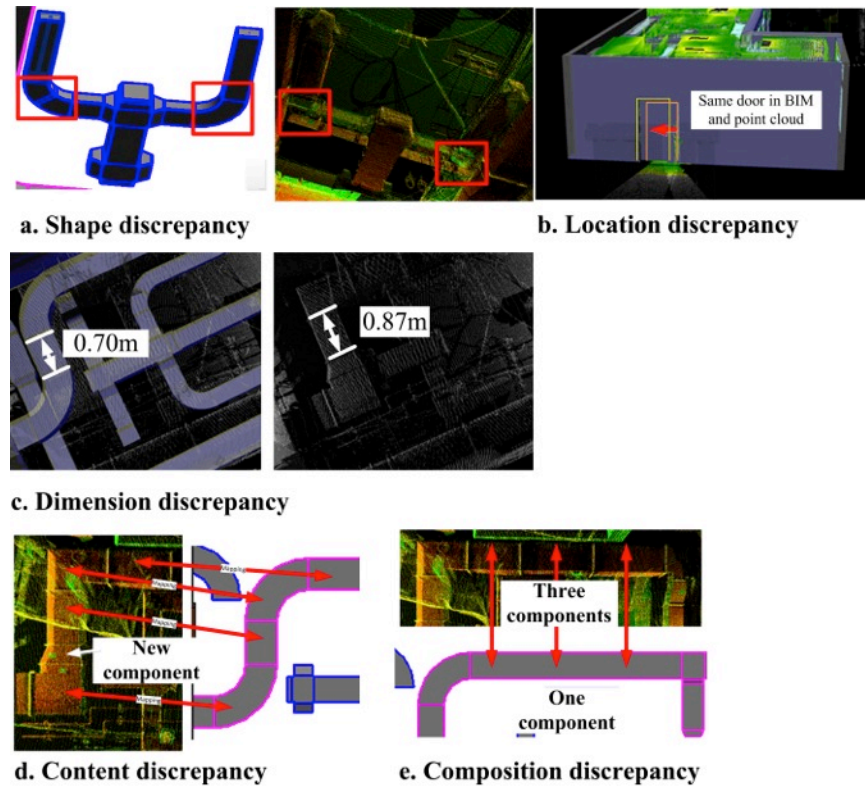


Figure 5. Examples of different types of discrepancies

While these categories are not mutually exclusive, they challenge the matching process and could mislead the matching. A matching approach based on a single type of feature might not always be able to identify all correct matches between point clouds and BIMs. Chapter 2 introduces a list of features that can be applied to match point cloud data to BIMs. Building on this feature list, I first developed four matching approaches that reason with different type of features. Second, I conducted an experimental analysis that evaluates the performance of the four matching approaches under different types of discrepancy situations. Third, I developed two matching approaches that combine different types of features and I analyzed how

those combined approaches could improve the matching results under different discrepancy conditions.

### **3.1 Background Research Studies**

Many approaches have been developed in the computer vision, robotic navigation and remote sensing domains to recognize objects from point cloud data or classify 3D objects into different categories (Bustos et al. 2007; Frome et al. 2004; Golovinskiy et al. 2009; Mahmoudi and Sapiro 2009; Pu and Vosselman 2006). Based on the features utilized, different object recognition approaches can be grouped into two categories: (a) spatial-based object recognition approaches, which recognize objects from point cloud data based on their relative location, distance and relationship with the other objects shown in the scene, and (b) shape-based object recognition approaches, which recognize objects from point clouds based on their shape-related features, such as dimension, shape, volume and size.

#### **3.1.1. Spatial-based object recognition approaches**

The spatial-based object recognition approaches take point cloud data as the input, and classify points or geometric primitives extracted from the point cloud into different classes. The classification is made according to the spatial-related features, such as connectivity, relative distances and angles between different objects, derived from point cloud data, and the heuristics or design models (e.g., BIM, floor plan, 3D design model) that reflect the nature of the collected point clouds. (Adan and Huber 2011; Duan et al. 2010; Eich et al. 2010; Elberink and Vosselman 2009; Huber et al. 2011; Koppula et al. 2011; Nüchter and Hertzberg 2008; Nuchter et al. 2003;

Pangercic et al. 2012; Pu and Vosselman 2009; Schnabel et al. 2008; Verma et al. 2006; Xiong et al. 2013).

Based on the spatial-related features and prior work in this area, I implemented two matching approaches: (a) 2D overlap area matching approach and (b) spatial relationship based graph-matching approach. The 2D overlap area matching approach measures the relative distances between a segment extracted from a point cloud and a component modeled in a BIM based on the overlap area of their 2D projections. The spatial relationship-based graph matching approach utilizes a graph to encode the spatial relationships (i.e., connectivity) between different components in a BIM and segments extracted from a point cloud, and it matches the point cloud to the BIM based on a graph-matching algorithm. The 2D overlap area matching approach simplifies 3D objects into a collection of 2D projections. The features associated with 2D objects are faster to compute than the 3D features (Chen et al. 2003; Jiantao et al. 2004; Wang et al. 2003). On the contrary, the spatial relationship based graph-matching approach requires high computational power, since the graph-matching problem is a well-known NP-complete problem, which cannot be solved in polynomial time (Caetano et al. 2009; Conte et al. 2004; Hofman and Jarvis 2000; Zaslavskiy et al. 2009). Despite the high computation power required by graph matching algorithms, the nature of a graph structure makes it suitable to represent the spatial relationships contained in a point cloud or a BIM. Graph matching algorithms are capable of fully utilizing the spatial relationships to increase the matching accuracy (Chao et al. 2011; Hofman and Jarvis 2000; Van Treeck and Rank 2004).

### 3.1.2 Shape-based object recognition approaches

The shape-based object recognition approaches recognize objects from point cloud data by reasoning with their sizes, shapes, dimensions, and other shape-related features. These approaches can be further categorized into four: (1) region-based object recognition, (2) primitive-based object recognition, (3) distribution-based object recognition and (4) view-based object recognition.

Region-based object recognition approaches reason with regional geometric-related features derived from a point and its neighboring points, such as point density, surface normal and spin image (Chen and Bhanu 2007; Frome et al. 2004; Johnson 1997; Johnson and Hebert 1999; Lai and Fox 2010). Johnson and Huber (1999) defined a descriptor named spin-image to encode the properties of local surfaces fitted to a point's neighborhood. Chen and Bhanu (2007) developed a local surface descriptor, which is calculated based on the angles between the surface normal of a reference point to the points in its neighborhood. The regional geometric-related features are less sensitive to noise and clutters shown in the scene (Bimbo and Pala 2006; Tangelder and Velkamp 2004).

The primitive-based object recognition approaches reason with the geometric properties associated with segments (e.g., planes, planar surfaces) or clusters extracted from point cloud data. The geometric properties include orientation (i.e., surface normal of a planar surface or direction of the principle axis), area, volume, dimension and so on (Gilsinn et al. 2005; Golovinskiy et al. 2009; Huber et al. 2011; Paquet et al. 2000; Rusu et al. 2009).

The distribution-based object recognition approaches represent and compare 3D shapes captured by a point cloud or a 3D model using probability distributions of certain geometric properties (Belongie et al. 2001; Belongie et al. 2002; Ip et al. 2002; Mahmoudi and Sapiro 2009; Tang et al. 2010). Belongie et al. (2001, 2002) defined a shape descriptor named shape context to measure the similarity between different 3D shapes. For every point on a 3D shape, the shape context captures the logarithmic distribution of the distances between a specific point to all other points. Osada et al. (2002) represented a 3D shape as a shape distribution sampled from the properties measured on a 3D object. The properties include distances or angles between a pair of points randomly selected on the surfaces of a 3D object. The advantages of using the shape distribution as a descriptor is that it takes the entire 3D object into consideration, which makes the approach robust to small geometrical distortion and noises in a scene (Osada et al. 2002). In addition, comparing to the other geometric-related features, such as volume, surface area and dimension, shape distribution is more descriptive and is able to distinguish components that have the same volume or surface areas and yet completely different shapes.

The view-based object recognition approaches decompose a 3D object into a series of 2D projections at different angles or cross-sections along different directions (Chen et al. 2003; Funkhouser et al. 2003; Jiantao et al. 2004; Nüchter and Hertzberg 2008). Chen et al. (2003) developed a view-based approach that matches similar 3D shapes based on visual similarity of their 2D projections at different angles. Funkhouser et al. (2003) designed a web-based 3D shape search engine, which allows users to search for 3D shapes that match the 2D sketch they inputted. Jiantao et al. (2004)



represented a 3D shape as a series of cross-sections generated along a given direction. The view-based object recognition approaches simplify the problem of 3D shape matching into measuring the similarity between 2D shapes.

Since the distribution-based features have several advantages over other shape-related features, I selected the shape distribution as the feature to be used to match point cloud data to a BIM. I developed two matching approaches: (a) distribution-based 3D shape matching approach and (b) distribution-based 2D shape matching approach, which is built upon the combination of shape distribution and 2D projections of 3D objects.

### **3.2 Feature Based Matching Approaches**

Built on the previous research studies, I developed four matching approaches: (a) 2D overlap area matching approach, (b) spatial relationship based graph-matching approach, (c) distribution-based 2D shape matching approach, and (d) distribution-based 3D shape matching approach. These four matching approaches reason with different types of features that have different characteristics. Testing the performance of these approaches would be able to help us to understand which features contribute more to the matching process under which discrepancy conditions.

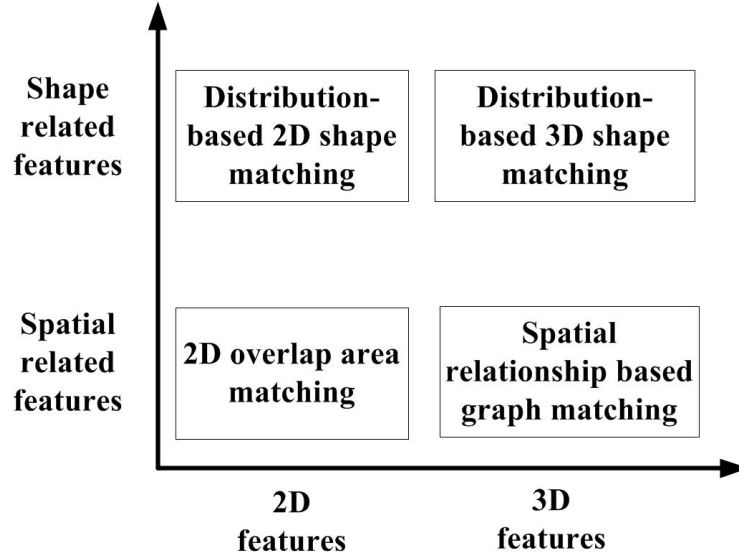


Figure 6. Classifications of features required by the four matching approaches

The four matching approaches take a BIM and a segmented point cloud as input and identify possible matches between segments in the point cloud and components in the BIM. The raw point clouds are pre-segmented using the state-to-art segmentation approaches (i.e., RANSAC and region growing approach). Since the focus of the research described in this thesis is to evaluate how well different features support the matching process, I assumed that the input point clouds have been segmented using RANSAC and region growing approach (Roggero 2002; Schnabel et al. 2007) Each matching approach is detailed in the following sections.

### 3.2.1 2D overlap area matching approach

Reasoning with the spatial proximity between point cloud data and a BIM helps to identify matches between the two data sets. The 2D overlap area matching approach first aligns a point cloud and a BIM into the same coordinate system. Second, it projects the aligned point cloud and BIM into a 2D plane and calculates the ratio of

the overlapping area between the segments extracted from a point cloud and the components modeled in a BIM. The 2D plane is selected based on the type and location of building components. For a building, orthogonal projection is a common way to project 3D components into the 2D space (Okorn et al. 2010; Pu and Vosselman 2009). As seen **Figure 7**, dotted lines represent orthogonal projection of segments from a point cloud and solid lines represent orthogonal projection of components modeled in a BIM. Component  $B_2$  in the BIM is spatially overlapped with three segments of the point cloud.

This matching approach is based on the logic that if segment  $P_i$  in the point cloud overlaps with component  $B_j$  in the BIM, then it is likely that  $P_i$  and  $B_j$  are the same component and should be matched together. If segment  $P_i$  is spatially overlapped with multiple components (e.g.,  $B_1, B_2, B_3 \dots B_n$ ), then the 2D overlap area matching approach compares calculated overlap ratios to a pre-defined threshold in order to determine the final matches. The equation to determine the matches between a point cloud and a BIM is shown in Equation 1. Since information represented in a point cloud is accurate and reflects the as-built condition, I use segments captured by the point cloud as the target objects, and match these segments to building components modeled in the BIM. As the result, the area of orthogonal projection of segments captured by a point cloud is used as the denominator in Equation 1.

Assuming  $P_i$  is overlapped with  $B_j$ :

$$\text{OverlapRatio}(P_i, B_j) = \frac{\text{Overlap surface area}(P_i, B_j)}{\text{Surface area}(P_i)} \quad \text{Equation 1}$$

$P_i$  is matched to  $B_j$  when  $\text{OverlapRatio}(P_i, B_j) \geq \text{threshold } \alpha$

where  $\alpha$  is the threshold for minimum overlap area ratio for the two matched components. Threshold  $\alpha$  is ranging from 0 to 1. As seen in **Figure 7**, if I set the threshold as 36% (the value is chosen just for demonstration purpose), then component  $B_2$  is matched to segment  $P_2$ . The two matches  $(B_2, P_1)$  and  $(B_2, P_3)$  are eliminated from the matching candidate list since their overlap ratio is smaller than the pre-defined threshold. If more than one component exceed the determined threshold, these components are all considered as the mapping candidates.

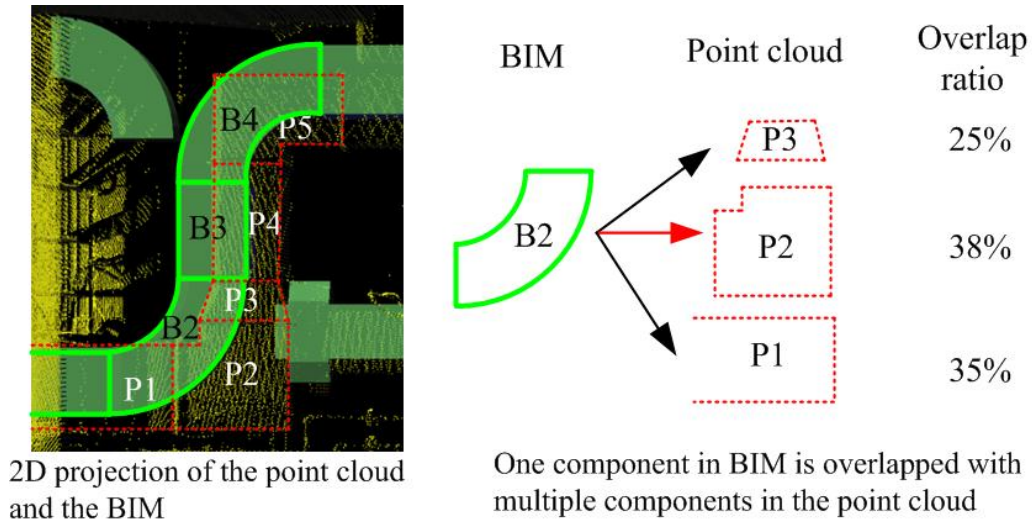


Figure 7. 2D projection of the registered point cloud and BIM

### 3.2.2 Spatial relationship based graph-matching approach

The spatial relationships of segments captured by a point cloud and components modeled in a BIM play an important role in the applications, such as 3D shape matching and point cloud object recognition (Biasotti et al. 2003; Tangelder and Velthkamp 2004). The spatial relationships can be extracted and represented as graphs. A graph  $G = (V, E)$  is composed of edges ( $E$ ) and vertices ( $V$ ). Vertices, in this approach, represent a set of segments extracted from a point cloud or a set of components modeled in a BIM. Edges represent connectivity (i.e., one edge represents one connectivity) and relative distance between two vertices. The relative distance between two vertices is calculated as the Euclidean distance between the centroids of the two components, to which the vertices correspond.

Once graphs are generated using both data sets (i.e., the BIM and the point cloud), vertices of these two graphs are matched to each other using a graph-matching algorithm. Given two graphs  $G_1 = (V_1 E_1)$  and  $G_2 = (V_2 E_2)$ , a graph-matching algorithm aims to find matches  $M: v_i \rightarrow v_j$ , where  $v_i \in V_1$  and  $v_j \in V_2$ . I selected the factorized graph matching (FGM) approach to match two graphs together. The reasons to utilize the FGM approach are that: (a) it reduces the demand for the computational power by converting affinity matrix  $K$  (the matrix that encodes the pair-wise similarity between vertices and edges) into two smaller matrices, vertex affinity matrix  $K_p$  and edge affinity matrix  $K_q$ ; (b) it produces better matching accuracy (e.g., Zhou and De la Torre 2012); and (c) it is more robust to noise.

The first step of the spatial relationship based graph-matching approach is to convert point clouds and BIMs into graphs. As shown in Figure 8, the approach assigns the centroids of the segments/components extracted from the point cloud and the BIM as vertices, and creates edges to link adjacent vertices together. The output of this step is two graphs, one represents the spatial relationships amongst point cloud segments and the other represents the spatial relationships amongst the components in a BIM. In the next step, the approach applies the Factorized Graph Matching (FGM) algorithm to match vertices from one graph to the other. The two types of features that FGM algorithm uses are: (a) relative distances between connecting vertices, and (b) connectivity of vertices.

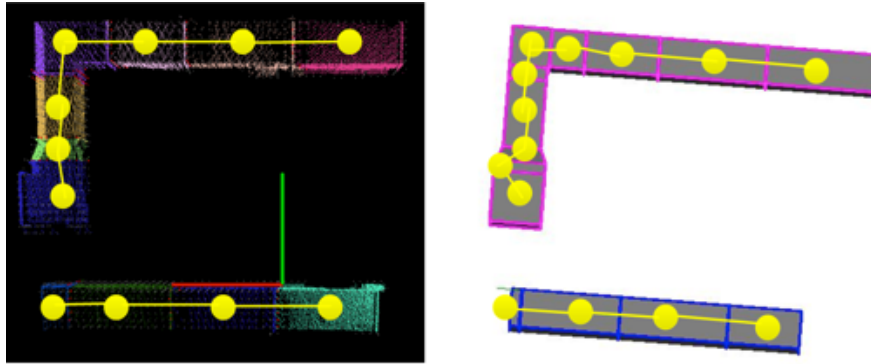


Figure 8. Graph constructed from the point cloud and the BIM

### 3.2.3. Distribution-based 3D shape matching approach

Segments from a point cloud can be matched to components modeled in a BIM based on their shape similarities. The idea behind this approach is that if a segment captured by a point cloud has similar shape as the component modeled in a BIM, then it is likely that these two objects can be matched to each other. This approach

uses the 3D shape distribution (i.e., probability distribution of the geometric properties sampled from segments in point cloud data and components modeled in a BIM) as the feature, and calculates the shape similarity between the components modeled in a BIM and segments extracted from a point cloud.

Figure 9 shows the flow chart for the distribution-based 3D shape matching approach. The approach takes a point cloud and an as-designed BIM as inputs. It first calculates the 3D shape distribution for the point cloud segments and BIM components. For component  $B_i$  modeled in a BIM, the approach measures the Euclidean distance between two random points on the surfaces of  $B_i$ . For segment  $P_j$  captured by a point cloud, the approach randomly picks two points within the point cloud segment, and measures the Euclidean distance between the two selected points. The measured distance is called as D2 distance in Osada et al. 2002. In the following paragraphs, I will refer to the distance between two randomly points sampled from the surface of components in a BIM or point cloud segments as D2 distance. The above step is iterated for  $k$  times ( $k$  = sample size). Generally, higher the  $k$  is, the more accurate the shape distribution is for describing the shape of a 3D object. But in the meantime, more computational power is required to calculate and process the shape distribution. In this research, I set  $k$  as 100,000 to balance the accuracy of the shape distribution with the computation power. After  $k^{\text{th}}$  iteration, the approach generates a  $k \times (m + n)$  matrix, represented as  $M$ . Each column of the matrix  $M$  refers to the probability distribution of D2 distances selected from a segment/component.

In order to compute the similarity between two shape distributions, each shape distribution is represented as a histogram. To create a histogram for the shape distribution (i.e., a column in matrix  $M$ ), I extract the maximum value of D2 distances and divide the space ( $0 \sim$  maximum value of D2 distances) into  $b$  intervals with fixed width. For each interval, I then count how many points in a column of matrix  $M$  are within the interval. The number of points is equal to the height of the interval. The differences between two histograms are then calculated using  $L_2$  normal distance.  $L_2$  normal distance is calculated using equation 2 below (Cha 2007):

$$L_2 \text{ normal distance} = \sqrt{\sum_{i=1}^b |P_i - Q_i|^2} \quad \text{Equation 2}$$

where  $d$  is  $L_2$  normal distance,  $b$  is the total number of intervals,  $P_i$  and  $Q_i$  are the number of samples within the interval  $i$ .



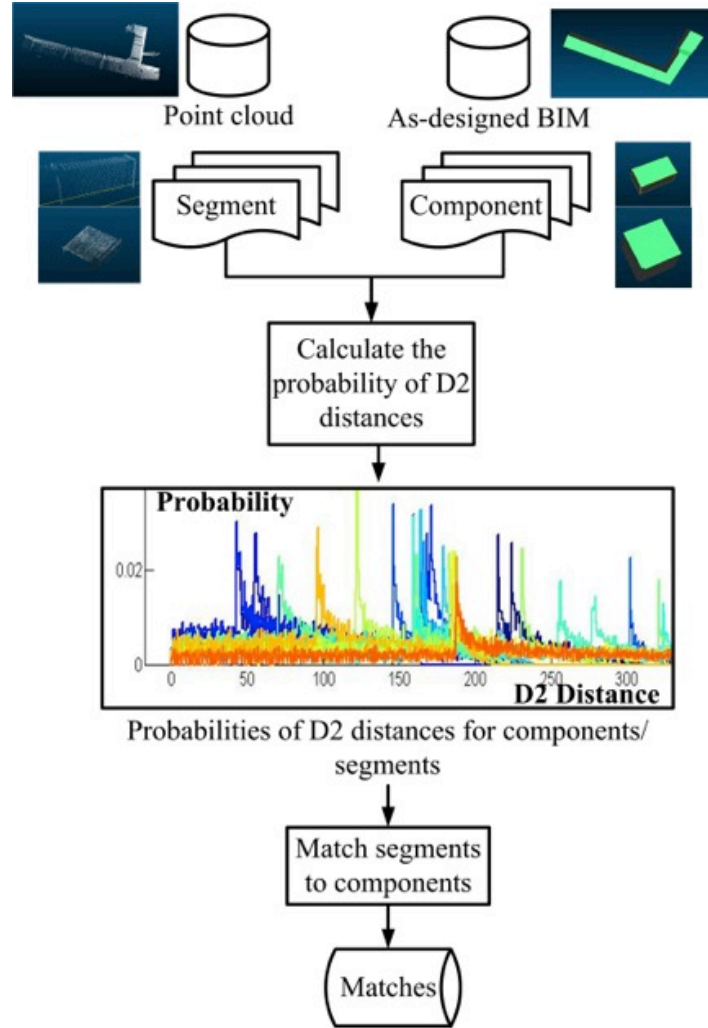


Figure 9. A flowchart depicting the distribution-based 3D shape matching approach

Assuming there are  $n$  components in a BIM and  $m$  segments extracted from a point cloud, based on  $L_2$  norm distance, I constructed a  $m \times n$  3D shape dissimilarity matrix, represented as  $D$ :

$$D = \begin{bmatrix} d_{11} & \dots & d_{1n} \\ \dots & \dots & \dots \\ d_{m1} & \dots & d_{mn} \end{bmatrix} \quad \text{Equation 3}$$

$d_{ij}$  is the  $L_2$  normal distance between the probability distribution of  $P_i$  and  $B_j$ . It represents the 3D shape dissimilarity between  $P_i$  and  $B_j$ .

I then transformed the dissimilarity matrix  $D$  so that each element in the matrix can be compared at the same scale of 0 to 1. The transformation is shown as follows:

$$D = \begin{bmatrix} d_{11} & \dots & d_{1n} \\ \dots & \dots & \dots \\ d_{m1} & \dots & d_{mn} \end{bmatrix} \xrightarrow{\text{normalized}} \begin{bmatrix} d_{11} & \dots & d_{1n} \\ \dots & \dots & \dots \\ d_{m1} & \dots & d_{mn} \end{bmatrix} \times \begin{bmatrix} \frac{1}{\sum_{i=1}^m d_{i1}} \\ \dots \\ \frac{1}{\sum_{i=1}^m d_{in}} \end{bmatrix} \quad \text{Equation 4}$$

According to the matrix,  $P_i$  is matched to  $B_j$  when  $d_{ij} < \alpha$ , where  $\alpha = [0:1]$  and is the threshold that defines the maximum shape dissimilarity between the two matched components.

### 3.2.4 Distribution-based 2D shape matching approach

This approach leverages the idea that the shape-related information associated with 2D views of a 3D object can be used as a descriptor to describe this object (Jiantao et al. 2004). The approach first projects a point cloud and a BIM into a 2D plane. The selection of the 2D plane is made depending on the type and location of building components. Second, the approach extracts the contours of objects from the 2D projection, and matches the two objects from two data sources based on their similarities in the 2D shape.

The similarity between two 2D shapes are measured based on the probability distribution of D2 distances. The pairwise distance of random points on the 2D contours are calculated for  $k$  times ( $k$  = sample size, representing the number of D2 distances calculated from a 2D contour), and are used to create a histogram that represents the distribution of these  $k$  distances. Similar to the distribution-based 3D shape matching approach, the dissimilarity between two histograms is calculated

using  $L_2$  norm distance. The output of distribution-based 2D shape matching approach is a 2D shape dissimilarity matrix, represented as  $D' = [d'_{ij}]$ , where  $d'_{ij}$  is the  $L_2$  norm distance between component  $i$  in a BIM and segment  $j$  in a point cloud.

### 3.3 Experiment Setup

This section discusses an experimental analysis that compares four different matching approaches described in the previous section in terms of how well they identify the correct matches between a point cloud and a BIM. The following sections discuss the details of the experiment including evaluation metrics, testbed, and tuning matching approaches.

#### 3.3.1 Experiment setting

The performances of the four matching approaches are compared in an experimental setup in terms of how well they can identify the correct matches between point cloud and a BIM. Two metrics, precision and recall are used to compare the performance of matching approaches.

$$Precision = \frac{\text{Correctly identified matches}}{\text{Total identified matches}} \quad \text{Equation 5}$$

$$Recall = \frac{\text{Correctly identified matches}}{\text{Total correct matches}} \quad \text{Equation 6}$$

The testbed used in this experiment is a research lab that has gone through a renovation process including the installation of a new mechanical system. I collected point cloud data during the renovation process, which depicts the as-is condition of

the research lab. I also created an as-designed BIM based on design drawings. In this experiment, I mainly focused on the ductworks that were newly installed in the research lab during the renovation process. The reason I targeted on the ductworks is that I observed different types of discrepancies occurred during the construction of the ductworks. Hence, it would be realistic to simulate different types of discrepancies into the corresponding data set and provide a good testbed for evaluating the performance of matching approaches under different discrepancy conditions. Figure 10 shows the point cloud and the as-designed BIM used in the experiment. The point cloud has been segmented using RANSAC and edge detection techniques. The segments of a point cloud are labeled as (P1, P2 ... P8). The ductworks modeled in the as-designed BIM are labeled as (B1, B2... B8).

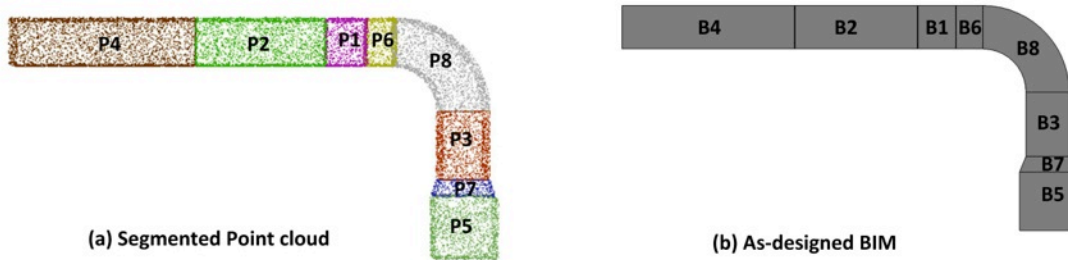


Figure 10 As-designed BIM and the corresponding point cloud

In order to evaluate the performance of different matching approaches under different types of discrepancies, I developed a simulation algorithm that runs on the testbed data and utilize a widely used building information modeling tool as the platform to randomly generate different types of discrepancies into a given set of components. The simulation algorithm is running as an add-on to the modeling tool,

and it takes a BIM as the input. For each building component modeled in the BIM, the algorithm randomly changes the dimension (e.g., length, height and width) and shape of the component. For a group of components that are connected to each other, the algorithm randomly changes the location and the connectivity of these components as well. To simulate composition discrepancy (i.e., the topological change), the algorithm combines multiple components together or divides one component into several new components. I determined the ranges of values for discrepancies to be used during the simulation (e.g., length, width, height, distance) so that they are close to real world scenarios. For instance, while changing the location of the ductworks, they cannot be moved to other rooms or installed on the floors.

As seen in Figure 11, the selected ductworks installed in the renovated space is used as the baseline model and then different types of discrepancies are introduced to the baseline model to generate different versions of the model with deviations different than what is captured in the point cloud. Using the simulation algorithm, I generated 39 models, which covered different types of discrepancies, including shape, dimension, location, and composition discrepancies as compared to the baseline model. In total, the testbed included 40 pairs of point cloud (the point cloud remain the same) and the corresponding BIM (each pair is called as one scenario) for the experiments. Among the 40 scenarios, 30% of them have shape discrepancies, 65% have dimension discrepancies, 75% have location discrepancies and 10% of them have composition discrepancies.

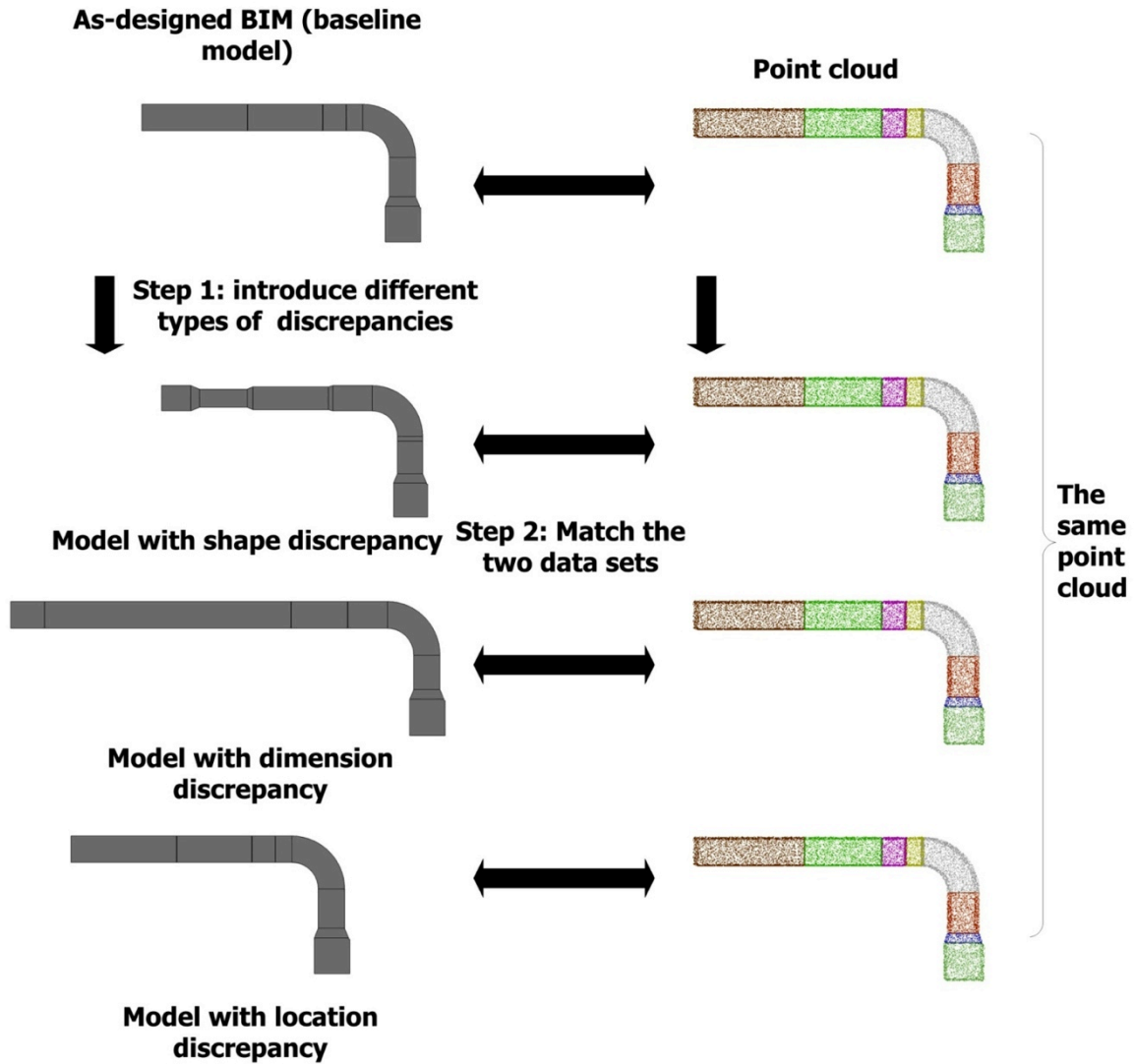


Figure 11. Example set of models with different discrepancies for the experiment

### 3.3.2. Tuning the matching approaches

The three matching approaches (i.e., 2D overlap area matching, distribution based 2D shape matching and distribution based 3D shape matching) need to set up a value for the threshold in order to perform the matching task. The value of the threshold could impact the result of these three matching approaches. In order to determine the near-optimal value for the thresholds required by these matching approaches, the

testbed data were randomly divided into a training set (25 scenarios) and a testing set (15 scenarios). The proportion chosen for the training set is 62.5% and the proportion for the testing set is 37.5%. The choice of 62.5 to 37.5 proportion is to ensure that there were sufficient training data to determine the near-optimal value of the thresholds, but not too much training data that might cause the over-fitting. In this research, I used F1 measure to choose the near-optimal value of the thresholds. The F1 measure (also called as F1 score) is a measure of the accuracy for a test. It considers both precision and recall and is calculated as Equation 7 (Hripcsak and Rothschild 2005; Huang et al. 2005).

Equation 7:

$$F = \frac{(1 + \beta^2) \times \text{precision} \times \text{recall}}{(\beta^2 \times \text{precision}) + \text{recall}}$$

where  $\beta$  is the balance between precision and recall. In this research, precision and recall are evenly weighted. Hence,  $\beta = 1$ . The best F1 measure has the value as 1 and the worse F1 measure has a value as 0.

Figure 12 shows the change of precision and recall for the three matching approaches when the thresholds are changing from 0 to 1. For each matching approach, four different curves are shown in Figure 12, which are: (1) precision vs. threshold, (2) recall vs. threshold, (3) F1 measure vs. threshold and (4) summation of precision and recall vs. threshold. As shown in Figure 12, the value of threshold significantly impacts the performance of the matching approaches. In this experiment setting, same weights are assigned to the precision and recall, and the near-optimal threshold

value has been determined as the value that is able to generate the highest F1 measure.

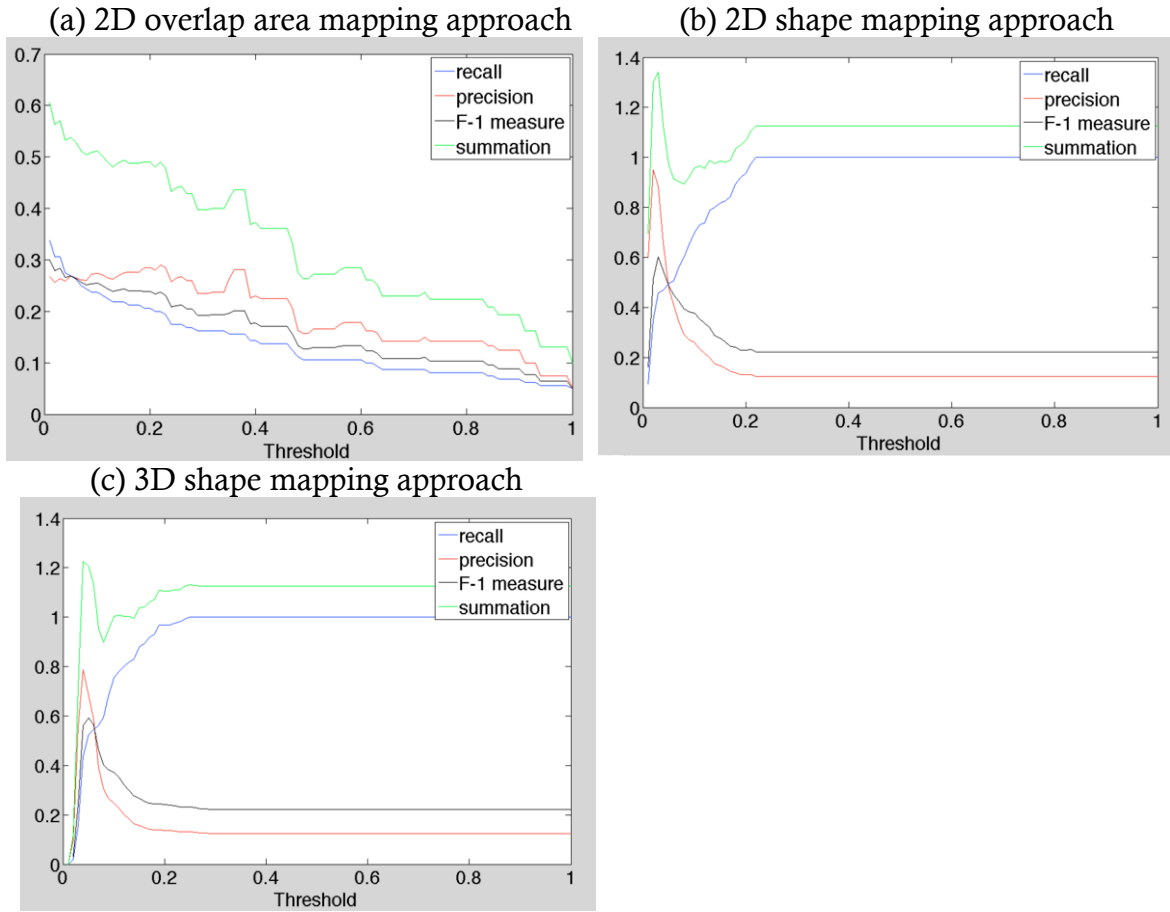


Figure 12. Tradeoff of the precision and recall when threshold is changing

The near optimal values for the thresholds of the three matching approaches are shown in Table 4. For the 2D overlap area matching approach, the threshold controls how sensitive the approach responds to location displacements. When the threshold is equal to 1, the overlap ratio between two matched components needs to be 100%. In the real world, a component might not be constructed at the exact location specified by the as-designed BIM. Hence, there is a need to reduce the value of the threshold in order to add the tolerance for location displacements. As seen in



Figure 12(a), with the increase of the threshold for the 2D overlap area ratio, recall reduces and precision first grows and then declines. The decline of the precision happens when the threshold for the 2D overlap area ratio is greater than the size of components. In that case, the 2D overlap area matching approach cannot extract any matches between the point cloud and the BIM since no overlap ratio can meet the threshold.

The values of the thresholds for the distribution-based 2D and 3D shape matching approaches control the minimum level of shape similarity required for two matched components. As seen in Figure 12(b) and (c), when the values of thresholds for 2D and 3D shape are equal to 0, the approaches are extremely sensitive to shape and dimension discrepancies and two components need to be identical to each other in order to be matched together. When the values of thresholds for 2D and 3D shape are equal to 1, the two approaches are not sensitive to shape and dimension discrepancies at all, and two components can be matched even though their shapes are completely different. As shown in Figure 12(b) and (c), with the increase of the thresholds, the recall values of the two approaches increase while the precision values reduce.

Table 4. The definitions and optimal values for the thresholds

Matching approach	Definition	Trend	Near-optimal value
2D overlap	Pi matches to Bj when OverlapRatio (Pi, Bj) $\geq$ threshold $\alpha_1$	When $\alpha$ increases, two matched components need to have more overlapping area	0.01 (the overlap ratio = 1%)
2D shape	Pi matches to Bj when distribution distance(Pi, Bj) $<$ threshold $\alpha_2$	When $\alpha$ increases, the approach is less sensitive to the 2D shape differences	0.03 (the difference of shape distribution = 3%)
3D shape	Pi matches to Bj when distribution distance(Pi, Bj) $<$ threshold $\alpha_3$	When $\alpha$ increases, the approach is less sensitive to the 3D shape differences	0.05 (the difference of shape distribution = 5%)

### 3.4 Experiment Results

After tuning the matching approaches, I evaluated the performance of the four matching approaches using the testing data. The testing data is composed of 15 scenarios with different types of discrepancies. For every scenario, the output of each matching approach is a matching matrix  $M = [m_{ij}]$ , where  $i$  indicates the segments in a point cloud ( $P_1, P_2 \dots P_n$ ), and  $j$  indicates the components modeled in a BIM ( $B_1, B_2 \dots B_m$ ). As shown in equation 8,  $m_{ij}$  can be 0 or 1. When  $m_{ij}$  is equal to 1, segment  $P_i$  from the point cloud is matched to component  $B_j$  modeled in the BIM. The matching results for the four different matching approaches are summarized in Table 5.

$$m_{ij} = \begin{cases} 1 & P_i \text{ matches } B_j \\ 0 & P_i \text{ and } B_j \text{ are not the same component} \end{cases} \quad \text{Equation 8}$$

The matching matrix is visualized by a binary image, as shown in Figure 13. The binary image is divided into  $n \times m$  grids,  $n$  = the number of segments in the point

cloud and  $m$  = the number of components modeled in the BIM. If  $grid_{ij}$  is filled in black, then  $P_i$  is mapped to  $B_j$ . The x-axis in the binary image shows the segments extracted from a point cloud (labeled as “original”) and the y-axis shows the components modeled in a BIM (labeled as “target”).

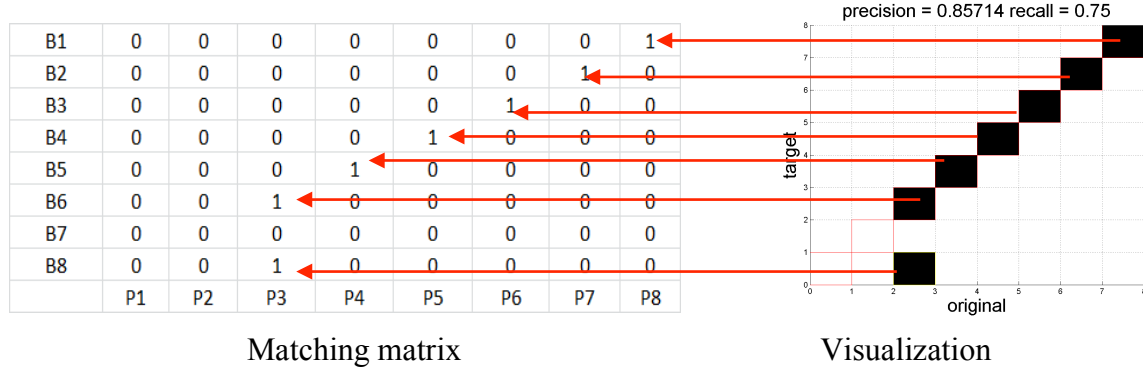


Figure 13. Visualization of a matching matrix

Appendix A shows the matching results of the four matching approaches. The results are presented in a table, where each row corresponds to one specific scenario (i.e., a pair of point cloud and BIM). Let’s take one scenario as an example. In the given scenario, there are no shape and dimension discrepancies between the point cloud and the simulated BIM. A small location displacement has been added into the simulated BIM. Figure 14 shows the results of the four matching approaches. Spatial relationship-based graph matching approach achieved 100% precision and recall, since the connectivity among components remains the same in the point cloud and the BIM. 3D shape based matching approach was able to identify all the correct matches between the point cloud and the BIM but it also incorrectly matched point cloud segment  $P_2$  to BIM component  $B_4$ . 2D shape based matching approach achieved 100% recall and 80% precision. It generated two incorrect matches  $P_2 \leftrightarrow B_4$

and  $P_4 \leftrightarrow B_2$ . As seen in Figure 14, component  $P_2$  and  $P_4$  are both modeled and constructed as rectangle ductworks. They have the same width and height but slightly different length (2" difference). Due to the similar shape and dimension, the 2D and 3D shape based matching approaches have difficulty to distinguish these two components from the point cloud and the BIM. The 2D overlap area matching approach had 100% recall and 66.67% precision. The incorrect matches generated by the 2D overlap area mapping approach are caused by the location displacement.

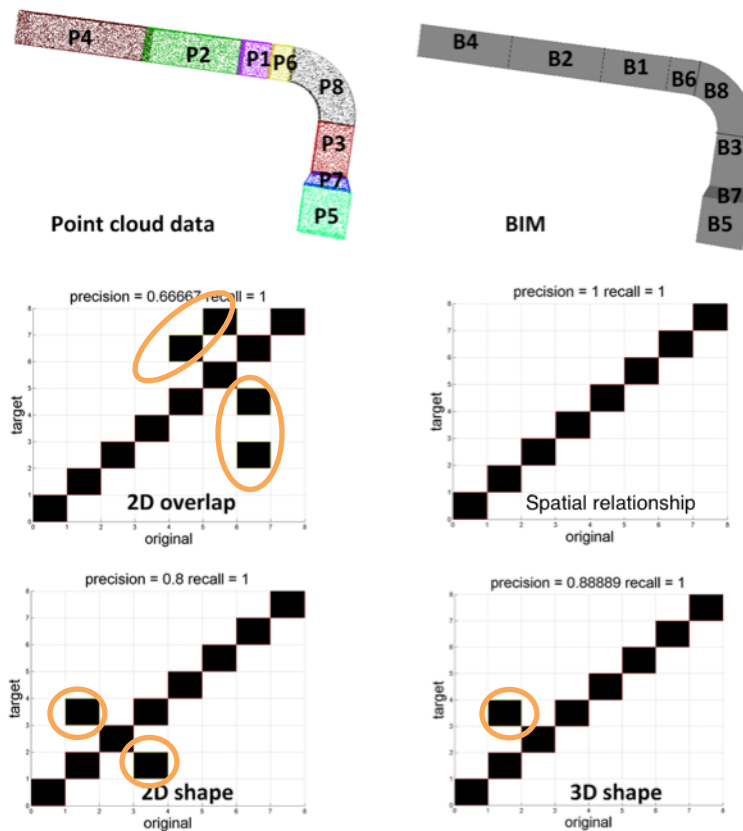


Figure 14. Mapping results for the four mapping approach in a case where there is a slight (2") difference in the length of a component between a point cloud and a BIM

The 2D overlap area matching approach is sensitive to location displacements. One matching result of the 2D overlap area matching approach is shown in Figure 15. Components that have the overlapping 2D surface area are matched together. However, due to the location, shape, and dimension discrepancies, one segment in the point cloud might be spatially overlapped with multiple components in the BIM. Hence, the approach identifies a set of matching candidates, and further analysis is required in order to eliminate the incorrect candidates, so as to increase the precision of the approach.

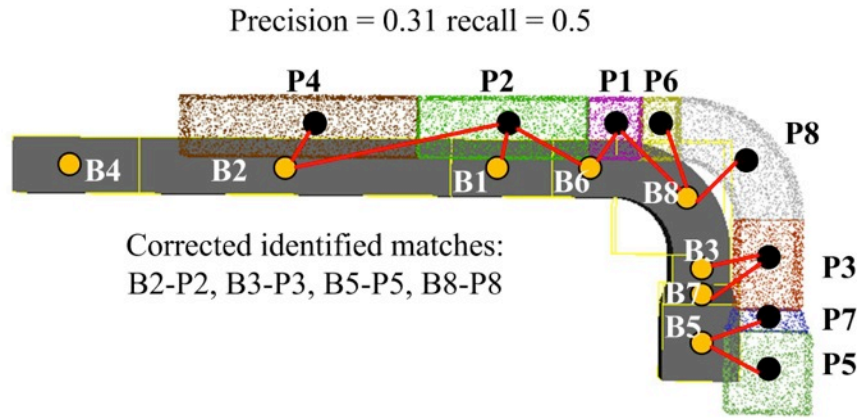


Figure 15. Matches identified by the 2D overlap area matching approach for one example case

The performance of the distribution-based 2D and 3D shape matching approaches are not impacted by the location displacements. Figure 16 describes how shape and dimension discrepancies could impact the 2D and 3D shape matching results. The red arrows represent the matches identified by both 2D and 3D shape matching approaches, the blue arrows represent the matches solely identified by 3D shape

matching approach, and the black arrows represent the matches solely identified by 2D shape matching approach. Since 2D and 3D shape matching approaches both target on shape related features, they showed the similar performance in the experiment. The arrows labeled as “X” indicate incorrect matches identified by the two matching approaches. The incorrect matches are caused by the shape discrepancies between the point cloud and the BIM.

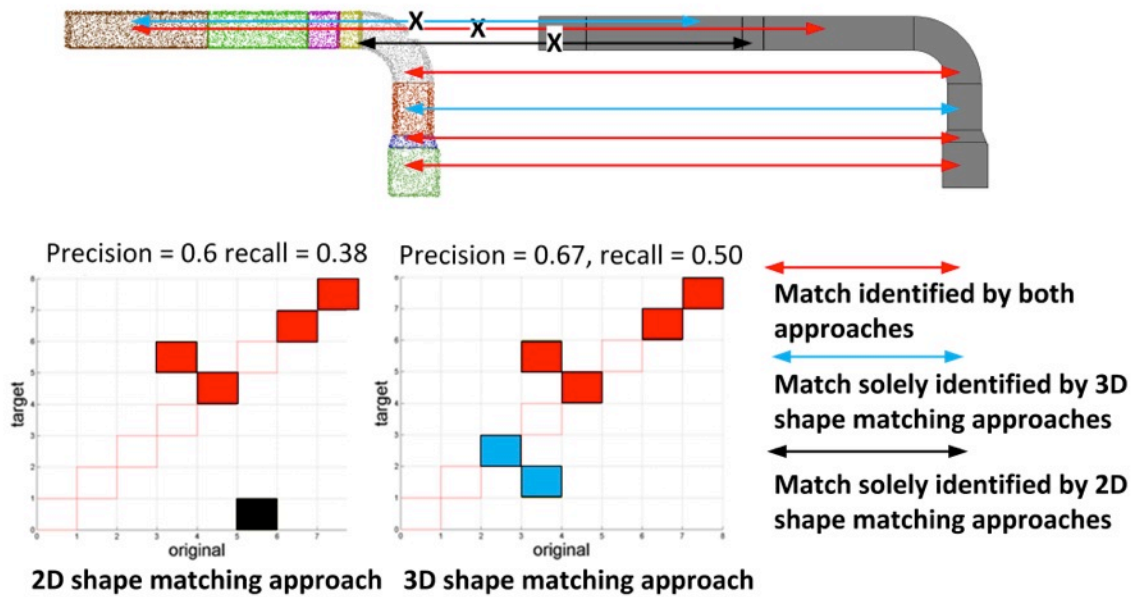


Figure 16. The matches identified by the 2D and 3D shape matching approaches or one example case

Spatial relationship based graph-matching approach is able to identify the correct matches regardless of shape, dimension and location discrepancies. However, in scenarios where composition discrepancies exist, the precision and recall of the spatial relationship-based graph matching approach will reduce.

Table 5. Results of matching process for the four matching approaches for testing and training data set

Approach	Training		Testing	
	Precision	Recall	Precision	Recall
2D shape	0.89	0.46	0.80	0.54
2D overlap	0.27	0.34	0.36	0.44
3D shape	0.68	0.53	0.76	0.59
Spatial relationship based	1	0.82	1	0.89

In summary, a single matching approach that is solely based on one type of feature does not identify all correct matches in all cases. For the cases within which the connectivity of building components remain the same in a point cloud and a BIM, the spatial relationship based graph matching approach has the best performance since it is robust to shape, dimension and location discrepancies. The shape based matching approaches are robust to location and composition discrepancies. However, under circumstances when multiple components are designed and modeled with same shape or a building component is not manufactured or constructed as design specifies, the shape based matching approaches might perform worse than the spatial relationship and 2D overlap area based matching approaches. Taking the entire training and testing data into consideration, the 2D overlap area matching approach has the worst performance (i.e., low precision and low recall). It is because this approach is sensitive to location displacements caused by shape, location, and dimension discrepancies. There is no absolute answer in terms of which is the best feature to match point clouds to BIMs, since all the features tested in this research are sensitive to certain types of discrepancies and are robust to others. It is possible to combine different types of features to improve matching results. The

next section describes two approaches that I developed to combine different types of features for matching.

### **3.5 Combination Of Different Features To Improve Matching Results**

In this section, I combined the spatial and shape related features together using two approaches: (1) re-aligning point cloud and BIM based on 3D shape distribution to reduce the impacts of location discrepancies on the area approach and (2) filtering the matching candidates generated by the 2D overlap area matching approach using 3D shape distribution feature (called as 3D filter in the following paragraphs).

#### **3.5.1. Re-aligning point cloud and BIM based on 3D shape feature**

To reduce the negative impacts of location discrepancies on the performance of 2D overlap area matching approach, I developed an approach that combines the 2D overlap area with 3D shape features. The details of this approach are provided in Figure 17. This approach first applies 3D shape matching approach to identify the correspondences between a point cloud and a BIM. Second, it chooses one component in the BIM and selects the point cloud with highest similarity as the reference and re-aligns the point cloud and the BIM together in order to remove the location displacements. After the point cloud and the BIM are aligned together, the 2D overlap area matching approach is then applied to identify matches between the two data sets.



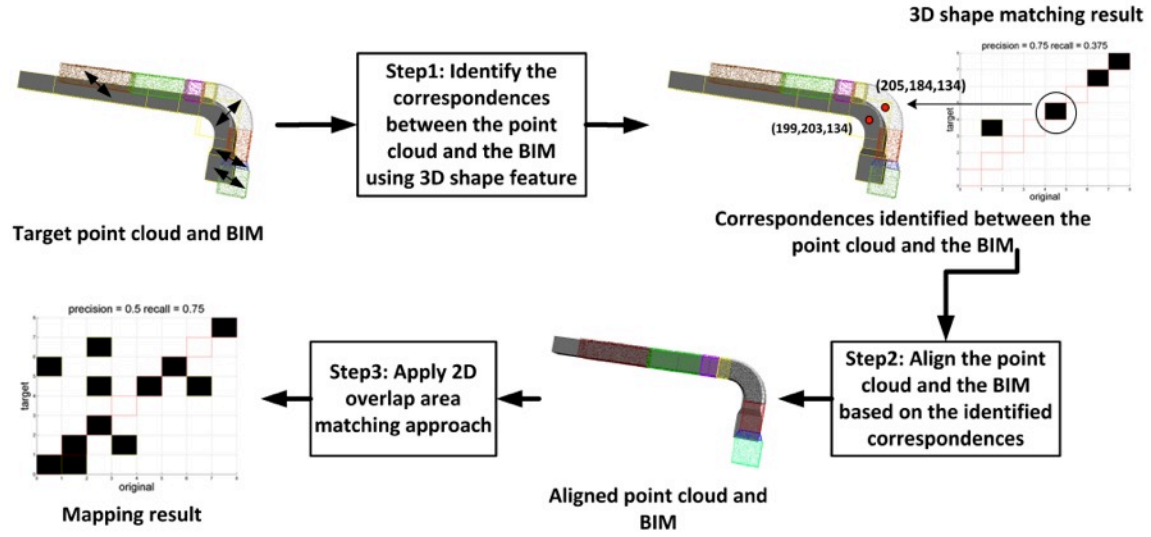


Figure 17. The process of 3D shape plus 2D overlap area matching approach

### 3.5.2. Filtering the matching results generated by the 2D overlap area by the result of matching approach using 3D shape features

As discussed in the previous section, the location discrepancy can be reduced by re-aligning the point cloud with the BIM according to the correspondences identified by 3D shape feature. However, even after the re-alignment, one segment from the point cloud might still be overlapped with multiple components modeled in the BIM due to the dimension and shape discrepancies. The matching candidates obtained by the 2D overlap area matching approach can be further filtered based on the 3D shape features.

After running 2D overlap area matching approach, one segment in the point cloud is matched to multiple components in the BIM. In order to eliminate the incorrect matches, for each segment  $P_i$  in the point cloud, the shape filter first extracts all the matching candidates ( $B_i \dots B_j$ ) that are associated with this segment. If the segment  $P_i$

has more than one matching candidate, the shape filter then calculate the shape distribution for each of the matching candidates. Third, the shape filter selects the matching candidate that has the most similar shape distribution as segment  $P_i$ , and matches them together. By using the 3D shape filter, I can filter out the incorrect matches so as to increase the precision of the matching approach. However, the shape filter could also eliminate the correct matches when the same components are modeled and constructed in different shapes and dimensions. Table 6 shows the matching results by combining features together. The combination of different types of features together is capable of identifying more correct matches as compared to when these features are used separately. The precision of the 2D overlap area matching approach after re-alignment based on 3D shape feature approach is much higher than the precision of the 2D overlap area matching approach, since the re-alignment reduces the location discrepancy. But, the precision of the approach is still lower than the precision of the 3D shape matching approach. The reason is that the performance of the 2D overlap area matching approach is bounded by its sensitivity to location displacements caused by shape and dimension discrepancies. Adding a shape filter is similar to adding a constraint on the 2D overlap area matching approach, which specifies the maximum tolerance for the shape differences between two matched components. As a result, the shape filter could filter out incorrect matches and improve the precision. However, it also increases the sensitivity to the shape discrepancies, which could reduce the recall.

Table 6. The precision and recall comparison for 2D overlap & 3D shape approaches with the combination of the two approaches

Matching approaches	Precision	Recall
2D overlap alone	0.36	0.44
3D shape alone	0.76	0.49
Re-aligning point cloud and BIM using 3D shape feature	0.66	0.78
Filtering 2D overlap matching results with 3D shape features	0.71	0.67

### 3.6 Validation

To evaluate that the performances of the matching approaches developed in this research can be generalized to different configuration of mechanical components and different types of discrepancies, I used three different sections of ductworks (see Figure 18) as the validation dataset. The three sections of ductworks contain mechanical components with different shapes, dimensions and spatial relationships in order to make ensure that the results are not biased to a specific scenario.

The developed simulation algorithm has been used to simulate different types of discrepancies (i.e., dimension, location, shape and composition discrepancies) into the validation dataset. The validation dataset contains 60 pairs of point clouds and BIMs. 40 pairs were used as the training and 20 pairs were used as the testing set. The thresholds for the matching approaches are determined using the training data. In this validation, the threshold of 2D overlap area matching approach is 6%, the threshold of 2D shape matching approach is 3% and the threshold of 3D shape matching approach is 5%.

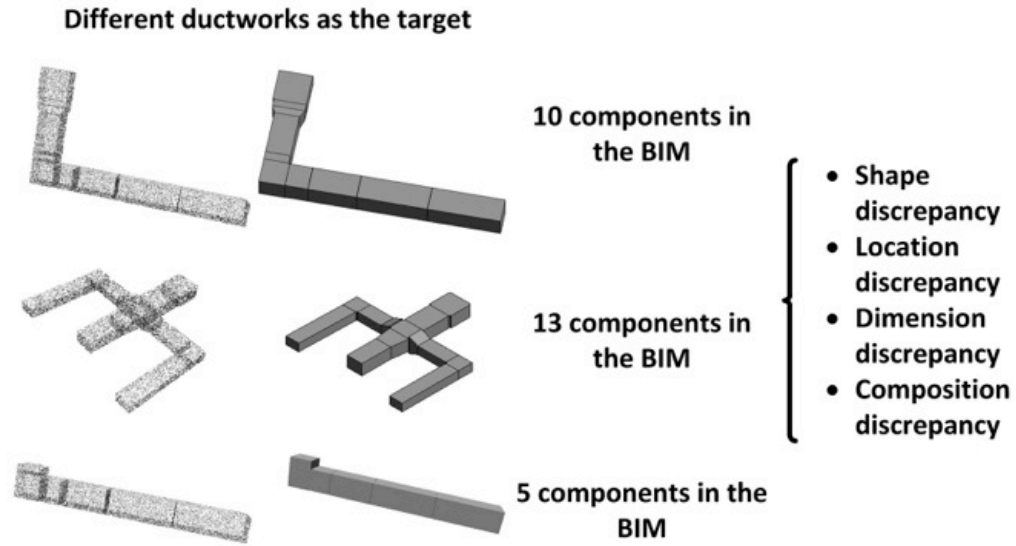


Figure 18. Validation data set includes different ductworks and different types of discrepancies applied to all three sets of ductworks

The matching results gained from the validation dataset are shown in Table 7. The performances of different matching approaches on the validation dataset show similar patterns as in the experimental dataset.

Table 7. Matching results for the validation dataset

Approaches	Precision	Recall
2D overlap area	0.35	0.31
2D shape	0.45	0.57
3D shape	0.3	0.59
Spatial relationship	0.89	1
Re-alignment using 3D shape feature	0.79	0.46
3D shape filter	0.41	0.64

### **3.7 Conclusions**

Matches between point cloud and a BIM can be identified based on different types of features associated with these two data sets. In this chapter, I implemented six matching approaches that reason with and combine different types of features to identify matches between segments in a point cloud and components modeled in a BIM. The performance of a matching approach is impacted by various factors, such as discrepancies existing between point cloud and a BIM, and the features used in the matching approach. I conducted an experimental analysis that evaluates the performance of the developed matching approaches. The experimental analysis reveals that different features are robust to different types of discrepancies and the combination of features could improve the matching results. The object-level matches identified from the matching process associate the design information contained in an as-designed BIM to the actual geometric information captured by point clouds. The matches will allow construction professionals to identify the discrepancies between an as-designed BIM and the as-built condition so as to update the as-designed BIM and remove such discrepancies.

## **CHAPTER 4. AN APPROACH FOR LEVERAGING PROGRESSIVE LASER SCANS**

Point cloud data captured by laser scans play an important role in the process of BIM update by providing accurate geometric information that depicts the as-built building conditions. However, point cloud data collected at a given point in time typically is not able to provide all the geometric information required for creating/updating a BIM, due to three major reasons. First, temporary work, machinery, laborers, and material periodically occlude a construction scene. Similarly, permanent building components occlude other building components and hinder capturing of complete view of the components within the range of the laser scanner. Examples of such instances include ductwork being hidden behind ceiling tiles or wall rough-ins being hidden behind walls. Second, building components might not be installed/constructed at the time of scanning. Third, when a building component is under construction, a laser scan performed at a single point in time might not be able to capture the final form of the component. In current practice, laser scanning is typically done before and/or after construction to capture as-built condition of construction projects (Liu et al.). However, due to the reasons given in the previous paragraph, not all of the geometric information for all building components can be captured.

With the goal of achieving accurate and comprehensive geometric information about all components being installed during construction, Chapter 4 presents an approach developed to evaluate the information contained in a point cloud so as to support the

decision of which subset of point clouds should be registered together in order to provide a complete set of geometric properties for building components with less storage size and processing time. The developed approach evaluates the geometric information associated with surfaces of building components that are shown in a point cloud data and assesses the additional geometric information generated by combining more point clouds together. While doing this assessment, the approach utilizes two metrics: (a) coverage ratio, which is used to evaluate the quantity of captured geometric information per component per point cloud, and (b) dissimilarity ratio, which identifies the additional information that can be generated by adding a point cloud to another point cloud.

#### **4.1 Related Research Studies**

The research work presented in this chapter builds on the previous research studies in relation to (1) evaluation of the quality and quantity of information captured by point cloud data, and (2) point cloud comparison approaches.

##### **4.1.1. Evaluation of the quality and quantity of information captured by a point cloud data**

The studies that evaluate the quality and quantity of information captured by point cloud data can be grouped into two: (a) Approaches that assess the accuracy of point cloud data by comparing the measurements on the real physical objects to the measurements taken in the point cloud data, and (b) Approaches that compare the point cloud data to other reference data, such as building information models.

The first group of research studies focuses on assessing the accuracy of geometric information captured by point clouds. To support the assessment, these research studies conducted measurements on the real physical objects and compared the measurements to the measurements taken from objects captured in point clouds (Boehler et al. 2003; Boehnen and Flynn 2005; Fröhlich and Mettenleiter 2004; Golparvar-Fard et al. 2011). Such accuracy analysis is used to support different engineering applications, including (a) identifying the factors (e.g., resolution and range of laser scanners, reflectivity of scanning surface, etc.) that could impact the accuracy of captured point cloud data (Boehler et al. 2003; Boehnen and Flynn 2005; Clark and Robson 2004; Fröhlich and Mettenleiter 2004; Kersten et al. 2005; Mechelke et al. 2007; Tang et al. 2009), and (b) evaluating the accuracy of point cloud data captured by laser scanners and assessing the applicability of using laser scans to support a specific task (e.g., 3D survey, quality control, surface flatness detection, etc.) (Golparvar-Fard et al. 2011; Park et al. 2007; Tang et al. 2010).

Such studies are well aligned with the work presented in this research in terms of assessing the quality of geometric information captured by laser scanners. However, one drawback of the approaches in this group is that such studies do not evaluate the completeness of data provided by a point cloud (i.e., how many building components and their associated geometric information can be provided by a point cloud to support the update of a BIM). The research presented in this thesis differs from these previous studies with respect to quantifying the geometric information captured by a point cloud.



The second group of research studies formulated a deviation-based approach in order to identify the differences between point cloud data with a reference model, such as building information model (Akinci et al. 2006; Anil et al. 2011; Bosché et al. ; Oude Elberink and Vosselman 2011; Tang et al. 2011; Tang et al. 2010). The patterns of deviations can be used to indicate the quality of a point cloud data and how much information is captured by the point cloud data, assuming that the reference models represent the correct information. Synthesis of the studies in this field suggests that three major concepts can be used to characterize the information contained in a point cloud (Tang et al. 2010, Anil et al. 2011(Oude Elberink and Vosselman 2011). These are: (1) the point density, which defines the number of points within in a unit area, (2) the uncertainty, which refers to the standard deviation of the shortest distance between points and the surfaces fitted to the points, and (3) the occlusion, which defines the regions of the object surface that have no corresponding points in the point cloud. The point density determines the level of detail of the geometric information that a point cloud is able to provide for a given object. A high-density point cloud is capable of providing geometric information with greater details. The uncertainty is used to define the reliability of the measurements taken in a certain region within a point cloud. The uncertainty could be caused by noise in the data, edge losses (i.e., the mixed-pixels at the edge of two surfaces), low reflectivity of a surface, and irregularities of surfaces (Tang et al. 2009). Among these three concepts, the occlusion is capable of providing the information regarding which parts of a building surface is occluded in a point cloud. Hence, the research presented in this

thesis extends the concept of occlusion to evaluate the completeness of the geometric information captured by point cloud data.

#### **4.1.2. Comparing two point clouds and identifying the differences**

Multiple point clouds can be combined together using the technique called registration. There are different registration approaches, which can be grouped into two classes: coarse registration and fine registration (Salvi et al. 2007). The coarse registration methods align a set of point clouds using the correspondences among these data sources. For instance, the corresponding points can be manually selected from the point clouds, and the methods will align the set of point clouds in order to minimize the distances between the selected corresponding point clouds. As the result, whether or not the selected point pairs are equivalent to each other will determine the accuracy of the final alignment. To improve the accuracy of the coarse registration approaches, targets (e.g., labels, markers, etc.) have been placed in the scene during the scanning process. The equivalent targets shown in different point clouds are used as the correspondences to guide the registration process.

The fine registration takes the entire point cloud into consideration (Chen and Medioni 1992; Rabbani et al. 2007; Salvi et al. 2007). Iterative closest point (ICP) is a well-known fine registration algorithm. ICP algorithm iteratively adjusts the alignment of a set of point clouds until the distances between points in one point cloud and their closet points in the others are minimized (Mitra et al. 2004; Sharp et al. 2002). The ICP algorithm is fully automated and no points need to be pre-selected for the registration purpose. But its performance could be impacted by the

deformation existed in the point clouds (i.e., the appearances of building components are changing in different point clouds). On the contrary, the registration based on the equivalent targets is not influenced by the deformation between different point clouds. In this research, targets were present in the scene when laser scans were progressively performed during the renovation process. Hence, point clouds captured in this study are registered together using the equivalent target-based approach.

#### **4.1.3. Next-best-view planning problem**

The next-best-view (NBV) problem is a well-known problem in the computer vision domain. The NPV problem is to find minimum number of viewpoints, where a range sensor is placed, to scan all the surfaces of an object (Connolly 1985; Maver and Bajcsy 1993; Pito 1996). Various next-best-view (NPV) planning approaches have been developed in the computer vision domain to address the NPV problem. These approaches are usually composed of two parts: (a) a method to represent and determine the visibility of an object surface from different viewpoints, and (b) a viewpoint selection algorithm that optimize the coverage of sensors with smallest number of views (Banta et al. 2000; Maver and Bajcsy 1993; Pito 1999; Scott et al. 2003; Wong et al. 1999). Various approaches have been used to represent the viewing volume of an object from a given viewpoints. Examples of such approaches include occupancy grids (or called as voxel grid), octrees and triangle meshes (Tarabanis et al. 1995). One of the commonly used representation approach is the occupancy grid, which encodes an object surface using a voxel grid. In a voxel grid, each voxel can be labeled as occupied (i.e., the voxel is captured by the scanner from a given viewpoint) or unoccupied. By counting the number of occupied voxels, the

approach is capable of determining the coverage of a range image captured from a given location and orientation (Banta et al. 2000; Massios and Fisher 1998).

One of the research objectives in research question 3 is to develop an approach that selects a minimum number of point clouds captured at different times to provide more geometric information for 3D BIM update. It is similar to the NPV problem as they all target on finding a subset of point clouds or range images to cover all the target object surfaces. The difference is that the NPV problem selects range images different in spatial viewpoints whereas the approach developed in research question 3 select point clouds different in scanning time. Therefore, the NPV planning approaches provide a starting point for developing an approach to select and combine point clouds captured at different times. The visibility metrics (i.e., occupied/unoccupied) and the occupancy grids proposed in the NPV planning approaches can be also extended and used in the point cloud selection approach proposed in research question 3.

#### **4.2 Motivating Case Study: an Analysis of Completeness of Geometric Information in Progressively Captured Point Clouds**

For a detailed analysis of completeness of geometric information contained in progressive point clouds, I conducted a case study using a renovation project of a 100-year old university campus building. The renovated space was scanned from multiple locations to capture the interior of the research lab at six different times during the renovation process that went from May to August 2012. A pulsed time-of-flight (PTOF) scanner was used in the case study. The progressively captured point

clouds are used to analyze the information provided by combining point clouds progressively.

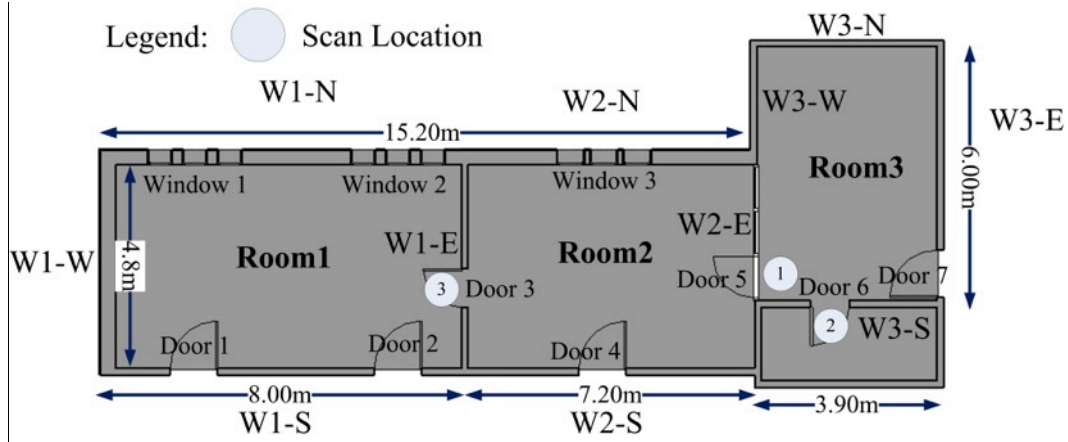


Figure 19. Project site layout and the scan locations

In this analysis, I targeted on the point clouds captured at six different times at location 3. These point clouds are referred to as P1, P2, P3, P4, P5 and P6. To evaluate the geometric information contained in the progressively captured point clouds, I selected the ductworks in room 2 as the testbed (as seen in Figure 20), and analyzed how much information each point cloud is able to provide to update these ductworks in a BIM. The reason to make this selection is that these ductworks were progressively installed during the renovation, and as a result point clouds captured at different times might provide different information regarding these ductworks. To update a ductwork segment manufactured in a rectangular shape, width, height and length of the segment need to be measured from the point cloud. The width, height and length of a ductwork segment are referred to as geometric properties of that ductwork segment. The analysis included 22 duct segments and their 66 geometric

properties (i.e., width, height and length) that were measured in all of the six point clouds.

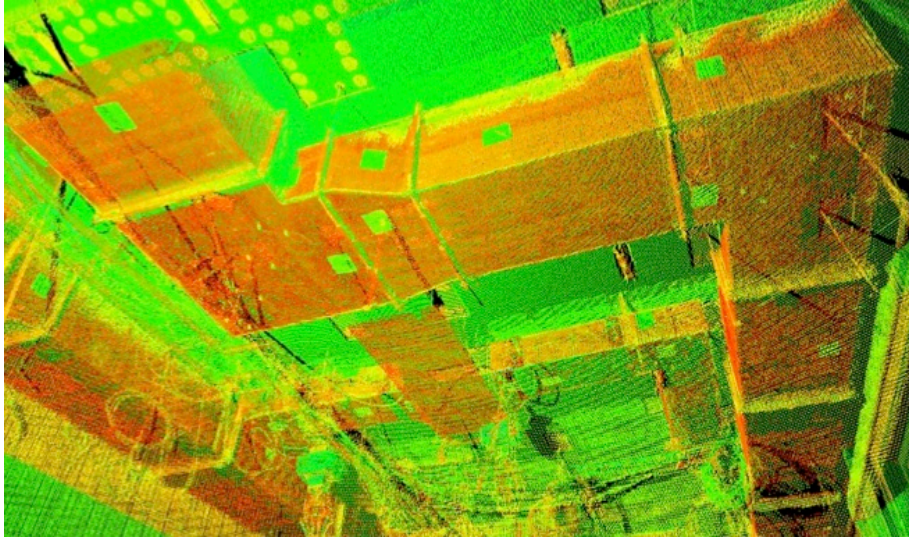


Figure 20. The testbed for the motivation case study: ductworks captured by the point cloud data

A geometric property can have three statuses in a point cloud:

- **Occluded:** Occlusions block the view of a geometric property, which prevent getting measurements from the point cloud for that property.
- **Not installed:** The target geometric property cannot be measured, as the corresponding component had not been installed when the point cloud was captured.
- **Measurable:** The target geometric property can be measured from the point cloud.

In order to evaluate the information added by the combination of point clouds captured at different times, the point clouds were progressively registered and analyzed for the status of the same 66 geometric properties shown in the registered point clouds. Targets were present in the scene when laser scans were progressively performed during the renovation process. A screenshot of targets presented in a point cloud is shown in Figure 21. The targets were used to register and combine point clouds captured at different times together. In total, five registered point cloud sets were generated, labeled as “P1+P2”, “P1 to P3”, “P1 to P4”, “P1 to P5”, and “P1 to P6”.



Figure 21. Targets shown in a point cloud

Figure 22 shows the status of geometric properties captured in the point clouds captured at a single point in time (i.e., P1, P2, etc.) as well as the progressively registered point clouds. In Figure 22, each row represents a ductwork segment in the testbed. The blue, gray and black boxes represent whether a corresponding geometric property is measurable, not installed or occluded in the point cloud, respectively.

Geometric Property	P1	P2	P3	P4	P5	P6	P1+P2	P1 to P3	P1 to P4	P1 to P5	P1 to P6
Height	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
Length	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
Width	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue
	Gray	Blue	Black	Blue	Black	Blue	Black	Blue	Blue	Blue	Blue

Occluded

Not installed

Measurable

Figure 22 Status of geometric properties captured by individual point clouds (P1, P2...P6) and the combination of them

As shown in Figure 22, none of the six progressively captured point clouds is capable of solely capturing all the geometric properties for all these segments. The screenshots for the six point clouds captured at different times are shown in Figure 23. Combing Figure 22 with Figure 23, it can be found that no ductwork segments were installed by the time P1 was captured, and thus all the 66 geometric properties are gray- hence have “not installed” status in P1. 42% of the geometric properties



were “not installed” in P2. By the time P3 was captured, all the ductworks in Room 2 were installed (i.e., no segments with not-installed status). Occlusions shown in the scene could prevent a point cloud from capturing a complete view of target components and their associated geometric properties. In P3, 11% of the geometric properties were “occluded”. 23% of the geometric properties captured by P4 were “occluded” and 14% of the geometric properties captured by P5 were “occluded”. By the time P6 was captured, ceiling tiles and lighting fixtures were installed. They occluded most of the ductworks from laser scans. As a result, the percentage of “occluded” geometric properties jumped to 67% in P6.

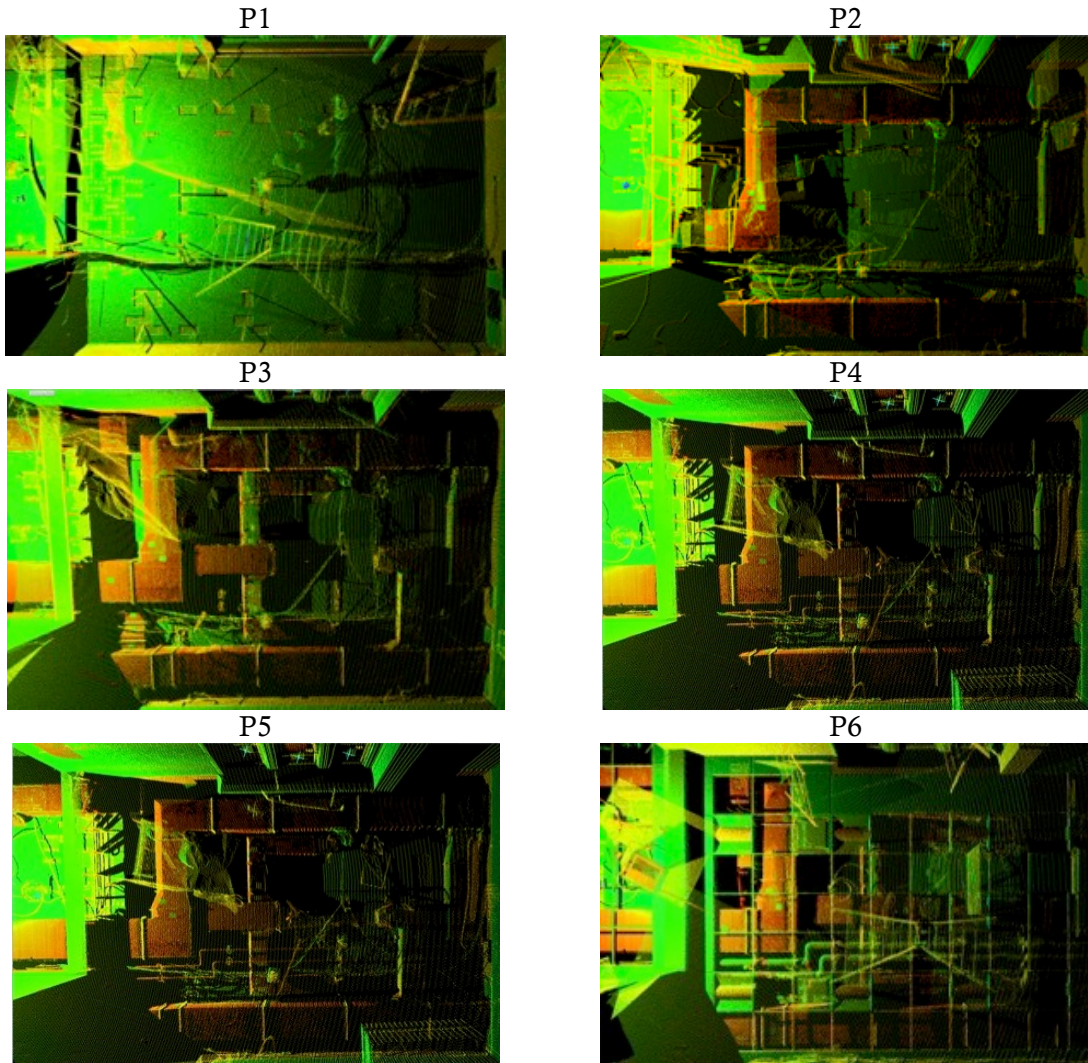


Figure 23. Screenshots of the point clouds captured at different times

As evident from the figure, it was possible to extract more geometric properties from the merged point clouds than the point clouds captured at a single point in time. In the combined point clouds “P1 to P3”, “P1 to P4”, “P1 to P5” and “P1 to P6”, 96% of the geometric properties were “measurable” and 4% of them were “occluded”. The occluded geometric properties in the combined point clouds (shown as black boxes in Figure 22) were either “occluded” or “not installed” in all of the six progressively captured point clouds. As seen in Figure 22, combining P1 to P3

provided the same number of geometric properties as compared to combining all six point clouds together. Adding P4, P5 and P6 into the registered point cloud did not increase the number of “measurable” geometric properties. It shows that combining more point clouds does not always mean that the registered point clouds would provide more geometric properties in relation to the target components.

In addition, as observed from Figure 22 and Figure 23, the occlusions are randomly distributed in the six point clouds. However, as renovation process progresses, it is likely that the probability of having the same occlusions occurring at exactly the same locations will decrease. Some of the geometric properties were “occluded” or “not installed” in some of the point clouds but they could be measured from the other point clouds. As a result, when multiple point clouds captured in different times are combined together, it is possible to retrieve geometric properties that were occluded or not installed in some of the point clouds. However, the growth of the geometric information provided by a combined point cloud is not proportional to the number of point clouds that are combined together. Adding more point clouds is likely to increase the size of the final data set though it might not always add more geometric information. Considering large file size of a point cloud, combining multiple point clouds together would generate a combined point cloud that requests a large storage space. For instance, six point clouds were collected at different points in time in the motivating case study, and the average size of a point cloud is around 320MB. Combining these six point clouds together generates a 2GB point cloud. It would be a challenge to store, process and edit the combined point cloud if more point clouds are combined together (de Haan 2009; Ravada et al. 2010; Song and

Feng 2009). Therefore, having a pre-selected set of point clouds could reduce the size of final point cloud data and also potentially improve the usage for the final point cloud data. In order to support the decision on which point clouds to be registered, there is a need to develop an approach to evaluate the information contained in each point cloud and assess the information gain when adding registering more point clouds. The next section provides the details of an approach developed for this purpose.

### **4.3. An Approach to Evaluate Information Contained in Progressively Captured Point Clouds and Select Point Clouds to Be Combined**

The developed approach is composed of two modules: (a) content assessment module, which quantifies the geometric information contained in each point cloud for updating/modeling target building components in a BIM, and (b) content improvement module, which calculates the information gained by adding new point cloud data to an initial baseline point cloud.

#### **4.3.1. Content assessment module for quantifying the information contained in a point cloud**

Figure 24 shows the flowchart of the content assessment module. The content assessment module takes two inputs: a set of point clouds captured at different points in time and a set of target components modeled in the BIM. For specific surfaces associated with the target components, this module applies a grid-based evaluation approach to quantify the geometric information provided by a point cloud. The grid-based approach overlays a grid on the surface of a target component and divides the

target surface into a finite number of cells. The grid-based point cloud evaluation approach can be divided into two steps: (a) grid construction, which overlays a grid onto the surfaces of target building components modeled in a BIM, and (b) point cloud geometric information assessment, which divides a point cloud using the same grid generated in Step 1 and evaluate how much information a point cloud can provide to model/update the targeted components in the BIM.

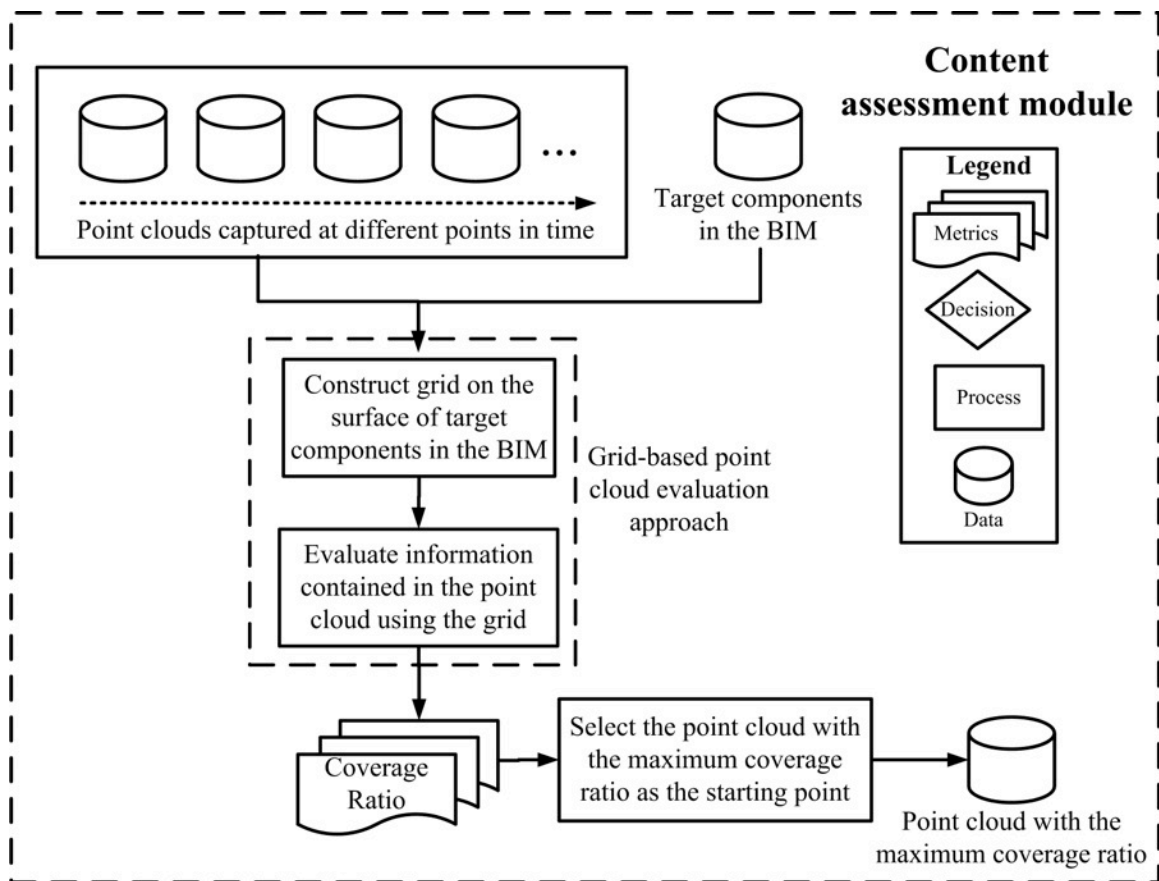


Figure 24. The flowchart of content assessment module

**Step 1: Grid construction:**

This step overlay a grid onto the surface of targeted components modeled in a BIM.

The grid divides the surface of target components into a finite number of cells. The size of a cell is determined according to the size of the surfaces and the level of detail required by the assessment conducted in the next step. Figure 25 (a) shows a grid that is generated over the surface of ductworks modeled in a BIM, which divides the surface into a number of cells.

### ***Step 2: Point cloud geometric information assessment***

This step analyzes a point cloud using the same grid generated in Step 1. It first aligns the point cloud and the BIM into the same coordinate system using the ICP algorithm, so that the grid can also be aligned with the point cloud. After the grid is overlaid with the point cloud, each cell in the grid contains a set of points. When a cell does not contain any points, the geometric information is missing in this cell due to following reasons: (a) the part of the building surface is occluded, (b) the part of the building surface is not installed, and (c) due to the reflectivity of the building surface, no laser beam were reflected and returned to the scanner. Given that, in this step, a coverage image is created, which uses color-coding to indicate whether, a cell contains any information related to target surfaces. Figure 25 (b) shows a point cloud and the coverage image created based on it. The coverage image is used to quantify the information contained in each grid. For this purpose, a metric has been defined and used, namely as coverage ratio. The coverage ratio refers to the ratio of a surface that is clearly presented in a point cloud without any occlusions and is calculated for each surface fitted in a point cloud, using Equation 9, as shown below.

Equation 9:

$$\text{Coverage ratio} = \frac{\text{number of cells occupied on the grid overlaid on point cloud on a given surface}}{\text{total number of cells available on the grid on a given surface}}$$

A low coverage ratio means that a given point cloud cannot provide enough geometric information to model/update target surfaces and suggests that additional point clouds need to be added in order to cover the missing portion of the components for accurate measurements.

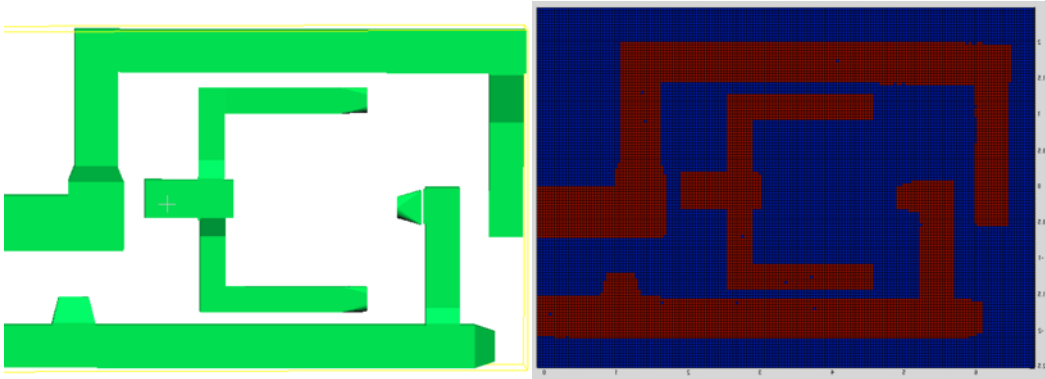


Figure 25. Ductworks modeled in a BIM and the grid overlaid on the surfaces of ductworks (the red box represents each cell in the grid)

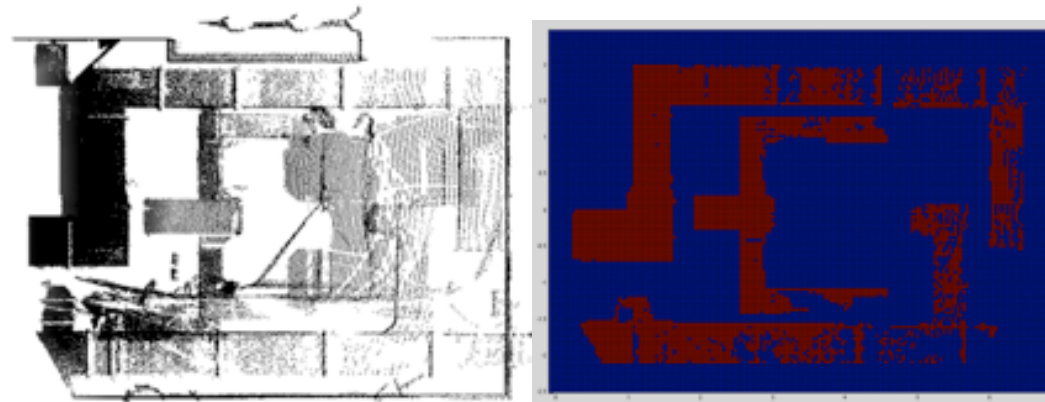


Figure 25(b) Ductworks captured by a point cloud and the coverage image, where the red cells indicate that the cells contain the points associated with the target surface.

The measure of the coverage ratio also depends on the selection of grid resolution. The coverage ratio is an estimate of how much geometric information provided by a point cloud. It is important to balance the grid size. When the grid size is too large, the coverage ratio might only provide a rough estimate for the geometric information contained in a point cloud. On the other hand, when the grid size is too small, the measure of the coverage ratio is sensitive to the noises presented in a point cloud and takes more time to compute.

To demonstrate the impacts of grid resolution, I conducted a sensitivity analysis using the example shown in Figure 25. I incrementally changed the grid size from 0.01m to 1m, and calculated the coverage ratio for the example shown in Figure 25. In addition, I also recorded the calculation time for each of the grid size and calculated the average number of points belonging to a point cloud within a cell. The sensitivity analysis is conducted using a computer with 2.4 GHz CPU and 8GB memory. Figure 26 (a) shows the values of calculation time and average number of points within a cell when the grid size is changing from 0 to 1. As seen in Figure 26(a), the calculation time increases while the grid size decreases. When the grid size decreases from 0.1m to 0.01m, the calculation time increases significantly. When the grid size is equal to 0.01m, the calculation time is about 610 seconds, which is 100 times greater than the calculation time at grid size equal to 0.1m. In the meantime, the average number of points per cell decreases along with the decrease of the grid size. Having too many points within a cell could indicate that the grid based evaluation approach can only provide a rough estimate for the geometric information contained in a point cloud. Decreasing the grid size would result in a more accurate



estimation for the geometric information contained in a point cloud, but it would also increase the calculation time significantly. A good selection of the grid size needs to balance the calculation time and the average number of points per cell. Figure 26 (b) shows the relationships between the grid size and the coverage ratio. Shown in Figure 26 (b), dividing a building surface with a large grid size could lead to an overestimate for the information contained in a point cloud. Considering the calculation time and the estimation accuracy, in this thesis, I select the grid size as 0.04m, based on which the calculation time is 40 seconds and the average number of points per cell is 8.

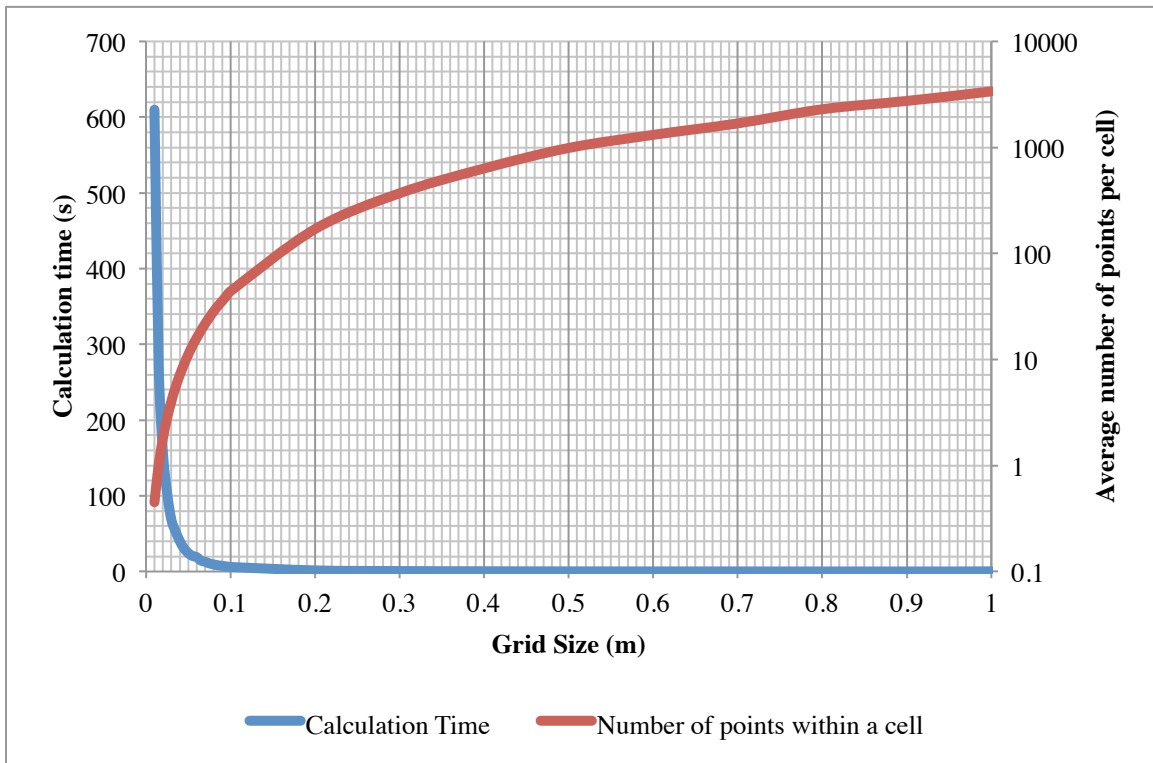


Figure 26 (a). Processing time and point cloud density vs. grid size

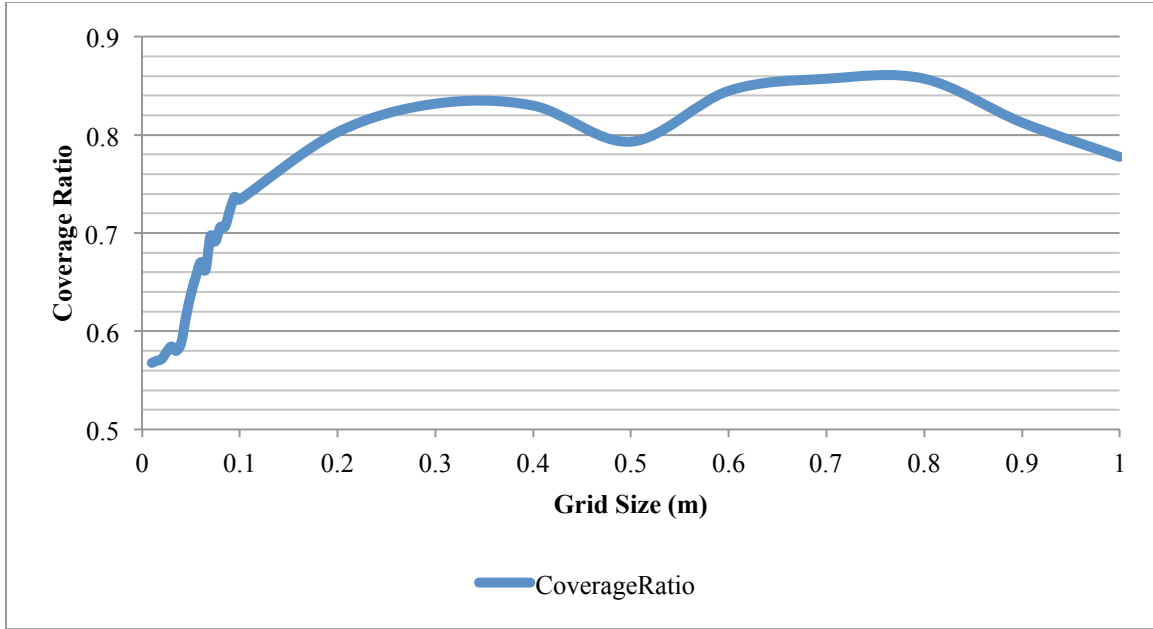


Figure 26 (b) Coverage ratio vs. grid size

The content assessment module calculated the coverage ratio for all the inputted point clouds. The point cloud with the maximum coverage ratio is the one that is capable of providing the most complete geometric information for the specific surfaces of the target components and is marked as the baseline point cloud. To better demonstrate how the content assessment module works, I used four point clouds captured at four different times as an example, and selected the ductworks shown in the point clouds as the target building components. Figure 27 shows four point clouds captured at different points in time and their coverage images for ductworks. According to the coverage ratios, P3 has the highest coverage ratio as 67.6 % for the surfaces of ductworks and hence should be used as the baseline point cloud. The remaining point clouds need to be evaluated in order to determine which point clouds should be combined with the baseline set. This ties to the second module, which is the content improvement module.

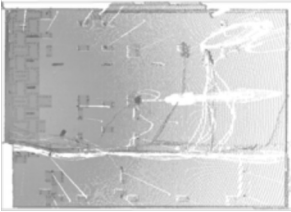

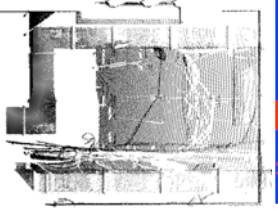

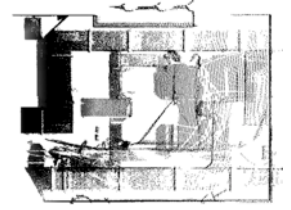
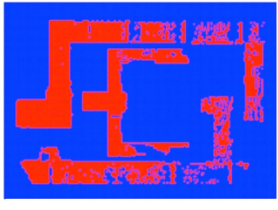

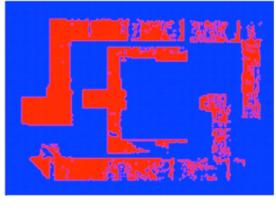
Point cloud	Coverage Image and Coverage Ratio	Point cloud	Coverage Image and Coverage Ratio
P1	P1 – 0%	P2	P2 – 48.1%
			
P3	P3 – 67.6%	P4	P4 – 65.4%
			

Figure 27 Point cloud data captured at different points in time and their occlusion images for duct segments

#### 4.3.2. Content improvement module: Assessing the information gain by combining additional point clouds

This module takes the baseline point cloud identified by the content assessment model as the input, and determines which remaining point clouds should be combined with the baseline point cloud. The flowchart of this process is shown in Figure 28.

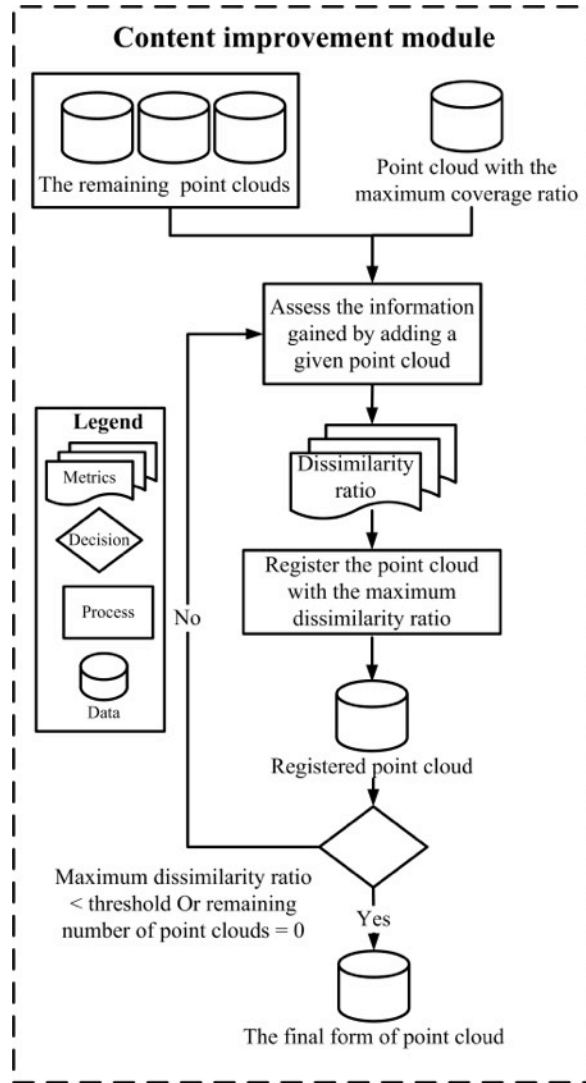


Figure 28. Flowchart of the content improvement module

Assuming that there are a set of point clouds ( $P_1, P_2, P_3 \dots P_n$ ), and  $P_1$  is selected as the baseline point cloud since it has the highest coverage ratio. This module first compares each of the remaining point clouds to the baseline point cloud, and assesses how much information can be gained by adding each of the remaining point clouds into  $P_1$  separately. When comparing  $P_i$  ( $1 < i \leq n$ ) with the baseline point cloud -  $P_1$ , the module utilizes the same grid generated in the content assessment

module and overlays the coverage images of  $P_1$  and  $P_i$  together. The content improvement module then exams the coverage images and identifies the cells that are occupied by  $P_i$ , but not  $P_1$ . These cells are referring to the target surface areas that were shown in  $P_i$ , but not captured or occluded by  $P_1$ . To quantify this additional geometric information, a metric, named as dissimilarity ratio, has been introduced. The dissimilarity ratio represents the percentage of information gained by adding the new point cloud into the baseline point cloud, and is calculated using Equation 10.

Equation 10:

$$\text{Dissimilarity ratio } (P_i, P_j) = \frac{\text{number of cells occupied } P_i \text{ but not } P_j \text{ on a given surface}}{\text{total number of cells available on the grid on a given surface}}$$

Figure 29 shows the result of the assessment for the same set of ductworks used as an example in the previous module. The additional information generated by adding P2 into the P1 is shown in the third column.

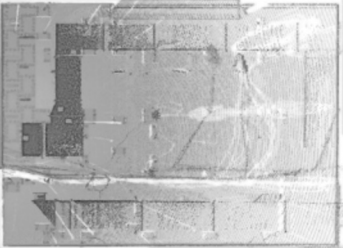
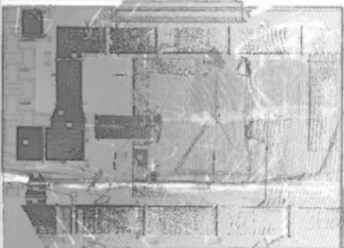
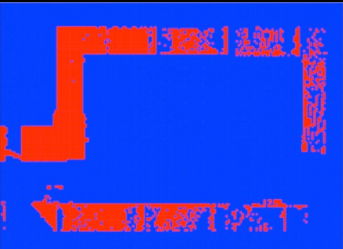
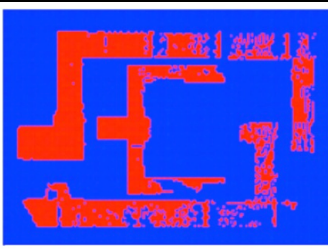
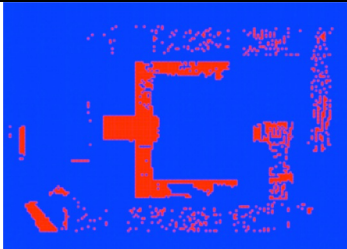
P1 – baseline point cloud	P2 – added point cloud	A coverage image shows cells occupied by P2 but not P1. Dissimilarity ratio = 25.96%.
		
P1 – coverage image	P2 – coverage image	
		

Figure 29. Dissimilarity ratio by adding P2 to P1

This assessment is repeated until all the point clouds at hand are compared to the baseline point cloud, as shown in Figure 28. The point cloud with the maximum dissimilarity ratio is defined as the one that has the highest information to add to the baseline point cloud. This point cloud is then registered with the baseline point cloud and considered as the new baseline point cloud for the next iteration. The model iteratively combines the remaining point clouds to the baseline point cloud till either all the point clouds have been registered together or the maximum dissimilarity ratio is smaller than a threshold, whichever occurs earlier. Users determine the value of the threshold. If the dissimilarity ratio of a point cloud is smaller than the threshold, it is not worth adding this point cloud to the baseline point cloud set since the combination will increase the file size without bringing enough additional geometric information for the model update/construction. The output of the content improvement module is a point cloud that is composed by the selected point clouds.

#### **4.4. Validation**

I validated the performance of the developed approach in terms of whether the approach is able to identify the right combination of point clouds in order to get a more complete set of geometric information with less file size. The performance of the developed approach has been compared to two other baseline approaches (i.e., approach 1 and approach 2). In approach 1, all the point clouds are registered together. In approach 2, the point clouds are progressively registered, in the order that they were acquired, until the dissimilarity ratio reaches to a preset threshold, which is the same threshold used in the developed approach. The approach developed in this research (i.e., approach 3), however, first uses the coverage ratios to

select the baseline point cloud and then determines which point clouds should be combined with the baseline point cloud using the dissimilarity ratios. The combined point clouds generated by the three approaches were compared. Three metrics were used in the validation, which are: (a) coverage ratio, (b) number of geometric properties that can be measured in a baseline point cloud, and (c) size of the final point cloud data.

The validation data includes five point clouds progressively captured during the renovation of a research lab, labeled as ( $P_1$ ,  $P_2$ ,  $P_3$ ,  $P_4$ , and  $P_5$ ). I focused on four different building components, which are (1) surface of ductworks installed in room 2, (2) interior wall surface Wall2-W, (3) interior wall surface Wall2-S, and (4) the surface of ceiling in the room. A set of screen shots of the components used in the validation is shown in Figure 30. These four building components are selected to ensure that the approach developed in this thesis can be generalized to different types of building components.

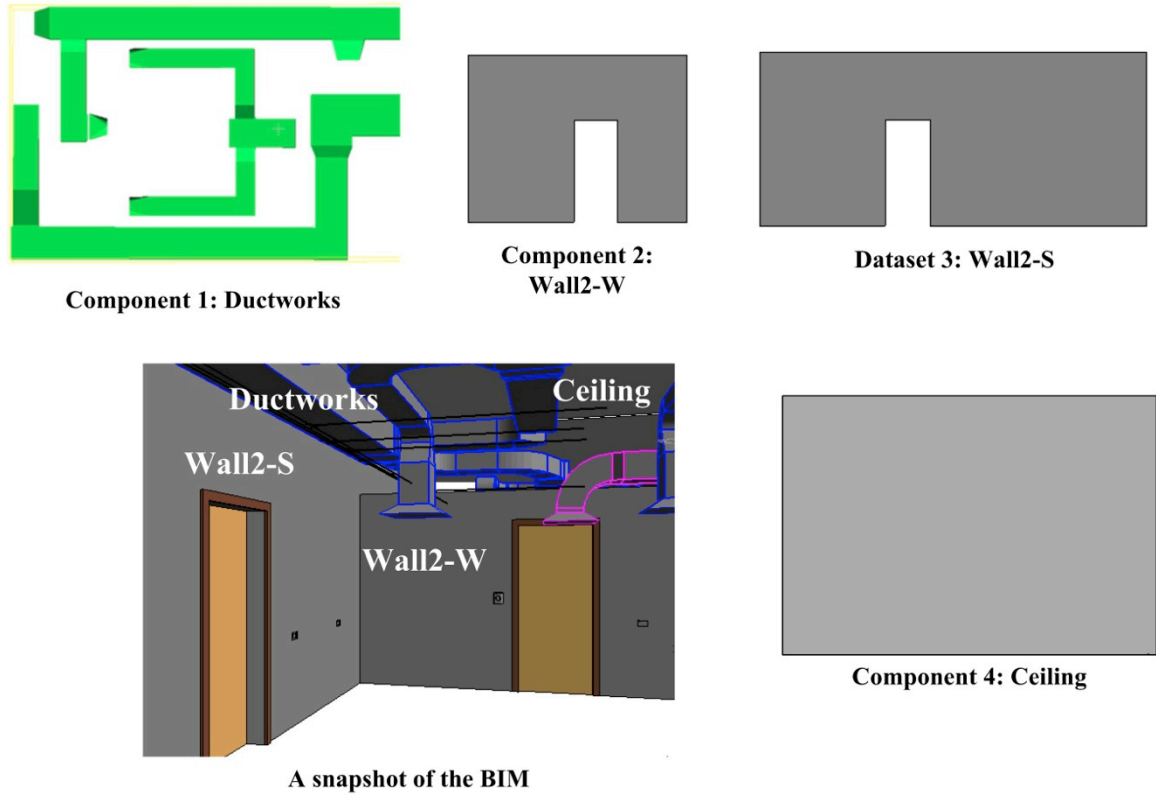


Figure 30. The four different building components targeted by the validation

Each approach has been implemented in Matlab and tested using the validation data. The resulting selections of point clouds to be registered following the three approaches are shown in Table 8. For the four building components, approach 3, which is the developed approach, selects the least number of point clouds to be combined.

Table 8 The resulting subset of point clouds to be registered following the three approaches

Approach	Component 1: Ductworks	Component 2: Wall2-W	Component 3: Wall2-S	Component 4: Ceiling
1	$P_1$ to $P_5$	$P_1$ to $P_5$	$P_1$ to $P_5$	$P_1$ to $P_5$
2	$P_1$ to $P_3$	$P_1$ to $P_4$	$P_1$ to $P_3$	$P_1$ to $P_4$
3	$P_3+P_4$	$P_4+P_1+P_2$	$P_1+P_3$	$P_1+P_2+P_5$



The comparison of the metrics used for performance evaluation of these approaches is provided in Table 9. The coverage ratio increases when additional point clouds are added into the baseline point cloud and at the same time the rate of growth of the coverage ratio decreases. The geometric properties remain the same in the combined point clouds generated by the three approaches. The developed approach (approach 3) achieved the smallest file size in all of the four validation datasets. For the ductworks, the coverage ratio of the registered point clouds generated by approach 1 is 2.3% higher than the registered point clouds generated by approach 2. However, the size of the final point cloud generated by approach 1 is 68.5% higher than approach 2. It shows that when registering all of the point clouds together without any pre-selection, the increase of file size might overweight the improvement in relation to the completeness of the geometric information. For the interior surface of Wall 2-W, the combined point cloud gained by approach 3 is 309.3MB and has 90.8% coverage ratio. Approach 2 added one more point cloud into the registered point cloud, which increases the coverage ratio by 1.5%, but also increases the file size by 32.3%. Approach 1 combined all the point clouds together. This approach increases the coverage ratio by 2% and file size by 66.2%, as compared to the third approach. I also counted the geometric properties that can be measured from the point clouds. The geometric properties for the four different types of building components include length and width of the building surfaces. The measurable properties remain the same for the four building components and the three approaches. The result showed that using approach 3 it is possible to reduce the number of point clouds that need to be registered so as to reduce the file size. As

shown in Table 9, as compared to the other two baseline approaches, I can retrieve more geometric information (i.e., higher coverage ratio) using the least number of point clouds (i.e., less file size), which validates the effectiveness of our approach.

Table 9. Comparison of the developed approach (approach 3) with respect to other baseline approaches

Approach	Component 1: Ductworks			Component 2: Wall2-W			Component 3: Wall2-S			Component 4: Ceiling		
	C	M	S	C	M	S	C	M	S	C	M	S
1	75.7 %	41	514	92.5 %	4	514	94.0 %	4	514	96.5 %	4	514
2	74.0 %	41	304.9	92.1 %	4	409.3	93.9 %	4	304.9	96.5 %	4	409.3
3*	75.0 %	41	204.4	90.8 %	4	309.3	93.7 %	4	199.9	96.5 %	4	309
C: Coverage ratio				M: measurable geometric properties				S: file size (MB)				

In the validation calculations, the threshold was set as 1% for all the datasets. The value of threshold could also impact the performance of the developed approach. When threshold is set as 0, approach 1, 2, and 3 gave the same result, since they all combines all of the point clouds together. When the threshold gets increased, it means that the size of the registered point cloud has gained more weight. As a result, less number of point clouds will be combined together. The impacts of the threshold on the developed approach and the selection of the optimal threshold will be addressed in the future work.

## 4.5 Conclusion

Progressively captured and registered point clouds provide opportunities to capture a more complete view of the building components over time. A unique challenge of using registered point clouds is that registering point clouds that contains repetitive

information might increase the difficulty of storing and processing registered point clouds due to the file size. Instead of registering all the point clouds together, only combining the point clouds that contain less repetitive geometric information could effectively reduce the file size of the final dataset and increase the usability of the data. One of the contributions of this thesis is an approach that evaluates the information contained in point clouds and supports the decision on which point clouds should be combined.

## CHAPTER 5 CONCLUSION AND FUTURE WORK

This section concludes this thesis with discussions of the contribution, practical implications and future research directions.

### 5.1 Contributions

This research has four main contributions: (a) Identification of a general list of features that can be applied to recognize the correspondences between a point cloud and a BIM, (b) Formalization of a set of matching approaches that reason with one or more than one type of features to identify the matches between a point cloud and a BIM, (c) Identifying the relationships between features and discrepancies and evaluating how different types of features perform under different discrepancies, and (d) Formalization of an approach to support the decision on selecting point clouds for combination.

#### **(1) Identification of a general list of features that can be applied to identify the correspondences between a point cloud and a BIM**

Matching a point cloud to a BIM is one of the most important steps for the process of updating BIM. By matching a point cloud to a BIM, the accurate geometric information captured by the point cloud can be linked to the semantic and geometric information contained in the BIM. The linkages between the two data sets allow users to identify the differences between the design information represented in an as-designed BIM and the as-built site condition captured by point clouds so as to update the as-designed model accordingly. Matches between a point cloud and a BIM can

be identified based on the features associated with these two data sets. Chapter 2 of this thesis provides a general list of features that can be applied to identify the correspondences between a point cloud and a BIM. Based on the previous BIM and point cloud related research studies, I identified 21 features associated with BIMs and 29 features of point clouds. In the feature list, features are extracted at different levels of detail (e.g., point to entity, then to global) from point clouds and BIMs. Hence, these features can serve as a bridge to identify the correspondences between point clouds and BIMs even though these two data sets are available in different levels of detail (point clouds are at the point level and BIM are at the object level). This list of features could serve as the basis for developing matching approaches that utilize/combine different feature to map point clouds to BIMs.

**(2) Formalization of a set of matching approaches that reason with one or more than one type of features to identify the matches between a point cloud and a BIM**

Building on the features identified in Chapter 2, I developed four matching approaches, which are (a) 2D overlap area matching approach, (b) spatial relationship based graph-matching approach, (c) distribution-based 2D shape matching approach, and (d) distribution-based 3D shape matching approach. Each matching approach is built on one specific type of features associated with point clouds and BIMs.

Based on the experimental analysis conducted in Chapter 3, none of these four matching approaches are capable of identifying all the correct matches between a point cloud and a BIM. Each matching approaches is robust to certain types of

discrepancy, but is sensitive to the others. In order to improve the matching result and reduce the impacts of discrepancies on the matching process, I formalized two combination approaches that combine multiple types of features together. The combination approaches are (a) re-aligning point cloud and BIM based on 3D shape distribution to reduce the impacts of location discrepancies on the area approach and, (b) filtering the matching candidates generated by the 2D overlap area matching approach using 3D shape distribution feature (called as 3D filter in the following paragraphs).

### **(3) Identifying the relationships between features and discrepancies and evaluating how different types of features perform under different discrepancies**

The performance of a matching approach is impacted by various factors, such as discrepancies existing between a point cloud and a BIM, the types of building components, the complexity of the site environment and the features used in the matching approach. The experimental analysis conducted in Chapter 3 provides a general knowledge on how different types of features perform when they are used to match point cloud segments to BIM components under various discrepancy conditions. The knowledge on how different types of features performance under different types of discrepancies could be used to guide the future research studies to understand the behavior of different features towards to different types of discrepancies, and develop more advanced matching approaches that utilize or combine different features to eliminate the negative impacts of discrepancies on matching results.

#### **(4) Formalization of an approach to support the decision on selecting point clouds for combination**

By combining point clouds progressively captured during the construction process, it is possible to get a more complete set of geometric information for the BIM update. However, the large file size of the combined point cloud makes it hard to process and store. Chapter 4 introduces an approach that is developed to pre-select point clouds collected at different points in time based on unique data sets that each point cloud brings in order to ensure completeness of the geometric information while reducing the file size resulted by combining point clouds with repetitive information. The approach is composed of two modules, which are: (a) content assessment module, it quantifies the geometric information contained in each point cloud for updating/modeling target building components in a BIM; and (b) content improvement module, it calculates the information gained by adding new point cloud data to an initial baseline point cloud. Based on the validation given in Chapter 4, the developed approach is effective in identifying the right combination of point clouds in order to get a more complete set of geometric information.

## **5.2 Practical Implication**

This research can possibly have a number of practical implications for information handover, construction management and facility operation and maintenance.

### **(1) Supporting the information handover by delivering a complete and accurate as-built BIM**

As a digital representation that captures and exchanges building information between different stakeholders, building information models (BIMs) can be used as an information repository to store and deliver as-built information. The framework developed in this research facilitates the process of updating an as-designed BIM into an as-built BIM. The updated as-built BIM can be delivered to owners or facility management teams as a part of information handover document.

### **(2) Providing foundation for developing advanced matching algorithm to match point clouds to BIMs**

Currently, there is a limited understanding of what features of point clouds and BIMs can be applied to match these two data sets together, let alone how the features would perform under different discrepancy conditions. In this research, I evaluated the performance of different features in terms of how they are able to identify the correct matches between point cloud segments and BIM components when different types of discrepancies exist. The result gained from this evaluation provides the foundation for developing more advanced matching approaches that



utilize or combine different features to eliminate the negative impacts of discrepancies on matching results.

### **(3) Supporting the future usage of progressive point clouds**

This research validates the value of the using progressive point clouds for the BIM update, which promotes the idea of progressively scanning a job site in order to capture more complete geometric information. However, one challenge that prevents progressive point clouds to be used in construction projects is that it is impossible to combine all the point clouds together considering the large file size of each point cloud. Without an approach to evaluate the information provided by point clouds captured at different times, construction professionals need to manually exam each point cloud and determine which one to use. The approach developed in this research allows construction professionals to rapidly evaluate the information contained in progressive point clouds and selectively combine progressive point clouds to provide a complete set of geometric information with file size. As a result, the developed approach makes the future usage of progressive point clouds possible.

### **5.3 Future Research Directions**

The results of this research point to several directions for future work, which are summarized as follows:

#### **(1) Extending the scope of the research and targeting on different types of building components**

First, the matching experiment conducted in this research mainly targets on the mechanical components installed in a building. It would be interesting to look at how the features performance for the other types of building components, such as walls, windows, doors and so on, and identify what features might be most applicable for a specific type of building components. Second, it would be worthwhile to further extend the feature combination approaches in terms of: (a) combining more features together, and (b) adjusting the way that how different features are combined together (e.g., identify matches using 3D shape feature first, and filter the matches based on overlap area space). In addition to 2D overlap area and 3D matching approaches, I can also include spatial relationships to further improve the matching results in the future work.

#### **(2) Testing the approaches developed in this research in a large-scale construction project with more realistic experimental settings**

In this research, I used a 4-month renovation project as the testbed to test and validate the matching approaches and point clouds combination approach developed in this research. A certain assumptions and simplifications have been made in the

experimental settings. For instance, I assumed that the point cloud inputted into the matching approaches are segmented and noises and temporary components are removed from the point cloud. It would be interesting to test the performance of the approaches developed in this research in a large-scale construction project with more realistic experimental settings and evaluate how the approaches would response in these situations.

#### **(5) Automatically selecting features using feature selection technique**

In this research, I identified a general list of features associated with point clouds and BIMs and developed a categorization to classify these features. I then selected features from each category and evaluate their performances. Future research studies could apply feature selection techniques to evaluate the features in an automatic fashion and remove the irrelevant and redundant features from the mapping process. The objective of selecting features is not to get a minimum set of features, but to acquire a better understanding on what features are more important and how features are correlated among each other.

#### **(6) Automatically accomplishing the update process to remove discrepancies between a point cloud and a BIM**

As discussed in Chapter 1.3 vision, this research mainly focuses on leveraging the progressive point clouds and mapping point clouds to BIMs. The actual update of an as-designed BIM based on the mapping results is out of the scope and is now up to modelers to accomplish. The future research studies could formulize a mechanism

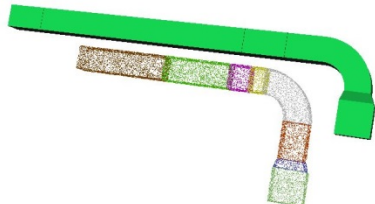
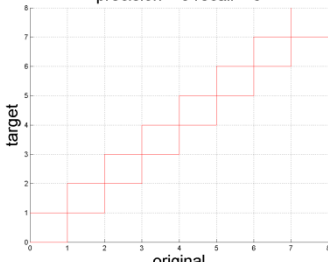
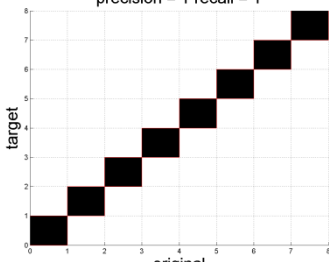
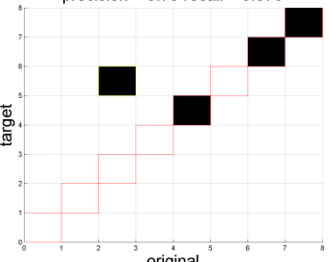
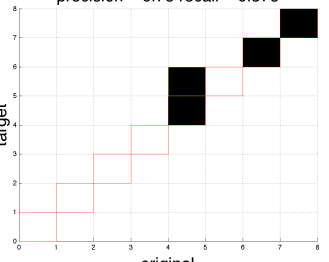
that automatically update the BM based on the matches between the BIM and the point cloud.

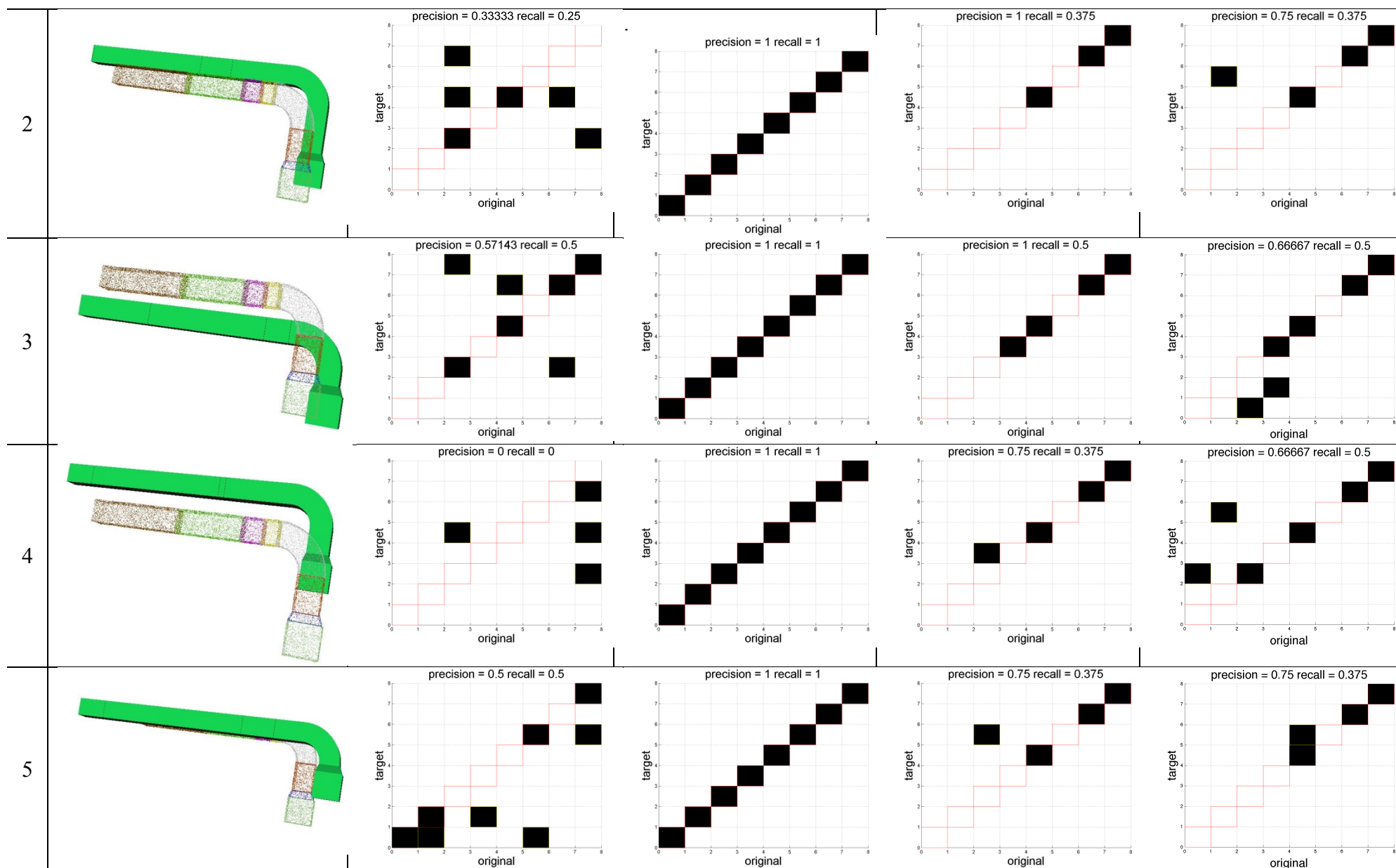
#### **(5) Developing an information repository to integrate BIM and point cloud together**

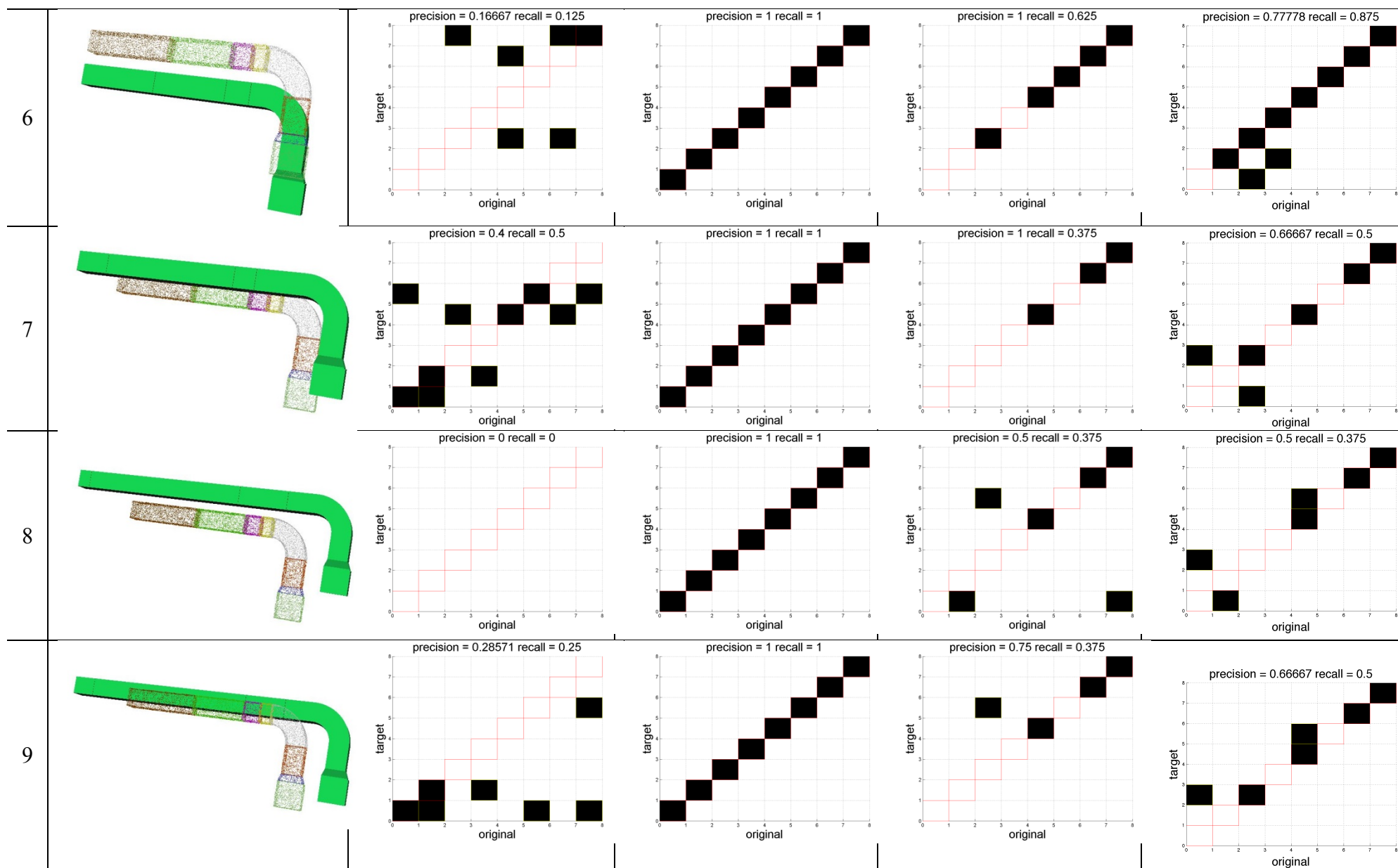
In the current practice, it relies on the manual approach to update a BIM using the 3D authoring tools (e.g., Revit, Sketchup, etc.). This manual practice is time-consuming and error prone. Therefore, instead of manually updating the as-designed BIM, the future research studies could target on using the point cloud as the baseline model and fuse the semantic information contained in the BIM to the point cloud. In this case, the effects spent in updating the model can be saved, and the point cloud could be enriched by the semantic information contained in the BIM.

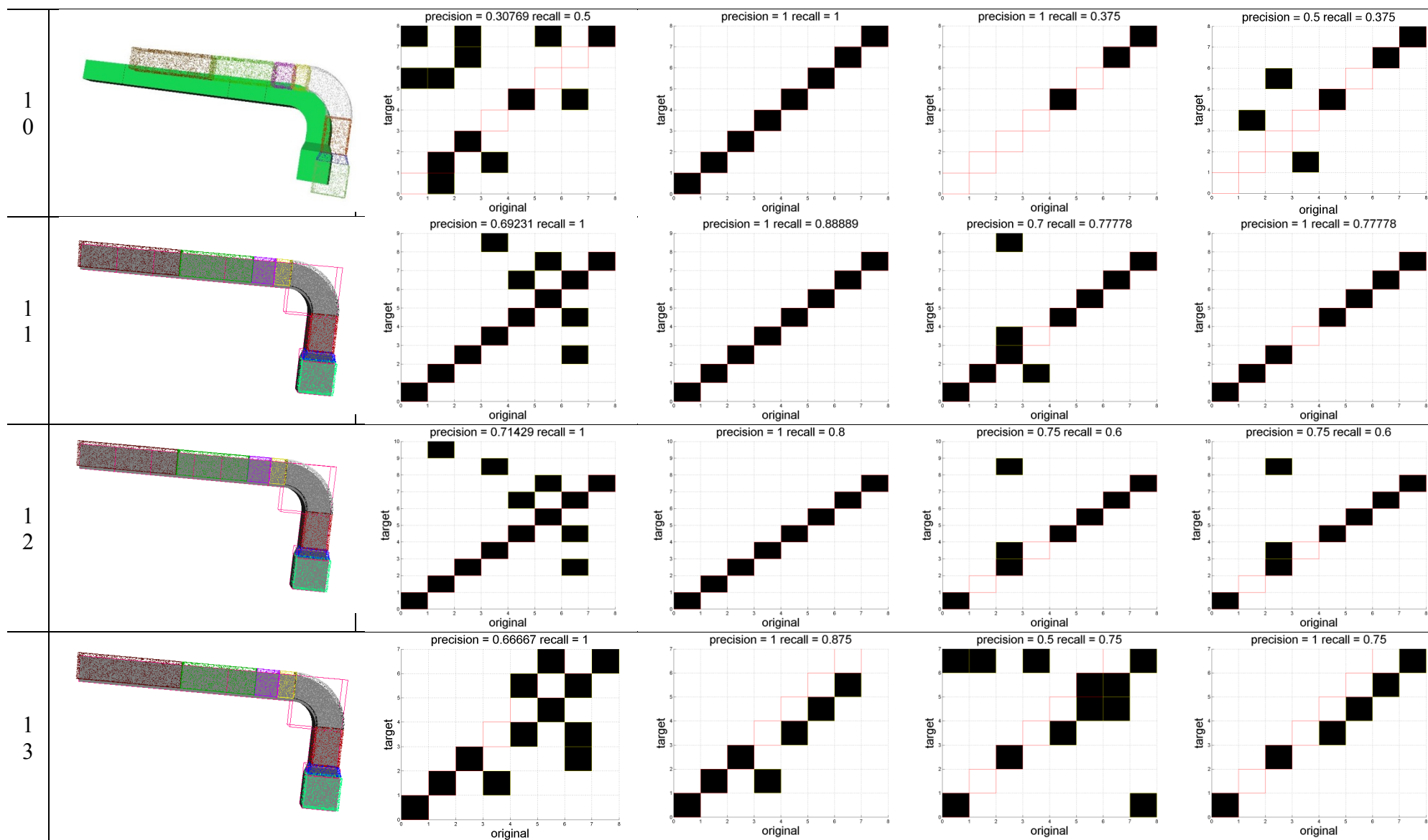
## Appendix A: Matching Results for the Four Matching Approaches

This appendix shows the experiment results of the four matching approaches on the experiment described in Chapter 3. The testbed used in this experiment is a research lab that has gone through a renovation process including the installation of a new mechanical system. In this experiment, I mainly focused on the ductworks that were newly installed in the research lab during the renovation process. The selected ductworks installed in the renovated space is used as the baseline model and different types of discrepancies are introduced to the baseline model to generate different versions of the model with deviations different than what is captured in the point cloud. Using the simulation algorithm, I generated 39 models, which covered different types of discrepancies, including shape, dimension, location, and composition discrepancies as compared to the baseline model. In total, the testbed included 40 pairs of point cloud (the point cloud remain the same) and the corresponding BIM (each pair is called as one scenario) for the experiments. Among the 40 scenarios, 30% of them have shape discrepancies, 65% have dimension discrepancies, 75% have location discrepancies and 10% of them have composition discrepancies.

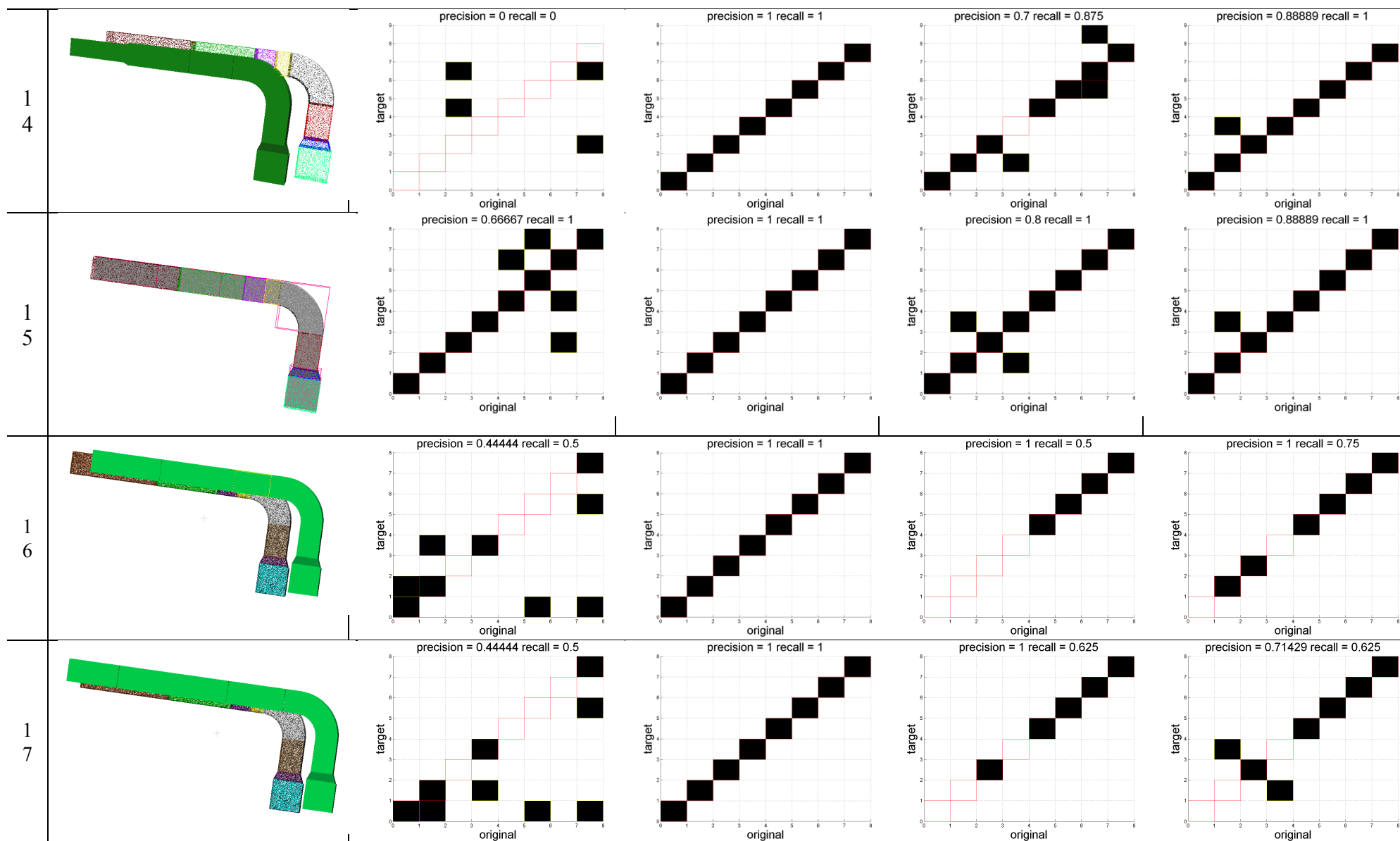
#	Scenario	2D overlap precision = 0 recall = 0	Topology-based precision = 1 recall = 1	2D shape precision = 0.75 recall = 0.375	3D shape precision = 0.75 recall = 0.375
1					

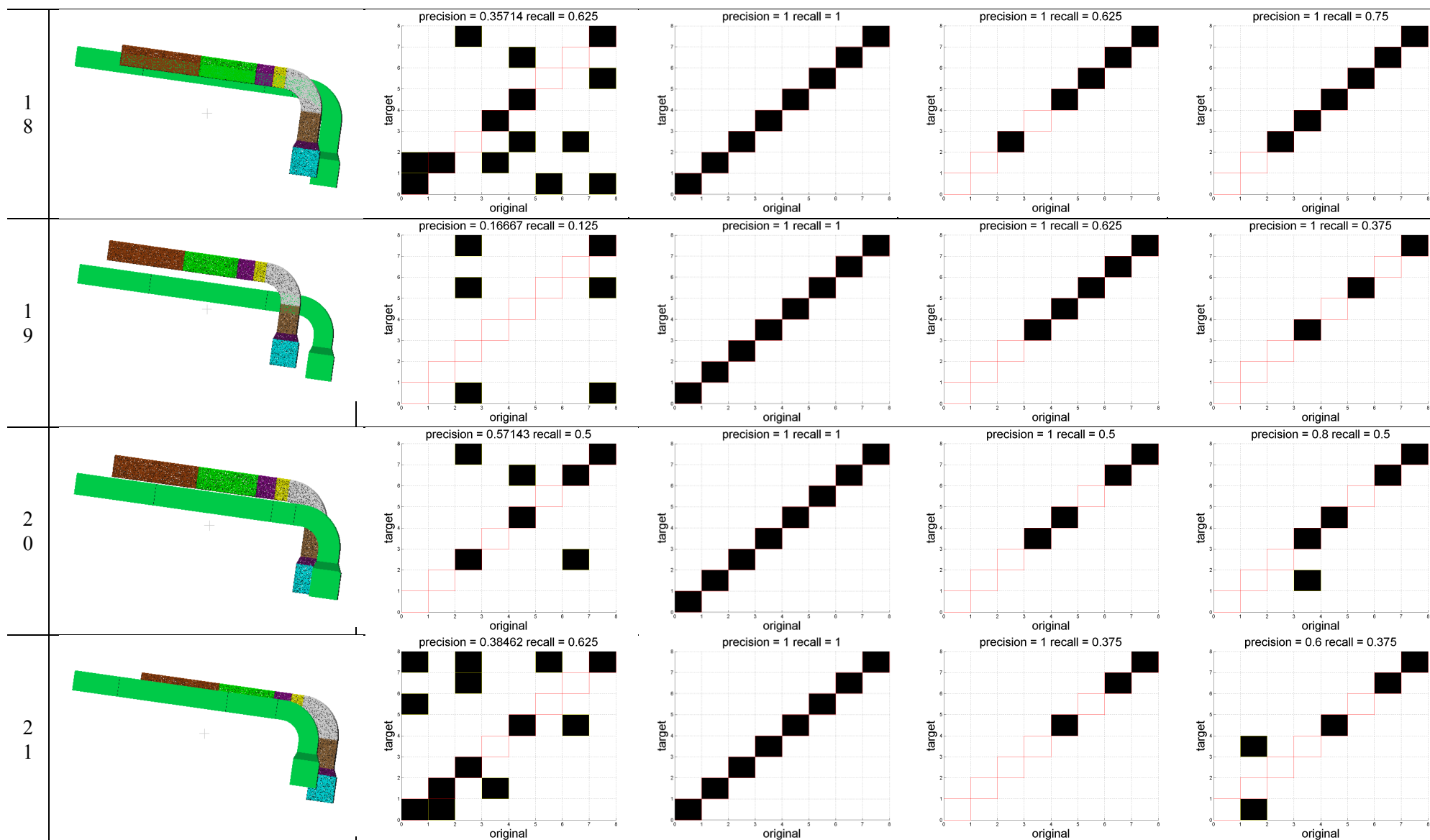


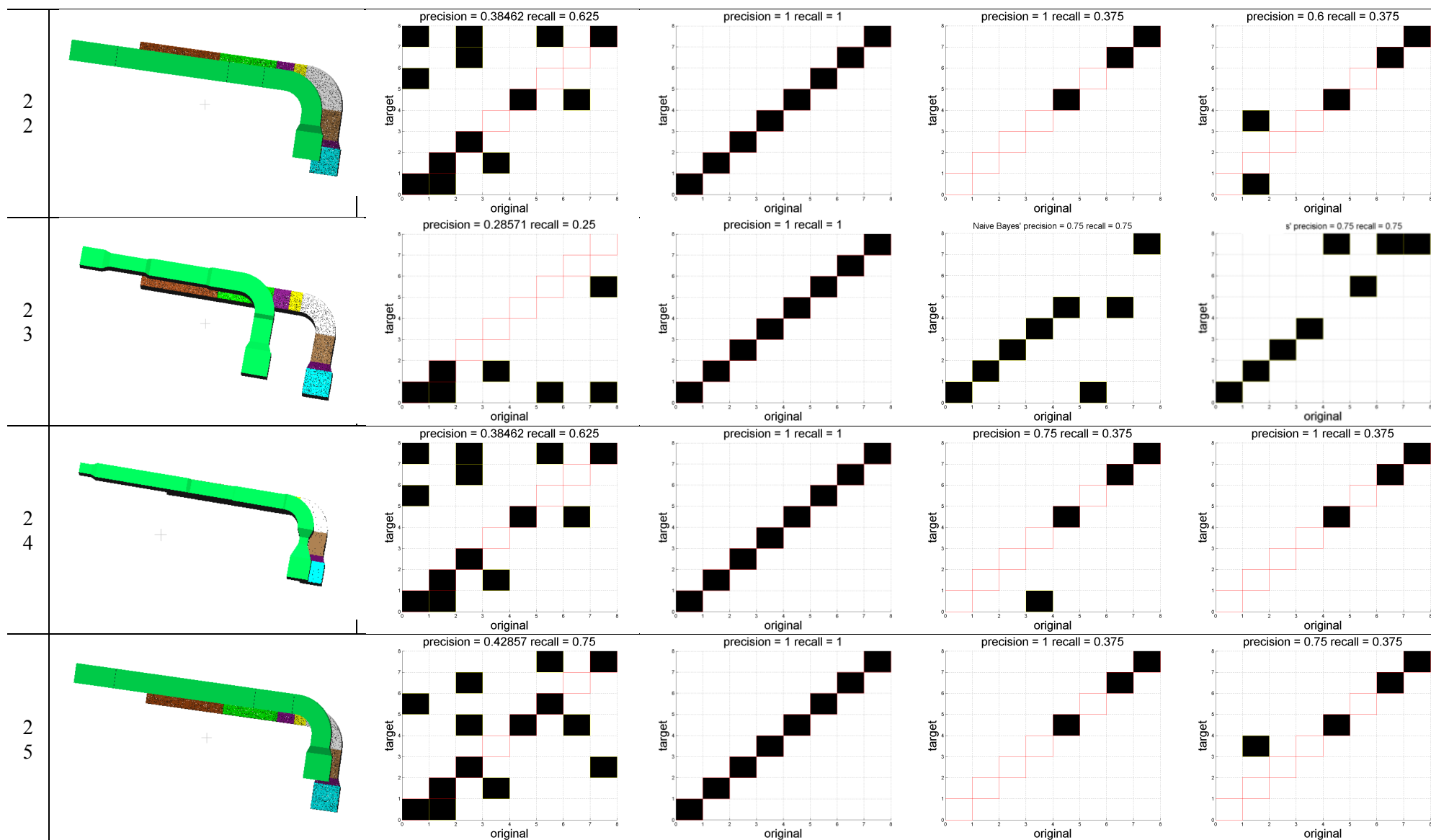


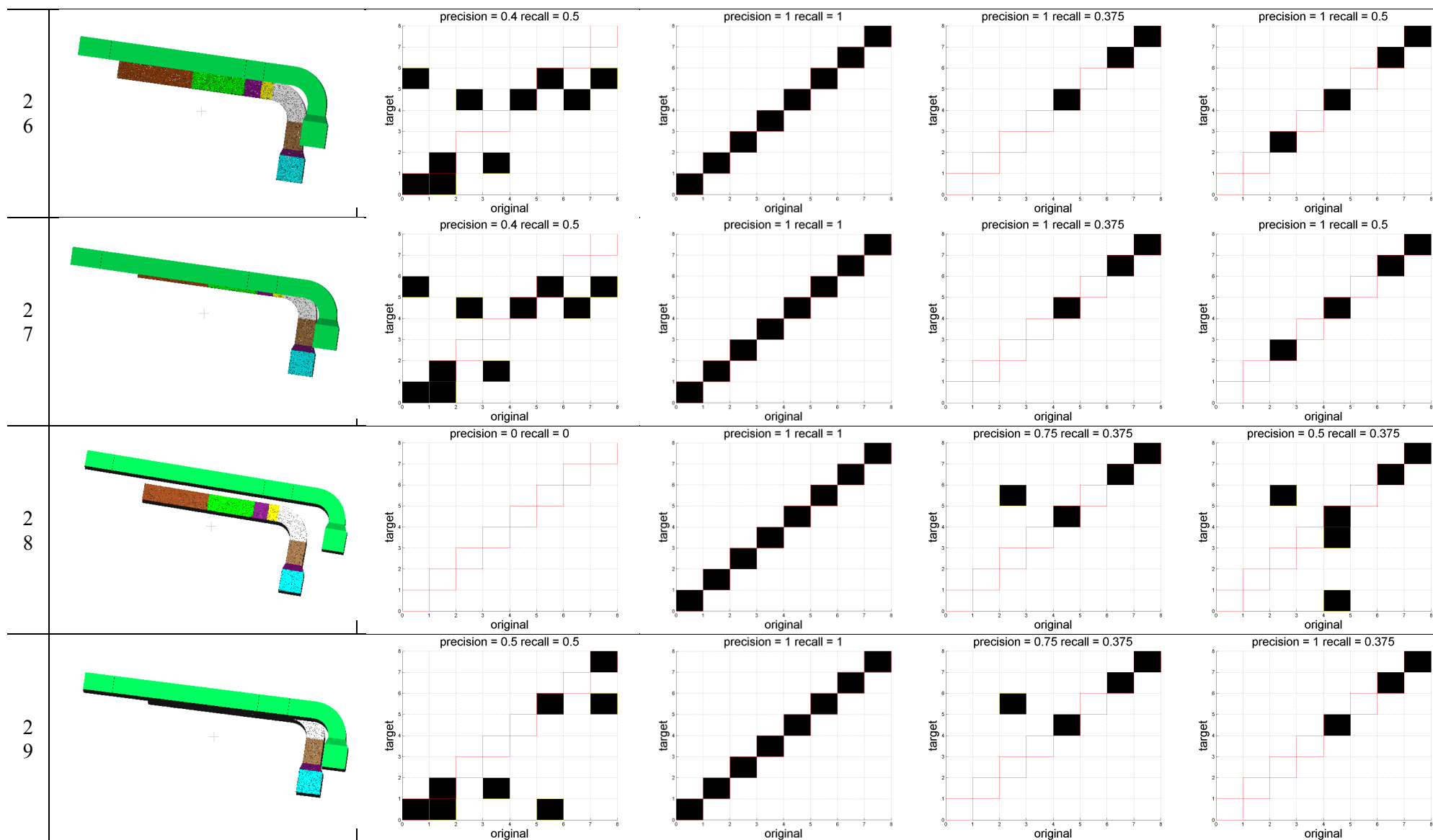


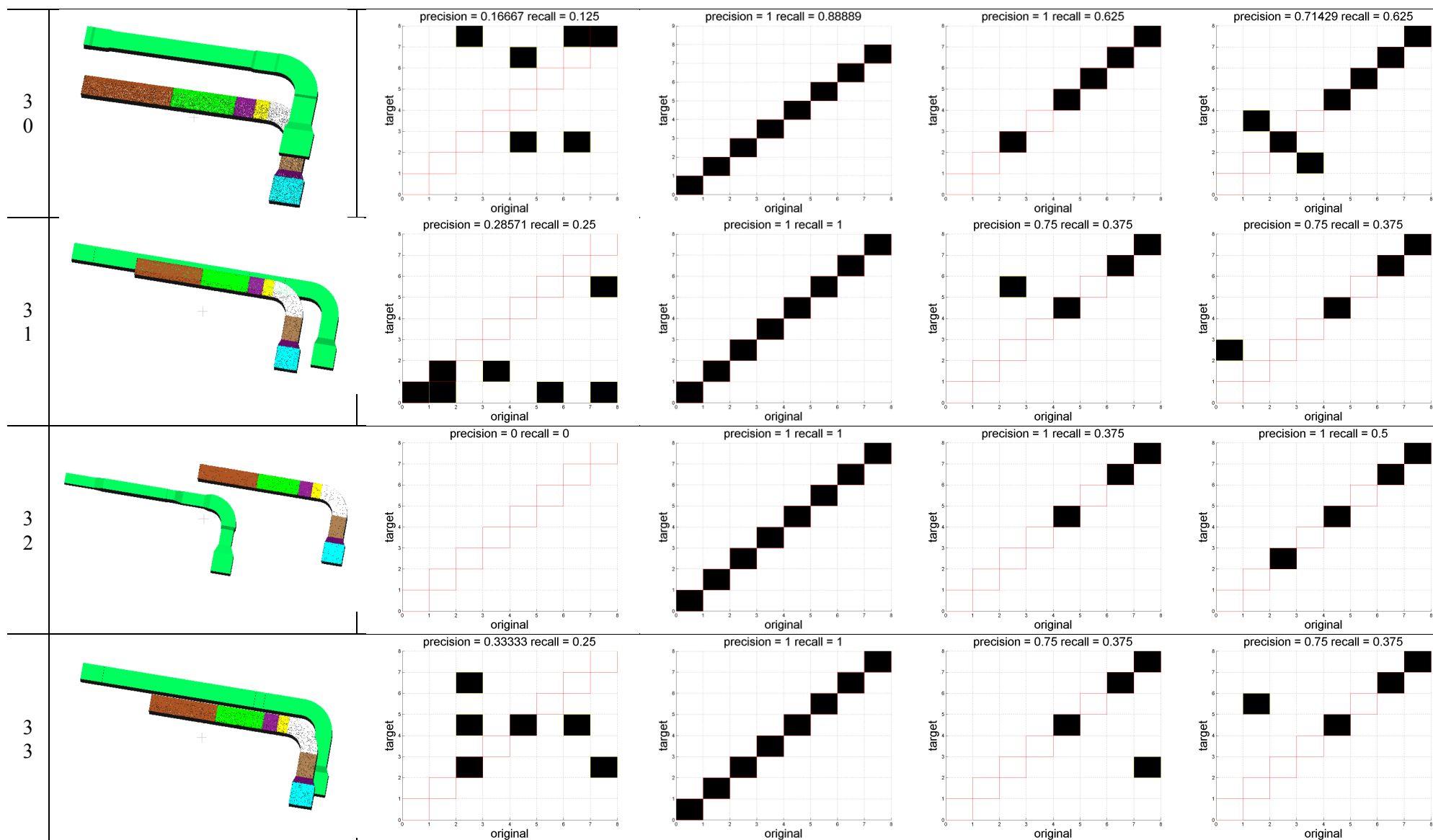


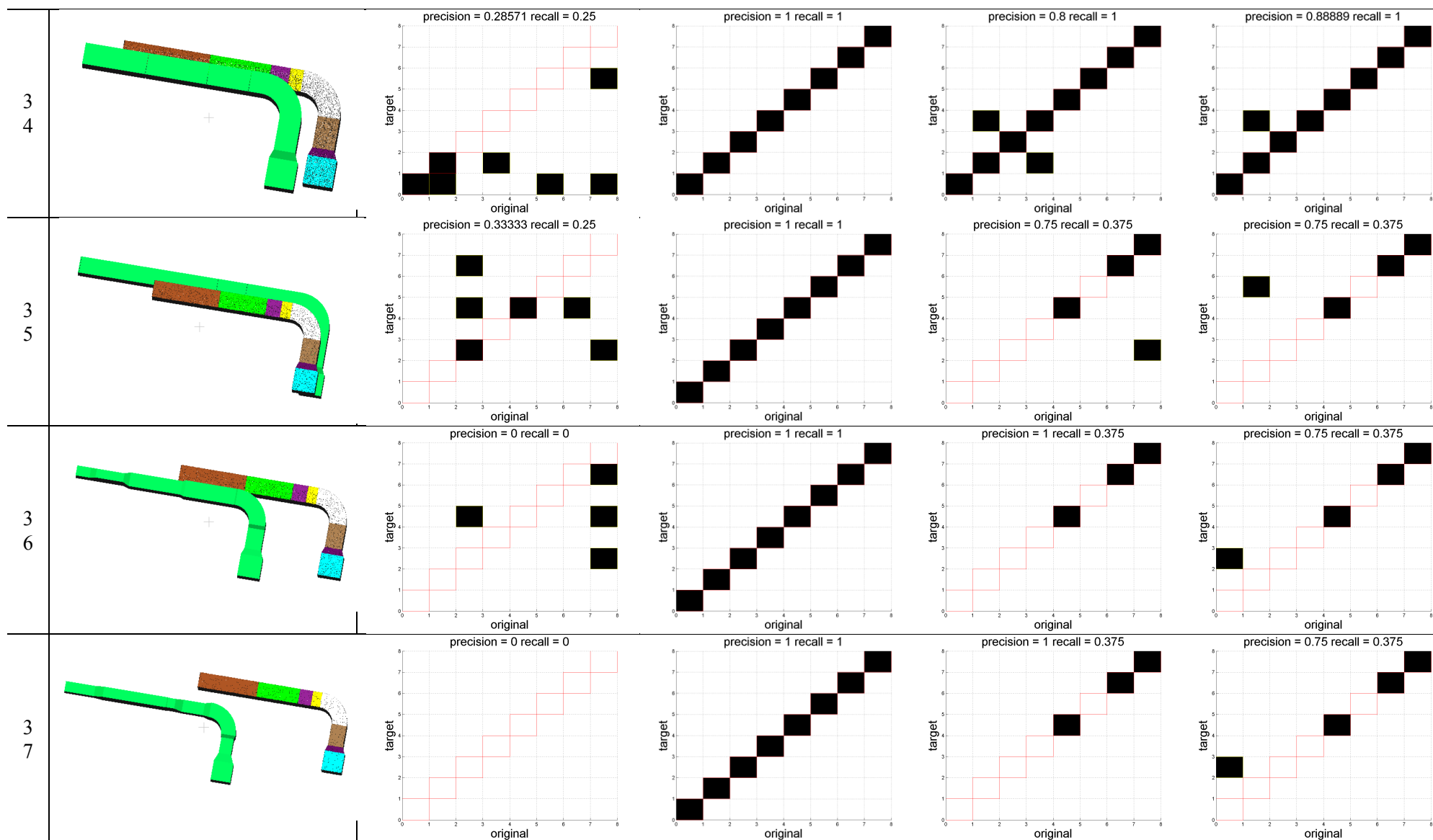


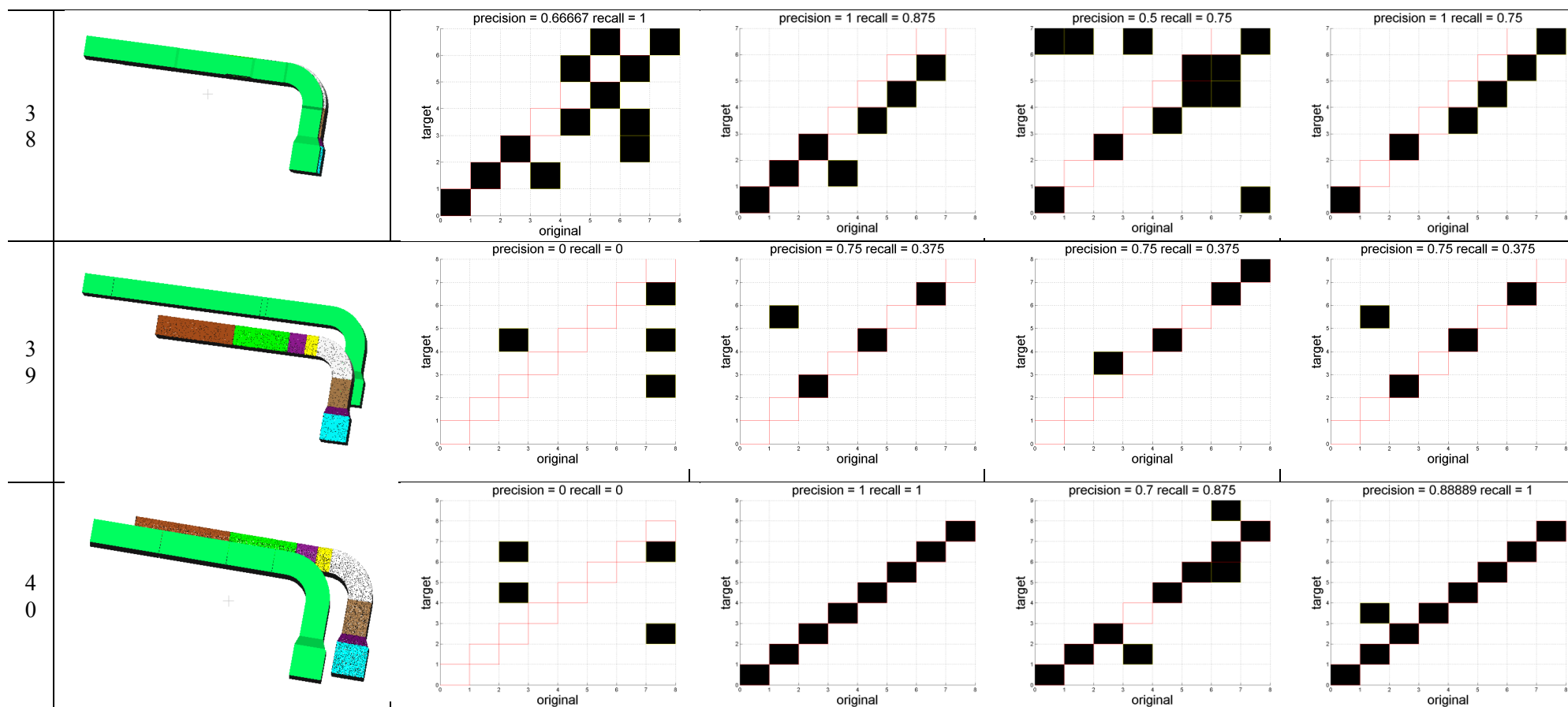












## Appendix B. Codes for implementing the four matching approaches

The four matching approaches are developed using Matlab. This appendix includes the codes for implementing the four matching approaches and conducting the experiment. Due to the space limit, I only include the partial codes here. Please refer to the DVD attached with the thesis for the complete codes.

```
-----
% This script is used to run the different mapping algorithms and
% collect the results.
% The input variables include: totalSet, trainingSet, testingSet,
density,
% sampleSize, noBin, inputPath, outputPath
-----

%% Get the inputs from the targeted folders
disp('Get the inputs');
% define the number of testing data and the number of training data
totalSet = 60; % total number of dataset
trainingSet = 45; % number of training data
testingSet = 15; % number of testing data
density = 400;
sampleSize = 100000;
noBin = 1024;

% define two path:
% path 1 is the reading path
inputPath = 'C:\Users\tgao\Documents\Test 1\';
outputPath = 'C:\Users\tgao\Documents\Test 1\';
% path 2 is the save path

% get the input for the baseline
folderName = 'Baseline';
PCD = getInput(folderName,inputPath);

% add 3D shape mapping
PCD_distance = getInput3D(folderName,inputPath, density,sampleSize);
sizeComponent = size(PCD_distance,1);
PCD_element_inBin = zeros(sizeComponent,noBin);
for j = 1:sizeComponent
    % PCD_element_inBin[componentNo, noBin]
    PCD_element_inBin(j,:) = hist(PCD_distance(j,:),noBin);
end

correctMatching = eye(PCD.NoComponent);

% get the other dataset and perform the matching algorithms
folderName_default = 'Scenario';
src = cell(totalSet,1); % src is used to store all the dataset

for i = 1:totalSet
```



```

disp(['get the data set',num2str(i)]);
folderName = [folderName_default,num2str(i)];
src{i} = getInput(folderName,inputPath);

% overlap area matching:
disp('starting 2D overlap matching');
overlapRatio = calOverlapRatio(PCD.image,src{i}.image);
src{i}.overlapRatio = overlapRatio;
disp('Finishing 2D overlap matching');

% 2D shape matching:
disp('starting 2D shape matching');
[difference_norm, difference_no] =
match_2D(PCD.boundary,src{i}.boundary);
src{i}.difference_no_2d = difference_no;
disp('Finishing 2D shape matching');

% area matching
disp('starting area matching');
differences = calAreaDifference(PCD.image,src{i}.image);
src{i}.difference_area = differences;
disp('Finishing area matching');

% graph matching
addpath('C:\Users\tgao\Documents\MATLAB\Point cloud processing
function\fgm_modified');
asgFgm = graphMatching(PCD, src{1});
end

for i = 1:totalSet
    folderName = [folderName_default,num2str(i)];
    % distance [componentNo, sampleSize]
    src{i}.distance = getInput3D(folderName,inputPath, density,
sampleSize);
    element_inBin = zeros (sizeComponent,noBin);
    for j = 1:sizeComponent
        % element_inBin [componentNo, noBin]
        element_inBin (j,:) = hist(src{i}.distance(j,:),noBin);
    end
    src{i}.element_inBin = element_inBin;
    diff = calHistDifference (PCD_element_inBin,element_inBin);
    src{i}.diff3D = diff;
end

% data analysis
[precision, recall, map, perm, threshold, index] = performAnalysis(
src, correctMatching, trainingSet, testingSet);
% take threshold
map.shape_optimal = cell(1,trainingSet);
map.overlap_optimal = cell(1,trainingSet);
map.area_optimal = cell(1,trainingSet);
map.shape_3D_optimal = cell(1,trainingSet);

%% done the calculation, this step just to output all the requiried
figures for the analysis
for i = 1:trainingSet
    map.shape_optimal{1,i} = map.shape_train{index.shape,i};

```

```

        out1 = [outputPath, 'shape_optimal', num2str(i)];
        outputMappingMatrix(map.shape_optimal{1,i},out1,
precision.shape_optimal(1,i),recall.shape_optimal(1,i));
        map.overlap_optimal{1,i} = map.overlap_train{index.overlap,i};
        out2 = [outputPath, 'overlap_optimal', num2str(i)];

        outputMappingMatrix(map.overlap_optimal{1,i},out2,precision.overlap_opt
imal(1,i),recall.overlap_optimal(1,i));
        map.area_optimal{1,i} = map.area_train{index.area,i};
        out3 = [outputPath, 'area_optimal', num2str(i)];

        outputMappingMatrix(map.area_optimal{1,i},out3,precision.area_optimal(1
,i),recall.area_optimal(1,i));
        map.shape_3D_optimal{1,i} = map.shape_3D_train{index.shape_3D,i};
        out4 = [outputPath, 'shape_3D_optimal', num2str(i)];

        outputMappingMatrix(map.shape_3D_optimal{1,i},out4,precision.shape_3D_o
ptimal(1,i),recall.shape_3D_optimal(1,i));
    end

    for i = 1:testingSet
        out1 = [outputPath, 'shape_test', num2str(i)];
        outputMappingMatrix(map.shape_test{i,1},out1,
precision.shape_test(i,1),recall.shape_test(i,1));
        out2 = [outputPath, 'overlap_test', num2str(i)];
        outputMappingMatrix(map.overlap_test{i,1},out2,
precision.overlap_test(i,1),recall.overlap_test(i,1));
        out3 = [outputPath, 'area_test', num2str(i)];
        outputMappingMatrix(map.area_test{i,1},out3,
precision.area_test(i,1),recall.area_test(i,1));
        out4 = [outputPath, 'shape_3D_test', num2str(i)];
        outputMappingMatrix(map.shape_3D_test{i,1},out4,
precision.shape_3D_test(i,1),recall.shape_3D_test(i,1));
    end

    %% draw the intersection and union figure
    precision.union = zeros (testingSet,1);
    recall.union = zeros (testingSet,1);
    for i = 1:testingSet
        out1 = [outputPath, num2str(i)];
        map.union{1,i}=outputOrAnd (map.overlap_test{i,1},
map.area_test{i,1}, map.shape_test{i,1},
map.shape_3D_test{i,1},out1,correctMatching);
        [precision.union(i,1),recall.union(i,1)] = calRecallPrecision
(map.union{1,i},correctMatching);
    end

    %% try the new combination approach
    move = cell (totalSet,1);
    for i = 1:totalSet
        move{i} = shape_overlap(src{i}, PCD ,12);
        savePath = [outputPath, 'shiftedImage', num2str(i)];
        imwrite (move{i}.completeImage,[savePath, '.jpg']);
    end

    % find the threshold that generates maximum results
    threshold_value = (0.01:0.01:1);

```

```

iter = length (threshold_value);
for j = 1:trainingSet
    for i = 1:iter
        set = perm (j);

[recall.overlap_shape_train(i,j),precision.overlap_shape_train(i,j),map
.overlap_shape_train{i,j}] = overlap_Analysis(move.overlapRatio,
correctMatching, threshold_value(i));
        end
    end
[threshold.overlap_shape,index.overlap_shape_optimal,precision.overlap_
shape_optimal,recall.overlap_shape_optimal] = findThreshold
(recall.overlap_shape_train,precision.overlap_shape_train,threshold_val
ue);
for a = 1:testingSet
    current = trainingSet+a;
    set = perm(current);
    [recall.overlap_shape_test(a),
precision.overlap_shape_test(a),map.overlap_shape_test{a}] =
overlap_Analysis(move(Ullrich et al.).overlapRatio, correctMatching,
threshold.overlap_shape);
end

for i = 1:trainingSet
    map.overlap_shape_optimal{1,i} =
map.overlap_shape_train{index.overlap_shape_optimal,i};
    out5 = [outputPath, 'shape_overlap_optimal', num2str(i)];

outputMappingMatrix(map.overlap_shape_optimal{1,i},out5,precision.overl
ap_shape_optimal(1,i),recall.overlap_shape_optimal(1,i));
end

for i = 1:testingSet
    out5 = [outputPath, 'shape_overlap_test', num2str(i)];
    outputMappingMatrix(map.overlap_shape_test{1,i},out5,
precision.overlap_shape_test(1,i),recall.overlap_shape_test(1,i));
end

%% further filter out the mapping candidate get from the overlap_shape
for a = 1:testingSet
    current = trainingSet+a;
    set = perm(current);
    map.filter{a} =
shape_overlap_filter(map.overlap_shape_test{1,a},src(Ullrich et al.));
    [ precision.filter(a),recall.filter(a) ] = calRecallPrecision(
map.filter{a},correctMatching );

    out6 = [outputPath, 'shape_overlap_test_filter', num2str(a)];
    outputMappingMatrix(map.filter{a},out6,
precision.filter(a),recall.filter(a));
end

%% draw the union figure for the total five approaches
precision.union_total = zeros (testingSet,1);
recall.union_total = zeros (testingSet,1);
map.union_total = cell(1,testingSet);

```

```

for i = 1:testingSet
    savePath = [outputPath, 'union_total', num2str(i)];
    h = figure;
    out1 = [outputPath, 'total', num2str(i)];
    map.union_total{1,i} = map.union{1,i}+map.overlap_shape_test{1,i};
    map.union_total{1,i} = map.union_total{1,i}>0;
    [nb,np] = size (map.union_total{1,i});
    for k = 1:nb
        for j = 1:np
            if (map.union_total{1,k}(k,j) ==1)
                rectangle ('Position',[k-1,j-
1,1,1], 'FaceColor', 'black', 'EdgeColor', 'red');
            end
        end
    end
    [precision.union_total(i,1),recall.union_total(i,1)] =
calRecallPrecision(map.union_total{1,i},correctMatching);
    grid on
    axis([0 nb 0 np]);
    xlabel ('original', 'FontSize', 25);
    ylabel ('target', 'FontSize', 25);
    title ([ 'union_total: precision =
', num2str(precision.union_total(i,1)), ' recall =
', num2str(recall.union_total(i,1))], 'FontSize', 22);
    saveas(h,[savePath, '-union'], 'png');
    close(h);
end

-----
% This script is used for two purpose:
% (1) Get the optimal threshold
% (2) Calculate the precision and recall for the four matching
approaches
%
%Input:  src {i}, i = number of component in one scenario
%        src.overlapRatio, src.difference_no_2d, src.difference_area,
%        src.diff3D
%
%output: precision, recall, map and the order of perm
%        precision.test(testingcase, 1), prcision.train(1,trainingcase)
%
% Map => x original, y target
-----

function [ precision, recall, map , perm, threshold, index] =
performAnalysis( src, correctMatching, trainingSet, testingSet)

% Determine the thresholds for different mapping algorithms
threshold_value = (0.01:0.005:1); % the vector that store the possible
value of threshold
iter = length (threshold_value);

% four variables - precision, recall and result for storing the results
of mapping
% algorithm
precision.overlap_train = zeros (iter,trainingSet);
recall.overlap_train = zeros (iter,trainingSet);

```

```

map.overlap_train = cell (iter,trainingSet);

precision.shape_train = zeros (iter,trainingSet);
recall.shape_train = zeros (iter,trainingSet);
map.shape_train = cell (iter,trainingSet);

precision.area_train = zeros (iter,trainingSet);
recall.area_train = zeros (iter,trainingSet);
map.area_train = cell (iter,trainingSet);

precision.shape_3D_train = zeros (iter,trainingSet);
recall.shape_3D_train = zeros (iter,trainingSet);
map.shape_3D_train = cell (iter,trainingSet);

% testing and training dataset should be ranomly generated
totoalSet = trainingSet+testingSet;
perm = randperm(totoalSet);

for j = 1:trainingSet
    for i = 1:iter
        % get the precision and recall for 2D overlap
        set = perm(j);

[recall.overlap_train(i,j),precision.overlap_train(i,j),map.overlap_train{i,j}] = overlap_Analysis(src(Ullrich et al.).overlapRatio,
correctMatching, threshold_value(i));

[recall.shape_train(i,j),precision.shape_train(i,j),map.shape_train{i,j}] = mapping_Analysis(src(Ullrich et al.).difference_no_2d,
correctMatching, threshold_value(i));

[recall.area_train(i,j),precision.area_train(i,j),map.area_train{i,j}] = mapping_Analysis(src(Ullrich et al.).difference_area,correctMatching,
threshold_value(i));

[recall.shape_3D_train(i,j),precision.shape_3D_train(i,j),map.shape_3D_train{i,j}] = mapping_Analysis(src(Ullrich et al.).diff3D,
correctMatching, threshold_value(i));
    end
end

disp ('find the optimal threhsold');
% find the maximum threshold
[threshold.overlap,index.overlap,precision.overlap_optimal,recall.overlap_optimal] = findThreshold
(recall.overlap_train,precision.overlap_train,threshold_value);
[threshold.shape,index.shape,precision.shape_optimal,recall.shape_optimal] = findThreshold (recall.shape_train,
precision.shape_train,threshold_value);
[threshold.area,index.area,precision.area_optimal,recall.area_optimal] = findThreshold
(recall.area_train,precision.area_train,threshold_value);
[threshold.shape_3D,index.shape_3D,precision.shape_3D_optimal,recall.shape_3D_optimal] = findThreshold
(recall.shape_3D_train,precision.shape_3D_train,threshold_value);

%% testing cases

```

```

disp ('start with the test cases');
precision.area_test = zeros (testingSet,1);
recall.area_test = zeros (testingSet,1);
map.area_test = cell (testingSet,1);

precision.shape_test = zeros (testingSet,1);
recall.shape_test = zeros (testingSet,1);
map.shape_test = cell (testingSet,1);

precision.overlap_test = zeros (testingSet,1);
recall.overlap_test = zeros (testingSet,1);
map.overlap_test = cell (testingSet,1);

precision.shape_3D_test = zeros (testingSet,1);
recall.shape_3D_test = zeros (testingSet,1);
map.shape_3D_test = cell (testingSet,1);

for a = 1:testingSet
    current = trainingSet+a;
    set = perm(current);
    [recall.overlap_test(a),
precision.overlap_test(a),map.overlap_test{a}] =
overlap_Analysis(src(Ullrich et al.).overlapRatio, correctMatching,
threshold.overlap);
    [recall.shape_test(a), precision.shape_test(a),map.shape_test{a}] =
mapping_Analysis(src(Ullrich et al.).difference_no_2d, correctMatching,
threshold.shape);
    [recall.area_test(a), precision.area_test(a),map.area_test{a}] =
mapping_Analysis(src(Ullrich et al.).difference_area, correctMatching,
threshold.area);
    [recall.shape_3D_test(a),
precision.shape_3D_test(a),map.shape_3D_test{a}] =
mapping_Analysis(src(Ullrich et al.).diff3D, correctMatching,
threshold.shape_3D);
end
disp ('finish testing');
end

```

```

-----
%% this function take recall, precision and the corresponding threshold
value and try to
% find the optimal value for the threshold
-----

```

```

function [value, i, precision_train, recall_train] =
findThreshold(recall,precision,threshold)
%recall_sum = sum (recall,2);
recall_mean = mean(recall,2);
precision_mean = mean(precision,2);
F = (2.*precision_mean.*recall_mean)./(recall_mean+precision_mean);

%precision_sum = sum (precision,2);
[~,i] = max(F);
value = threshold(i);
precision_train = precision(i,:);
recall_train = recall (i,:);
end

```

```

-----
% This function calculates the preision and recall for a scenario
% Input: difference - (i,j) the difference between component i in
original
% pcd and component j in target pcd
%       correctMapping - the ground truth
%       threshold
% output: recall, precision and final mapping result
-----

```

```

function [recall,precision,result] = mapping_Analysis(difference,
correctMapping, threshold )
%% Perform the data analysis
d_work = difference;
summary = sum (d_work);
[a, b] = size (difference);
summary = repmat (summary,[a,1]);
d_work = d_work./summary;
map = d_work <= threshold;
template = (1:a);
template = repmat (template', [1,b]);
result = template.*map;
% get the precision
% get the recall
temp = correctMapping.*map;
correct = sum(sum(temp));
allMapping = sum(sum(map));
allCorrect = sum(sum(correctMapping));
if (allMapping == 0)
    precision = 0;
else
    precision= correct/allMapping;
end
if (allCorrect ==0)
    recall = 0;
else
    recall= correct/allCorrect;
end
end

```

```

-----
% 2D shape matching
% input:
%   boundary_BIM(i,1): 2D boundary extracted from BIM, i - no of
component
%   boundary_PCD(j,1): 2D boundary extracted from PCD, j - no of
component
% option: 'normlization' or 'non_normalization'
% output:
%   dif_no (i,j): i - no of BIM component, j - no of PCD component
-----

```

```

function [dif_norm,dif_no] = match_2D(boundary_BIM,boundary_PCD)
%% define the prameters
sampleSize = 10000;
binSize = 1024;
nb = length (boundary_BIM);

```

```

np = length (boundary_PCD);
distance_BIM = zeros (nb,sampleSize);
for i = 1:nb
    boundary = boundary_BIM{i};
    index = size (boundary,1);
    r1 = randi(index,[sampleSize,1]);
    r2 = randi(index,[sampleSize,1]);
    for j = 1:sampleSize
        distance_BIM (i,j)= sqrt((boundary (r1(j),1) -
boundary(r2(j),1))^2+(boundary (r1(j),2) - boundary(r2(j),2))^2);
    end
end
distance_PCD = zeros (np,sampleSize);
for i = 1: np
    boundary = boundary_PCD{i};
    index = size (boundary,1);
    r1 = randi(index,[sampleSize,1]);
    r2 = randi(index,[sampleSize,1]);
    for j = 1:sampleSize
        distance_PCD (i,j)= sqrt((boundary (r1(j),1) -
boundary(r2(j),1))^2+(boundary (r1(j),2) - boundary(r2(j),2))^2);
    end
end
%% No normalization
% create a cell to store all the distance
distance = [distance_BIM;distance_PCD];
distance = num2cell (distance,2);
[t,element_inBin,X] = nhist (distance,'samebins','minbins', 1000);
%Compare N
difference = zeros (nb,np);
for i=1:nb
    a = element_inBin{i};
    for j = 1:np
        b = element_inBin{nb+j};
        difference (i,j) = sum(sum(abs(a - b)));
    end
end
[Y,I] = min (difference');
dif_no = difference;
%% Normlization
bin_no = 1000;
element_inBin_new = zeros (np+nb,bin_no);
for i = 1:np+nb
    element_inBin_new (i,:) = hist(distance{i,1},bin_no);
end
difference_new = zeros (nb,np);
for i=1:nb
    a = element_inBin_new(i,:);
    for j = 1:np
        b = element_inBin_new(nb(1)+j,:);
        difference_new (i,j) = sum(sum(abs(a - b)));
    end
end
[Y_new,I_new] = min (difference_new');
dif_norm = difference_new;
end

```



```

-----
% The function is used to calculate the 2D overlap ratio between two
components
% Input: b, the boundary of 2D BIM component
%       p, the boundary of 2D projected points
% output: overlapRatio (BIM, Point cloud)
-----

function overlapRatio = calOverlapRatio(b,p)
nb = size (b,1);
np = size (p,1);
overlapRatio = zeros (nb,np);
for i = 1:nb
    for j = 1:np
        areaB = sum(sum(b{i}));
        overlap = b{i}.*p{j};
        overlapArea = sum(sum(overlap));
        overlapRatio(i,j) = overlapArea/areaB;
    end
end
end

-----
% This function return the 3D shape distance matrix for a scenario
(component n)
% input: foldername - the name of the folder
%       inputPath - the path where the folder is saved
%       density - the density of trnasformed point cloud
%       sampleSize - the number of pairs of pairwise distance
% output: distance (i,j), i = component i, j = pairwise distance j
-----

function distance = getInput3D( folderName, inputPath, density,
sampleSize)
files = dir(fullfile(inputPath ,folderName, '*.stl'));
filePath = [inputPath,folderName, '\\'];
NoComponent = length(files);

num = zeros(NoComponent,1);
for i = 1:NoComponent
    num (i) = sscanf (files(i).name, 'ComponentNo%d.stl#');
end
[~, index] = sort(num); % index the correct index of components

% distance metrix: d[i ,j] - i is the no of component, j is the sample
size
distance = zeros (NoComponent, sampleSize);
for i = 1:NoComponent
    stlAddress = [filePath,files(i).name];
    [F, V, ~] = rndread(stlAddress);
    % samplePoints[i,1], i is the sample size
    samplePoints = sampleMesh(F,V,density);
    % determine the samping rate
    % calculate pairwise distanc
    distance(i,:) = cal_pairwiseDistance(samplePoints, sampleSize)';
end

```

end

```
% this function take two inputs: points and sample
% output the distance metrix
function [ distance ] = cal_pairwiseDistance( points, sample )
pointSize = length (points);
picks = 1+round((pointSize-1)*rand(2*sample,1));
count = 1;
distance = zeros (sample,1);
for i = 1:2:2*sample
    id1 = picks (i,1);
    id2 = picks (i+1,1);
    x1 = points(id1,1);
    y1 = points(id1,2);
    z1 = points(id1,3);
    x2 = points(id2,1);
    y2 = points(id2,2);
    z2 = points(id2,3);
    d = sqrt ((x1-x2)^2 + (y1-y2)^2 + (z1-z2)^2);
    distance (count,1) = d;
    count = count+1;
end
```

---

%This function is used to perform the shape + overlap filter

---

```
function [map_filter] = shape_overlap_filter(map, src)
[nb,np] = size (map);
map_filter = zeros (nb,np);
for i = 1:nb
    [~,index] = find (map(i,:)>0);
    p = length(index);
    if (p~=0)
        difference = src.diff3D(i,index);
        [~,match] = min(difference);
        map_filter(i,index(match)) = 1;
    end
end
```

---

% for each scenario, this function get the mapping matrix and output the mapping image

---

```
function [] = outputMappingMatrix( map, titleName, precision, recall )
%CREATEMETRIC Summary of this function goes here
% Detailed explanation goes here
standard = map > 0;
[nb,np] = size (standard);
h = figure;
for i = 1:nb
    for j = 1:np
        if (standard(i,j)==1)
            rectangle ( 'Position',[i-1,j-
1,1,1], 'FaceColor', 'black', 'EdgeColor', 'yellow');
        end
    end
end
```

```

        if (i==j)
            rectangle ('Position',[i-1,j-1,1,1],'EdgeColor','r');
        end
    end
end
grid on
axis([0 nb 0 np]);
xlabel ('original','FontSize', 25);
ylabel ('target','FontSize', 25);
title (['precision = ',num2str(precision),' recall = ',num2str(recall)],'FontSize', 25);
%print('-dpng','-r100',titleName);
saveas(h,titleName,'png')
close(h);

-----
% Perform the graph matching for the point cloud
% Extended from the research done by Feng Zhou
% Te Gao
-----

clear variables;
addPath;
prSet(5); % set the prompt level
%% Get the image and pick up the points for the nodes
% create a very simple UI for interact with the system
global footpath;
imgPath = sprintf ('%s/BIM_BIM/',footpath);
choice = menu ('Please select the option','load file','determine
nodes','simulate the location discrepancy');
if (choice ==1)
    [FileName,PathName] = uigetfile('*.mat','Select the MATLAB code
file');
    Path = [PathName, FileName];
    if (exist(Path,'file') == 0)
        fprintf ('The file is not existed');
    else
        load (Path);
        loadPath = FileName;
        fprintf ('use the pre-saved file.\n');
    end
end

if (choice ==2)
    %BIM_Path = [imgPath,'ori_BIM.jpg'];
    %PCD_Path = [imgPath,'new_PCD.jpg'];
    BIM_Path = [imgPath,'BIM.png'];
    PCD_Path = [imgPath,'PCD.png'];
    BIM = imread (BIM_Path);
    PCD = imread (PCD_Path);
    figure ('name','Select nodes from BIM');
    imshow (BIM);
    [xBIM yBIM] = ginput;
    hold on
    scatter (xBIM,yBIM,100,'fill');
    hold off
    figure ('name','Select nodes from point cloud')
    imshow (PCD);
end

```

```

[xPCD yPCD] = ginput;
hold on
scatter (xPCD,yPCD,100,'fill');
hold off
xTs = {[xBIM,yBIM]','[xPCD,yPCD]'};
prompt = {'Save the file, Y/N?','Enter the file path for save'};
dig_title = ('Input');
num_lines = 1;
answer = inputdlg(prompt, dig_title,num_lines);
if (answer{1} == 'Y' )
    save (answer{2},'xTs');
    loadPath = answer{2};
end
fprintf ('The file has been saved.\n');
end

if (choice ==3)
    [FileName,PathName] = uigetfile('*.mat','Select the MATLAB code
file');
    Path = [PathName, FileName];
    if (exist(Path,'file') == 0)
        fprintf ('The file is not existed');
    else
        load (Path);
        loadPath = FileName;
        fprintf ('use the pre-saved file.\n');
    end
    %% simulate the discrepancy
    BIM = xTs{1};
    PCD = xTs{2};
    displacement = (1:3:1000);
    PCD(2,2) = PCD(2,2)- displacement(100);
    xTs{2} = PCD;
end

%% read src
nb = size (xTs{1},2);
np = size (xTs{2},2);
tag = 'pointCloud'; pFs = [1 2]; nIn = [nb nb];
% the important field in wsSrc is point coordinate
% in my case, 2X13 is enough
% wsSrc = pcdAsgSrc(tag, pFs, nIn,loadPath,'svL',1);
wsSrc = simAsgSrc(tag, pFs, nIn,xTs,'svL',1);
% asgT the attribute to save the final output?
asgT = wsSrc.asgT;

%% read feature
% define two parameters and store them in a structure
parG = st('link', 'predefine');
parF = st('smp', 'n', 'nBinT', 4, 'nBinR', 3);
% obtain features: two important files - gphs, and fs
wsFeat = pcdAsgFeat(wsSrc, parG, parF, 'svL', 1);
[gphs Fs] = stFld(wsFeat, 'gphs','Fs');

ns = cellDim(wsSrc.Pts, 2);

%% affinity

```

```

% need to re-defined the affinity
parKnl = st('alg', 'cmum');
[KP, KQ] = conKnlGphPQ(gphs, parKnl);
K = conKnlGphK(KP, KQ, gphs);

%% parameter
pars = gmPar(1);

%% SM
asgSm = gm(K, ns, asgT, pars{3}{:});

%% FGM
% here is the actual implementation of fgm algorithm
% need to define KP, KQ
% don't worry about Fs
asgFgm = fgm(KP, KQ, gphs, asgT, pars{3}{1});

%% show correpondence matrix
asgs = {asgT, asgSm, asgFgm};
algs = {'Truth', 'SM', 'FGM'};
rows = 1; cols = 3;
Ax = iniAx(1, rows, cols, [250 * rows, 250 * cols]);
shAsgX(asgs, Ax, algs);
%%-----the above code is able to work
%% show correspondence result
%rows = 2; cols = 1;
%Ax = iniAx(2, rows, cols, [400 * rows, 900 * cols]);
parCor = st('mkSiz', 7, 'cls', {'y', 'b', 'g'});

%shAsgImg(Fs, gphs, asgSm, asgT, parCor, 'ax', Ax{1});
%title(sprintf('SM : acc %.2f, obj %.2f', asgSm.acc, asgSm.obj));
shAsgImg(Fs, gphs, asgFgm, asgT, parCor);
title(sprintf('FGM: acc %.2f, obj %.2f', asgFgm.acc, asgFgm.obj));

```

## Appendix C. Codes for the discrepancy simulation algorithm

The simulation algorithm developed in this research runs as an add-on to a widely used building information modeling tool. The algorithm is developed using C#.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
using STL_Export;

using Autodesk.Revit.UI;
using Autodesk.Revit.DB;
using Autodesk.Revit.Attributes;
using Autodesk.Revit.UI.Selection;
using Autodesk.Revit.DB.Mechanical;
using Autodesk.Revit.ApplicationServices;

// define a technique to store the discrepancy for correction analysis:
// each element - 1 = length discrepancy; 2 = location discrepancy; 3 = width
// discrepancy, 4 = height discrepancy
namespace Test
{
    [TransactionAttribute(TransactionMode.Manual)]
    public class DiscrepancyIntroduction : IExternalCommand
    {
        // define the two main variables that will be used throughout different
        // functions
        Document document;
        Application app;
        string record = "";
        public Result Execute(ExternalCommandData commandData, ref string message,
        ElementSet elements)
        {
            UIDocument doc = commandData.Application.ActiveUIDocument;

            // assign values to document and app
            document = doc.Document;
            app = document.Application;

            try
            {
                /*****The Discrepancy
                Creation*****/
                // iterate for 100 times
                // we also need to define a variable to store the simulation
                result
                ElementSet set = new ElementSet();
                set = GetAllModelElements();
                Random random = new Random();
                //string savePath =
                "C:\\Users\\tgao\\Desktop\\SimulationResult_5\\Baseline";
                string savePath =
                "C:\\Users\\tgao\\Desktop\\SimulationResult_5\\Senario4";
            }
        }
    }
}
```

```

DirectoryInfo di = Directory.CreateDirectory(savePath);
Transaction trans = new Transaction(document);
trans.Start("baseline");
exportImg(savePath + "\\complete");
//TaskDialog.Show("Revit", "The complete figure is exported");
trans.Commit();
exportImgs(set, savePath, trans);
trans.Start("unhide");
document.ActiveView.Unhide(set);
trans.Commit();
var ductTypeCollector1 = new FilteredElementCollector(document);
ductTypeCollector1.WherePasses(new
ElementClassFilter(typeof(Duct)));
IList<Element> ducts = ductTypeCollector1.ToElements();
record = "Baseline"+"\\n";
record = outputInfo(ducts, record);
// give a number for the dimension simulatoin as well, when i is
greater than 30
int whenDimensionChange = 45;
for (int i = 1; i <= 0; i++)
{
    //Step 1: change the dimension of the ducts

    //String info = "Get all the ducts";
    //TaskDialog.Show("Revit", info);
    //ShowElementList(ducts, "show element type");
    int flag = 0;
    //trans.Start("change");
    foreach (Element e in ducts)
    {
        Duct duct = (Duct)e;
        flag = random.Next(0, 2);
        if (flag == 0)
        {
            // TaskDialog.Show("Whether to change the component or
not ", duct.Name + "No!");
            //do nothing

        }
        else
        {
            //change the profile
            //trans.Start();
            ModifyElementLengthWithAttach(e, trans);
            //trans.Commit();
        }
    }

    if (i > whenDimensionChange)
    {
        int flag_Width = random.Next(0,3);
        if (flag_Width == 1)
        {
            double change = 0.75 + random.NextDouble();
            ModifyElementWidth(e, change,trans);
        }
        int flag_Height = random.Next(0, 3);
        if (flag_Height == 1)
    }
}

```

```

        {
            double change = 0.75 + random.NextDouble();
            ModifyElementHeight(e, change, trans);
        }
    }
    // step 2: move the ducts
    flag = random.Next(0, 2);
    if (flag == 0)
    {
        //TaskDialog.Show("Warning", "no movement");
        //do nothing
    }
    else
    {
        //trans.Start();
        trans.Start("change");
        generateLocationDiscrepancy();
        trans.Commit();
    }
    //trans.Commit();
    // step 3: output the model:

    savePath =
"C:\\Users\\tgao\\Desktop\\SimulationResult_4\\Scenario" + i;
    di = Directory.CreateDirectory(savePath);
    trans.Start("export");
    exportImg(savePath + "\\complete");
    trans.Commit();
    //TaskDialog.Show("Revit", "The complete figure is exported");
    set = GetAllModelElements();
    exportImgs(set, savePath, trans);
    trans.Start("unhide");
    document.ActiveView.Unhide(set);
    trans.Commit();
    record += "scenario" + i.ToString() + "\n";
    record = outputInfo(ducts, record);
}

System.IO.File.WriteAllText(@"C:\Users\tgao\Documents\test.txt",
record);
}
catch (Exception e)
{
    message = e.Message;
    return Result.Failed;
}
return Result.Succeeded;
}

public void exportImgs(ElementSet allComponents, string path, Transaction
tran)
{
    // Step 4: add the visibility control
    //Transaction tran1 = new Transaction(document);

    int i = 1;
    foreach (Element elem in allComponents)

```



```

{
    tran.Start();
    document.ActiveView.Hide(allComponents);
    tran.Commit();

    ElementSet unhide = new ElementSet();
    string temPath = "";
    string temPathStl = "";
    if (elem.Name.Equals("Experiment"))
    {
        tran.Start();
        unhide.Insert(elem);
        document.ActiveView.Unhide(unhide);
        tran.Commit();
        temPath = path + "\\ComponentNo" + i + ".png";
        temPathStl = path + "\\ComponentNo" + i + ".stl";
        tran.Start();
        exportImg(temPath);
        export_Stl(temPathStl);
        tran.Commit();
        i++;
    }
}

public void export_Stl(string fileName)
{
    DataGenerator STL_Generator = new DataGenerator(app, document,
document.ActiveView);
    SaveFormat saveFormat = SaveFormat.ASCII;
    ElementsExportRange exportRange;
    exportRange = ElementsExportRange.OnlyVisibleOnes;
    ExportSettings aSetting = new ExportSettings(saveFormat, exportRange);
    DataGenerator.GeneratorStatus succeed =
STL_Generator.SaveSTLFile(fileName, aSetting);
}

public string outputInfo(ICollection<Element> ducts, string s)
{
    foreach (Element elem in ducts)
    {
        s += elem.Id.ToString() + ",";
        string width = elem.get_Parameter("Width").AsValueString();
        string height = elem.get_Parameter("Height").AsValueString();
        string length = elem.get_Parameter("Length").AsValueString();
        s += width + ",";
        s += length + ",";
        s += height + ",";
        // location
        Duct duct = (Duct)elem;
        LocationCurve ductCurve = (LocationCurve)duct.Location;
        XYZ pt1 = ductCurve.Curve.get_EndPoint(0);
        XYZ pt2 = ductCurve.Curve.get_EndPoint(1);
        s += pt1.X + "," + pt1.Y + "," + pt1.Z + ",";
        s += pt2.X + "," + pt2.Y + "," + pt2.Z + "\n";
    }
}

```

```

        return s;
    }

    public void exportImg(string path)
    {
        ImageExportOptions m_exportOptions = new ImageExportOptions();
        m_exportOptions.ExportRange = ExportRange.VisibleRegionOfCurrentView;
        m_exportOptions.ZoomType = ZoomFitType.FitToPage;
        m_exportOptions.FilePath = path;
        m_exportOptions.PixelSize = 4000;
        document.ExportImage(m_exportOptions);
    }

    public void generateLocationDiscrepancy()
    {
        ICollection<ElementId> allComponents = GetAllModelElementIds();
        Random random = new Random();
        Double X = -3 + 6 * random.NextDouble();
        Double Y = -3 + 6 * random.NextDouble();
        Double Z = -3 + 6 * random.NextDouble();
        XYZ direction = new XYZ(X,Y,Z);
        ElementTransformUtils.MoveElements(document, allComponents,
direction);
    }

    public void ShowElementList(IList<Element> elems, string header)
    {
        string s = " - Class - Category - Name (or Family: Type Name) - Id -
\r\n";
        foreach (Element e in elems)
        {
            s += ElementToString(e);
        }
        TaskDialog.Show(header + "(" + elems.Count.ToString() + "):", s);
    }

    // Helper function: summarize an element information as a line of text,
    // which is composed of: class, category, name and id.
    // name will be "Family: Type" if a given element is ElementType.
    // Intended for quick viewing of list of element, for example.

    public string ElementToString(Element e)
    {
        if (e == null)
        {
            return "none";
        }

        string name = "";

        if (e is ElementType)
        {
            Parameter param = e.get_Parameter(
                BuiltInParameter.SYMBOL_FAMILY_AND_TYPE_NAMES_PARAM);
            if (param != null)
            {
                name = param.AsString();
            }
        }
    }

```

```

        }
    }
    else
    {
        name = e.Name;
    }
    return e.GetType().Name + "; "
        + e.Category.Name + "; "
        + name + "; "
        + e.Id.IntegerValue.ToString() + "\r\n";
}

public void outputElementInfo(Element e)
{
    string s = "";
    s += s;

}
// get the selected element parameters
public void showBasicElementInfo(Element elem)
{
    string s = "You Picked:" + "\n";
    s += " Class name = " + elem.GetType().Name + "\n";
    s += " Category = " + elem.Category.Name + "\n";
    s += " Element id = " + elem.Id.ToString() + "\n" + "\n";
    // and, check its type info.
    ElementId elemTypeId = elem.GetTypeId();
    ElementType elemType = (ElementType)document.get_Element(elemTypeId);
    s += "Its ElementType:" + "\n";
    s += " Class name = " + elemType.GetType().Name + "\n";
    s += " Category = " + elemType.Category.Name + "\n";
    s += " Element type id = " + elemType.Id.ToString() + "\n";
    TaskDialog.Show("Basic Element Info", s);
}

public void ShowParameters(Element elem, string header)
{
    ParameterSet paramSet = elem.Parameters;
    string s = string.Empty;
    foreach (Parameter param in paramSet)
    {
        string name = param.Definition.Name;
        string val = ParameterToString(param);
        s += name + " = " + val + "\n";
    }
    TaskDialog.Show(header, s);
}

public ElementSet GetExportedElements()
{
    //get ducts
    ElementSet set = new ElementSet();
    ElementSet elems = app.Create.NewElementSet();
    var ductTypeCollector1 = new FilteredElementCollector(document);
    ductTypeCollector1.WherePasses(new ElementClassFilter(typeof(Duct)));
    ICollection<Element> ducts = ductTypeCollector1.ToElements();
}

```

```

        foreach (Element a in ducts)
        {
            set.Insert(a);
        }

        //get duct fittings
        ElementClassFilter familyInstanceFilter = new
ElementClassFilter(typeof(FamilyInstance));
        ElementCategoryFilter ductFittingCategoryFilter = new
ElementCategoryFilter(BuiltInCategory.OST_DuctFitting);
        LogicalAndFilter ductFittingInstancesFilter = new
LogicalAndFilter(familyInstanceFilter, ductFittingCategoryFilter);
        FilteredElementCollector collector = new
FilteredElementCollector(document);
        ICollection<Element> ductfittings =
collector.WherePasses(ductFittingInstancesFilter).ToElements();
        foreach (Element a in ductfittings)
        {
            if (a.GetType().Equals("Experiment"))
                set.Insert(a);
        }
        //TaskDialog.Show("duct no", set.Size.ToString());
        return set;
    }

    // The function is used to get all the model Element -> for the moving
function
    public ICollection<ElementId> GetAllModelElementIds()
    {
        //get ducts
        ElementSet elems = app.Create.NewElementSet();
        var ductTypeCollector1 = new FilteredElementCollector(document);
        ductTypeCollector1.WherePasses(new ElementClassFilter(typeof(Duct)));

        ICollection<ElementId> ducts = ductTypeCollector1.ToElementIds();

        //get duct fittings
        ElementClassFilter familyInstanceFilter = new
ElementClassFilter(typeof(FamilyInstance));
        ElementCategoryFilter ductFittingCategoryFilter = new
ElementCategoryFilter(BuiltInCategory.OST_DuctFitting);
        LogicalAndFilter ductFittingInstancesFilter = new
LogicalAndFilter(familyInstanceFilter, ductFittingCategoryFilter);
        FilteredElementCollector collector = new
FilteredElementCollector(document);
        ICollection<ElementId> ductfittings =
collector.WherePasses(ductFittingInstancesFilter).ToElementIds();
        foreach (ElementId a in ductfittings)
        {
            ducts.Add(a);
        }
        //TaskDialog.Show("duct no", ducts.Count.ToString());
        return ducts;
    }

    public ElementSet GetAllModelElements()

```

```

{
    //get ducts
    ElementSet set = new ElementSet();
    ElementSet elems = app.Create.NewElementSet();
    var ductTypeCollector1 = new FilteredElementCollector(document);
    ductTypeCollector1.WherePasses(new ElementClassFilter(typeof(Duct)));
    ICollection<Element> ducts = ductTypeCollector1.ToElements();
    foreach (Element a in ducts)
    {
        set.Insert(a);
    }

    //get duct fittings
    ElementClassFilter familyInstanceFilter = new
ElementClassFilter(typeof(FamilyInstance));
    ElementCategoryFilter ductFittingCategoryFilter = new
ElementCategoryFilter(BuiltInCategory.OST_DuctFitting);
    LogicalAndFilter ductFittingInstancesFilter = new
LogicalAndFilter(familyInstanceFilter, ductFittingCategoryFilter);
    FilteredElementCollector collector = new
FilteredElementCollector(document);
    ICollection<Element> ductfittings =
collector.WherePasses(ductFittingInstancesFilter).ToElements();
    foreach (Element a in ductfittings)
    {
        set.Insert(a);
    }
    //TaskDialog.Show("duct no", set.Size.ToString());
    return set;
}

public static string ParameterToString(Parameter param)
{
    string val = "none";

    if (param == null)
    {
        return val;
    }

    // to get to the parameter value, we need to pause it depending
    // on its strage type
    switch (param.StorageType)
    {
        case StorageType.Double:
            double dVal = param.AsDouble();
            val = dVal.ToString();
            break;

        case StorageType.Integer:
            int iVal = param.AsInteger();
            val = iVal.ToString();
            break;

        case StorageType.String:
            string sVal = param.AsString();

```

```

        val = sVal;
        break;

    case StorageType.ElementId:
        ElementId idVal = param.AsElementId();
        val = idVal.IntegerValue.ToString();
        break;

    case StorageType.None:
        break;

    default:
        break;
}

return val;
}

public void IdentifyElement(Element elem)
{
    // An instance of a system family has a designated class.
    // You can use it identify the type of element.
    // e.g., walls, floors, roofs.
    //
    string s = "";

    if (elem is Wall)
    {
        s = "Wall";
    }
    else if (elem is Floor)
    {
        s = "Floor";
    }
    else if (elem is RoofBase)
    {
        s = "Roof";
    }
    else if (elem is Duct)
    {
        s = "Duct";
    }

    else if (elem is FamilyInstance)
    {
        // An instance of a component family is all FamilyInstance.
        // We'll need to further check its category.
        // e.g., Doors, Windows, Furnitures.
        if (elem.Category.Id.IntegerValue ==
            (int)BuiltInCategory.OST_Doors)
        {
            s = "Door";
        }
        else if (elem.Category.Id.IntegerValue ==
            (int)BuiltInCategory.OST_Windows)
        {
            s = "Window";
        }
    }
}

```

```

    }
    else if (elem.Category.Id.IntegerValue ==
(int)BuiltInCategory.OST_Furniture)
    {
        s = "Furniture";
    }

    else if (elem.Category.Id.IntegerValue ==
(int)BuiltInCategory.OST_DuctFitting)
    {
        s = "Duct Fitting";
    }
    else
    {
        // e.g. Plant
        s = "Component family instance";
    }
}
else if (elem is HostObject)
{
    // check the base class. e.g., CeilingAndFloor.
    s = "System family instance";
}
else
{
    s = "Other";
}

s = "You have picked: " + s;

// show it.
TaskDialog.Show("Identify Element", s);
}

public void ModifyElementWidth(Element elem, double n, Transaction trans)
{
    Parameter param = elem.get_Parameter("Width");
    //double currentWidth = param.AsDouble();
    if (param != null)
    {
        trans.Start("ModifyWidth");
        param.Set(n);
        trans.Commit();
    }
}

public void ModifyElementHeight(Element elem, double n, Transaction trans)
{
    Parameter param = elem.get_Parameter("Height");
    if (param != null)
    {
        trans.Start("ModifyHeigh");
        param.Set(n);
        trans.Commit();
    }
}
}

```

```

public void ModifyElementLengthWithAttach(Element elem, Transaction trans)
{
    try
    {
        trans.Start("Modify_Length");
        Duct duct = (Duct)elem;
        LocationCurve ductCurve = (LocationCurve)duct.Location;
        XYZ pt1 = ductCurve.Curve.get_EndPoint(0);
        XYZ pt2 = ductCurve.Curve.get_EndPoint(1);
        double Xdiff = System.Math.Abs(pt1.X - pt2.X);
        double Ydiff = System.Math.Abs(pt1.Y - pt2.Y);
        double Zdiff = System.Math.Abs(pt1.Z - pt2.Z);
        //string s = "X difference " + Xdiff + "\n" + "Y difference " +
Ydiff + "\n" + "Z difference " + Zdiff + "\n";
        //TaskDialog.Show("Checking", s);
        double X = 0, Y = 0, Z = 0;
        Random random = new Random();
        double change = 2*random.NextDouble();
        int flag = random.Next(0, 2);
        if (flag == 0)
            change = change * 1;
        else
            change = change * -1;
        if (Xdiff >= 0.001)
            X = change;
        if (Ydiff >= 0.001)
            Y = change;
        if (Zdiff >= 0.001)
            Z = change;
        XYZ v = new XYZ(X, Y, Z);
        move(elem, v);
        trans.Commit();

    }
    catch (Exception e)
    {
        TaskDialog.Show("error", e.Message);
    }
}

public void ModifyElmentLength(Element elem, double value)
{
    Duct duct = (Duct)elem;
    LocationCurve ductCurve = (LocationCurve)duct.Location;
    XYZ pt1 = ductCurve.Curve.get_EndPoint(0);
    XYZ pt2 = ductCurve.Curve.get_EndPoint(1);

    XYZ newPt1 = new XYZ(pt1.X - value, pt1.Y, pt1.Z);
    XYZ newPt2 = new XYZ(pt2.X - value, pt2.Y, pt2.Z);

    // create a new line bound.
    Line newDuctLine = app.Create.NewLineBound(newPt1, pt2);

    // finally change the curve.
    ductCurve.Curve = newDuctLine;

    // message to the user.

```



```

        String msg = "Location: start point moved " + value + "in X-direction"
+ "\n";
        TaskDialog.Show("Revit Intro Lab", msg);
    }

    public void ModifyDuctfittingLength(Element elem, double value)
    {
        FamilyInstance d = (FamilyInstance)elem;
        LocationCurve dCurve = (LocationCurve)d.Location;
        XYZ pt1 = dCurve.Curve.get_EndPoint(0);
        XYZ pt2 = dCurve.Curve.get_EndPoint(1);

        XYZ newPt1 = new XYZ(pt1.X - value, pt1.Y, pt1.Z);
        XYZ newPt2 = new XYZ(pt2.X - value, pt2.Y, pt2.Z);

        // create a new line bound.
        Line newDuctLine = app.Create.NewLineBound(newPt1, pt2);

        // finally change the curve.
        dCurve.Curve = newDuctLine;

        // message to the user.
        String msg = "Location: start point moved " + value + "in X-direction"
+ "\n";
        TaskDialog.Show("Revit Intro Lab", msg);
    }

    public void move(Element e, XYZ v)
    {
        ElementTransformUtils.MoveElement(document, e.Id, v);
    }
}

```

## REFERENCES

- Adan, A., and Huber, D. (2011). "3D reconstruction of interior wall surfaces under occlusion and clutter." *Proc., International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, IEEE, Hangzhou, China, 275-281.
- Adan, A., Xiong, X., Akinci, B., and Huber, D. (2011). "Automatic creation of semantically rich 3D building models from laser scanner data." *Proc., International Symposium on Automation and Robotics in Construction (ISARC)*, Seoul, Korea, 342-347.
- Akinci, B., Boukamp, F., Gordon, C., Huber, D., Lyons, C., and Park, K. (2006). "A formalism for utilization of sensor systems and integrated project models for active construction quality control." *Automation in Construction*, 15(2), 124-138.
- Anand, A., Koppula, H. S., Joachims, T., and Saxena, A. (2013). "Contextually guided semantic labeling and search for three-dimensional point clouds." *The International Journal of Robotics Research*, 32(1), 19-34.
- Anderson, D. C., and Chang, T. C. (1990). "Geometric reasoning in feature-based design and process planning." *Computers & Graphics*, 14(2), 225-235.
- Anguelov, D., Taskarf, B., Chatalbashev, V., Koller, D., Gupta, D., Heitz, G., and Ng, A. (2005). "Discriminative learning of markov random fields for segmentation of 3d scan data." *Proc., IEEE Computer Society Conference on Computer Vision and Pattern Recognition* IEEE, San Diego, CA, 169-176.
- Anil, E., Akinci, B., and Huber, D. (2011). "Representation requirements of as-is building information models generated from laser scanned point cloud data." *Proc., International Symposium on Automation and Robotics in Construction (ISARC)* Seoul, Korea.
- Banta, J. E., Wong, L., Dumont, C., and Abidi, M. A. (2000). "A next-best-view system for autonomous 3-D object reconstruction." *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 30(5), 589-598.
- Beetz, J., Van Leeuwen, J., and De Vries, B. (2006). "Towards a topological reasoning service for IFC-based Building Information Models in a Semantic Web Context." *Proc., 11th International Conf. on Computing in Civil and Building Engineering ICCCBE-XI* Montreal, Canada, 3426-3435.
- Belongie, S., Malik, J., and Puzicha, J. (2001). "Shape context: A new descriptor for shape matching and object recognition." *Advances in neural information processing systems*, 831-837.
- Belongie, S., Malik, J., and Puzicha, J. (2002). "Shape matching and object recognition using shape contexts." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4), 509-522.
- Ben Hmida, H., Cruz, C., Nicolle, C., and Boochs, F. (2011). "Toward the Automatic Generation of a Semantic VRML Model from Unorganized 3D Point Clouds." *Proc., IARIA Conference*, Venice/Mestre, Italy, 35-41.

- Biasotti, S., Marini, S., Mortara, M., Patane, G., Spagnuolo, M., and Falcidieno, B. (2003). "3D shape matching through topological structures." *Proc., Discrete Geometry for Computer Imagery*, Springer, Berlin Heidelberg, 194-203.
- Bimbo, A. D., and Pala, P. (2006). "Content-based retrieval of 3D models." *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)* 2(1), 20-43.
- Boehler, W., Vicent, M. B., and Marbs, A. (2003). "Investigating laser scanner accuracy." *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34(Part 5), 696-701.
- Boehnen, C., and Flynn, P. (2005). "Accuracy of 3D scanning technologies in a face scanning scenario." *Proc., Fifth International Conference on 3-D Digital Imaging and Modeling IEEE*, Ottawa, Ontario, Canada, 310-317.
- Borrmann, A., and Beetz, J. (2010). "Towards spatial reasoning on building information models." *Proc., 8th European Conference on Product and Process Modeling (ECPPM)*, Taylor & Francis Group, Cork, Ireland.
- Borrmann, A., and Rank, E. (2009). "Topological analysis of 3D building models using a spatial query language." *Advanced Engineering Informatics*, 23(4), 370-385.
- Bosché, F. (2009). "Automated recognition of 3D CAD model objects and calculation of as-built dimensions for dimensional compliance control in construction." *Adv. Eng. Informatics*, 24, 107-118.
- Bosché, F., Guillemet, A., Turkan, Y., Haas, C., and Haas, R. "Tracking the Built Status of MEP Works: Assessing the Value of a Scan-vs-BIM System." *Journal of Computing in Civil Engineering*
- Bosche, F., and Haas, C. (2008). "Automated retrieval of 3D CAD model objects in construction range images." *Automation in Construction*, 17(4), 499-512.
- Bosche, F., and Haas, C. T. (2008). "Automated retrieval of project three-dimensional CAD objects in range point clouds to support automated dimensional QA/QC." *Information Technologies in Construction*, 13, 71-85.
- Brenner, C. (2005). "Building reconstruction from images and laser scanning." *International Journal of Applied Earth Observation and Geoinformation*, 6(3), 187-198.
- BuildingSMART (2012). "IFC2x Edition 3 Technical Corrigendum 1." <<http://buildingsmart-tech.org/ifc/IFC2x3/TC1/html/index.htm%3E>. (April 2012).
- buildingSMART (2012). "The IFC Specification." <<http://www.buildingsmart-tech.org/specifications%3E>. (August, 2012).
- Bustos, B., Keim, D., Saupe, D., and Schreck, T. (2007). "Content-based 3D object retrieval." *Computer Graphics and Applications, IEEE*, 27(4), 22-27.
- Caetano, T. S., McAuley, J. J., Cheng, L., Le, Q. V., and Smola, A. J. (2009). "Learning graph matching." *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(6), 1048-1058.
- Campbell, R. J., and Flynn, P. J. (2001). "A survey of free-form object representation and recognition techniques." *Computer Vision and Image Understanding*, 81(2), 166-210.

- Cantzler, H., Fisher, R. B., and Devy, M. (2002). "Improving architectural 3D reconstruction by plane and edge constraining." *Proc., British Machine Vision Conference*, Citeseer, Cardiff, UK, 43-52.
- Cha, S.-H. (2007). "Comprehensive survey on distance/similarity measures between probability density functions." *International Journal of Mathematical Models and Methods in Applied Science*, 1(2), 300-307.
- Chao, M. W., Lin, C. H., Chang, C. C., and Lee, T. Y. (2011). "A graph - based shape matching scheme for 3D articulated objects." *Computer Animation and Virtual Worlds*, 22(2 - 3), 295-305.
- Chen, D. Y., Tian, X. P., Shen, Y. T., and Ouhyoung, M. (2003). "On visual similarity based 3D model retrieval." *Computer graphics forum*, 22(3), 223-232.
- Chen, H., and Bhanu, B. (2007). "3D free-form object recognition in range images using local surface patches." *Pattern Recognition Letters*, 28(10), 1252-1262.
- Chen, Y., and Medioni, G. (1992). "Object modelling by registration of multiple range images." *Image and vision computing*, 10(3), 145-155.
- Choi, N., Son, H., Kim, C., Kim, C., and Kim, H. (2009). "Rapid 3D object recognition for automatic project progress monitoring using a stereo vision system." *Proc., The 25th International Symposium on Automation and Robotics in Construction*.
- Chua, C. S., and Jarvis, R. (1997). "Point signatures: A new representation for 3d object recognition." *International Journal of Computer Vision*, 25(1), 63-85.
- Clark, J., and Robson, S. (2004). "Accuracy of measurements made with a Cyrax 2500 laser scanner against surfaces of known colour." *Survey Review*, 37(294), 626-638.
- Clementini, E., and Di Felice, P. (1996). "A model for representing topological relationships between complex geometric features in spatial databases." *Information sciences*, 90(1), 121-136.
- Connolly, C. (1985). "The determination of next best views." *Proc., IEEE International Conference on Robotics and Automation*, IEEE, St. Louis, Missouri, 432-435.
- Conte, D., Foggia, P., Sansone, C., and Vento, M. (2004). "Thirty years of graph matching in pattern recognition." *International journal of pattern recognition and artificial intelligence*, 18(03), 265-298.
- Cruz, C. (2007). "Ontology-driven 3D reconstruction of architectural objects." *VISAPP (Special Sessions)*, 47-54.
- Cybenko, G., Bhasin, A., and Cohen, K. D. (1997). "Pattern recognition of 3D CAD objects: Towards an electronic yellow pages of mechanical parts." *International Journal of Smart Engineering System Design*, 1(1), 1-13.
- de Haan, G. (2009). "Scalable visualization of massive point clouds." *Nederlandse Commissie voor Geodesie KNAW*, 49, 59.
- Douillard, B., Fox, D., Ramos, F., and Durrant-Whyte, H. (2011). "Classification and semantic mapping of urban environments." *The International Journal of Robotics Research*, 30(1), 5-32.
- Douillard, B., Underwood, J., Vlaskine, V., Quadros, A., and Singh, S. (2010). "A pipeline for the segmentation and classification of 3D point clouds." *Proc., The International Symposium on Experimental Robotics (ISER)* Delhi, India.

- Duan, Y., Cruz, C., and Nicolle, C. (2009). "Propose Semantic Formalization for 3D Reconstruction of Architectural Objects." *Proc., 7th ACIS International Conference on Software Engineering Research, Management and Applications*, IEEE, Haikou, China, 78-85.
- Duan, Y., Cruz, C., and Nicolle, C. (2010). "Architectural reconstruction of 3D building objects through semantic knowledge management." *Proc., 11th ACIS International Conference on Software Engineering Artificial Intelligence Networking and Parallel/Distributed Computing (SNPD)*, IEEE, London, United Kingdom, 261-266.
- Eastman, C. M. (1999). *Building product models: computer environments supporting design and construction*, CRC press, Boca Raton, FL.
- Eich, M., Dabrowska, M., and Kirchner, F. (2010). "Semantic labeling: Classification of 3d entities based on spatial feature descriptors." *Proc., IEEE International Conference on Robotics and Automation, ICRA* Anchorage, Alaska.
- Elberink, S. O., and Vosselman, G. (2009). "Building reconstruction by target based graph matching on incomplete laser data: analysis and limitations." *Sensors*, 9(8), 6101-6118.
- Fischler, M. A., and Bolles, R. C. (1981). "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography." *Communications of the ACM*, 24(6), 381-395.
- Fröhlich, C., and Mettenleiter, M. (2004). "Terrestrial laser scanning—new perspectives in 3D surveying." *International archives of photogrammetry, remote sensing and spatial information sciences*, 36(Part 8), W2.
- Frome, A., Huber, D., Kolluri, R., Bülow, T., and Malik, J. (2004). "Recognizing Objects in Range Data Using Regional Point Descriptors  
Computer Vision - ECCV 2004." T. Pajdla, and J. Matas, eds., Springer Berlin / Heidelberg, 224-237.
- Funkhouser, T., and Kazhdan, M. (2004). "Shape-based retrieval and analysis of 3D models." *Proc., ACM SIGGRAPH 2004 Course Notes*, ACM, Los Angeles, CA, USA, 16.
- Funkhouser, T., Min, P., Kazhdan, M., Chen, J., Halderman, A., Dobkin, D., and Jacobs, D. (2003). "A search engine for 3D models." *ACM Transactions on Graphics (TOG)*, 22(1), 83-105.
- Gilsinn, D., Cheok, G. S., Witzgall, C., and Lytle, A. (2005). *Construction object identification from LADAR scans: an experimental study using I-beams*, US Department of Commerce, Technology Administration, National Institute of Standards and Technology, Washington DC.
- Golovinskiy, A., and Funkhouser, T. (2009). "Min-cut based segmentation of point clouds." *Proc., IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops)*, Ieee, Barcelona, Spain, 39-46.
- Golovinskiy, A., Kim, V. G., and Funkhouser, T. (2009). "Shape-based recognition of 3d point clouds in urban environments." *Proc., IEEE 12th International Conference on Computer Vision*, IEEE, Kyoto, Japan, 2154-2161.
- Golparvar-Fard, M., Bohn, J., Teizer, J., Savarese, S., and Pena-Mora, F. (2011). "Evaluation of image-based modeling and laser scanning accuracy for

- emerging automated performance monitoring techniques." *Automation in Construction*, 20(8), 1143-1155.
- Golparvar-Fard, M., Pena-Mora, F., and Savarese, S. (2011). "Monitoring changes of 3D building elements from unordered photo collections." *Proc., 2011 IEEE International Conference on Computer Vision Workshops*, IEEE, Barcelona, Spain, 249-256.
- Gong, J., and Caldas, C. H. (2008). "Data processing for real-time construction site spatial modeling." *Automation in Construction*, 17(5), 526-535.
- Hilaga, M., Shinagawa, Y., Kohmura, T., and Kunii, T. L. (2001). "Topology matching for fully automatic similarity estimation of 3D shapes." *Proc., The 28th annual conference on Computer graphics and interactive techniques*, ACM, Los Angeles, CA, USA, 203-212.
- Hofman, I., and Jarvis, R. (2000). "Object Recognition Via Attributed Graph Matching." *Proc., Australasian Conference on Robotics and Automation, ACRAMelbourne*, Australia.
- Hripcsak, G., and Rothschild, A. S. (2005). "Agreement, the f-measure, and reliability in information retrieval." *Journal of the American Medical Informatics Association*, 12(3), 296-298.
- Huang, H., and Brenner, C. (2011). "Rule-based roof plane detection and segmentation from laser point clouds." *Proc., Joint Urban Remote Sensing Event (JURSE)*, IEEE, Munich, Germany, 293-296.
- Huang, Y. J., Powers, R., and Montelione, G. T. (2005). "Protein NMR recall, precision, and F-measure scores (RPF scores): structure quality assessment measures based on information retrieval statistics." *Journal of the American Chemical Society*, 127(6), 1665-1674.
- Huber, D., Akinci, B., Oliver, A., Anil, E., Okorn, B. E., and Xiong, X. (2011). "Methods for automatically modeling and representing As-built Building Information Models." *Proc., The NSF CMMI Research Innovation Conference*, Atlanta, GA.
- Huber, D., Akinci, B., Tang, P., Adan, A., Okorn, B., and Xiong, X. (2010). "Using laser scanners for modeling and analysis in architecture, engineering, and construction." *Proc., 44th Annual Conference on Information Sciences and Systems (CISS)*, IEEE, Princeton, New Jersey, USA, 1-6.
- Huber, D., Kapuria, A., Donamukkala, R., and Hebert, M. (2004). "Parts-based 3d object classification." *Proc., IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE, Washington, DC 2004 II-82-II-89 Vol. 82.
- Ip, C. Y., Lapadat, D., Sieger, L., and Regli, W. C. (2002). "Using shape distributions to compare solid models." *Proc., The seventh ACM symposium on Solid modeling and applications*, ACM, Saarbrücken, Germany, 273-280.
- Iyer, N., Jayanti, S., Lou, K., Kalyanaraman, Y., and Ramani, K. (2005). "Three-dimensional shape searching: state-of-the-art review and future trends." *Computer-Aided Design*, 37(5), 509-530.
- Jiantao, P., Yi, L., Guyu, X., Hongbin, Z., Weibin, L., and Uehara, Y. (2004). "3D model retrieval based on 2D slice similarity measurements." *Proc., 2nd*

- International Symposium on 3D Data Processing, Visualization and Transmission*, IEEE, Thessaloniki, Greece, 95-101.
- Johnson, A. E. (1997). "Spin-images: A representation for 3-D surface matching." PhD thesis, Carnegie Mellon University.
- Johnson, A. E., and Hebert, M. (1999). "Using spin images for efficient object recognition in cluttered 3D scenes." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5), 433-449.
- Kersten, T. P., Sternberg, H., and Mechelke, K. (2005). "Investigations into the accuracy behaviour of the terrestrial laser scanning system Mensi GS100." *Proc., Conference in Optical 3D Measurement Techniques* Vienna, Austria, 122-131.
- Kohavi, R. (1995). "A study of cross-validation and bootstrap for accuracy estimation and model selection." *Proc., International Joint Conference on Artificial Intelligence (IJCAI)* Montreal, Quebec, Canada, 1137-1145.
- Koppula, H. S., Anand, A., Joachims, T., and Saxena, A. (2011). "Semantic labeling of 3d point clouds for indoor scenes." *Proc., Advances in Neural Information Processing Systems* Granada, Spain, 244-252.
- Körtgen, M., Park, G. J., Novotni, M., and Klein, R. (2003). "3D shape matching with 3D shape contexts." *Proc., The 7th central European seminar on computer graphics*, Citeseer, Budmerice, Slovakia, 5-17.
- Lai, K., and Fox, D. (2010). "Object Recognition in 3D Point Clouds Using Web Data and Domain Adaptation." *The International Journal of Robotics Research*, 29(8), 1019-1037.
- Liu, X., Eybpoosh, M., and Akinci, B. "Developing As-Built Building Information Model Using Construction Process History Captured by a Laser Scanner and a Camera." *Construction Research Congress 2012* West Lafayette, Indiana, 1232-1241.
- Liu, Y., Zhang, D., Lu, G., and Ma, W. Y. (2007). "A survey of content-based image retrieval with high-level semantics." *Pattern Recognition*, 40(1), 262-282.
- Mahmoudi, M., and Sapiro, G. (2009). "Three-dimensional point cloud recognition via distributions of geometric distances." *Graphical Models*, 71(1), 22-31.
- Massios, N. A., and Fisher, R. B. (1998). "A best next view selection algorithm incorporating a quality criterion." *Proc., British Machine Vision Conference* Southampton, UK, 1-10.
- Maver, J., and Bajcsy, R. (1993). "Occlusions as a guide for planning the next view." *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(5), 417-433.
- Mechelke, K., Kersten, T. P., and Lindstaedt, M. (2007). "Comparative investigations into the accuracy behaviour of the new generation of terrestrial laser scanning systems." *Proc., Optical 3-D Measurement Techniques VIII* Gruen and Kahmen, Zurich, 319-327.
- Min, P., Chen, J., and Funkhouser, T. (2002). "A 2D sketch interface for a 3D model search engine." *Proc., ACM SIGGRAPH 2002 conference abstracts and applications*, ACM, San Antonio, Texas, USA, 138-138.
- Mitra, N. J., Gelfand, N., Pottmann, H., and Guibas, L. (2004). "Registration of point cloud data from a geometric optimization perspective." *Proc., 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, ACM, Nice, France, 22-31.

- Mozos, O. M., Stachniss, C., and Burgard, W. (2005). "Supervised learning of places from range data using adaboost." *Proc., IEEE International Conference on Robotics and Automation, ICRA*, IEEE, Barcelona, France, 1730-1735.
- Nepal, M. P., Zhang, J., Webster, A., Staub-French, S., Pottinger, R., and Lawrence, M. (2009). "Querying IFC-based building information models to support construction management functions." *Proc., Construction Research Congress "Building a Sustainable Future"* Seattle, Washington, 506-515.
- Novotni, M., and Klein, R. (2003). "3D Zernike descriptors for content based shape retrieval." *Proc., Proceedings of the eighth ACM symposium on Solid modeling and applications*, ACM, Seattle, Washington, USA, 216-225.
- Nüchter, A., and Hertzberg, J. (2008). "Towards semantic maps for mobile robots." *Robotics and Autonomous Systems*, 56(11), 915-926.
- Nuchter, A., Surmann, H., and Hertzberg, J. (2003). "Automatic model refinement for 3D reconstruction with mobile robots." *Proc., Fourth International Conference on 3-D Digital Imaging and Modeling*, IEEE, Banff, Canada, 394-401.
- Nüchter, A., Surmann, H., Lingemann, K., and Hertzberg, J. (2003). "Semantic Scene Analysis of Scanned 3D Indoor Environments." *Proc., Semantic Scene Analysis of Scanned 3D Indoor Environments* Munich, DE, 215-221.
- Okorn, B., Xiong, X., Akinci, B., and Huber, D. (2010). "Toward automated modeling of floor plans." *Proc., The Symposium on 3D Data Processing, Visualization and Transmission* Paris, France.
- Osada, R., Funkhouser, T., Chazelle, B., and Dobkin, D. (2001). "Matching 3D models with shape distributions." *Proc., SMI 2001 International Conference on Shape Modeling and Applications* IEEE, Genoa, Italy, 154-166.
- Osada, R., Funkhouser, T., Chazelle, B., and Dobkin, D. (2002). "Shape distributions." *ACM Transactions on Graphics (TOG)*, 21(4), 807-832.
- Oude Elberink, S., and Vosselman, G. (2011). "Quality analysis on 3D building models reconstructed from airborne laser scanning data." *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(2), 157-165.
- Pangercic, D., Pitzer, B., Tenorth, M., and Beetz, M. (2012). "Semantic object maps for robotic housework-representation, acquisition and use." *Proc., 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Vilamoura, Algarve, Portugal, 4644-4651.
- Paquet, E., Rioux, M., Murching, A., Naveen, T., and Tabatabai, A. (2000). "Description of shape information for 2-D and 3-D objects." *Signal Processing: Image Communication*, 16(1), 103-122.
- Park, H., Lee, H., Adeli, H., and Lee, I. (2007). "A new approach for health monitoring of structures: terrestrial laser scanning." *Computer - Aided Civil and Infrastructure Engineering*, 22(1), 19-30.
- Pito, R. (1996). "A sensor-based solution to the "next best view" problem." *Proc., 13th International Conference on Pattern Recognition*, IEEE, Vienna, Austria, 941-945.
- Pito, R. (1999). "A solution to the next best view problem for automated surface acquisition." *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(10), 1016-1030.



- Posner, I., Cummins, M., and Newman, P. (2009). "A generative framework for fast urban labeling using spatial and temporal context." *Autonomous Robots*, 26(2-3), 153-170.
- Posner, I., Schroeter, D., and Newman, P. (2008). "Using scene similarity for place labelling." *Proc., Experimental Robotics*, Springer, Berlin Heidelberg, 85-98.
- Pu, S., and Vosselman, G. (2006). "Automatic extraction of building features from terrestrial laser scanning." *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(5), 25-27.
- Pu, S., and Vosselman, G. (2009). "Knowledge based reconstruction of building models from terrestrial laser scanning data." *ISPRS journal of Photogrammetry and Remote Sensing*, 64(6), 575-584.
- Rabbani, T., Dijkman, S., van den Heuvel, F., and Vosselman, G. (2007). "An integrated approach for modelling and global registration of point clouds." *ISPRS journal of Photogrammetry and Remote Sensing*, 61(6), 355-370.
- Rabbani, T., van Den Heuvel, F., and Vosselmann, G. (2006). "Segmentation of point clouds using smoothness constraint." *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(5), 248-253.
- Ravada, S., Horhammer, M., Kazar, B. M., and Spatial, O. (2010). "Point Cloud: Storage, Loading, and Visualization." <[http://www.cigi.illinois.edu/cybergis/docs/Kazar\\_Position\\_Paper.pdf%3E](http://www.cigi.illinois.edu/cybergis/docs/Kazar_Position_Paper.pdf%3E) . (2013).
- Refaeilzadeh, P., Tang, L., and Liu, H. (2009). "Cross-validation." *Encyclopedia of Database Systems*, Springer, 532-538.
- Requicha, A. G. (1980). "Representations for rigid solids: Theory, methods, and systems." *ACM Computing Surveys (CSUR)*, 12(4), 437-464.
- Roggero, M. (2002). "Object segmentation with region growing and principal component analysis." *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34(3/A), 289-294.
- Rusu, R. B., Blodow, N., and Beetz, M. (2009). "Fast point feature histograms (fpfh) for 3d registration." *Proc., IEEE International Conference on Robotics and Automation (ICRA'09)* IEEE, Anchorage, USA, 3212-3217.
- Rusu, R. B., Bradski, G., Thibaux, R., and Hsu, J. (2010). "Fast 3d recognition and pose using the viewpoint feature histogram." *Proc., IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Taipei, Taiwan, 2155-2162.
- Rusu, R. B., Marton, Z. C., Blodow, N., Dolha, M., and Beetz, M. (2008). "Towards 3D Point cloud based object maps for household environments." *Robotics and Autonomous Systems*, 56(11), 927-941.
- Rusu, R. B., Marton, Z. C., Blodow, N., Holzbach, A., and Beetz, M. (2009). "Model-based and learned semantic object labeling in 3D point cloud maps of kitchen environments." *Proc., IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009*, IEEE, St. Louis, MO, USA, 3601-3608.
- Salomons, O. W., van Houten, F. J., and Kals, H. (1993). "Review of research in feature-based design." *Journal of Manufacturing Systems*, 12(2), 113-132.

- Salvi, J., Matabosch, C., Fofi, D., and Forest, J. (2007). "A review of recent range image registration methods with accuracy evaluation." *Image and Vision Computing*, 25(5), 578-596.
- Schnabel, R., Wahl, R., and Klein, R. (2007). "Efficient RANSAC for Point - Cloud Shape Detection." *Computer Graphics Forum*, 26, 214-226.
- Schnabel, R., Wessel, R., Wahl, R., and Klein, R. (2008). "Shape recognition in 3d point-clouds." *Proc., Conference in Central Europe on Computer Graphics, Visualization and Computer Vision* Citeseer, Bory, Czech Republic.
- Schneider, M., and Behr, T. (2006). "Topological relationships between complex spatial objects." *ACM Transactions on Database Systems (TODS)*, 31(1), 39-81.
- Scott, W., Roth, G., and Rivest, J.-F. (2003). "View Planning for Automated 3D Object Reconstruction Inspection." *ACM Computing Surveys*, 35(1).
- Sharp, G. C., Lee, S. W., and Wehe, D. K. (2002). "ICP registration using invariant features." *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(1), 90-102.
- Son, H., and Kim, C. (2010). "3D structural component recognition and modeling method using color and 3D data for construction progress monitoring." *Automation in Construction*, 19(7), 844-854.
- Song, H., and Feng, H.-Y. (2009). "A progressive point cloud simplification algorithm with preserved sharp edge data." *The International Journal of Advanced Manufacturing Technology*, 45(5-6), 583-592.
- Staub-French, S., Fischer, M., Kunz, J., and Paulson, B. (2003). "An ontology for relating features with activities to calculate costs." *Journal of Computing in Civil Engineering*, 17(4), 243-254.
- Tang, P., Akinci, B., and Huber, D. (2009). "Quantification of edge loss of laser scanned data at spatial discontinuities." *Automation in Construction*, 18(8), 1070-1083.
- Tang, P., Anil, E. B., Akinci, B., and Huber, D. (2011). "Efficient and effective quality assessment of as-is building information models and 3D laser-scanned data." *Proc., ASCE International Workshop on Computing in Civil Engineering* Miami, Florida.
- Tang, P., Huber, D., and Akinci, B. (2010). "Characterization of laser scanners and algorithms for detecting flatness defects on concrete surfaces." *Journal of Computing in Civil Engineering*, 25(1), 31-42.
- Tang, P., Huber, D., Akinci, B., Lipman, R., and Lytle, A. (2010). "Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques." *Automation in Construction*, 19(7), 829-843.
- Tangelder, J. W., and Veltkamp, R. C. (2004). "A survey of content based 3D shape retrieval methods." *Proc., The International Conference on Shape Modeling and Applications*, IEEE, Genova, Italy., 145-156.
- Tarabanis, K. A., Allen, P. K., and Tsai, R. Y. (1995). "A survey of sensor planning in computer vision." *Robotics and Automation, IEEE Transactions on*, 11(1), 86-104.
- Teizer, J., Caldas, C. H., and Haas, C. T. (2007). "Real-time three-dimensional occupancy grid modeling for the detection and tracking of construction

- resources." *Journal of Construction Engineering and Management*, 133(11), 880-888.
- Triebel, R., Kersting, K., and Burgard, W. (2006). "Robust 3D scan point classification using associative Markov networks." *Proc., IEEE International Conference on Robotics and Automation*, IEEE, Orlando, Florida, USA, 2603-2608.
- Truong, H., Boochs, F., Habed, A., and Voisin, Y. (2012). "A knowledge-based approach to the automatic algorithm selection for 3D scene annotation." *Proc., 2012 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA)*, IEEE, Montreal, QC, Canada, 225-230.
- Ullrich, T., Settgast, V., and Fellner, D. W. (2008). "Semantic fitting and reconstruction." *Journal on Computing and Cultural Heritage (JOCCH)*, 1(2), 12.
- Valero, E., Adan, A., Huber, D., and Cerrada, C. (2011). "Detection, modeling, and classification of moldings for automated reverse engineering of buildings from 3D data." *Proc., 28th International Symposium on Automation and Robotics in Construction* Seoul, Korea.
- Van Leeuwen, J. P., and Wagter, H. (1997). "Architectural design-by-Features." *CAAD futures*, Springer, Kluwer, Dordrecht, 97-115.
- Van Treeck, C., and Rank, E. (2004). "Analysis of building structure and topology based on graph theory." *Proc., International conference on computing in Civil and Building engineering* Weimar, German.
- Verma, V., Kumar, R., and Hsu, S. (2006). "3d building detection and modeling from aerial lidar data." *Proc., IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE, New York, NY, USA, 2213-2220.
- Vosselman, G., and Dijkman, S. (2001). "3D building model reconstruction from point clouds and ground plans." *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences* 34(3/W4), 37-44.
- Wang, C. C., Wang, Y., and Yuen, M. M. (2003). "Feature based 3D garment design through 2D sketches." *Computer-Aided Design*, 35(7), 659-672.
- Wong, L., Dumont, C., and Abidi, M. (1999). "Next best view system in a 3D object modeling task." *Proc., 1999 IEEE International Symposium on Computational Intelligence in Robotics and Automation* IEEE, Monterey, CA, 306-311.
- Xiong, X., Adan, A., Akinci, B., and Huber, D. (2013). "Automatic creation of semantically rich 3D building models from laser scanner data." *Automation in Construction*, 31, 325-337.
- Xiong, X., and Huber, D. (2010). "Using context to create semantic 3D models of indoor environments." *Proc., British Machine Vision Conference (BMVC)*, Aberystwyth, UK, 1-11.
- Zaslavskiy, M., Bach, F., and Vert, J.-P. (2009). "A path following algorithm for the graph matching problem." *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(12), 2227-2242.
- Zhang, C., and Chen, T. (2001). "Indexing and retrieval of 3D models aided by active learning." *Proc., ninth ACM international conference on Multimedia*, ACM, Ottawa, Ontario, Canada, 615-616.
- Zhang, D., and Lu, G. (2004). "Review of shape representation and description techniques." *Pattern recognition*, 37(1), 1-19.

