From LiDAR point cloud towards digital twin city: Clustering city objects based on Gestalt principles

Fan Xue[1]*, Weisheng Lu[2], Zhe Chen[3], and Christopher J. Webster[4]

Abstract

Recent advancement of remote sensing technologies has brought in accurate, dense, and inexpensive city-scale Light Detection And Ranging (LiDAR) point clouds, which can be utilized to model city objects (e.g., buildings, roads, and automobiles) for creating Digital Twin Cities (DTCs). However, processing such unstructured point clouds is very challenging, epitomized by high cost, movable objects, limited object classes, and high information inadequacy/redundancy. We noticed that many city objects are not in random shapes; rather, they have invariant cross-sections following the Gestalt design principles, including proximity, connectivity, symmetry, and similarity. In this paper, we present a novel unsupervised method, called Clustering Of Symmetric Cross-sections of Objects (COSCO), to process urban LiDAR point clouds to a hierarchy of objects based on their characteristic cross-sections. First, city objects are segmented as connected patches of proximate 3D points. Then, symmetric cross-sections are detected for symmetric city objects. Finally, the taxonomy and groups of city objects are recognized from a hierarchical clustering analysis of the dissimilarity matrix. Experimental results showed that COSCO detected the correct taxonomy and types of 12 cars from 24,126 LiDAR points in 8.28s. Based on the cross-sections and taxonomy, a digital twin was created by registering online free 3D car models in 29.58s. The contribution of this paper is twofold. First, it presents an effective unsupervised method for understanding and developing DTC objects in LiDAR point clouds by harnessing innate Gestalt design principles. Secondly, COSCO can be an efficient LiDAR pre-processing tool for recognizing symmetric city objects' cross-sections, positions, heading directions, dimensions, and possible types for smart city applications in GIScience, Architecture, Engineering, Construction and Operation (AECO), and autonomous vehicles.

[1] Assistant Professor, Department of Real Estate and Construction, The University of Hong Kong. Email: xuef@hku.hk, Tel.: (852) 3917 4174, ORCiD: 0000-0003-2217-3693
[2] Professor, Department of Real Estate and Construction, The University of Hong Kong. Email: wilsonlu@hku.hk, Tel.: (852) 3917 7981, ORCiD: 0000-0003-4674-0357
[3] Research Assistant, Department of Real Estate and Construction, The University of Hong Kong. Email: 1810332089@email.szu.edu.cn, Tel.: (852) 9705 2240, ORCiD: 0000-0002-8720-2638
[4] Chair Professor, Faculty of Architecture, The University of Hong Kong. Email: cwebster@hku.hk, Tel.: (852) 3917 2125, ORCiD: 0000-0002-2171-7495

Keywords: LiDAR, Digital Twin City (DTC), Gestalt principles, hierarchical clustering, symmetry, cross-section.

1 Introduction

A digital twin, according to UK National Infrastructure Commission (2017), is a "virtual representation of a physical object or system across its lifecycle, using real-time data to enable understanding, learning, and reasoning." Digital twinning, as its name implies, is a methodology that is implemented in modeling and monitoring complex systems and objects such as aircrafts, factory production lines, and offshore drilling rigs without close adjacency to the physical systems and objects. Analytical and simulation results from digital twins can reveal emergent behaviors and mitigate unpredictable and undesirable consequences to the physical systems and objects (Grieves & Vickers 2017; Du et al. 2020; Fan et al. 2020).

Likewise, a Digital Twin City (DTC) is a virtual representation of physical systems and assets across their respective lifecycles in a city. The opportunity window for DTC was opened by both demand and data. First, there is an increasing demand for real-time semantically rich city models to ensure urban resilience and smart city (Kitchin 2014; Fan et al. 2020). Furthermore, recent advancements in the Internet of Things (IoT) and remote sensing technologies provide large-scale, accurate, and—most importantly—affordable measurement data at a city scale. Creating a DTC involves dynamic objects, real-time sensors, multimodal reasoners, and heterogeneous application frameworks (Austin et al. 2020; Du et al. 2020). Examples are a smart meter data-driven DTC for energy management (Francisco et al. 2020), a photogrammetric point cloud-based digital twin of trees for urban forestry (Chen et al. 2020), a wearable stress sensor-based digital twin for elderly's mobility (Lee et al. 2020), and Light Detection And Ranging (LiDAR) based digital twin Zurich for urban planning (Schrotter & Hürzeler 2020). In general, 3D point clouds offered high-quality geometric measurements and limited non-geometric semantics, such as object type and topological relationships, while sensor networks like smart meters mainly targeted non-geometric semantics such as energy consumption in a DTC. The output DTCs thus connected many fields including Building Information Modeling (BIM), Geographic Information Systems (GIS), transportation, resources and energy, disaster management, and robotics and self-driving cars (Burrough et al. 2015; Xu et al. 2014; NIBS 2015; Sacks et al. 2018).

This paper focuses on the LiDAR data of 4D dynamic city objects such as construction sites, airports, street lights, overhead advertisement boards, and vehicles. From a time-stamped LiDAR point cloud, a 'frame' of DTC can be created as a set of 3D models. However, understanding the relatedness between the objects and tracking them in the time-series LiDAR data are thus essential but challenging tasks (Andreopoulos & Tsotsos 2013). Because of the complicated shapes and dynamic status changes of city objects, together with the data amount and high update frequency, it is too tedious, time-consuming, and costly to

create and update digital city objects solely by human hands. Thus, automated DTC creation is highly desired. From a technical point of view, such DTC creation is equivalent to devising algorithms to parse and regress 3D geometry, rank the probability of being a whole, classify the type, and predict future changes of each object in a LiDAR point cloud. By definition, the automated DTC creation is a machine learning task (Mohri et al. 2018).

Nevertheless, the complexity of city objects made it challenging for automated or semi-automated DTC creation. Even the up-to-date deep learning models that were proven successful in computer vision competitions were found constrained by expensive training examples, limited accuracy, and scalability of trained models (Xue et al. 2018; Tian et al. 2019). In summary, the state-of-the-art automated and semi-automated methods are still limited for (i) processing massive point cloud to complex geometries of city objects (Wang & Kim 2019), (ii) understanding sophisticated relationships between unknown objects in city scenes (Xue et al. 2019a), or (iii) creating volumetric and semantically rich DTC (Lehner & Dorffner 2020; Boje et al. 2020). Exploring the effective automated DTC creation for dynamic city objects epitomizes one of the most explored research frontiers.

We noticed that most of the man-made objects in urban habitats, such as architectures, vehicles, and street furniture, embrace the same general Gestalt design principles regardless of the complexity in geometry (Todorovic 2008). Gestalt principles, such as the laws of proximity, connectivity, symmetry, and similarity, were introduced as design principles by Arnheim (1965) from Gestalt psychology about visual perceptions, where patterns and configurations were emphasized over individual components in perception. The Gestalt designs and cross section plans that represent the designs in 2D views can be universally found in various city objects, as shown in Figure 1, of which many have unique symmetric cross-sections in designs. The individual principles were extensively discussed and applied to reconstruct 3D shapes in the literature (Li et al. 2004; Du et al. 2016; Xie et al. 2020). However, the relatedness among such characteristic cross-sections were not well discussed as a key geometric feature in the existing studies. One explanation was the challenging complexity of identifying symmetries in LiDAR data. The symmetric cross-section generation is computationally inefficient in processing city-scale data unless novel symmetry detection algorithms are developed (Xue et al. 2019b).

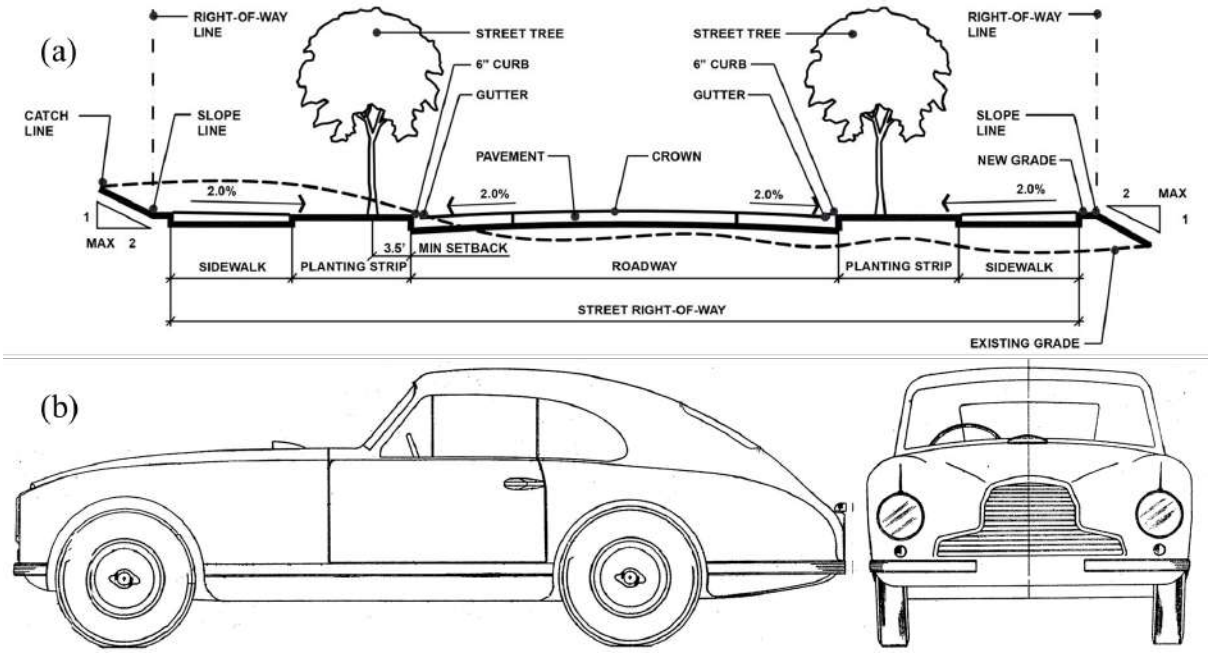

Figure 1. Example cross-sections of symmetric city objects. (a) Typical city road design (Source: City of Seattle Government); (b) A car design (Aston Martin DB2/4 Mk1) in 1951

This study aims to explore a novel unsupervised method, called Clustering Of Symmetric Cross-sections of Objects (COSCO), to process LiDAR data for recognizing symmetric city objects for DTC. Rather than relying on annotated training data for supervised learning, our COSCO method exploits the intrinsic characteristics—including proximity, connectivity, symmetry, and similarity—of city objects for automating urban point cloud processing. The proposed approach first detects 3D city objects from 3D point clouds, extracts the cross-sections of the unknown city objects, then understands the hierarchical relationships between the unknown objects, and finally organizes unknown city objects in a taxonomy. The contribution of this paper is twofold. First, COSCO is an effective unsupervised method to understand city objects in LiDAR point clouds by exploiting the free innate urban design principles rather than expensive training examples. Secondly, COSCO can be an efficient LiDAR pre-processing tool for recognizing symmetric city objects' positions, heading directions, cross-sections, and types for DTCs and more smart city applications in GIScience, Architecture, Engineering, Construction and Operation (AECO), and autonomous vehicles.

2 Literature review

Although DTC is a relatively new concept, a wealth of automated and semi-automated DTC creation methods have been developed (Huber et al. 2011; Wang & Kim 2019). From a machine learning perspective, all the methods of processing LiDAR point clouds to a DTC can be classified into four classes, i.e., non-learning, supervised learning, reinforcement learning, and unsupervised learning, according to the taxonomy of machine learning methods (Mohri et al. 2018). Each class of methods has notable pros, cons, and prerequisites.

The first class is non-learning, in which the correlation model is structured universal expert knowledge on the built environment, such as rules about geometric primitives and their joints on building envelopes. Examples include Valero et al.'s (2012) floor and ceiling segmentation from *z*-slices, Previtali et al.'s (2014) indoor space segmentation from incomplete boundaries, Jung et al.'s (2018) opening detection for the automated 3D volumetric reconstruction of indoor environments, Xue et al.'s (2018) rooftop element rectification based on as-designed parallelity and perpendicularity, and Pan et al.'s (2019) rule-based reasoning based on normal values. By utilizing such structured expert knowledge, non-learning methods correlate approximate semantic information in a short computational time and with perfect interpretability (Hong et al. 2015). However, the applications of this class were limited by the extractability of expert knowledge at first (Wang & Kim 2019); the methods were also limited in dealing with complicated scenes and noise filtering (Pătrăucean et al. 2015). That was why most methods in this class worked together with other methods to handle noisy point clouds in the literature.

The second class consists of supervised learning-like methods, in which a supervised model fitting (or training) is performed to correlate the given labels in training samples to the measurement features (Rosser et al. 2019). A well-known subclass is called semantic segmentation which processes point cloud and photo data to building and city models (Xiong et al. 2013; Babacan et al. 2017; Czerniawski et al. 2018; Wang et al. 2018). Examples in other subclasses include Zou et al.'s (2018) 3D room layout modeling based on a supervised deep learning model, He et al.'s (2018) adoption of three cities as training samples to identify various combinations of building group features, Rosser et al.'s (2019) research of predicting residential building age from map data, and Huang et al.'s (2020) spatial joint embedding of deep learning features. Some studies like Li et al. (2019) and Ahmed and Chew (2020) employed a basket or series of supervised learning methods for striking a balance between the accuracy and time cost, while some research like Yang et al. (2020b) linked 3D points to 2D topographic maps. However, the supervised learning task itself is also very challenging in sophisticated urban scenes (Babacan et al. 2017). For example, the optimization algorithms driving the learning can become complex (Ullah et al. 2020) and accuracy of learning results can be problematic (Xiong et al. 2013) once there is a large deviation in a small sample-based classification model and recognition of a variety of features in big data (Yu et al. 2020), which is difficult to avoid in large-scale urban datasets. Besides, the preparation of the training labels also costs a fortune, especially when the sample size goes wild. In practical applications, it was essential to do an adequate survey with extensive manual work to assign its precision so as to obtain training samples; this was prohibitively expensive (Rosser et al. 2019). Furthermore, it accomplished the function of memory and knowledge recombination without interaction with the real environment (Ma et al. 2020).

The third class is reinforcement learning, in which the correlation gradually converged after

trial iterations of possible targets. Hidaka et al. (2018) applied the iterative closest point (ICP) algorithm to register previously known bridge pier models to a point cloud; Xue et al. (2018; 2019a) developed a Semantic Registration approach to reconstruct 3D building models automatically from point cloud data and online open 3D component models. Other examples in this category include Sarmad et al. (2019), who applied it to point cloud shape completion, and BuHamdan et al. (2020) who focused on decision-making during the construction design phase. In contrast with supervised learning, reinforced learning depends on a reward function rather than training labels, so that the agent (or algorithm) interplays with the environment and becomes rewarded for making correct decisions (e.g., 3D model registration) and penalized for incorrect ones (BuHamdan et al. 2020). In essence, reinforced learning is a trial-and-error process; an agent needs to estimate which actions to select to maximize the reward in the environment (Ullah et al. 2020). Thus, due to the lack of direct guidance or knowledge, it is necessary for an agent to make continuous attempts in order to acquire optimal strategies (Yang et al. 2020a). For example, Xue et al. (2019a) took over 15 minutes to automatically generate an auditorium seat model with about 90% precision and recall. The slow responsiveness may become a problem in fulfilling the timely performance required for digital twinning (Son et al. 2015). Furthermore, the methods in the reinforcement learning group were limited to the availability of standardized 3D components for matching the input point clouds (Xue et al. 2019a), and the reward (or payoff) functions are sometimes unfeasible or unethical to design such as when it involves privacy data.

In comparison, unsupervised learning requires no annotated training examples or reward functions and components. It is highly automated in grouping objects into clusters in terms of learned features (Armesto-González 2010). Objects from the same group are more similar to some learned features than those that come from different groups. Examples of conventional clustering methods in order to process LiDAR point clouds include region growing to approximate point cloud geometry (Pauly et al. 2002), *k*-means of normal directions to simplify point cloud (Shi et al. 2011), and proximity-based query of points (El-Mahgary et al. 2020). Modern unsupervised methods also employed other three classes of sub-processors for novel learned features. For instance, Papon et al. (2013) developed supervoxel clustering, which generates connected patches based on the closeness and normal directions of 3D points' supervoxels; Xu et al. (2018a; 2018b) utilized the connectivity rules between the supervoxels to merge surfaces greedily. Zeng et al. (2020) utilized a supervised deep network for deep point-level feature extraction; then, they computed the feature correlation to distinguish precious elements even for complex buildings. Xue et al. (2019c) showed the possibility to cluster patches based on the reinforcement learned feature of geometric dissimilarity. The novel learned features and the LiDAR data quality led to increased accuracies of unsupervised clustering methods in the literature. Besides, independence from training examples made the unsupervised methods cost much less than supervised learning. Yet, some unsupervised methods like the *k*-means are sensitive to outliers (Arora et al. 2016),

and some methods like the hierarchical agglomerative clustering (HAC) are limited to small databases due to the computational time complexity (e.g., $O(n^3)$ for HAC).

The advantages of unsupervised learning, as summarized in Table 1, pinpointed a unique position of pre-processing LiDAR point clouds to patches, objects, and clusters, for a DTC. Laws of proximity, connectivity, symmetry, and similarity—as Gestalt design principles of many city objects—have been adopted respectively as intrinsic characteristics in existing unsupervised methods (Shi et al. 2011; Poux & Billen 2019). Features that reflect the combinations of the laws, such as supervoxels, footprints, and cross-sections, have been employed as critical features in processing LiDAR point clouds (Xu et al. 2018a; 2018b). For example, the clustering of cross-sections in LiDAR points was vital to the understanding and modeling of forest (Wang et al. 2017), ships (Mi 2015), tunnel segments (Sun et al. 2020), and 3D printing parts (Samie Tootooni et al. 2017). For many movable city objects like cars and airplanes, symmetric cross-sections are also invariant in translations and rotations. Therefore, detecting and utilizing the symmetric cross-sections in LiDAR point clouds could intuitively accelerate the creation and synchronization of a DTC. However, symmetric cross-sections were not well discussed for DTC as far as we are concerned.

Table 1: Comparison of machine learning methods for processing LiDAR point clouds to a DTC

	Non-learning	Supervised	Reinforcement	Unsupervised
Pre-requisites	Structured expert knowledge	Annotated training examples	A reward or penalty function	Intrinsic characteristics or learned features
Usages for city objects	Filtering, reasoning	Labeling, recognition	Registration, matching	Grouping, pre-processing
Pros	Shortest time, and good interpretability	Adaptivity to domains, capability of learning from big data, and new AI methods	Active interaction, and optimized decision-making	Low cost, training-free, and increasing accuracy
Cons	Limited by knowledge extractability, and limited applicability	Unsatisfactory accuracy, high cost, and lack of training data	Slow speed, time-delayed feedback, availability of reward (or penalty) function	Limited application in small database, sensitive to outliers
Examples	Z-slicing, rules (Pan et al. 2019), planar parallelity, and perpendicularity (Valero et al. 2012)	PointNet++, ShapeNet, Semantic segmentation, (Ma et al. 2020)	Improved RANSAC, ICP, and Semantic registration (Xue et al. 2019d)	Supervoxel segmentation, and connectivity-based merging (Xu et al. 2018a; 2018b)

3 The proposed method

Figure 2 demonstrates the flow chart of the proposed Clustering Of Symmetric Cross-sections of City Objects (COSCO) method. COSCO aims to process large-scale urban LiDAR point clouds to derive a hierarchical understanding of city objects. It involves three steps: (i) connectivity-based object detection, (ii) symmetric cross-section detection, and (iii) dissimilarity matrix-based hierarchical clustering, based on Gestalt design principles.

Sections 3.1 and 3.2 describe the three steps, respectively, and Section 3.4 introduces DTC creation as an application of COSCO output.

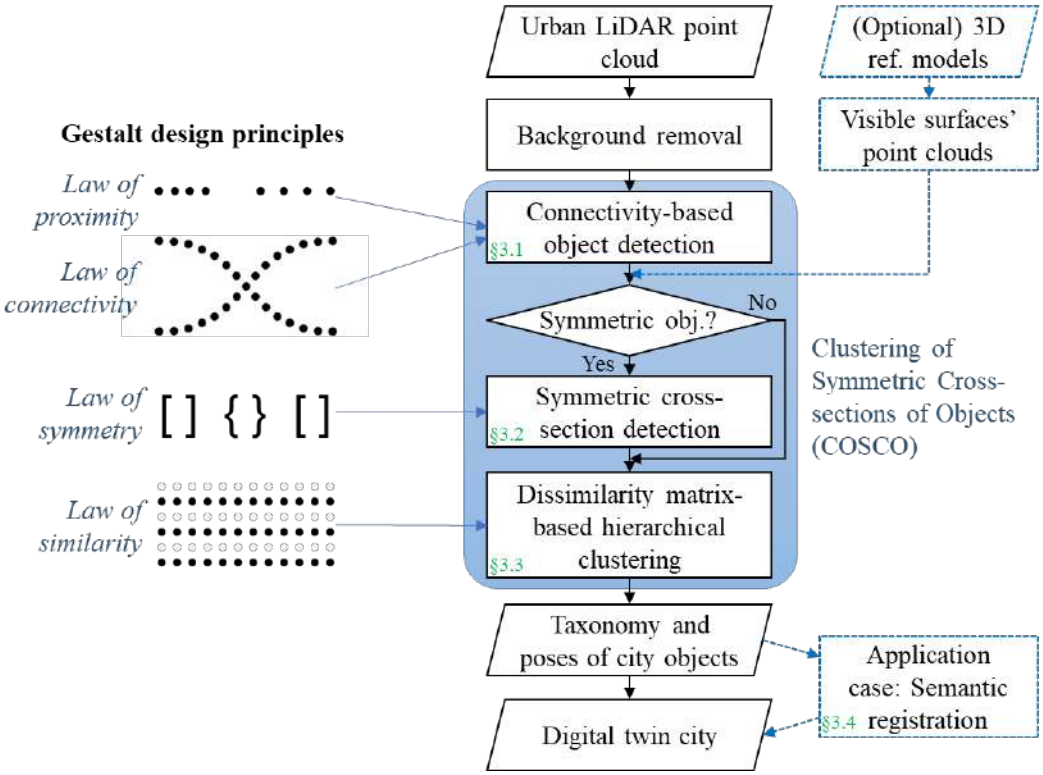

Figure 2: Flow chart of the proposed COSCO method

3.1 Connectivity-based object detection

Step 1 of COSCO consists of two stages, as shown in Figure 2. The first stage is Papon et al.'s (2013) point cloud over-segmentation method, which computes the adjacency graph (proximity) of all 26-adjacent voxels in the 3×3×3 grids for each voxel. The super voxels are clustered from voxels based on the 3D position, color values, and the FastPoint feature (Rusu et al. 2009). The results of Papon et al. (2013) can reflect the proximity of small patches such as continuous surfaces. However, the patches are very often over-segmented at the object level.

The second stage aims to group the over-segmented patches to the object level based on the law of connectivity. A filtering restriction on the number of inter-connected super voxels with a lower bound and upper bound can filter the target size of objects. It was discovered that a background removal operation, as shown in Figure 2, can considerably improve city object detection at this stage. For example, removing street and earth surface point clouds can well disconnect the city into street blocks and subsequently save the computational load substantially without loss of accuracy. Each output patch is stored in the Stanford Polygon (.ply) format and labeled as a unique object. Overall, the two-stage connectivity-based object detection can cluster point clouds to the object level and remove noise data.

3.2 Symmetric cross-section detection

Step 2 of COSCO in Figure 2 identifies the cross-sections for symmetric objects in two stages. The objective in this stage is to detect the reflective symmetrical axis (e.g., as a 2D plane for a 3D reflective symmetry). As shown in Figure 3, COSCO first adopts Xue et al.'s (2019b) efficient Optimization-based Detection of Architectural Symmetries (ODAS) library (https://github.com/ffxue/odas) with default parameters for automated symmetry detection in urban LiDAR point clouds. Although ODAS was proposed for buildings, it also works well for other urban objects. The results of detection include a point corresponding ratio s_m after reflective symmetry and the reflective symmetrical axis A_m. If there are several symmetries, the perfect one with the maximum s_m is expected. The characteristics of airborne LiDAR data, such as high density on the top and balanced details on the sides, is helpful for detecting reflective symmetries of city objects than other 3D point clouds, such as total station or ground vehicle-borne LiDAR. The reason is that airborne LiDAR is very good at measuring a city's convex objects (except for the bottom side). It returns much more isotropic surfaces in terms of horizontal directions than ground vehicle-borne.

```
procedure sym_cross_sect_detect
input: P                                          // Patch of points of an unknown object
1    s_m, A_m ← symmetry_detect ( P );            // To call ODAS to detect the symmetry
2    if s_m > threshold then
3        P_l ← { p | p ∈ P, d ( p, A_m ) < ε };   // Points near the symmetrical axis A_m
4        CS_l ← project_to ( P_l, A_m );          // The longitudinal section
5        A_t ← orthogonalize (A_m, center (CS_l) );
6        P_t ← { p | p ∈ P, d ( p, A_t ) < ε };
7        CS_t ← project_to ( P_t, A_t );          // The transverse section
8        CS ← voxelize (CS_l ∪ CS_t, res);        // Voxelized cross-sections
9        return CS, center (CS_l), rot (A_m);     // Cross-sections, position, and heading
10   end if
11   return P, null, null;                        // Not found
```

Figure 3: Pseudo code of the symmetric cross-section detection

In the second stage, the symmetry is applied to the patch to form the longitudinal and the transverse sections of an object. The point clouds near (Euclidean distance d within a threshold ε) the axis A_m are filters as a slice P_l, then projected on to the plane of axis A_m as the longitudinal section CS_l. Then, a vertical plane A_t at the centroid of CS_l perpendicular to A_m is computed as the base plane of the transverse section. A similar projection process maps the near point clouds to form the transverse section CS_t. To minimize the random noises, we apply a voxelization at a resolution *res* for a regularized contour-like cross-sections. As shown in Line 9 in Figure 3, the two sections are returned and saved in the Stanford Polygon format, and the centroid position and heading direction of the patch are saved in a data

spreadsheet. From the contour-like cross-sections, the patch's 3D size can be extracted apart from its position and rotation. The 3D size can be a leading indicator to cluster similar objects for the next step. If the object in the patch is asymmetric (not meeting the *threshold* of minimal symmetric corresponding ratio), Step 2 of COSCO returns the input patch P and a "not found" message.

3.3 Dissimilarity matrix-based hierarchical clustering

Step 3 of COSCO also involves two stages based on the law of similarity. The first stage aims to measure the dissimilarity (or similarity equivalently) matrix between the objects' patches. Given the two patches P_i and P_j, the most well-known dissimilarity metric is the minimal Root-Mean-Square Error (RMSE):

$$dissimilarity\ (P_i, P_j) = \min_{r,\ t \in \mathbb{R}^3} RMSE\ (P_i, trans\ (rot\ (P_j, r), t)), \quad (1)$$

where the *RMSE* function measures the error between the point clouds in the two patches, $r = [r_x, r_y, r_z]$ is the set of 3D rotation parameters defined on $\mathbb{R}^3$, and $t = [t_x, t_y, t_z]$ is the set of 3D translation parameters. For the patches with cross-sections detected in Step 2, the 6 variables in Equation (1) can be dramatically reduced to 2, one continuous t_z and one discrete r_z:

$$r_{sym} = [0,\ 0,\ r_z],\ t_{sym} = [0,\ 0,\ t_z],\ r_z \in \{\ 0, \pi/2, \pi, 3\pi/2\ \}. \quad (2)$$

We applied the ODAS as the solver by defining the objective function as Equation (1). Given N patches of objects, the pairwise comparison leads to an $N \times N$ dissimilarity matrix. In the matrix, the diagonal entries are all 0, while the lower triangle is identical to the upper one. Therefore, $(N - 1) \times N / 2$ pairwise comparisons are needed. Furthermore, the two main algorithmic parameters of Step 3 of COSCO are inherited from ODAS, i.e., the depth of weighted octree down-sampling (δ) and the number of iterations (k).

The second stage is a hierarchical clustering based on the dissimilarity matrix. A hierarchical structure can be created from the matrix by gradually listing the most similar items together. The hierarchical clustering (*hclustering*) function in the *scipy* package (ver. 0.19, in Python 3.6) was employed to cluster the dissimilarity matrix to a hierarchy. In the hierarchy, a threshold can filter groups of object patches. Those object patches in one group have similar or even the same type of city objects. And those in the near subtree are also very close in geometry.

3.4 Mapping the unsupervised results to digital twin city

The result of COSCO can be utilized in many ways for creating a DTC. One approach is the semantic registration approach of Xue et al. (2019a). The semantic registration approach, initially proposed for a segmentation-free 3D model reconstruction, encountered a combinatorial explosion when facing too many models for large-scale point clouds. The intrinsic characteristics of city objects, especially the symmetric cross-sections, are vital for processing large-scale urban point clouds for the semantic registration approach. First, the poses, including translation, rotation, and scaling in the symmetric cross-sections, can

directly register 3D models to an object. Furthermore, it is now able to coarsely filter the possible models using the 3D size of the target patch. The latter is also true for asymmetric city objects. As a result, the COSCO can help the semantic registration reduce the execution time significantly based on the hierarchical understanding of the LiDAR data as well as the reference objects.

The updated semantic registration approach based on COSCO acts as follows. First, a few semantically rich 3D models are given as possible references. The 3D models can be extracted from industrial as-designed modes or previous versions of DTCs. Their visible, continuous surfaces are down-sampled to point clouds at the point density equivalent to the city point clouds. Then, the reference models' point clouds also go through Steps 2 and 3 of COSCO for the symmetric cross-sections and similarity hierarchy. Thirdly, the cross-sections of the reference objects are registered to the cross-sections of the detected objects in the point cloud of the target urban area, instead of registering the 3D models to the unsegmented point cloud as a whole. The optimized pose variables in the cross-section registration can transform the 3D reference models to the target objects. The final DTC contains both an abstracted concept hierarchy and the rich semantics of the city objects for enabling reasoning and simulating smart city development.

4 Experimental tests

4.1 Experimental settings

The COSCO method was experimented as an in-house developed computer program, based on three existing scientific software libraries for the three steps. In the first step, we employed the supervoxel over-segmentation function from the open-source software library *point cloud library* (*pcl*, version 1.8) and developed the connectivity-based object filters. In the second step, we adopted the ODAS library with the algorithm and parameters recommended in Xue et al. (2019b). In the last step, a minimum dissimilarity computation module is developed based on the algorithm and parameters in the ODAS library. The programming language was C++ for all the three steps complied under the C++11 standard on a Ubuntu 16.04.

A pilot study was conducted on a case of LiDAR data scanned from a small car park near O'Connell Street Upper, Dublin, Ireland (Laefer et al. 2017), as shown in Figure 4. The LiDAR point cloud, consisting of 112,999 points (6.78MB compressed on disk), included 12 city cars of various models, parking locations, and orientation. In terms of height, the cars included 8 'short' cars (height < 1.5m), 3 'tall' cars (1.5m ≤ height < 1.9m), 1 full-size SUV (sport utility vehicle, height ≥ 1.9m); in terms of types, there are 1 supermini car (length < 3.7m), 5 hatchbacks with trunks, 5 hatchbacks without trunks, and 1 full-size SUV. A preprocess of planar removal removed almost all point clouds of the ground. The cloud after the ground removal consisted of only 24,126 points. The experimental study involved four tests on (i) the COSCO method, (ii) the effectiveness of symmetric cross-sections, (iii) the

parameters sensitivity of the COSCO, and (iv) the application to semantic registration. The tests were conducted on a desktop computer with Intel E5-2690 CPU (2.6 GHz), 64GB memory, and Ubuntu 16.04 system. The main performance metrics were accuracy and computational time in the single-threaded mode.

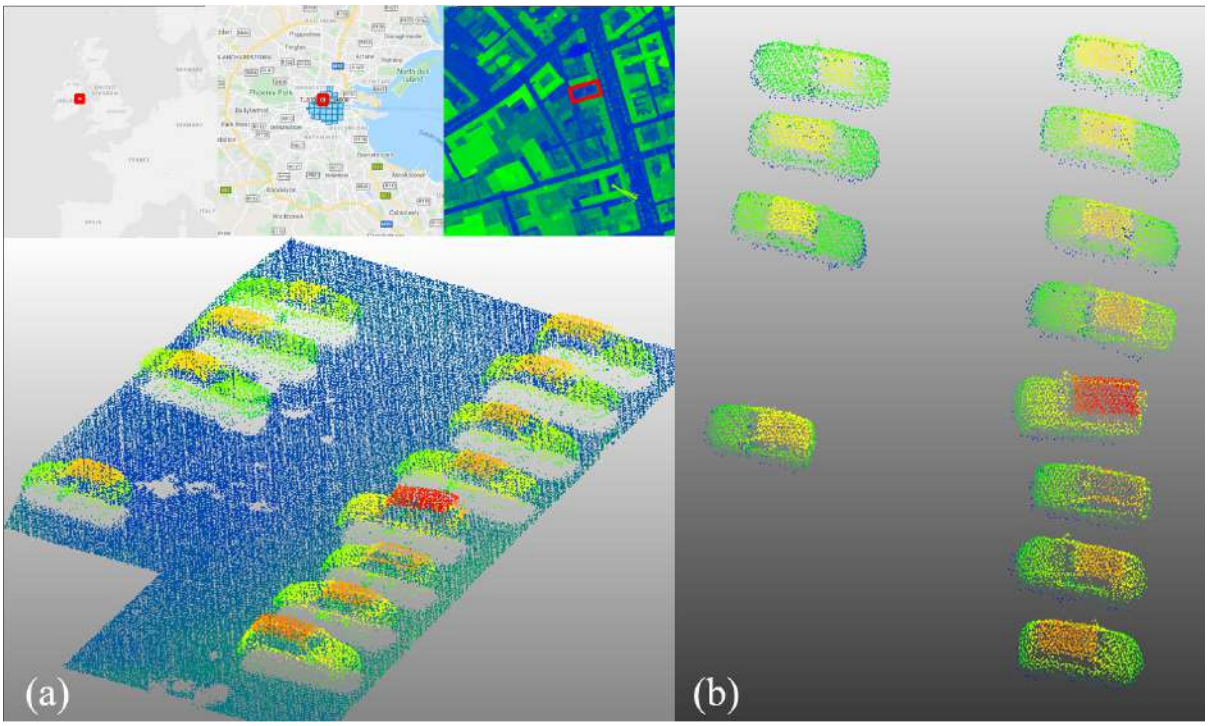

Figure 4: The pilot case of a car park scene. (a) 112,999 airborne LiDAR point clouds (color indicates height); (b) 24,126 point clouds after ground removal

4.2 Experimental results

4.2.1 Results of the COSCO methods

Figure 5 shows the results of the first step of implementing COSCO. The execution time of the supervoxel over-segmentation was 1.30 seconds. It yielded 368 over-segmented supervoxels and the connectivity among them, as shown in Figure 5a. The parameter "voxel seed size" was 15cm, so that a typical supervoxel represented a surface area of about 0.09m² (30cm×30cm). We set the target objects' surfaces (glass excluded) to be between 1 m² and 20 m², equivalent to supervoxel connectivity between 10 and 220. Based on the connectivity range, 12 patches of objects were filtered instantly from the connected subgraphs. The object patches were named based on their centroids along the x-axis, as obj_1, obj_2, … to obj_{12}, as shown in Figure 5b.

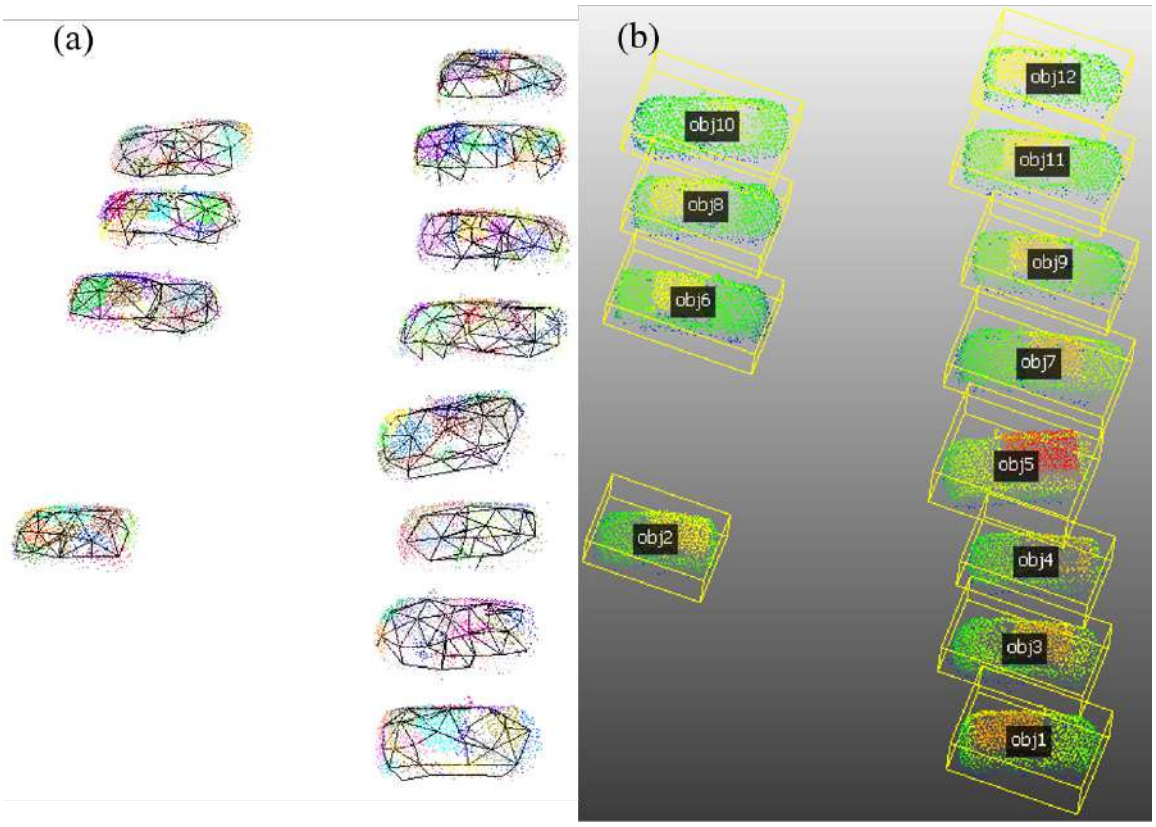

Figure 5: Twelve objects detected based on connectivity. (a) 368 small patches (by color) and the connectivity (lines) in 1.30s, (b) 12 object patches by connectivity filtering

The second step is the symmetric cross-section detection. Table 2 lists the results of the first stage, i.e., the automated symmetry detection for the 12 objects. The first column is the object ID, while the second to fourth columns are the results, i.e., the point correspondence ratio (PCR) of the symmetry, the symmetric centroid of the object, and the rotation of the symmetry axis, of the automated symmetry detection using the ODAS method with default parameter values. It can be seen that all the PCR values were no less than 0.93, which stood for good symmetries for all the objects. It should be noted that an object's symmetric centroid is slightly different from its geometric centroid.

Table 2: Results of 3D poses and symmetric cross-sections of the 12 objects

Obj.	Sym. PCR	Sym. centroid (t_x, t_y)	Rotation (r_z)	Cross-sections 3D size (w×d×h)	Time (s)
1	0.967	-9.39, 6.02	0.617π	4.1×1.8×1.6	0.52
2	0.944	-7.10, 18.36	1.580π	3.6×1.6×1.4	0.49
3	0.965	-6.56, 7.22	1.583π	4.3×1.8×1.5	0.48
4	0.936	-3.72, 7.88	0.592π	3.8×1.7×1.5	0.55
5	0.962	-1.06, 9.40	0.636π	4.3×1.8×1.8	0.47
6	0.975	0.71, 19.97	0.581π	4.6×1.8×1.3	0.48
7	0.979	2.56, 9.94	0.575π	4.9×1.9×1.4	0.49
8	0.966	3.83, 20.67	0.603π	4.3×1.8×1.3	0.40
9	0.968	5.69, 10.75	0.578π	4.5×1.8×1.5	0.45
10	0.972	6.41, 21.44	0.611π	4.6×1.8×1.3	0.37
11	0.977	8.82, 12.00	0.596π	4.6×1.8×1.4	0.39
12	0.959	11.56, 12.72	0.606π	4.0×1.7×1.4	0.39

Figure 6 exhibits the second stage of symmetric cross-section modeling. The projected point clouds of the 12 objects, as shown in Figure 6a, correctly reflected the cross-sections. Figure 6b demonstrates the voxel modeling of cross-sections, wherein noises are mitigated, and contours became rectified. Then, the width and height of an object can be easily read from the longitudinal section, similar to the depth from the transverse section. Table 2 lists 3D sizes, in terms of width × depth × height, of all the objects. The twelve objects were very close when it comes to sizes. The computational time of Step 2, as listed in Table 2, was short, around 0.4 to 0.5 seconds for each object.

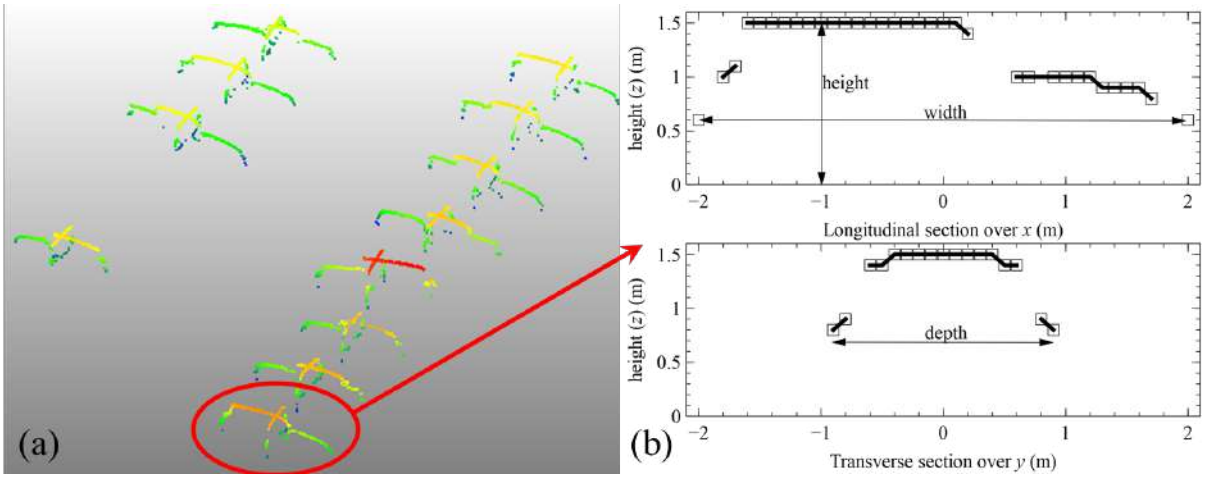

Figure 6: Symmetric cross-sections detected using COSCO. (a) Associated point clouds (color indicates height), (b) Cross-section modeling of obj_1 in 10cm voxels

Figure 7 manifests the results of Step 3 after the implementation of COSCO. First, the dissimilarity matrix was calculated between the objects' cross-sections, as shown in Figure 7a. It consumed 4.96s to complete the upper triangle, which was copied to the lower triangle immediately. The parameter δ was set to 4 and k to 50. In the visualized matrix in Figure 7a, the most dissimilar pair was 19.3cm between obj_2 (the supermini) and obj_7 (a hatchback with trunk), while the most similar was 4.9cm between two hatchbacks (obj_9 and obj_{11}). Besides, there were a few warm and cold color zones, which indicated inter-similar objects. Such objects were hierarchically clustered based on the matrix, as shown in Figure 7b. We set the color threshold to 6.5cm and got two groups of objects and two isolated instances. The group in green consisted of five objects, i.e., obj_6, obj_7, obj_9, obj_{10}, and obj_{11}, which were the five hatchbacks with trunks. The other group in red included obj_1, obj_3, obj_4, obj_8, and obj_{12}, which were the five hatchbacks without trunks. The supermini and the SUV were isolated. Moreover, the two groups were detected to be more similar (as 10 hatchbacks) than the supermini and the SUV. The SUV was identified as the most dissimilar one in all the 12 objects. The taxonomy was correct and meaningful.

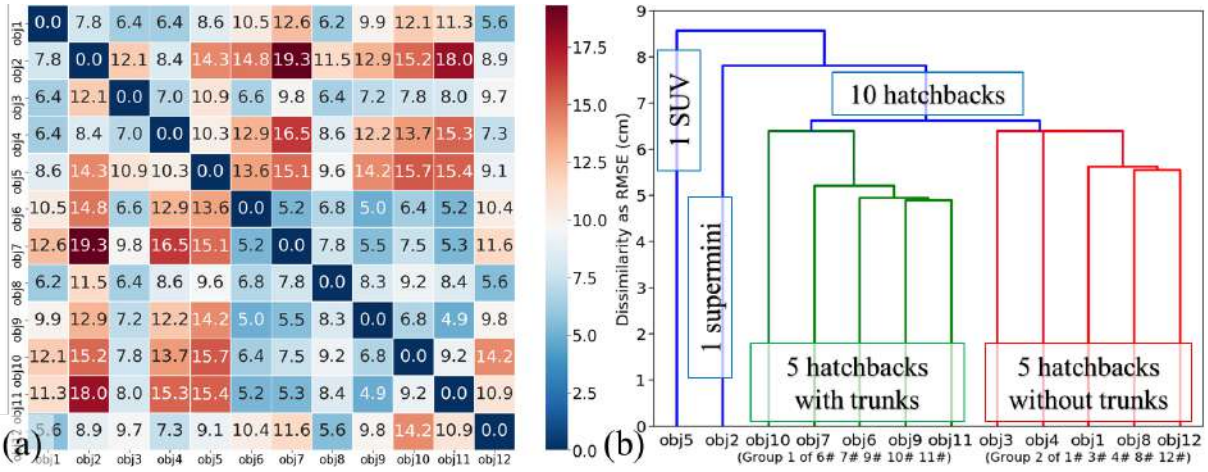

Figure 7: Results of Step 3. (a) Dissimilarity matrix (unit: cm, cold color indicates similar) in 4.96s, (b) Dendrogram of hierarchical clustering (color stands for group) in 0.05s

Overall, the COSCO spent 8.28s to process the 24,126 LiDAR points in three steps. The results of COSCO included (i) 12 objects as shown in Figure 5, (ii) the centroid, rotation, 3D size, and cross-sections of each symmetric object as shown in Table 2 and Figure 6, and (iii) a hierarchical understanding of the objects as shown in Figure 7. The experimental results showed that COSCO was efficient (short in time) and effective (accurate in recognition and understanding) in processing unstructured urban LiDAR point clouds. The understanding of the city objects, such as the cross-sections and hierarchical cluster, was in line with how they are designed and manufactured, thus, providing a useful guiding framework in creating a DTC.

4.2.2 Comparison to previous work

We compared the COSCO method to the patch clustering method in Xue et al. (2019c), which bypassed COSCO's Step 2 by assuming asymmetry equivalently. Figure 8 shows the results of the two methods. Both methods detected 12 objects. However, the dissimilarity matrices were different. First, the maximum dissimilarity in Xue et al. (2019c) was 36.0cm between the SUV and the supermini, where the value was much higher than that of COSCO, and the entry was different. Some parts of hot/cold color zones, as highlighted in Figure 8, were also different. One key reason for the difference was that the car roofs produced much more point clouds in aerial LiDAR scanning so that the pairwise comparison in Figure 8a overweighed the car roofs.

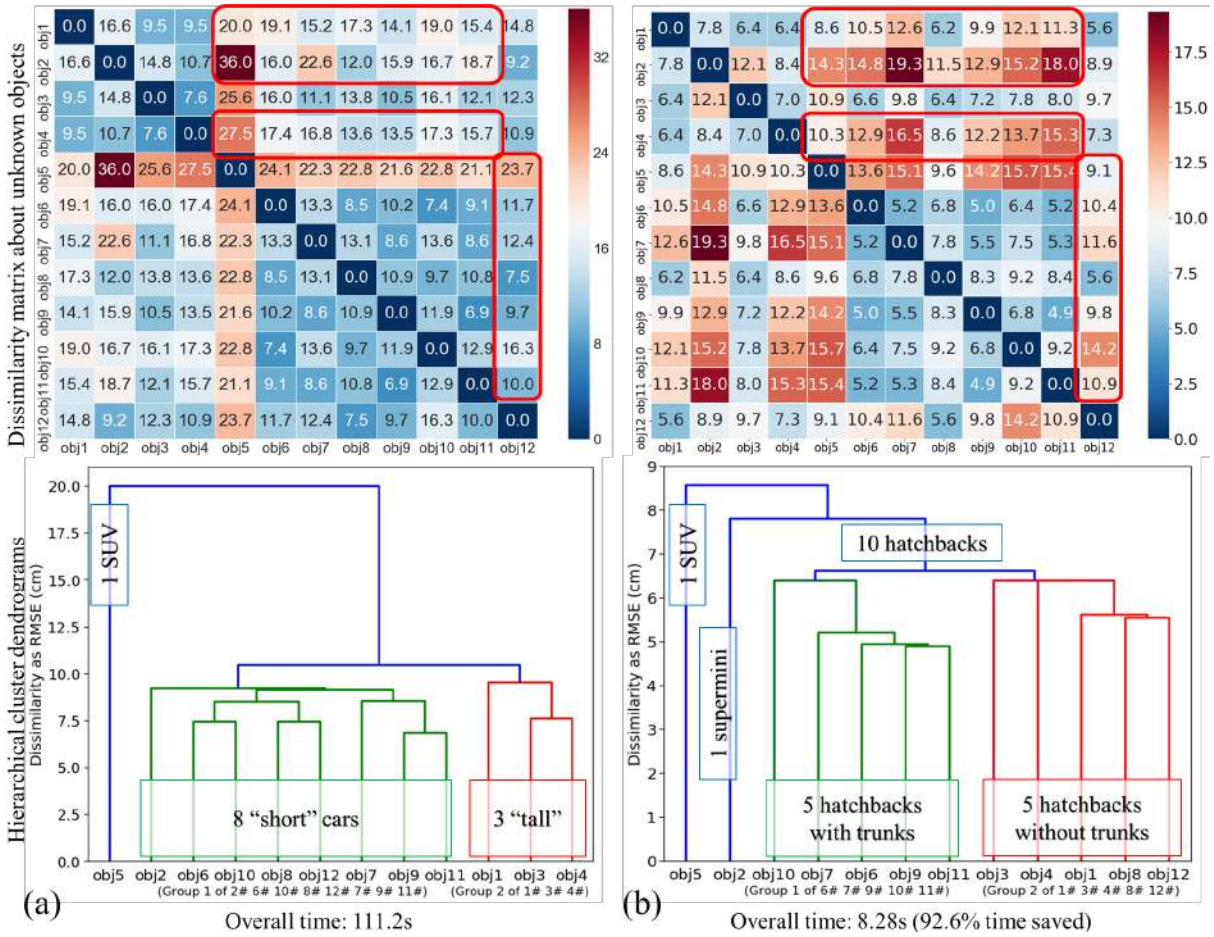

Figure 8: Comparison on the matrix, accuracy, meanings, and time of unsupervised hierarchical clustering. (a) patch-based in Xue et al. (2019c), (b) the COSCO.

Consequently, the hierarchical clusters in Figure 8 were different. The cluster in Figure 8a emphasized the dissimilarity on the height, where the 8 'short' cars were in the green group, and the 3 'tall' cars were in the red group. In contrast, the COSCO's results in Figure 8b emphasized the contours of cross-sections. Nevertheless, the SUV (obj_5) was the most dissimilar object in both. Moreover, the overall computation of COSCO saved 92.6% time from Xue et al. (2019c), which proved the efficacy of symmetric cross-sections.

4.2.3 Parameter sensitivity analysis

In order to gauge the parameter sensitivity of COSCO, we tested a group of combinations of the two major parameters δ and k. Stage 1 of Step 3 was the most time-consuming process in COSCO, due to the $(N-1) \times N/2$ pairwise comparisons for dissimilarity matrix computation. In the tests, δ increased from 2 to 5, and k changed independently from 10 to 1,000 in an exponential order. Figure 9 shows the results on the average dissimilarity (in cm) and computational time (in logarithmic scale). It can be seen from Figure 9a that the dissimilarity computation is more sensitive to k, i.e., the number of iterations. The dissimilarity values came to an optimum plateau of 10.628cm, as shown in the dark blue area in Figure 9a, when $k \geq 50$. Meanwhile, Figure 9b shows that the computational time increased

with either δ or k. Therefore, we set k to 50. After examining the minor excess of average dissimilarity over the optimum plateau and the time cost, we chose δ as 4 for the best accuracy-time trade-off. In comparison with the setting k = 1,000 and δ = 5, the selected parameters can save 96.2% of computational time, i.e., 0.03s for each comparison on average, at a cost of ignorable 0.02cm average excess (about 0.2% error) over the optimum dissimilarity matrix.

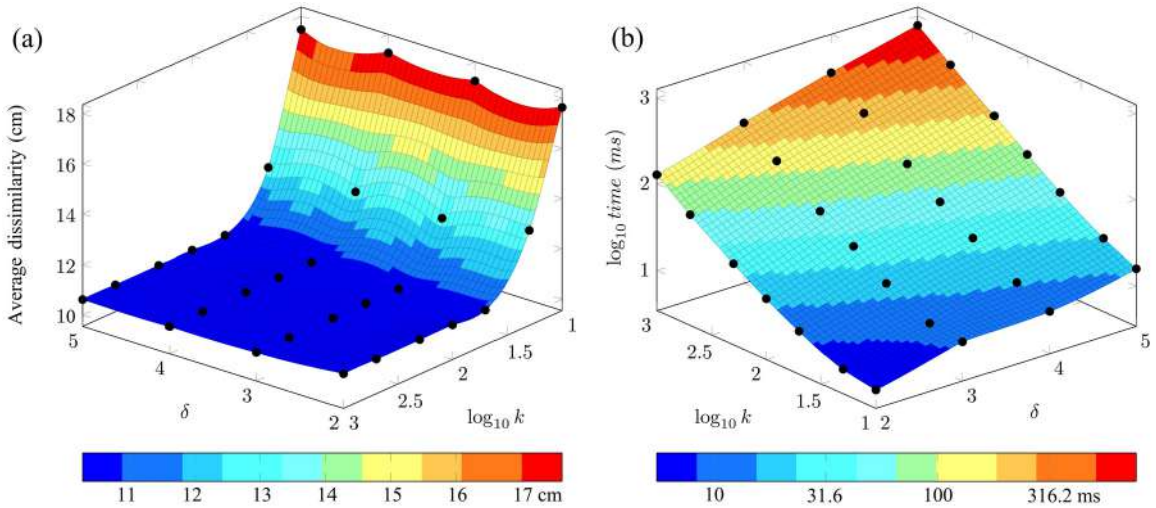

Figure 9: Results of the dissimilarity matrix computation under different COSCO parameter settings. (a) Average dissimilarity in the matrix, (b) Average computational time

4.3 Application to semantic registration for creating DTC

The tests in this section applied the hierarchical understanding of COSCO for creating a DTC from reference 3D CAD (Computer-Aided Design) models. The approach was Xue et al. (2019a)'s semantic registration for fitting known semantically rich 3D models to a point cloud. We randomly downloaded 24 CAD models of city car models freely available from the manufacturers' and fans' websites. Table 3 lists the brands and models of cars. The models included popular brands such as Alfa, Audi, Honda, Toyota, Mercedes-Benz, and Volvo. The COSCO method was applied to the surface (glass excluded) point clouds of the 3D CAD models and obtained their cross-sections and 3D sizes. As listed in Table 3, COSCO returned the approximate sizes for the car models. After a comparison with the car dimensions queried from manufacturer websites, Wikipedia, or automobiledimension.com, we validated the correctness of the models. By setting a dimensional tolerance at 0.2m, we can preliminarily match the car models to potential objects detected in Section 4.1. For example, an Audi TT, a hatchback model without a trunk, has a 3D size 4.0×1.7×1.4 and thus matched to three candidate objects obj_1, obj_4, and obj_{12}. In this way, the number of model-object semantic registration trials was reduced drastically from 24 × 12 = 288 to 87, or 69.8% trials saved.

Table 3: List of 24 reference car CAD models

ID	Brand	Model	Cross-sections 3D size (w×d×h)		Candidate objects (dim. tolerance = 0.2m)
			Web query*	by COSCO	
1	Alfa	Brera	4.41×1.83×1.34	4.4×1.8×1.4	3, 6, 8, 9, 10, 11
2	Alfa	Romeo 8C	4.38×1.89×1.34	4.4×1.8×1.4	3, 6, 8, 9, 10, 11
3	Audi	A4	4.73×1.84×1.43	4.8×1.8×1.5	6, 7, 10, 11
4	Audi	A5	4.63×1.85×1.37	4.6×1.8×1.4	6, 9, 10, 11
5	Audi	TT	4.04×1.76×1.34	4.0×1.7×1.4	1, 4, 12
6	Chevy	Silverado 3500 HD	5.98×2.08×2.05	6.0×1.8×2.0	(No candidates)
7	Fiat	Grande Punto	3.99×1.69×1.49	4.0×1.7×1.5	1, 4, 12
8	Honda	Civic	4.52×1.80×1.43	4.5×1.7×1.5	3, 6, 8, 9, 10, 11
9	Honda	Jazz	3.95×1.69×1.54	3.9×1.7×1.5	1, 4, 12
10	Jeep	Cherokee Sport	4.40×1.82×1.64	4.4×1.7×1.7	3, 5, 9
11	Jeep	Wrangler	3.84×1.70×1.78	3.7×1.6×1.8	(No candidates)
12	Mercedes Benz	C63 AMG W204	4.73×1.77×1.45	4.8×1.8×1.5	6, 7, 10, 11
13	Mercedes Benz	Class C W204	4.58×1.77×1.47	4.5×1.7×1.5	3, 6, 8, 9, 10, 11
14	Mercedes Benz	Class C W204 Wagon	4.60×1.77×1.46	4.6×1.7×1.5	6, 9, 10, 11
15	Mercedes Benz	GL Class	5.12×1.93×1.85	5.1×1.9×1.9	(No candidates)
16	Mini	Coopper R50	3.63×1.69×1.42	3.6×1.7×1.5	2, 4
17	Opel	Astra Sport	4.37×1.81×1.49	4.7×1.8×1.5	6, 7, 9, 10, 11
18	Renault	Kangoo Express	4.04×2.03×1.80	4.0×1.8×1.8	1
19	Renault	Megane	4.35×1.88×1.44	4.3×1.8×1.5	1, 3, 8, 9
20	SEAT	IBIZA	4.06×1.78×1.44	4.3×1.8×1.5	1, 3, 8, 9
21	Toyota	Avensis	4.75×1.81×1.48	4.7×1.8×1.5	6, 7, 9, 10, 11
22	Toyota	Prius	4.54×1.76×1.49	4.5×1.7×1.5	3, 6, 8, 9, 10, 11
23	Volkswagen	Up!	3.60×1.65×1.50	3.6×1.6×1.5	2, 4
24	Volvo	S40	4.52×1.72×1.43	4.5×1.8×1.5	3, 6, 8, 9, 10, 11

*: From manufacturer websites, Wikipedia, or automobiledimension.com.

Figure 10 shows the results of the 87 semantic registration trials. The best-fit car model for each object is circled. For example, the most similar model of obj_1, out of the 24 cars, was Honda Jazz, a hatchback without a trunk. We also tested the pairwise registrations for non-candidate objects, the results of which are shown in the background heatmap. It can be found that filtering the registrations using cross-sections was significant so that the optimal registrations were found when the computational load was reduced dramatically. The average computational time for each pairwise registration was 0.340s, which saved 76.0% computational time from bypassing Step 2 (1.415s; (Xue et al. 2019c).

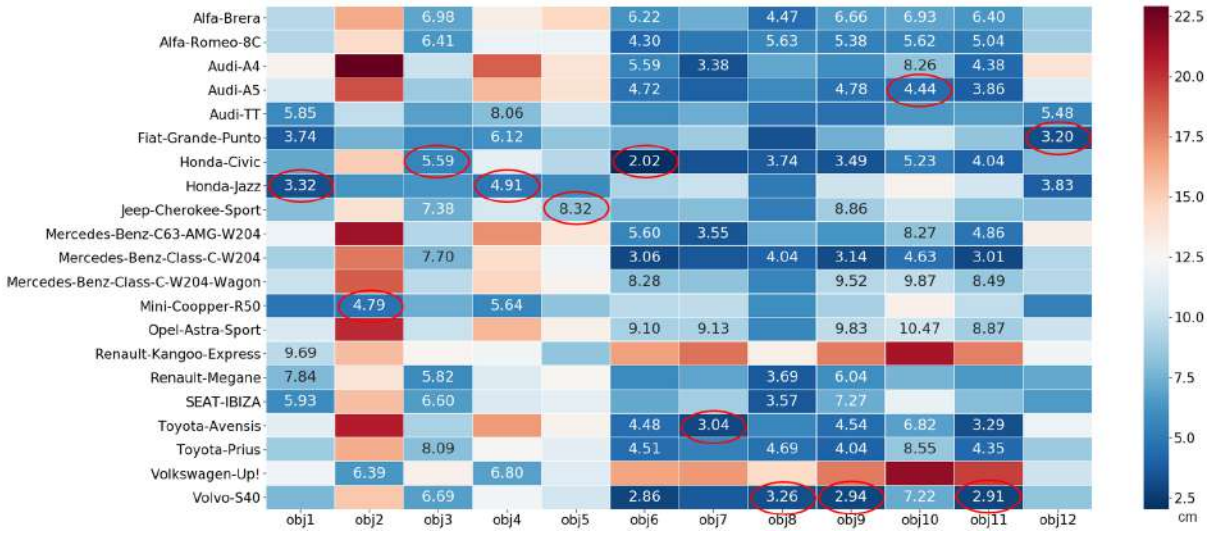

Figure 10: Results of COSCO-assisted model-to-object semantic registration (the best in circles).

A digital twin was then created as a 12.45MB Stanford Polygon (.ply) model from the optimal registrations, as shown in Figure 11. The location and heading direction of each car were referenced to the symmetric centroid and rotation (see Table 2) and the optimal shifts for the registration. Figure 11a shows that 12 car models were registered to the object patches. There were 5 hatchbacks with trunks, 5 hatchbacks without trunks, 1 supermini, and 1 SUV in Figure 11a. In the registration, the meaningful semantics, such as class, brand, model, production, and performance, can also be enriched to the objects. Due to the reference CAD models, the digital twin was with high-quality geometry—much higher than the input LiDAR. The digital twin can be rendered in many ways, as shown in Figure 11b. Then, the digital twin was geo-referenced to the location of the car park, i.e., latitude = 53.351160°, longitude = −6.261601° (WGS1984 system) in Dublin and displayed in the digital twin city view in 3D GIS, such as Google Earth (Figure 11c) or Cesium. It should be noted that some properties of the digital twin, including small geometric details and colors, can be lost during the multi-step format conversion (e.g., to a small .kmz file for Google Earth).

Figure 11: Digital twin results. (a) Registered 10 hatchbacks, 1 super mini, and 1 SUV models, (b) Rendered 3D view, (c) DTC view on Google Earth, (d) Ground truth.

Figure 11d shows the ground truth of the car park from an aerial view after the LiDAR data was collected. By comparing the digital twin with the actual scene, it can be found that:

(i) The location, direction, types, and sizes of the cars were correct;

(ii) Most of the car paint colors were incorrectly inherited from CAD models; and

(iii) Some models with more than 4cm registration errors in Figure 10, such as the Mini Cooper (to obj_2), were wrongly registered.

The first finding validated the COSCO approach. The reason for the second finding was that the input LiDAR was colorless. Otherwise, it is possible to change the paint colors automatically using parametric material parameters in the CAD models. The primary reason for the third finding was the limited number of CAD models, e.g., only two (i.e., Mini Cooper and Volkswagen Up!) in the 24 models were in the supermini class. A full library of city CAD objects can solve the issues as identified in observation (iii).

5 Discussion

The proposed COSCO approach can resolve a critical issue of the taxonomy of unknown city objects through exploiting the Gestalt design principles. The use of symmetric cross-sections for grouping city objects and creating DTC is an interpretable 'white-box' method that can be understood and verified by human. Furthermore, the efficiency and low-level pre-requisites of COSCO make it possible to pre-process city-scale point clouds to cross-sections for facilitating other DTC methods such as semantic registration. When a time-series LiDAR point cloud, or a 4D LiDAR point cloud video, is available, the output of each frame includes as-designed cross-sections as succinct geometric data, excluding the static background environment. A new frame of DTC can refer to the previous one from a cross-frame clustering of the cross-sections. Therefore, the motion trails of movable objects like cars and airplanes can be tracked with constant cross-section shapes and changing positions and heading directions, while the regular updates on construction sites and overhead advertisements are reflected as permanent positions with minor changes in cross-section shapes.

The proposed COSCO method has advantages in several aspects.

- First, the COSCO method applies a combination of Gestalt design principles to process LiDAR point clouds for DTC creation. The digital twinning process benefits from the LiDAR's observer perspective and the producer and designer perspective. Some types of noises, such as small Gaussian noises from sensors, can be corrected in part using the symmetry (see Figure 4b).
- Secondly, the results of COSCO include symmetric objects and their cross-sections from large-scale urban LiDAR data. The cross-sections can reveal their positions, poses, and coarse 3D geometric information with the minified volume of 2D data. These symmetric objects are more interesting than the asymmetric patches to many studies in AECO. For example, symmetries can guide the detection of buildings, archeological sites, and culture heritages (Schofield 2009; Xue et al. 2019b), facilitate urban design and planning (Park 2011; Salat et al. 2014), and correct data imperfections, such as random noises, small clutters, and wrong segmentations (Xue et al. 2019d). The associated properties, including positions, poses, and insights, are essential knowledge to a DTC and its applications.
- Thirdly, COSCO utilizes the symmetric cross-sections in innate designs of city

objects and the latest algorithmic development to achieve automated and fast processing. Experiments showed that symmetric cross-sections make COSCO about ten times faster.

- Finally, the unsupervised nature of COSCO makes it inexpensive and with an evolvable performance over time. The low requirements for the input data make it easy to adapt to new environments and integrate with other digital twinning methods. For example, the results of COSCO were applied to a semantic registration approach in Sect. 4.3.

There are also several drawbacks in this paper:

- First, COSCO requires symmetry of city objects in LiDAR point clouds and utilizes the symmetry within the host object's geometry. However, the possible higher order of symmetry and regularities among multiple city objects, such as regular rows of street trees and matrix of building blocks, is not yet covered.
- Furthermore, COSCO is limited to process symmetric city objects with incomplete or auxiliary geometries scanned in LiDAR point clouds. For example, half of a car is occluded by tree canopies from an aerial LiDAR scanner; a truck was scanned with asymmetric goods loaded on the trailer.
- The hierarchical clustering in this paper is 'cross-sectional' between city objects; while the longitudinal clustering, which can relate a city object to its forms in previous versions of DTCs, is not mentioned.
- The experiments were in a small scale for the proof of the concept. Thus, the parameters analyzed in the experiments may subject to change in other experimental settings.
- The detected cross-sections and semantics of objects may be slightly inconsistent between multiple and distributed observers or stakeholders of the city objects. The inconsistency can lead to collisions in distributed DTC updates and possible disputes like authorships over co-created objects. Blockchain is an emerging distributed ledger technology that can safeguard the semantics. Xue and Lu (2020) proved a semantic difference approach could blockchain large-scale BIM through the lifecycle by extracting the full semantic changes between BIM versions. Likely, a semantic difference study between 'frames' of a DTC can contribute to blockchaining the DTC.

6. Conclusion

Developing the digital twins of cities from LiDAR point clouds is one of the most heated research fields in the era of smart city. On the one hand, thriving remote sensing technologies have enabled rich, accurate, and inexpensive measurement data (e.g., 3D LiDAR point clouds) of cities. On the other hand, the scale of the problem, unknown taxonomy, and dynamic features of city objects (e.g., construction sites, roads, street light, and vehicles) all challenge the task of Digital Twin City (DTC) creation and near-real-time updates. Our

research offered an unsupervised Clustering Of Symmetric Cross-sections of Objects (COSCO) to recognize symmetric city objects by harnessing the power of Gestalt design principles. It is positioned in the stream of research on LiDAR data processing. It follows the notion of using city objects' intrinsic and conventional features (e.g., symmetries) to optimize the task. Nonetheless, the research goes beyond the individual city features by harnessing the wider Gestalt design principles, such as the laws of proximity, connectivity, symmetry, and similarity.

In a nutshell, COSCO incorporates the Gestalt design principles in three key steps: (i) connectivity-based object detection, (ii) symmetric cross-section detection, and (iii) dissimilarity matrix-based hierarchical clustering. The outputs include the taxonomy, as well as the poses of the city objects. By applying the taxonomy, the unknown city objects can be grouped into types with relatedness between them identified. Furthermore, a group of objects can be mapped to the same parametric 3D reference model with some parameters, such as color specialization. A DTC can be effectively created by engaging other methods, such as semantic registration, with the output taxonomy and poses. Experimental results showed that COSCO detected the correct taxonomy and types of 12 cars in 24,126 LiDAR points in 8.28s. Based on the cross-sections and taxonomy, a digital twin was created by registering online free 3D car models in 29.58s. The COSCO method has several advantages, including harnessing the power of Gestalt design principles, recognizing symmetric cross-sections, taxonomy and grouping of unknown objects, efficiency in terms of computational time, and low requirement and easy adaption to new tasks. Yet, the research also has its limitations in levels of symmetries, LiDAR data quality, time-series LiDAR processing, and scale of experiments.

The contribution of this paper is twofold. First, COSCO is an effective unsupervised method to understand city objects in LiDAR point clouds by exploiting the free innate urban design principles rather than expensive training examples. Secondly, COSCO can be an efficient pre-processor for recognizing symmetric city objects' positions, heading directions, cross-sections, and types in LiDAR data for DTCs and smart city applications. Future work of this study includes (i) expanding the symmetry cross-sections to higher-order urban regularities in the systems of city objects, (ii) handling incomplete and auxiliary LiDAR data, (iii) processing time series LiDAR data to 4D DTCs, (iv) automatically selecting and adapting algorithms and parameters for various scenes, (v) integrating into existing data standards and software related to DTC, and (vi) blockchaining DTCs by ledgering the high-level semantics rather than big data of city objects .

Acknowledgments

The study was supported by the General Research Fund from Hong Kong Research Grant Council (Grant No. 17200218). We would like to express our gratitude to the anonymous

reviewers for the constructive suggestions to improve the quality of this paper.

References

Ahmed, S. M. & Chew, C. M. (2020). Density-Based Clustering for 3D Object Detection in Point Clouds. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 10608-10617). IEEE.

Andreopoulos, A. & Tsotsos, J. K. (2013). 50 years of object recognition: Directions forward. *Computer Vision and Image Understanding, 117*(8), 827-891. doi:10.1016/j.cviu.2013.04.005

Armesto-González, J. B.-R.-A.-B. (2010). Terrestrial laser scanning intensity data applied to damage detection for historical buildings. *Journal of Archaeological Science, 37*(12), 3037-3047. doi:10.1016/j.jas.2010.06.031

Arnheim, R. (1965). *Art and visual perception: A psychology of the creative eye.* Oakland, CA, USA: University of California Press.

Arora, P., Deepali, D. & Varshney, S. (2016). Analysis of k-means and k-medoids algorithm for big data. *Procedia Computer Science, 78*, 507-512. doi:10.1016/j.procs.2016.02.095

Austin, M., Delgoshaei, P., Coelho, M. & Heidarinejad, M. (2020). Architecting Smart City Digital Twins: Combined Semantic Model and Machine Learning Approach. *Journal of Management in Engineering, 36*(4), 04020026. doi:10.1061/(ASCE)ME.1943-5479.0000774

Babacan, K., Chen, L. & Sohn, G. (2017). Semantic segmentation of indoor point clouds using Convolutional Neural Network. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, IV-4*(W4), 101-108. doi:10.5194/isprs-annals-IV-4-W4-101-2017

Boje, C., Guerriero, A., Kubicki, S. & Rezgui, Y. (2020). Towards a semantic Construction Digital Twin: Directions for future research. *Automation in Construction, 114*, 103179. doi:10.1016/j.autcon.2020.103179

BuHamdan, S., Alwisy, A. & Bouferguene, A. (2020). Explore the application of reinforced learning to support decision making during the design phase in the construction industry. *Procedia Manufacturing, 42*, 181-187. doi:10.1016/j.promfg.2020.02.068

Burrough, P. A., McDonnell, R., McDonnell, R. A. & Lloyd, C. D. (2015). *Principles of geographical information systems.* Oxford, UK: Oxford university press.

Chen, M., Feng, A., McAlinden, R. & Soibelman, L. (2020). Photogrammetric Point Cloud

Segmentation and Object Information Extraction for Creating Virtual Environments and Simulations. *Journal of Management in Engineering, 36*(2), 04019046. doi:10.1061/(ASCE)ME.1943-5479.0000737

Czerniawski, T., Sankaran, B., Nahangi, M., Haas, C. & Leite, F. (2018). 6D DBSCAN-based segmentation of building point clouds for planar object classification. *Automation in Construction, 88*, 44-58. doi:10.1016/j.autcon.2017.12.029

Du, J., Zhu, Q., Shi, Y., Wang, Q., Lin, Y. & Zhao, D. (2020). Cognition digital twins for personalized information systems of smart cities: Proof of concept. *Journal of Management in Engineering, 36*(2), 04019052. doi:10.1061/(ASCE)ME.1943-5479.0000740

Du, S., Luo, L., Cao, K. & Shu, M. (2016). Extracting building patterns with multilevel graph partition and building grouping. *ISPRS Journal of Photogrammetry and Remote Sensing, 122*, 81-96. doi:10.1016/j.isprsjprs.2016.10.001

El-Mahgary, S., Virtanen, J. P. & Hyyppä, H. (2020). A Simple Semantic-Based Data Storage Layout for Querying Point Clouds. *ISPRS International Journal of Geo-Information, 9*(2), 72. doi:10.3390/ijgi9020072

Fan, C., Jiang, Y. & Mostafavi, A. (2020). Social Sensing in Disaster City Digital Twin: Integrated Textual–Visual–Geo Framework for Situational Awareness during Built Environment Disruptions. *Journal of Management in Engineering, 36*(3), 04020002. doi:10.1061/(ASCE)ME.1943-5479.0000745

Francisco, A., Mohammadi, N. & Taylor, J. E. (2020). Smart City Digital Twin–Enabled Energy Management: Toward Real-Time Urban Building Energy Benchmarking. *Journal of Management in Engineering, 36*(2), 04019045. doi:10.1061/(ASCE)ME.1943-5479.0000741

Grieves, M. & Vickers, J. (2017). Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems. In F.-J. Kahlen, S. Flumerfelt & A. Alves, *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches* (pp. 85-113). Springer.

He, X., Zhang, X. & Xin, Q. (2018). Recognition of building group patterns in topographic maps based on graph partitioning and random forest. *ISPRS Journal of Photogrammetry and Remote Sensing, 136*, 26-40. doi:10.1016/j.isprsjprs.2017.12.001

Hidaka, N., Michikawa, T., Motamedi, A., Yabuki, N. & Fukuda, T. (2018). Polygonization of point clouds of repetitive components in civil infrastructure based on geometric

similarities. *Automation in Construction, 86*, 99-117. doi:10.1016/j.autcon.2017.10.014

Hong, S., Jung, J., Kim, S., Cho, H., Lee, J. & Heo, J. (2015). Semi-automated approach to indoor mapping for 3D as-built building information modeling. *Computers, Environment and Urban Systems, 51*, 34-46. doi:10.1016/j.compenvurbsys.2015.01.005

Huang, R., Xu, Y., Hong, D., Yao, W., Ghamisi, P. & Stilla, U. (2020). Deep point embedding for urban classification using ALS point clouds: A new perspective from local to global. *ISPRS Journal of Photogrammetry and Remote Sensing, 163*, 62-81. doi:j.isprsjprs.2020.02.020

Huber, D., Akinci, B., Oliver, A. A., Anil, E., Okorn, B. E. & Xiong, X. (2011). Methods for automatically modeling and representing as-built building information models. *Proceedings of the NSF CMMI Research Innovation Conference.* Retrieved December 18, 2019, from https://ri.cmu.edu/pub_files/2011/1/2011-huber-cmmi-nsf-v4.pdf

Jung, J., Stachniss, C., Ju, S. & Heo, J. (2018). Automated 3D volumetric reconstruction of multiple-room building interiors for as-built BIM. *Advanced Engineering Informatics, 38*, 811-825. doi:10.1016/j.aei.2018.10.007

Kitchin, R. (2014). The real-time city? Big data and smart urbanism. *GeoJournal, 79*(1), 1-14. doi:10.1007/S10708-013-9516-8

Laefer, D. F., Abuwarda, S., Vo, A.-V., Truong-Hong, L. & Gharibi, H. (2017). *2015 Aerial Laser and Photogrammetry Survey of Dublin City Collection Record.* New York, USA: NYU Spatial Data Repository. doi:10.17609/N8MQ0N

Lee, G., Choi, B., Ahn, C. R. & Lee, S. (2020). Wearable Biosensor and Hotspot Analysis–Based Framework to Detect Stress Hotspots for Advancing Elderly's Mobility. *Journal of Management in Engineering, 36*(3), 04020010. doi:10.1061/(ASCE)ME.1943-5479.0000753

Lehner, H. & Dorffner, L. (2020). Digital geoTwin Vienna: Towards a Digital Twin City as Geodata Hub. *PFG, 88*, 63-75. doi:10.1007/s41064-020-00101-4

Li, Y., Chen, D., Du, X., Xia, S., Wang, Y., Xu, S. & Yang, Q. (2019). Higher-order conditional random fields-based 3D semantic labeling of airborne laser-scanning point clouds. *Remote Sensing, 11*(10), 1248. doi:10.3390/rs11101248

Li, Z., Yan, H., Ai, T. & Chen, J. (2004). Automated building generalization based on urban morphology and Gestalt theory. *International Journal of Geographical Information Science, 18*(5), 513-534. doi:10.1080/13658810410001702021

Ma, J. W., Czerniawski, T. & Leite, F. (2020). Semantic segmentation of point clouds of building interiors with deep learning: Augmenting training datasets with synthetic BIM-based point clouds. *Automation in Construction, 113*, 103144. doi:10.1016/j.autcon.2020.103144

Mi, C. S. (2015). Ship identification algorithm based on 3D point cloud for automated ship loaders. *Journal of Coastal Research*(73), 28-34. doi:10.2112/SI73-006.1

Mohri, M., Rostamizadeh, A. & Talwalkar, A. (2018). *Foundations of machine learning* (2nd ed.). Cambridge, MA, USA: MIT press.

NIBS. (2015). *National Building Information Modeling Standard (Version 3).* National Institute of Building Sciences. Retrieved from https://www.nationalbimstandard.org/

NIC. (2017). *Data for the Public Good.* London: National Infrastructure Commission, UK. Retrieved from https://www.nic.org.uk/publications/data-public-good/

Pan, Y., Dong, Y., Wang, D., Chen, A. & Ye, Z. (2019). Three-dimensional reconstruction of structural surface model of heritage bridges using UAV-based photogrammetric point clouds. *Remote Sensing, 11*(10), 1204. doi:10.3390/rs11101204

Papon, J., Abramov, A., Schoeler, M. & Worgotter, F. (2013). Voxel cloud connectivity segmentation-supervoxels for point clouds. *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2027-2034). IEEE. doi:10.1109/CVPR.2013.264

Park, J. J. (2011). Dynamics of urban centre and concepts of symmetry: Centroid and weighted mean. *Nexus Network Journal, 13*(2), 397-410. doi:10.1007/s00004-011-0073-5

Pătrăucean, V., Armeni, I., Nahangi, M., Yeung, J., Brilakis, I. & Haas, C. (2015). State of research in automatic as-built modelling. *Advanced Engineering Informatics, 29*(2), 162-171. doi:10.1016/j.aei.2015.01.001

Pauly, M., Gross, M. & Kobbelt, L. P. (2002). Efficient simplification of point-sampled surfaces. *Proceedings of the IEEE conference on Visualization'02* (pp. 163-170). IEEE. doi:10.1109/VISUAL.2002.1183771

Poux, F. & Billen, R. (2019). Voxel-based 3D point cloud semantic segmentation: unsupervised geometric and relationship featuring vs deep learning methods. *ISPRS International Journal of Geo-Information, 8*(5), 213. doi:10.3390/ijgi8050213

Previtali, M., Barazzetti, L., Brumana, R. & Scaioni, M. (2014). Towards automatic indoor reconstruction of cluttered building rooms from point clouds. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2*(5), 281-288.

doi:10.5194/isprsannals-II-5-281-2014

Rosser, J. F., Boyd, D. S., Long, G., Zakhary, S., Mao, Y. & Robinson, D. (2019). Predicting residential building age from map data. *Computers, Environment and Urban Systems, 73*, 56-67. doi:10.1016/j.compenvurbsys.2018.08.004

Rusu, R. B., Blodow, N. & Beetz, M. (2009). Fast point feature histograms (FPFH) for 3D registration. *Proceedings of the 2009 IEEE international conference on robotics and automation* (pp. 3212-3217). IEEE.

Sacks, R., Eastman, C. M., Lee, G. & Teicholz, P. (2018). *BIM Handbook: A guide to Building Information modeling for owners, designers, engineers, contractors, and facility managers* (3rd ed.). Hoboken, NJ, USA: John Wiley & Sons.

Salat, S., Bourdic, L. & Labbe, F. (2014). Breaking symmetries and emerging scaling urban structures: A morphological tale of 3 cities: Paris, New York and Barcelona. *Archnet-IJAR, 8*(2), 77-93. doi:10.26687/archnet-ijar.v8i2.445

Samie Tootooni, M., Dsouza, A., Donovan, R., Rao, P. K., Kong, Z. J. & Borgesen, P. (2017). Classifying the dimensional variation in additive manufactured parts from laser-scanned three-dimensional point cloud data using machine learning approaches. *Journal of Manufacturing Science and Engineering, 139*(9), 091005. doi:10.1115/1.4036641

Sarmad, M., Lee, H. J. & Kim, Y. M. (2019). RL-GAN-Net: A Reinforcement Learning Agent Controlled GAN Network for Real-Time Point Cloud Shape Completion. *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 5891-900). IEEE. doi:10.1109/CVPR.2019.00605

Schofield, J. (2009). Being autocentric: Towards symmetry in heritage management practices. In L. Gibson & J. Pendlebury, *Valuing Historic Environments* (pp. 93-114). Surrey, UK: Ashgate.

Schrotter, G. & Hürzeler, C. (2020). The Digital Twin of the City of Zurich for Urban Planning. *PFG, 88*, 99-112. doi:10.1007/s41064-020-00092-2

Shi, B. Q., Liang, J. & Liu, Q. (2011). Adaptive simplification of point cloud using k-means clustering. *Computer-Aided Design, 43*(8), 910-922. doi:10.1016/j.cad.2011.04.001

Son, H., Kim, C. & Kim, C. (2015). 3D reconstruction of as-built industrial instrumentation models from laser-scan data and a 3D CAD database based on prior knowledge. *Automation in Construction, 49*, 193-200. doi:10.1016/j.autcon.2014.08.007

Sun, H., Liu, S., Zhong, R. & Du, L. (2020). Cross-section deformation analysis and visualization of shield tunnel based on mobile tunnel monitoring system. *Sensors,*

20(4), 1006. doi:10.3390/s20041006

Tian, Y., Luo, A., Sun, X., Ellis, K., Freeman, W. T., Tenenbaum, J. B. & Wu, J. (2019). Learning to infer and execute 3d shape programs. *ArXiv preprint*, 1901.02875.

Todorovic, D. (2008). Gestalt principles. *Scholarpedia, 3*(12), 5345. doi:10.4249/scholarpedia.5345

Ullah, Z., Al-Turjman, F., Mostarda, L. & Gagliardi, R. (2020). Applications of Artificial Intelligence and Machine learning in smart cities. *Computer Communications, 154*, 313-323. doi:10.1016/j.comcom.2020.02.069

Valero, E., Adán, A. & Cerrada, C. (2012). Automatic method for building indoor boundary models from dense point clouds collected by laser scanners. *Sensors, 12*(12), 16099-16115. doi:10.3390/s121216099

Wang, D., Kankare, V., Puttonen, E., Hollaus, M. & Pfeifer, N. (2017). Reconstructing stem cross section shapes from terrestrial laser scanning. *IEEE Geoscience and Remote Sensing Letters, 14*(2), 272-276. doi:10.1109/LGRS.2016.2638738

Wang, Q. & Kim, M. K. (2019). Applications of 3D point cloud data in the construction industry: A fifteen-year review from 2004 to 2018. *Advanced Engineering Informatics, 39*, 306-319. doi:10.1016/j.aei.2019.02.007

Wang, W., Yu, R., Huang, Q. & Neumann, U. (2018). SGPN: Similarity group proposal network for 3D point cloud instance segmentation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 2569-2578). IEEE. doi:10.1109/CVPR.2018.00272

Xie, Y., Tian, J. & Zhu, X. X. (2020). Linking points with labels in 3D: A review of point cloud semantic segmentation. *IEEE Geoscience and Remote Sensing Magazine*, in press. doi:10.1109/MGRS.2019.2937630

Xiong, X., Adan, A., Akinci, B. & Huber, D. (2013). Automatic creation of semantically rich 3D building models from laser scanner data. *Automation in Construction, 31*, 325-337. doi:10.1016/j.autcon.2012.10.006

Xu, X., Ding, L., Luo, H. & Ma, L. (2014). From building information modeling to city information modeling. *Journal of Information Technology in Construction, 19*, 292-307. Retrieved from http://www.itcon.org/2014/17

Xu, Y., Tuttas, S., Hoegner, L. & Stilla, U. (2018a). Voxel-based segmentation of 3D point clouds from construction sites using a probabilistic connectivity model. *Pattern Recognition Letters, 102*, 67-74. doi:10.1016/j.patrec.2017.12.016

Xu, Y., Yao, W., Tuttas, S., Hoegner, L. & Stilla, U. (2018b). Unsupervised segmentation of

point clouds from buildings using hierarchical clustering based on Gestalt principles. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 11*(11), 4270-4286. doi:10.1109/JSTARS.2018.2817227

Xue, F. & Lu, W. (2020). A semantic differential transaction approach to minimizing information redundancy for BIM and blockchain integration. *Automation in Construction, 118*, 103270. doi:10.1016/j.autcon.2020.103270

Xue, F., Chen, K. & Lu, W. (2019c). Understanding unstructured 3D point clouds for creating digital twin city: An unsupervised hierarchical clustering approach. *Proceedings of the CIB World Building Congress 2019.* Retrieved from https://site.cibworld.nl/dl/publications/WBC19/WBC_Proceedings_June2019_Complete.pdf

Xue, F., Lu, W. & Chen, K. (2018). Automatic generation of semantically rich as-built building information models using 2D images: A derivative-free optimization approach. *Computer-Aided Civil and Infrastructure Engineering, 33*(11), 926-942. doi:10.1111/mice.12378

Xue, F., Lu, W. & Chen, K. (2019d). BIM reconstruction from 3D point clouds: A semantic registration approach based on multimodal optimization and architectural design knowledge. *Advanced Engineering Informatics, 42*, 100965. doi:10.1016/j.aei.2019.100965

Xue, F., Lu, W., Chen, K. & Zetkulic, A. (2019a). From semantic segmentation to semantic registration: Derivative-free optimization-based approach for automatic generation of semantically rich as-built building information models from 3D point clouds. *Journal of Computing in Civil Engineering, 33*(4), 04019024. doi:10.1061/(ASCE)CP.1943-5487.0000839

Xue, F., Lu, W., Webster, C. & Chen, K. (2019b). A derivative-free optimization-based approach for detecting architectural symmetries from 3D point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing, 148*, 32-40. doi:10.1016/j.isprsjprs.2018.12.005

Yang, Y., Chen, F., Wu, F., Zeng, D., Ji, Y. M. & Jing, X. Y. (2020a). Multi-view semantic learning network for point cloud based 3D object detection. *Neurocomputing, 397*, 477-485. doi:10.1016/j.neucom.2019.10.116

Yang, Z., Jiang, W., Lin, Y. & Elberink, S. O. (2020b). Using training samples retrieved from a topographic map and unsupervised segmentation for the classification of airborne laser scanning Data. *Remote sensing, 12*(5), 877. doi:10.3390/rs12050877

Yu, Y., Huang, Z., Li, F., Zhang, H. & Le, X. (2020). Point Encoder GAN: A deep learning model for 3D point cloud inpainting. *Neurocomputing, 384*, 192-199. doi:10.1016/j.neucom.2019.12.032

Zeng, S., Chen, J. & Cho, Y. K. (2020). User exemplar-based building element retrieval from raw point clouds using deep point-level features. *Automation in Construction, 114*, 103159. doi:10.1016/j.autcon.2020.103159

Zou, C., Colburn, A., Shan, Q. & Hoiem, D. (2018). LayoutNet: Reconstructing the 3D room layout from a single RGB image. *Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2051-2059). IEEE. doi:10.1109/CVPR.2018.00219