Automation in Construction 19 (2010) 829–843

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

Review

Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques

Pingbo Tang [a], Daniel Huber [b,*], Burcu Akinci [c], Robert Lipman [d], Alan Lytle [e]

[a] Western Michigan University, Civil and Construction Engineering Department, Kalamazoo, MI 49008, United States
[b] Carnegie Mellon University, Robotics Institute, 4105 Newell-Simon Hall, Pittsburgh, PA 15213, United States
[c] Carnegie Mellon University, Civil and Environmental Engineering Department, Porter Hall 123K, Pittsburgh, PA 15213, United States
[d] National Institute of Standards and Technology, 100 Bureau Drive, Stop 8630, Gaithersburg, MD 20899, United States
[e] National Institute of Standards and Technology, 100 Bureau Drive, Stop 8611, Gaithersburg, MD 20899, United States

ARTICLE INFO

Article history:
Accepted 7 June 2010

Keywords:
Building information models
Building reconstruction
Laser scanners
Object recognition
Geometric modeling
Relationship modeling
Shape representation

ABSTRACT

Building information models (BIMs) are maturing as a new paradigm for storing and exchanging knowledge about a facility. BIMs constructed from a CAD model do not generally capture details of a facility as it was actually built. Laser scanners can be used to capture dense 3D measurements of a facility's as-built condition and the resulting point cloud can be manually processed to create an as-built BIM — a time-consuming, subjective, and error-prone process that could benefit significantly from automation. This article surveys techniques developed in civil engineering and computer science that can be utilized to automate the process of creating as-built BIMs. We sub-divide the overall process into three core operations: geometric modeling, object recognition, and object relationship modeling. We survey the state-of-the-art methods for each operation and discuss their potential application to automated as-built BIM creation. We also outline the main methods used by these algorithms for representing knowledge about shape, identity, and relationships. In addition, we formalize the possible variations of the overall as-built BIM creation problem and outline performance evaluation measures for comparing as-built BIM creation algorithms and tracking progress of the field. Finally, we identify and discuss technology gaps that need to be addressed in future research.

© 2010 Elsevier B.V. All rights reserved.

Contents

* Corresponding author. Tel.: +1 412 268 2991.
E-mail addresses: tangpingbo@gmail.com (P. Tang), dhuber@cs.cmu.edu (D. Huber), bakinci@andrew.cmu.edu (B. Akinci), robert.lipman@nist.gov (R. Lipman), alan.lytle@nist.gov (A. Lytle).

0926-5805/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.autcon.2010.06.007

1. Introduction

Semantically rich digital facility models, known as building information models (BIMs), are gaining attention in the Architecture, Construction, Engineering, and Facility Management (AEC/FM) community due to their ability to enhance communication between the various stakeholders involved in the different stages of a facility's life cycle [34] and their multitude of potential uses, ranging from improved planning for renovations to more accurate modeling of a building's energy consumption [35]. A BIM represents a facility in a semantically rich manner, unlike a traditional CAD model. Whereas a CAD model would represent a wall as a set of independent planar surfaces, a BIM would represent the wall as a single, volumetric object with multiple surfaces, as well as adjacency relationships between the wall and other entities in the model, the identification of the object as a wall, and other relevant properties (material characteristics, cost, etc.). A BIM can be used, for example, to plan the modification of a process plant with a level of precision that would not be possible otherwise, reducing the possibility of unexpected and costly delays during the construction.

While it is possible to construct a BIM from a CAD-based model of a facility's design (as-designed condition), such a model does not generally capture detailed depictions of the state of a facility as it was actually built (as-built condition) or as it exists currently (as-is condition). A facility may not be constructed exactly as the design specified, changes may be made during subsequent renovations, or a design model of an existing facility may simply not exist. These observations point to the need for methods to create BIMs that represent a facility's as-built/as-is conditions. We use the term "as-built" to mean "as-built/as-is" for brevity hereafter.

The creation of an as-built BIM involves measuring the geometry and appearance of an existing facility and transforming those measurements into a high-level, semantically rich representation. The goals of this article are threefold: 1) to document the state-of-the-art methods for creating as-built BIMs, which are primarily manual processes; 2) to survey existing techniques within the relevant fields of civil engineering and computer science and evaluate their suitability for automating the as-built BIM creation process; and 3) to identify technology gaps and barriers to automating the process. BIMs incorporate many aspects of a facility's components, such as material characteristics, cost, and behavioral properties. In this article, we focus primarily on methods to model the geometry and functional properties of the components in a facility. The processes described in the remainder of this section are a synthesis of interviews conducted with professional laser scanning service providers, discussions with representatives of laser scanning hardware vendors and BIM creation software vendors, and formal collaborations with domain experts within government agencies, including the General Services Administration (GSA) and the National Institute of Standards and Technologies (NIST).

1.1. Creating as-built BIMs

Laser scanners are gaining acceptance and usage in the AEC/FM industry for creating as-built BIMs because they can rapidly and accurately measure the 3D shape of the environment. The process of creating an as-built BIM using laser scanners can be divided into three main steps: 1) *data collection*, in which dense point measurements of the facility are collected using laser scans taken from key locations throughout the facility; 2) *data preprocessing*, in which the sets of point measurements (known as point clouds) from the collected scans are filtered to remove artifacts and combined into a single point cloud or surface representation in a common coordinate system; and 3) *modeling the BIM*, in which the low-level point cloud or surface representation is transformed into a semantically rich BIM. These steps are described in more detail in the next sub-sections. The focus of this article is on the modeling step, but a brief overview of the first two steps is provided for completeness.

1.1.1. Data collection

Laser scanning technology has made the creation of as-built BIMs tractable. Other methods, such as using a total-station, measuring tapes, or even ordinary cameras, are too time-consuming or inaccurate to be practical on a large scale, though recent results using images are promising [29]. Laser scanners measure the distance from the sensor to nearby surfaces with millimeter to centimeter accuracy at speeds of thousands to hundreds of thousands of point measurements per second (Fig. 1). A single scan may contain several million 3D points. Since no single scanning location can visualize all surfaces within a facility, scans must be obtained from multiple locations. Often, a digital camera is used to capture images of the environment, which can be later fused with the 3D data to aid in the modeling process [6].

1.1.2. Data preprocessing

The point clouds from each scan are initially represented in the scanner's local coordinate frame. All of the data need to be aligned in a common, global coordinate system through a process known as registration. Although automated registration methods have been developed [43], in current practice, registration is still a semi-automated process. Typically, the user must manually identify, in the 3D data, the approximate locations of specialized targets that have been placed in the environment to aid in the registration [34]. Data preprocessing may also include manual or automated filtering to remove unwanted data, such as points from moving objects, reflections, or sensor artifacts. Depending on the modeling algorithm or software used, the point cloud data may be converted into surface data, usually in the form of a triangular surface mesh. This process may be performed at the individual scan level, and then the resulting surfaces are combined [94], or the point clouds can be registered first, and then the surfaces can be estimated from the combined point cloud [41].

1.1.3. Modeling the BIM

Given a point cloud of a facility, the modeling of a BIM involves three tasks: modeling the geometry of the components ("What is the shape of this wall?"), assigning an object category and material properties to a component ("This object is a brick wall."), and establishing relationships between components ("Wall1 is connected to Wall2 at this location."). These tasks do not necessarily take place sequentially, and depending on the workflow, they may be interleaved.

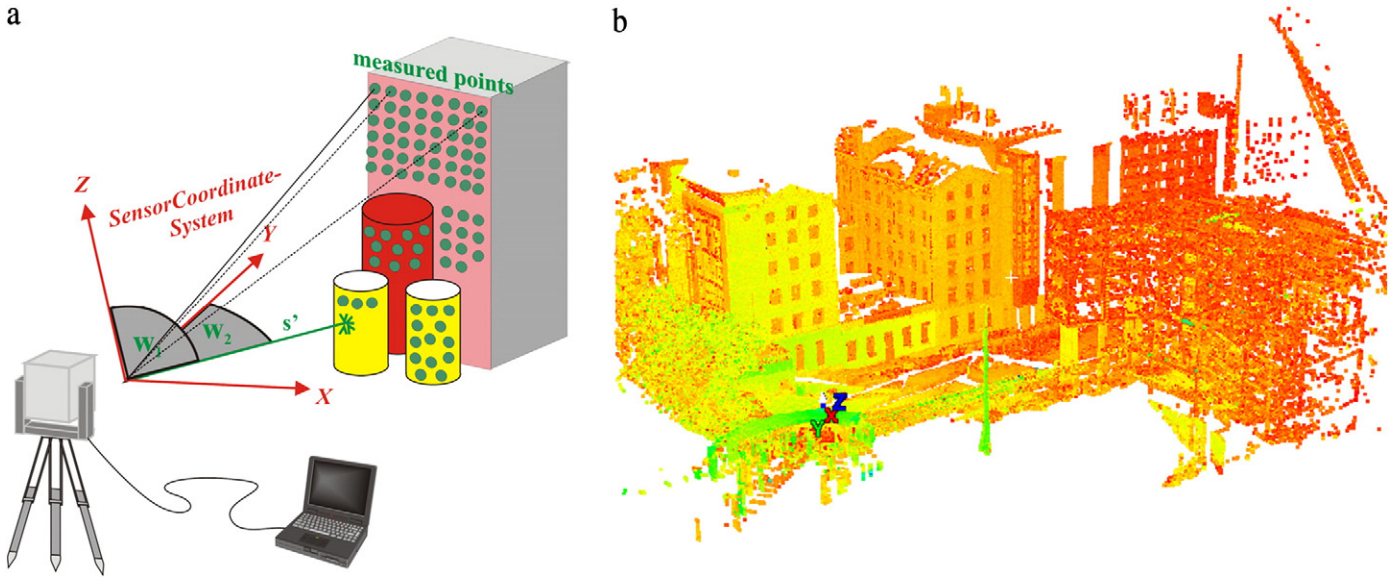

Fig. 1. (a) The laser scanning process for measuring 3D points (from [88], reproduced with permission of R. Staiger). (b) An example of laser-scanned data of a building under construction.

The goal of the geometric modeling task is to create simplified representations of building components by fitting geometric primitives to the point cloud data. Geometric primitives can be individual surfaces or volumetric shapes. For example, a simple wall can be modeled as a planar patch, or it can be a rectangular box (cuboid). Surfaces like moldings or decorative carvings may not be well modeled by a simple geometric primitive. In such cases, different modeling techniques can be used. For linear structures (e.g., moldings), a cross-section of the object can be modeled by fitting splines to the data and then sweeping the cross-section along a trajectory to form the object model [20]. More complex structures (e.g., decorative carvings) may be modeled non-parametrically, using triangle meshes, for example, or they can be modeled from a database of known object models [15]. Since BIMs are normally defined using solid shapes, surface-based representations need to be transformed into solid models.

The modeled components are labeled with an object category. Standard BIM categories include wall, roof, slab, beam, and column [2]. Additionally, custom object categories can be created based on individual project needs. Objects may be further augmented with other meta-data, such as material properties or links to specifications for custom components.

Topological relationships between components, and between components and spaces, are important in a BIM and must be established. Connectivity relationships indicate which objects are connected to one another and where they are connected. For example, adjacent walls will be connected at their boundaries, and walls will be connected to slabs at the bottom. Additionally, containment relationships are used to encode the locations of components that are embedded within one another, such as windows and doors embedded within walls [2,14].

1.2. Manual creation of as-built BIMs

In current practice, the creation of as-built BIMs is largely a manual process performed by service providers who are contracted to scan and model a facility [34]. A project may require several months to complete, depending on the complexity of the facility and modeling requirements [39,70].

Because as-built BIMs are a relatively new concept, commercial software supporting the process is somewhat fragmented and is in a state of rapid transition. No single software tool can accomplish all aspects of the process — reverse engineering tools excel at geometric modeling of surfaces, but lack volumetric representations and BIM-specific capabilities necessary for semantic modeling, while BIM design systems cannot handle the massive data sets from laser scanners. As a result, modelers often shuttle intermediate results back and forth between different software packages during the modeling process, giving rise to the possibility of information loss due to limitations of data exchange standards or errors in the implementation of the standards within the software tools [30].

The workflow for manual creation of as-built BIMs is best illustrated through a simple example. Here, we model a portion of a room using data collected from two scanning positions (Fig. 2). The raw point clouds are registered in a common coordinate system using the techniques described in Section 1.1.2, and then merged together into a single point cloud that serves as the input to the modeling process.

There are two main approaches for geometric modeling (Fig. 3). The first approach is to fit geometric primitives to the 3D data directly (Fig. 3a). Geometric modeling software typically includes tools for fitting geometric primitives, such as planes, cylinders, spheres, and cones to the data, as well as special-purpose tools for modeling pipes [73,76]. These tools are semi-automated and require significant user input. For example, to model a planar surface, the user selects a few points or a patch of data, and a plane will be fitted to the selected data. The planar patch may be extended using a region growing algorithm to the extent that contiguous data lie within a tolerance distance of the initial surface estimate [19]. In this way, approximate boundaries of the patch can be identified, but, in practice, these boundaries can be irregular and inaccurate (Fig. 3a). More regular boundaries can be obtained by intersecting multiple geometric primitives. For example, the intersection of three orthogonal planes representing two walls and the floor forms the corner of a room as well as straight line wall–wall and wall–floor boundaries. Depending on the software, geometric modeling may operate on point clouds or polygonal (usually triangular) surface meshes. At present, BIM design software does not have the capability to convert geometric primitives created with reverse engineering tools into BIM objects directly. Therefore, it is common practice to re-model the geometry within the BIM design environment using the reverse engineered model as a guide. The need to transfer models back and forth between several different software packages gives rise to data interoperability problems as well.

The second geometric modeling approach uses cross-sections and surface extrusion (Fig. 3b). First, horizontal and vertical cross-sections are extracted from the data, and lines are fit to the cross-sections to represent walls and slabs in plan views. Then, vertical cross-sections

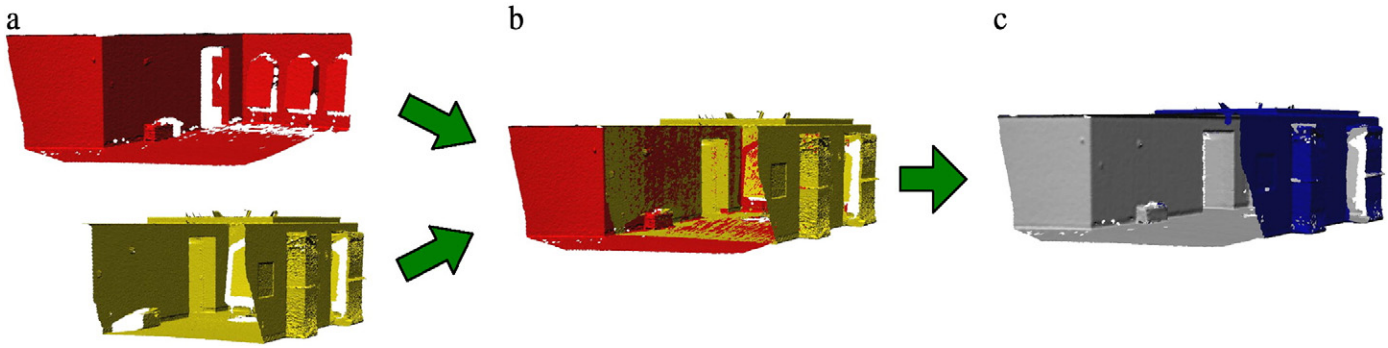

Fig. 2. Scans from individual viewpoints, one shown in red and a second shown in yellow, (a) are aligned in a common coordinate system (b) and then merged into a single representation that contains the data from all scans (c). In real applications, tens to hundreds of scans are needed to model a facility.

are extracted to determine the heights of walls, and any doors and windows, with respect to the floor and ceiling. Finally, walls are modeled by extruding the horizontal cross-section vertically based on the constraints of the vertical cross-sections. This approach is less computationally intensive than the surface-fitting approach, but it can lead to errors when the components do not follow their idealized geometries, for instance, if a wall is not truly vertical.

Various techniques can be used to speed up the modeling process. One instance of a repeated component (e.g., a window) can be modeled initially from data and used as a template to model additional instances. The risk is that different instances may contain slightly different geometry, which would cause geometric modeling errors. Prior knowledge about component geometry, such as the diameter of a column, can be used to constrain the modeling process [16], or the characteristics of known components may be kept in a standard component library.

The category for a BIM component is determined by the modeler when the object is first created within the BIM design software. Relationships between components are established either manually or in a semi-automated manner. For example, when components are created in positions that are touching, the software may automatically connect the two components.

1.3. Automating the creation of as-built BIMs

The manual process for constructing as-built BIMs is time-consuming, labor-intensive, tedious, subjective, and requires skilled workers. Even though modeling of individual geometric primitives can be fairly quick, modeling a facility may require thousands of primitives. The combined modeling time can be several months for an average-sized building [70], which is frequently the bottleneck in the completion

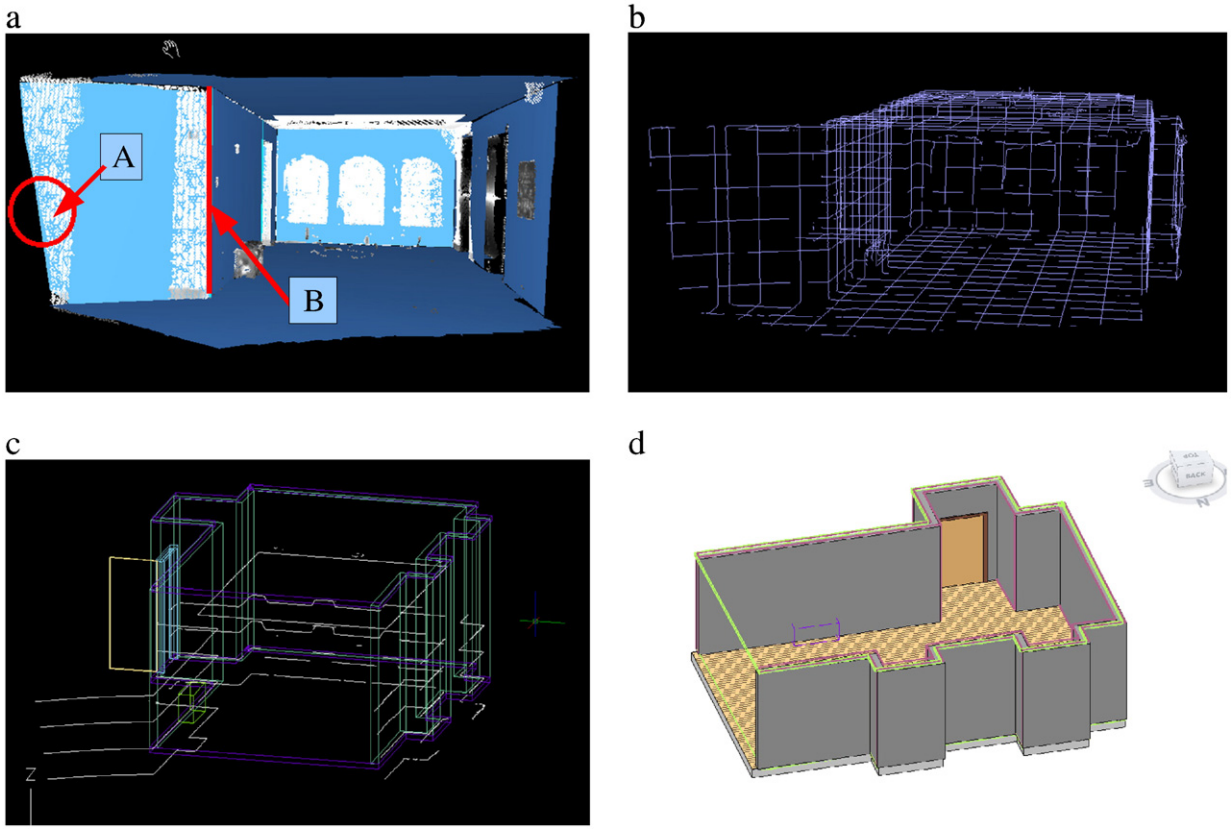

Fig. 3. Examples of methods for reconstructing an as-built BIM from laser scanner data (see text for details). (a) Geometric primitives are fit to the data. Region A highlights an irregular edge, while region B highlights an edge formed from region intersections. (b) Modeling using cross-sections of the data. (c) The results of cross-section modeling. (d) The geometry of the resulting as-built BIM.

of an as-built BIM creation project. Since the same types of primitives must be modeled throughout a facility, the steps are highly repetitive and tedious [36]. Yet, the modeling tools are complex, and unique situations occur frequently enough that modelers must be highly skilled and require specialized training to be proficient [70]. Even with training, the decisions about exactly what elements to model and how to model them are sufficiently subjective that there can be significant variability in the quality of the models produced by different personnel.

These observations illustrate the need to streamline the as-built BIM process using semi-automated and automated techniques. Ideally, a system could be developed that would take a point cloud of a facility as input and produce a fully annotated as-built BIM of the facility as output. This is a challenging problem for several reasons. Facilities can be complex environments, often with numerous unrelated objects, such as furniture and wall-hangings, which obscure the view of the components to be modeled. These unrelated objects (known as clutter) typically are not required to be included in a BIM, and the surfaces that are occluded result in incomplete BIM representations unless assumptions are made about them (e.g., walls extend until they touch the floor). Even without clutter and occlusions, the geometry of a facility can be very complex, geometrically, with window and door moldings, light fixtures, and other components.

Depending on the information requirements of project participants as well as the context of a project, the problem of as-built BIM reconstruction can have several variants in terms of available inputs and expected outputs. On the input side, additional information about a facility, beyond the raw point cloud data, may be available. This information may be a previously created as-built or as-designed model. We can distinguish between variants by the dimensionality (2D plans and elevations or full 3D CAD models) and the level of semantics in the *a priori* data (e.g., geometry, object labels, object relationships). Such prior information can simplify the BIM reconstruction process because the prior model can be aligned with the collected data, and knowledge gleaned from that prior model can serve as guidance. For example, it is much easier to detect and model a doorway if the design model gives its approximate location. In this article, we primarily focus on the most general case, but Section 4.4 discusses methods for leveraging prior information when it is available.

On the output side, the expected output of the process can vary in terms of the level of detail, types of objects, and level of semantics to be modeled. In some applications, such as spatial program validation, only major building components (e.g., walls and columns) need to be modeled, while in other applications, such as historic building documentation, the details of even small objects are important. Moreover, different use cases involve modeling of different types and complexities of objects, ranging from simple planar walls in an office building to complex assemblies of pipes and equipment in a process plant. Finally, in some cases, the full semantics of a BIM may not be needed, so simpler variations of the problem could omit the modeling of relationships or even object identities, in which case the resulting model would essentially be a CAD model. The modeling accuracy and level of detail required for a particular use case is still an open question, but the GSA provides guidelines for accuracy tolerances ranging from ± 51 mm down to ± 3 mm and artifact sizes ranging from 152 mm down to 13 mm [34].

The remainder of this article focuses on surveying promising techniques from civil engineering research and related fields, such as computer vision and robotics, and analyzing their potential and limitations for automated as-built BIM creation. Section 2 surveys the methods for knowledge representation throughout the BIM creation process. Sections 3 through 5 cover the three core operations: geometric modeling (Section 3), object recognition (Section 4), and relationship modeling (Section 5). We also motivate the need for standard test cases and formal performance evaluation measures to allow different algorithms to be compared objectively (Section 6). Finally, Section 7 concludes with a discussion of the technology gaps and directions for future research.

2. Knowledge representation

In the as-built BIM creation process, three types of knowledge need to be represented: knowledge about the object shapes, knowledge about the identities of objects, and knowledge about the relationships between objects (e.g., floor slabs are perpendicular to walls). This section overviews the range of representations that are commonly used for modeling and recognition, and discusses their technical tradeoffs independently of the algorithms that use them.

2.1. Shape representation

The representation of geometric shapes has been studied extensively [15]. In the context of as-built BIMs, shape representations can be classified along three independent dimensions: parametric versus non-parametric, global versus local, and explicit versus implicit. Parametric representations describe a shape using a model with a small number of parameters. For example, a cylinder may be represented by its radius, its axis, and the start and end points. Non-parametric representations do not have any parameters, per se. A cylinder can be represented non-parametrically using a triangle mesh. Global representations describe the shape of an entire object, while local representations may characterize only a portion. The dimension of explicit versus implicit representations is perhaps the most significant axis for distinguishing shape representations. Explicit representations directly encode the shape of an object (e.g., a triangle mesh), while implicit representations indirectly encode object shape using an intermediate representation (e.g., a histogram of surface normals). Explicit representations are well suited for modeling 3D objects, whereas implicit representations are most often used for 3D object recognition and classification [15,63].

2.1.1. Explicit shape representations

Many people are familiar with explicit shape representations because they are encountered frequently in 3D software packages. Most CAD packages and 3D reverse engineering environments utilize explicit representations to describe objects in terms of surfaces, lines, and points [49].

Explicit representations can be sub-divided into surface-based representations and volumetric representations. The most common surface-based representation is the boundary representation (B-rep), in which a shape is described by a set of surface elements (its surface boundary) along with connectivity information to describe the topological relationships between the elements [5]. A triangle mesh is a kind of B-rep. The faces of the mesh are the surface elements, and the adjacency of the faces encodes the topological relationships. Volumetric representations describe shapes by solid geometries. Constructive solid geometry (CSG) builds complex shapes from simple geometric primitives (e.g., cuboids, cylinders, and spheres) that can be combined using operations such as union and intersection (Fig. 4a) [58]. The main advantage of CSG over B-Rep is that solid blocks are a more intuitive means for manipulation by users [49]. However, CSG is not as flexible as B-rep, since the applicability of CSG is limited by the primitive library that is used [77]. Furthermore, B-rep allows efficient partial object representation, such as partially occluded objects, a situation that occurs commonly in as-built BIM creation [96]. There can be some overlap between surface and volumetric representations. For example, a sphere can be considered either a surface representation or a volumetric one.

Explicit representations may be parametric or non-parametric (Fig. 4b and c). Since parametric representations use a small number of parameters to describe shapes, these representations are inherently more compact than non-parametric representations and, therefore, have lower storage requirements and computational complexity. On the other hand, non-parametric approaches are more general and flexible because they can represent free-form objects. Parametric representations include linear geometric shapes like planes, lines, rectangles, and cuboids (3D boxes). Curved surfaces can also be represented parametrically. The

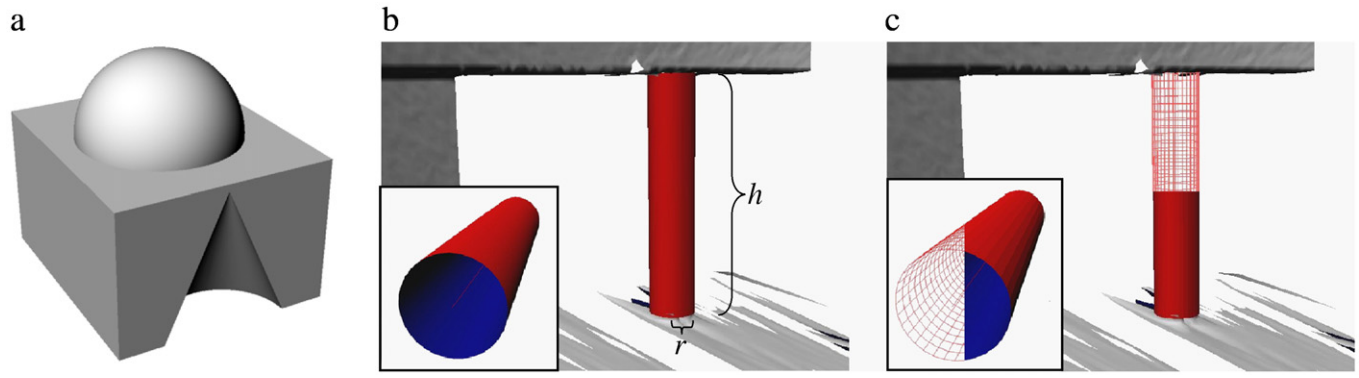

Fig. 4. Explicit shape representations. (a) Objects may be represented volumetrically using constructive solid geometry (CSG), for example. A surface, such as this column (b–c), may be represented parametrically (e.g., by a radius and height) (b) or non-parametrically by a mesh (c). The insets show close-ups of the column model.

simplest curved geometric primitives include circles, ellipses, and arcs in 2D, and spheres, cylinders, cones, and ellipsoids in 3D. More complex curved surfaces can be represented parametrically by Bezier splines or, more commonly, NURBS [60]. NURBS are a generalization of Bezier splines which represent smooth surfaces using a sparse set of control points to govern the surface shape. The most common non-parametric representation is the triangle mesh. Other types of polygonal meshes, such as quadrilateral meshes, are possible but less common. Recently, point-based representations have started to become popular for visualizing 3D data [80]. Since point clouds are the direct output of laser scanners, visualization of point data eliminates the need to create surface representations.

Along the dimension of global versus local, parametric representations are usually used as local representations, and complex shapes must be decomposed into parts using, for example, CSG to represent each part with one or more geometric primitives. In contrast, non-parametric representations, such as triangle meshes, are flexible enough to be used as global representations, since they can easily describe free-form objects in their entirety.

2.1.2. Implicit shape representations

While explicit shape representations can accurately describe the geometries needed for modeling as-built BIMs, they are not very well suited for algorithms that automatically segment or recognize building components or other objects. Consequently, alternate representations are frequently employed. These representations do not directly represent surface shape; instead, they encode shape through features derived from the data or an *a priori* library of shape models. Most implicit representations are non-parametric. They can be categorized as local, global, or an intermediate state that is sometimes called semi-local.

Local representations characterize differential properties of a surface at each location (Fig. 5a). Two common local representations are surface normals and surface curvature. Local representations are frequently used for segmenting a scene into surfaces [93]. For example, points on the same planar surface will all have approximately the same surface normal and will have curvature close to zero. Surface normal estimation can be performed directly on point clouds [41,54] or on polygonal meshes derived from a point cloud [51]. Computing surface normals can be challenging near boundaries, where the underlying surface is discontinuous or changes orientation, because data from other surfaces can bias the estimate [27]. Curvature estimation involves determining the principal curvature directions, which are the tangential surface directions of minimal and maximal curvature, or determining functions of the principal curvature, such as mean or Gaussian curvature [79]. Local representations are fairly tolerant to occlusions and clutter, since the region of support for computing them is usually small, but the small region of support makes them more susceptible to data noise [93].

Global representations describe the shape of an entire object (Fig. 5b). For example, a histogram of surface normals could be used as a descriptor to characterize the shape of a chair, and this descriptor would likely be different from an object with significantly different shape, such as a bookshelf [42]. Global representations are often aggregations of a local property, such as surface distance from the object centroid or surface curvature. Some additional examples of global representations include shape distributions (histograms of properties of point pairs) [69], spherical harmonics (a Fourier space representation) [48], and skeleton structures [45]. With global representations, the region of support includes the entire object, which makes them more susceptible to occlusions and clutter. Generally, methods that use global representations expect objects to be segmented from the background

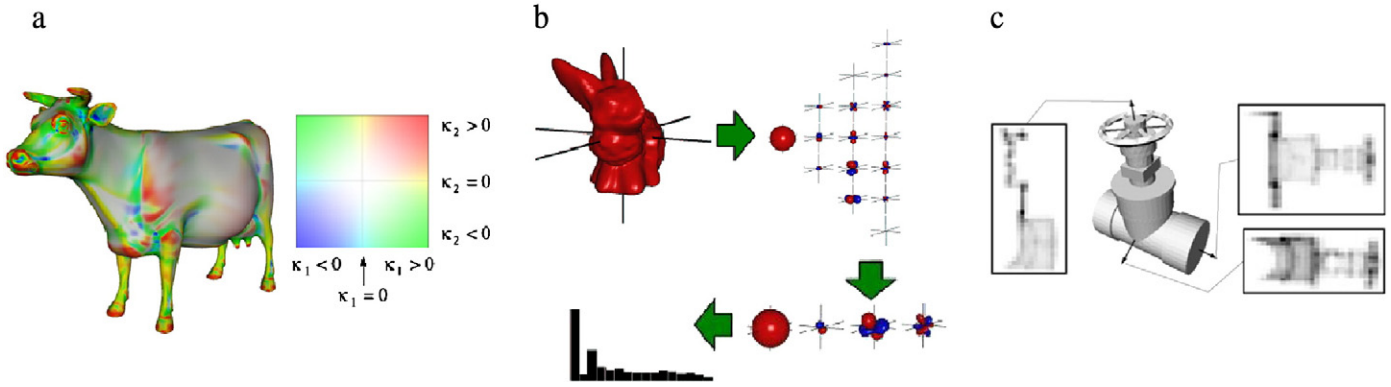

Fig. 5. Implicit shape representations can be categorized as local, global, or semi-local. (a) Local representations characterize differential properties, such as surface curvature (from [79], © 2004 IEEE). (b) Global representations, such as the spherical harmonics shown here, characterize entire shapes (from [48], © Eurographics Association 2003; reproduced with permission of the Eurographics Association). (c) Semi-local representations, such as spin images, characterize localized regions of a shape (from [47], © 1997 IEEE).

and observed from all sides. This requirement is not often achievable in laser scan data of real environments, which limits the practical usage of global representations in modeling as-built BIMs.

Semi-local representations strike a balance between global and local representations, taking the strengths of each to eliminate many of the limitations of the extremes (Fig. 5c). Semi-local representations encode object shape in the vicinity of a point that is typically located on the surface of the object. The region of support used for encoding such a shape descriptor is larger than the immediate vicinity that would be used for computing a local representation, and yet smaller than the entire object. This intermediate region of support makes semi-local descriptors less sensitive to data noise. At the same time, the limited extent of the region of support reduces the chance of the descriptor including occluded regions or clutter, reducing their sensitivity to these effects. Semi-local descriptors are designed so that similarly shaped surfaces will produce similar descriptors, making them well suited for surface matching and object recognition. One popular semi-local descriptor is the spin image, which uses a 2D histogram to encode surface shape in a cylindrical region around a basis point [46]. Other descriptors include point signatures [17], Euclidean 3D grid histograms [64], 3D shape contexts [28], and bitangent curves [98]. A survey by Mian offers a good overview of semi-local descriptors and their uses [63]. A more general overview of 3D shape representations can be found in [15].

2.2. Relationship representation

The relationships between objects are important information in a BIM because they are critical for many engineering tasks, such as navigation inside a building or diagnosis of building system faults (e.g. using the spatial relationships between valves, pipes, and rooms for identifying the valve to be turned off for stopping a water leak). In addition, previous studies have shown that spatial relationships between objects provide contextual information to assist in object recognition [16,68].

We can identify three categories of spatial relationships that are relevant to BIMs: aggregation relationships (e.g., part of), topological relationships (e.g., connectivity, inside, or outside), and directional relationships (e.g., above or below). Relationships are typically represented by a tree or graph structure. Aggregation relationships can be modeled using a tree-based hierarchical representation that encodes compositions in a local-to-global manner: nodes represent objects or geometric primitives, while arcs represent aggregation relationships between them [26,53]. Graph-based representations generalize the hierarchical approach by allowing arcs to represent not only aggregation relationships, but also topological and directional relationships. The B-rep representation (Section 2.1.1) uses a graph to encode spatial relationships between object parts [23]. Algorithms that use semantic networks for object recognition use graphs to encode the semantic network as well as the topological and directional relationships in the scene [16,68].

3. Geometric modeling

Having reviewed the different categories of knowledge representation, we now turn our attention to geometric modeling — the first of the three core operations of as-built BIM creation. Geometric modeling is the process of constructing simplified representations of the 3D shape of building components, such as walls, windows, and doors, from point cloud data. The output representation can be either parametric or non-parametric, as well as surface-based (e.g., boundary-based representation) or volumetric (e.g., CSG representation). Volumetric parametric methods are the most relevant to BIMs, since BIMs are normally represented primarily using parametric volumetric primitives. However, surface-based methods are more prevalent. Non-parametric geometric modeling reconstructs a surface, typically in the form of a triangle mesh [41], or a volume [18]. Surface-based non-parametric modeling, known as surface reconstruction, is a well studied field [1,41,57]. Such methods are useful for modeling complex geometries, such as statues or friezes. Here, we focus on parametric modeling methods, which are most relevant to construction of BIMs.

3.1. Modeling surfaces

Parametric surface modeling requires the detection and extraction of geometric primitives and their parameters. Common types of primitives occurring in facilities include planar surfaces, curved surfaces (e.g., cylinders and cones), and extrusions (e.g., decorative moldings and trim). Research on the first two categories is much more prevalent than the last category.

3.1.1. Modeling planar surfaces

Planar surfaces are often detected using bottom-up methods, using local estimates of surface shape, such as flatness [95] or surface curvature [82], to group data into locally similar patches (Fig. 6a). Patches that are sufficiently planar can be grouped together based on similarity of surface normals and co-planarity [92,95] or through more sophisticated schemes (e.g., tensor voting [62]). Other detection approaches include using the Hough transform or hypothesize and test methods like the RANSAC algorithm and its variants [91]. Once a planar region is detected, the parameters of the plane can be estimated using total least squares fitting [32] or robust methods that are less affected by points not belonging to the plane [65]. The method proposed by Thrun et al. combines detection and modeling into one algorithm that alternates between soft assignment of points to planes and estimation of the planes based on the points assigned to them [92]. Once a plane is modeled, the boundaries of the planar patch can be estimated by projecting the constituent points onto the modeled plane and estimating the outline in 2D [68,75]. Such boundaries can be irregular, though, due to the discrete sampling nature of laser scan data. More precise boundaries can be found by computing the intersection of neighboring planar patches [89].

3.1.2. Modeling curved surfaces

Curved surfaces can be detected using bottom-up or hypothesize and test methods similar to plane detection methods. Different types of curved surfaces can be classified based on local curvature features, specifically the mean and Gaussian curvature [7]. Regions with a specific shape signature can then be used as seed points for surface-fitting algorithms. Many curved surfaces can be represented by quadrics, including ellipsoids, paraboloids, cylinders, and cones. Two main classes of surface-fitting methods are algebraic fitting and geometric fitting. Algebraic fitting has a closed-form solution for quadrics, but has significant bias, while geometric fitting, which minimizes the Euclidean distance between points and the modeled surface, has no closed-form solution and is significantly slower but more accurate [25]. The survey by Petitjean offers a good introduction quadric fitting methods [71].

More complex curved surfaces can be modeled with flexible parametric representations, such as Bezier spline patches [21], NURBS patches [60], or subdivision surfaces [40]. These representations have been used for reverse engineering free-form surfaces and complex manufactured parts. In the context of BIM construction, these representations would be appropriate for smooth curved geometries that do not fit within the class of quadric surfaces, such as cove ceilings, archways, and spiral staircases.

3.1.3. Modeling extrusions

Modeling techniques for extracting linear structures, such as moldings, beams, and pipes, are relatively rare. Moldings can be modeled by fitting splines to a cross-section and sweeping along a trajectory, as proposed by De Luca, but this method is mostly manual [20]. Pipe

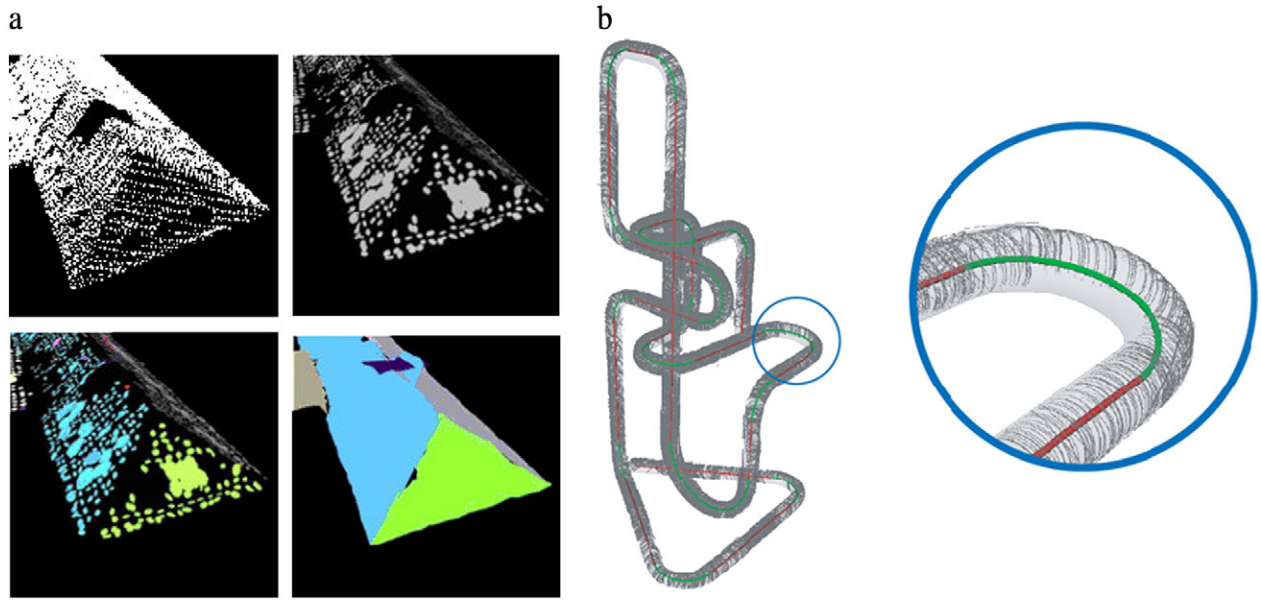

Fig. 6. (a) Planar surfaces can be extracted from laser scan data (top-left) by fitting local plane patches to the points (top-right), grouping points together based on surface normal (bottom-left), and then estimating the boundaries of each patch (bottom-right) (from [95], © 2006 IEEE). (b) Extruded structures, such as beams, moldings, and pipes, can be modeled by sweeping a constant shape along a trajectory. Bauer and Polthier estimate the spine curve of a pipe and then model the segments using splines (from [4], © 2009, with permission from Elsevier).

modeling is important in creating BIMs of process plants, where pipes occur frequently (Fig. 6b). These modeling algorithms first estimate the trajectory of the pipe center, which is known as the spine curve, and then model the curve using straight and curved segments [33] or splines [4,55].

3.2. Modeling volumes

Many objects and environments can be represented by a combination of a small set of volumetric primitives. Various researchers have proposed candidate sets of primitives, such as geons [9], superquadrics [3], and generalized cylinders [10]. Geons are a qualitative set of primitive shapes proposed by Biederman in his theory of Recognition by Components. Geons can represent cuboids and cylinders, as well as curved and tapered versions of these base shapes. Superquadrics are a quantitative form capable of representing spheres, cylinders, cuboids, octahedrons, and interpolations between these shapes. Extensions to the basic equations allow more complex shapes that bend or taper [59]. Generalized cylinders extend the concept of a cylinder by allowing the cross-section shape to change as a function of position along the axis. These models can be fit to 3D data by searching for the parameters that best align the model's surface with the data. The large number of parameters in these representations and potential for local minima in the optimization function presents challenges for this data fitting problem. Basic methods for extracting these volumetric primitives from 3D data assume that the data are already segmented — a challenging problem in itself [87,97]. One exception to this trend is the "recover and select" paradigm proposed by Leonardis, which simultaneously fits many different models to the data and dynamically selects the most promising candidates while filtering out those that match the data poorly [56].

So far, volumetric primitives have not been exploited much in creating as-built BIMs, though the methods have been used in related fields, such as building modeling from aerial data and reverse engineering of manufactured parts. You et al. use volumetric primitives (cylinders and spheres) to model complex roof shapes from aerial data [99], while Lin and Chen extract primitives (planes, cuboids, spheres, cones, and cylinders) from range data and describe objects using CSG as part of an object recognition algorithm [58].

3.3. Modeling complex structures — windows and doors

There has been some work on detecting and modeling more complex structures, such as windows and doors. These methods often include some aspects of object recognition or depend on *a priori* knowledge of object class. Faber and Fisher use constrained surface fitting and a genetic algorithm to fit parametric models of doors point clouds [22]. The method was demonstrated with rectangular and arched doorways but requires relatively high density data and assumes that the data have been pre-segmented. Since laser scanners cannot easily sense glass surfaces, the data density within windows (and some doors) is typically nearly zero. This fact allows window and door openings to be modeled by fitting geometric primitives to the boundaries of holes in wall surfaces [11,74]. Pu and Vosselman use a triangulation-based method to detect the boundaries of sparse regions within a building façade and then fit rectangles to the resulting regions (Fig. 7) [74]. Böhm et al. use density-based edge detection to find vertical and horizontal lines in the depth map of a building façade followed by a classification of the resulting rectangles as window or non-window [11]. They use visual information from stereo imagery to detect and model detailed cross-bar positions within the windows, and exploit regular repeated patterns of windows to constrain the estimated positions.

4. Object recognition

The second core task of as-built BIM construction is object recognition, the process of labeling a set of data points or geometric primitives extracted from the data with a named object or object class. Whereas the modeling task would find a set of points to be a vertical plane, the recognition task would label that plane as being a wall. Object recognition algorithms may label object instances of an exact shape (e.g. recognize instances of a specific I-beam), or they may recognize classes of objects, where the shape may vary among instances from the class (e.g. recognize all windows, which can vary in height, width, etc.).

Often, the knowledge describing the shapes to be recognized is encoded in a set of descriptors that implicitly capture object shape, either at a semi-local or global level (Section 2.1.2). Such approaches

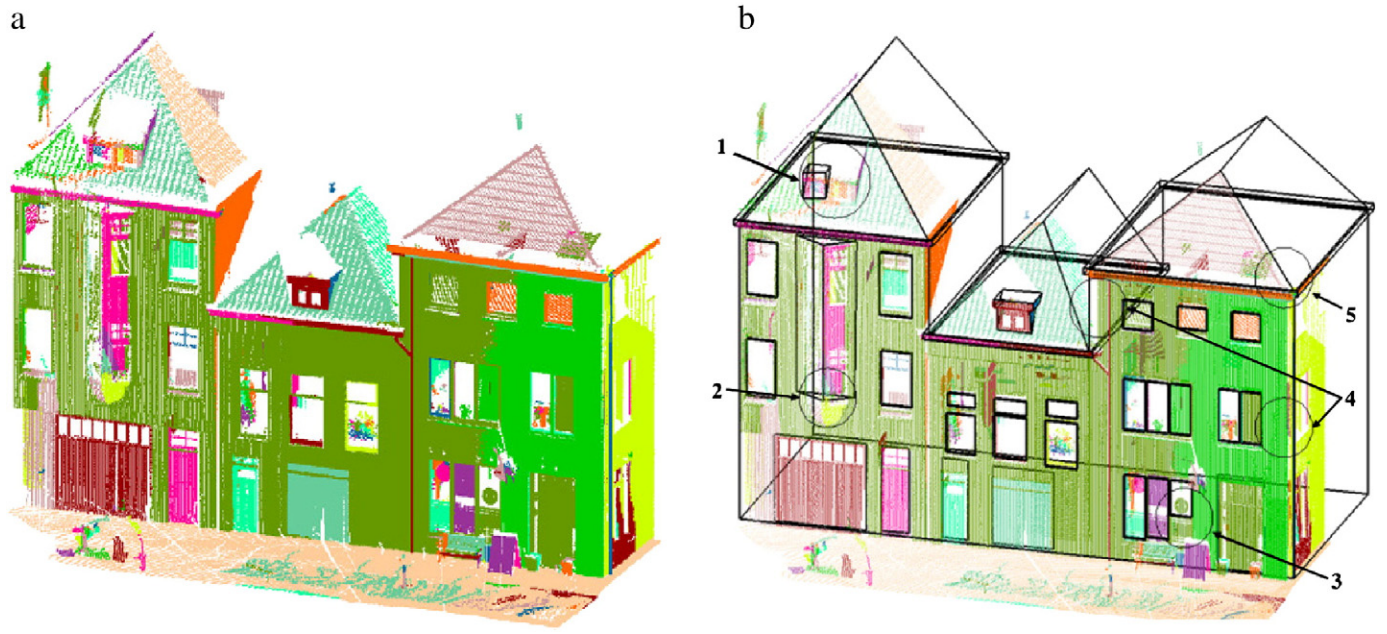

Fig. 7. Detection of windows and doors (from [75], reproduced with permission of the ISPRS). (a) Segmented point cloud showing each planar region in a different color; (b) Result of reconstructed model with detected windows and doors outlined in black.

are more typical of non-parametric models. For parametric representations, typically heuristics are used to encode knowledge about the object categories (e.g., rules like "walls are vertical"), though, in theory, this knowledge could be learned from examples.

4.1. Recognizing object instances

Among the different types of 3D object recognition within real-world data, the problem of recognizing specific instances of known objects is perhaps the most studied and has the best success rate. From early work on detecting objects on a conveyer belt — the "bin-picking" task [24] — to more recent work on recognizing vehicles and other objects in urban environments [31,61], a successful strategy to object instance recognition has evolved [15].

The recognition process has three general steps, the details of which vary. First, in an offline process, shape descriptors (often semi-local) are computed for each object to be recognized. For semi-local descriptors, samples are taken either uniformly over the surface or at salient points, such as local maxima of curvature. These descriptors are stored in a model database that is designed to facilitate rapid lookup of descriptors based on similarity to a query descriptor. Second, at runtime, the algorithm is presented with a scene in which instances of the target object are to be detected, or possibly with a pre-segmented data instance to be recognized. Shape descriptors are computed at locations in the query scene, either randomly or at salient points, and the most similar descriptors in the model database are retrieved. The descriptor similarity measure is designed so that similar shaped objects will result in similar descriptors. As a result, the closest matching descriptors from the database will give an indication of the object instance or its location in the scene. Finally, as a verification step, the object from the database is aligned with the scene to make sure that the shapes match well. This verification can help distinguish between a few closely matching instances. The alignment is accomplished by using the fact that the semi-local descriptors encode shape at a specific location on an object, and therefore matching descriptors from the database should correspond to approximately the same location on the query object. Depending on the descriptor, one or a few correspondences will be sufficient to determine the alignment in closed form. Usually, this coarse alignment is followed by a fine-tuning step to exactly align the two shapes.

The work of Matei et al. is typical of this approach to instance-based object recognition. In this case, the shape descriptor is spin images [61]. The model database is stored in a locality sensitive hash table, which makes lookup scale sub-linearly with the database size, allowing recognition from a database of several hundred objects. Verification is performed using MLESAC, a variant of RANSAC. Many other, similar approaches have been proposed, and Campbell and Flynn's survey offers a comprehensive overview [15].

In the context of modeling as-built BIMs, instance-based object recognition is mainly useful for recognizing objects with known shape, or objects that are repeated throughout a facility. Examples of such objects include machinery in process plants, pipes, valves, and I-beams (Fig. 8) [47].

4.2. Recognizing object classes

One disadvantage to the instance-based recognition approach is that it cannot handle significant variation in object shape. A number of approaches have been proposed for this more difficult task of recognizing classes of objects. The most common approach uses global shape descriptors, which are less discriminative than semi-local descriptors, but better adapted to shape variations. Alternatively, the instance-based method described above can be modified to relax the assumption of rigid instances.

The global descriptor method for object recognition is very similar to the aforementioned method for recognizing instances. The main difference is that query objects are matched against examples from the entire class, rather than just instances of a single object. The process is somewhat simpler, since there is no possibility of verifying the recognition by aligning the shapes [48,86]. One challenge is to recognize the object regardless of its pose. This capability can be achieved by designing the global descriptor to be invariant to pose or by estimating the object pose, for example, by computing its moments [48]. The Princeton Shape Benchmark was recently developed to provide a common data set and evaluation methodology for comparing 3D object classification algorithms. The corresponding paper summarizes and compares a large number of global descriptor methods [86]. One disadvantage of the global descriptor approach is that it is unable to handle partial data caused by occlusion or clutter, both of which are prevalent in laser-scanned data. In particular, the

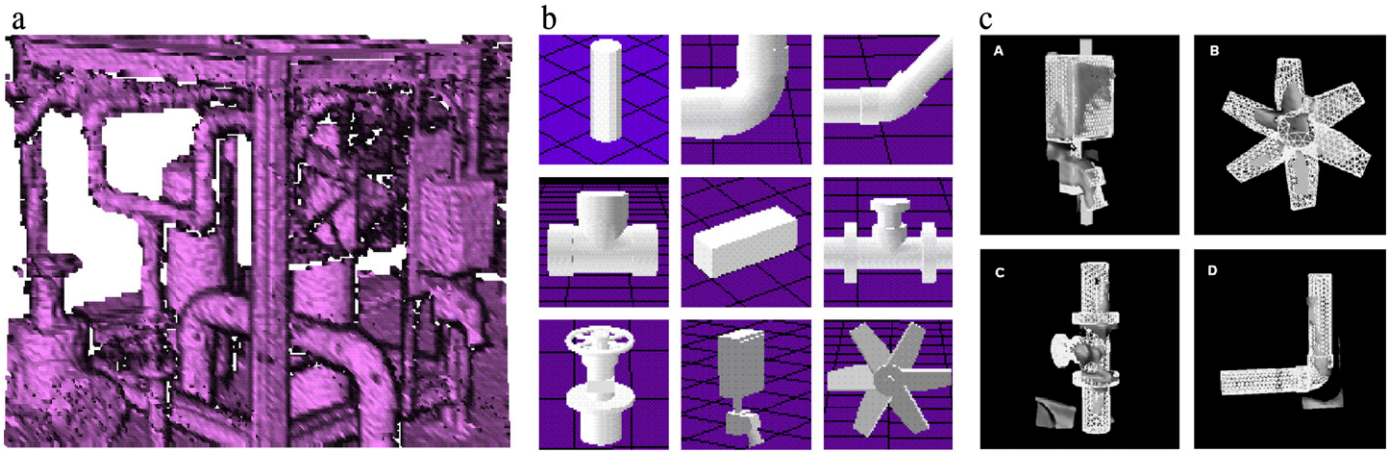

Fig. 8. Recognizing instances of objects (from [47], © 1997 IEEE). (a) Surface mesh input scene — a mockup of a process plant; (b) The database of objects to be recognized; (c) Examples of objects detected in the scene and extracted, with the model (wireframe) overlaid on the original data (shaded surface mesh).

back sides and bottoms of most objects are not visualized. Existing methods would need to be adapted to handle such situations before they could be practically applied in as-built BIM creation.

Instance-based methods have also been extended to allow recognition of non-rigid objects. One approach is to divide objects into parts that can be recognized individually. For instance Huber et al. used a parts-based method to classify types of vehicles [44], and Matei et al. used a parts-based method to allow recognition of shapes with articulations [61]. A second approach is to use descriptors that are themselves more robust to shape changes [78]. While these methods are more robust to occlusion and clutter, it is still unclear whether they could be applied to recognition of components relevant to BIM creation.

Research on recognition of BIM-specific components, such as walls, windows, and doors, is still in its early stages. Methods in this category typically perform an initial shape-based segmentation of the scene, into planar regions, for example, and then use features derived from the segments to recognize objects. This approach is exemplified by Rusu et al. who use heuristics to detect walls, floors, ceilings, and cabinets in a kitchen environment [81]. A similar approach was proposed by Pu and Vosselman to model building façades (see Fig. 7) [75].

4.3. Recognition using context

One of the challenges of recognition in the BIM context is that many of the objects to be recognized are very similar to objects of little relevance. For example, how does one differentiate between a wall and the side of a bookshelf? Both are vertical planar surfaces with little distinguishing texture. Motivated by this observation, some researchers have proposed leveraging the spatial relationships between objects or geometric primitives to reduce the ambiguity of recognition results. Such approaches generate semantic labels of geometric primitives, and test the validities of these labels with a spatial relationship knowledge base. Usually, such a knowledge model is represented by a semantic net [16,68].

For example, a semantic net may specify the relationships between entities such as “floors are orthogonal to walls and doors, and parallel with ceilings” (Fig. 9). During the recognition process, if a surface is recognized as “floor,” then the algorithm will identify that the valid semantic labels of a surface orthogonal to it can only be “wall” or “door,” but not “ceiling,” thereby reducing the search space [16]. Such validity checking approaches provide ways to integrate domain knowledge into the object recognition process.

Another approach for recognition is to first detect objects that are easily recognizable, and then use the context of these initial detections to facilitate recognition of more challenging structures. For example, Pu and Vosselman use characteristic features, such as size, orientation, and relationships to other prominent objects, to detect walls and roofs [75]. Then, a second stage detects windows within each of the detected walls.

4.4. Using prior knowledge

One strategy for reducing the search space of object recognition algorithms is to utilize knowledge about a specific facility, such as a CAD model or floor plan of the original design. For instance, Yue et al. overlay a design model of a facility with the as-built point cloud to guide the process of identifying which data points belong to specific objects and to detect differences between the as-built and as-designed conditions [100]. In such cases, object recognition problem is simplified to be a matching problem between the scene model entities and the data points. Another similar approach is presented in [13].

In the remote sensing and GIS domains, studies have also explored the effectiveness of utilizing 2D drawings or GIS databases for improving the speed and accuracy of object recognition algorithms [50,52,84,90].

5. Relationship modeling

Spatial relationships between objects in a BIM are useful in many scenarios, such as spatial reasoning for the diagnosis of the water distribution system of a building. Hence, it is promising to develop automated approaches for reconstructing spatial relationships between objects. Common relationships modeled in a building information model include: aggregation relationships (e.g., a window is *contained in* a wall), topological relationships (e.g., wall 1 is *connected to* wall 2), and directional relationships (e.g., the second floor is *above* the first floor).

Several spatial relationship models have been developed for automatically deriving topological relationships between objects. An example of such research studies is Nguyen et al., who utilize a 3D solid CAD model to automatically derive topological relationships between solid objects or geometric primitives [67]. The derived relationships include adjacency, containment, separation, intersection, face–face intersection, face–edge intersection, common edge intersection, and connectivity. Other examples include [37,38,66].

One approach for automatic directional relationship identification is presented in [12]. This approach introduces two models for identifying directional relationships between two extended solid objects (e.g. “NorthOf”, “Above”).

6. Performance evaluation

As new methods are developed and advances in the automated or semi-automated creation of as-built BIMs are achieved, it is important to

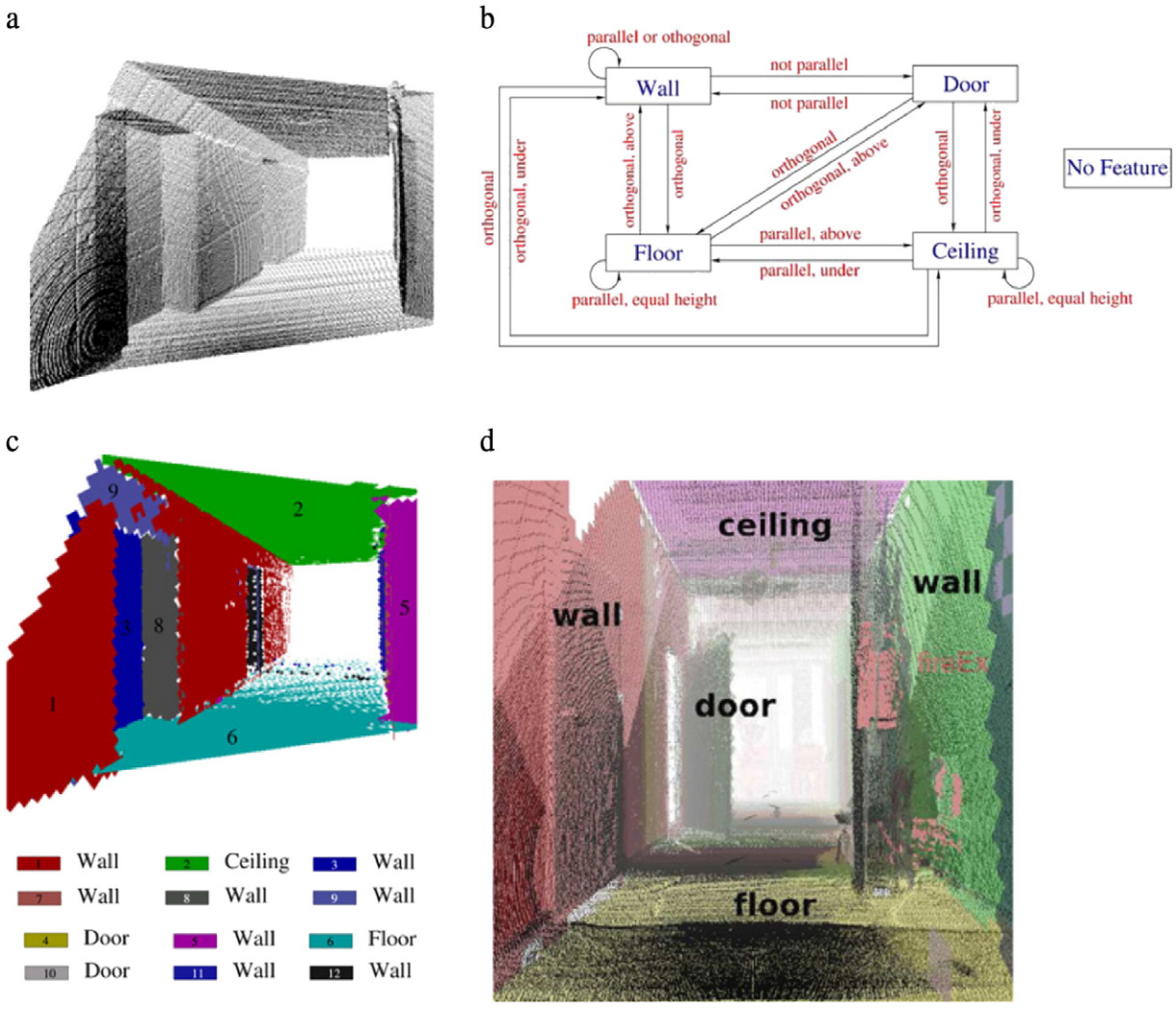

Fig. 9. An example of recognition using context (reprinted from [68], © 2008, with permission from Elsevier). (a) A point cloud data set of a hallway. (b) A semantic network showing relationships between different classes of surfaces. (c) The resulting segmented and labeled structures for the data set in (a). (d) Another example, showing the perspective of the sensor with embedded labels.

establish methodologies for performance evaluation to track the progress of the field and determine which methods perform best. In other fields, such as computer vision, standard test sets and performance metrics have been established [72,83], but no standard evaluation metrics have been established for as-built BIM creation as yet. Furthermore, existing research studies that do involve performance evaluation of related techniques lie mainly within the computer vision and remote sensing domains [8,63], and the performance measures used in those studies do not directly address the requirements of the AEC domain. As a result, it is difficult for AEC domain experts to use these measures for understanding the implications of selecting a specific algorithm in an application scenario (e.g., determine whether a given algorithm can achieve a certain level of detail for detecting construction defects).

Inspired by performance measures previously proposed for geometric modeling and object recognition algorithms [8,63], we now identify a set of performance measures for modeling as-built BIMs and relate them to the information requirements of the AEC domain (GSA 2007). These evaluation measures can be clustered into three categories: 1) measures related to the algorithm design; 2) measures related to environmental and sensing conditions; and 3) measures of modeling performance.

The design of any algorithm requires up-front decisions about numerous aspects of the algorithm such as what capabilities it should be able to achieve, what types of inputs and outputs it requires, and so forth. We have identified six dimensions of performance related to algorithm design:

- Degree of automation. How much human interaction is required during the modeling process? Is the algorithm fully automated, partially automated, or a completely manual process?
- Input and output assumptions and data types. What type of input does the algorithm require (e.g., point cloud only, 2D, or 3D design model)? What type of output does the algorithm produce (e.g., level of detail, types of objects, and level of semantics)? These variations of the problem space were discussed in Section 1.3.
- Computational complexity. What are the time and memory requirements of the algorithm? How does the method scale as the size of the data or number of object types increases?
- Extensibility to new environments. Can the algorithm be extended to handle new types of objects? Can the algorithm be applied to different types of environments, or is it specific to one or a certain class of spaces?
- Learning capabilities. Does the algorithm use learning to improve its performance? Does the algorithm depend on hand-coded rules to express knowledge, or does it learn from training data? If the algorithm learns, how many training examples are needed?
- Confidence-level and uncertainty modeling. Does the algorithm report its confidence in the modeling and recognition results? Does the algorithm estimate uncertainty in the geometric modeling accuracy?

Once an algorithm is designed and implemented, it must be tested and evaluated on real data, ideally using standard reference data sets. Test environments can vary considerably in form and function, ranging from home or office environments, which have many planar structures and repeated objects throughout the facility, to process plants, which often feature complex arrangements of pipes and special-purpose equipment. Within such varied environments, some commonalities do exist, however. We identified eight factors relating to environmental conditions:

- Types of objects present. Common object types include walls, floors, ceilings, windows, doors, columns, beams, stairways, railings, fixtures, pipes and conduits, railings, equipment and machinery, and clutter objects that are not part of the facility (e.g., furniture, accessories, etc.).
- Level of sensor noise. Different sensors have different noise characteristics. This measure can be expressed in terms of noise statistics for a given data set.
- Level of occlusion. What amount of the scene or an object in the scene is occluded? This can be expressed quantitatively as a percentage of the total surface area that is occluded along with statistics about the size of the occlusions.
- Level of clutter. What amount of the scene consists of clutter? This can be expressed quantitatively as a percentage of the total scene that is classified as clutter.
- Presence of moving objects. Does the scene contain moving objects, such as people or vehicles, that interfere with the integrity of the data?
- Presence of specular surfaces. Does the scene contain shiny surfaces, such as polished metal, which laser scanners typically have difficulty measuring accurately?
- Presence of dark (low-reflectance) surfaces. Does the scene contain dark-colored surfaces that poorly reflect laser light, leading to missing or very noisy data?
- Sparseness of data. How dense are the 3D measurements on the surface of the scene? This measure can be expressed using statistics of data density estimates.

An as-built BIM construction algorithm can be evaluated based on its ability to handle these various environmental conditions. For example, an algorithm could be tested on data sets with different levels of clutter to determine how the performance degrades as a function of clutter level.

Finally, given an algorithm that addresses a variant of the BIM creation problem, it is necessary to objectively evaluate the algorithm's performance for a given set of environmental conditions. We identify four aspects of algorithm performance:

- Geometric modeling accuracy. How accurately does the algorithm reproduce the geometry of the facility? This can be objectively expressed in terms of the error in position, orientation, and parameters of each component that is modeled. Additionally, some components may be missed entirely by the modeling algorithm, which can be quantified by the fraction of the data that is modeled correctly within some tolerance accuracy.
- Recognition/labeling accuracy. How accurately does the algorithm label the relevant components within the facility? An algorithm may have high geometric modeling accuracy, yet still perform poorly at identifying the type of some components. For example, a closed door could be mislabeled as a wall. Recognition accuracy can be quantified using precision–recall (PR) curves or receiver operator characteristic (ROC) curves, which are commonly used for measuring object recognition algorithm performance.
- Relationship modeling accuracy. To what extent does the algorithm correctly determine relationships between the components in the model? For example, does the algorithm correctly determine which walls are connected to one another? These concepts can be quantified in terms of the fraction of relationships that are correctly identified, similar to the methods for evaluating recognition accuracy.
- Level of detail. What is the smallest size component or feature that the algorithm can reliably model or recognize? Many AEC projects have explicit requirements on the level of detail that must be encoded in an as-built BIM. This measure spans geometric modeling and recognition, and one possible way of quantifying level of detail performance is to compute the aforementioned accuracy measures for different levels of detail. In this way, it would be possible to determine the level of detail where performance begins to break down.

There is still much work to do on designing and standardizing evaluation metrics. When a discrepancy in the reconstructed as-built model is detected, it can be challenging and even ambiguous to decide what type of error has occurred. For example, how does one determine the mapping between components in an algorithm's output model and the ground truth BIM? If an algorithm performs well, the mapping could be accomplished by considering overlap between the components in these two models. If an algorithm performs poorly, the modeled and ground truth components may not match up very well, and it could be difficult to determine which component in the ground truth BIM is being represented by a given component in the output model.

7. Discussion of technology gaps

Today, the as-built BIM creation process is largely a manual procedure, which is time-consuming and subjective. While there is a clear need for automated, or even semi-automated methods for aiding the creation of as-built BIMs, research on the subject is still in the very early stages. Our survey shows that many of the existing methods for geometric modeling, object recognition, and relationship modeling can be important building blocks that as-built BIM automation can leverage. Nevertheless, a number of technology gaps must be addressed as research on this subject moves forward.

The majority of existing modeling work focuses on modeling only the simplest aspects of a building — planar surfaces. Real facilities have many complex details in their geometry, which these methods do not yet capture. More recent work has begun to focus on modeling details, such as window and door openings on building façades [11,74]. Looking ahead, more effort is needed in modeling more complex structures like columns, structural steel, archways, and cove ceilings, and also on improving the performance of window and door detection and modeling. In the longer term, methods are needed for modeling fine details, like decorative moldings, light fixtures, and window details. Furthermore, existing parametric modeling algorithms primarily operate in application domains where there is little or no clutter or occlusion, such as the reverse engineering of manufactured parts. New methods are needed to address realistic environments, which are often cluttered and have significant occluded surfaces.

There is a significant disconnect between the surface-based modeling representations that dominate the prior art (as summarized in this article) and the volumetric representation used in practice by existing BIM tools. While there are methods for converting boundary representations into volumetric CSG representations [85], the process is not straightforward, and the conversion is not guaranteed to be unique. Further research is needed either in directly modeling components using volumetric primitives or in converting surface-based representations into volumetric BIM representations. Potentially, domain knowledge could be exploited to limit the ambiguities in this transformation.

The methods for modeling and recognition described in this article rely on hand-coded knowledge about the domain. Concepts like “walls are vertical” and “walls intersect with the ceiling” are encoded within the algorithms either explicitly, through sets of rules, or implicitly, through the design of the algorithm. Such hard-coded, rule-based approaches tend to be brittle and break down when tested in new and slightly different environments. One reason for this is that the rules are not universal — some walls are not vertical, and some walls do not intersect the ceiling. Also, it can be difficult to extend an

algorithm with new rules or to modify the rules to work in new environments. Methods that learn from experience can address these limitations. Learning-based methods learn the rules (or parameters for tuning the rules) from examples, which are hand-labeled. The rules themselves can be modeled more flexibly. For example, instead of encoding "walls are vertical," one could encode a probability distribution over the space of surface orientations, which might be centered on vertical, but would also afford the possibility of non-vertical walls. Based on these observations, we predict that more flexible representations and learning-based algorithms will open the way to significant gains in modeling capability and generality.

Research on modeling facilities is still in its early stages, and as such, methods and testbeds for evaluating algorithm performance have not been formalized. Most of the prior work primarily concentrates on qualitative assessment of performance, and typically the algorithms are only tested on one or a few examples. Moving forward, we need to work to develop reference testbeds that span the use cases for as-built BIMs. These testbeds need to include the full range of environmental factors that can affect BIM construction performance. Ideally, each environmental factor should be isolated, for example, by creating a series of test sets that have identical ground truth models but varying degrees of clutter. The corresponding algorithm performance measures also need to be standardized. Together, standardized test sets and standardized performance measures would enable different algorithms to be compared objectively. The Princeton shape modeling benchmark is a good example of where this approach has been used successfully in a related field [86].

Current BIM representation and data exchange formats assume idealized geometries that are rarely seen in real facilities. Walls are not exactly planar, and corners are rarely precisely 90 °C, for example. Furthermore, the imperfect nature of sensing data for as-built BIMs leads to other representation needs that are outside of the current capabilities of BIM formats, including representation of occluded regions and information about model uncertainty. New representations or extensions to existing representations are necessary in order to address these capability gaps.

Finally, work is needed to link the performance measures for as-built BIM creation to the needs of specific problems within the AEC domain. Currently, requirements for as-built BIM creation might specify a minimum sized feature that must be modeled or an accuracy requirement for feature position. However, the requirements are usually driven by a specific need, such as planning for a renovation or preserving the design of a historical building. Requirements that are too strict will unnecessarily increase the cost of as-built BIM creation, while those that are too lax will render the model useless for its intended purpose. Therefore, it is necessary to determine the relationship between the domain-independent requirements for as-built BIM creation with the domain-specific needs for a given problem.

8. Conclusions

The state-of-the-art in the creation of as-built BIMs is fundamentally a manual process. The time-consuming and subjective nature of this process motivates the need for automated, or at least semi-automated, tools for as-built BIM creation. Our survey of existing representations and algorithms suitable for automated as-built BIM creation shows that, while many promising techniques have been developed, there is relatively little research specifically addressing the problem in an AEC/FM context.

The systems that have been demonstrated often focus on special-purpose situations, such as modeling building façades [75] or kitchens [81], and it is not clear how well such methods would extend to the more general problem of modeling an entire facility. Other promising approaches have only been tested on limited and very simple examples, and it is equally difficult to predict how they would fare when faced with more complex and realistic data sets. For example, the semantic network methods for recognizing components using context work well for simple examples of hallways and barren, rectangular rooms [16,68], but how would they handle spaces with complex geometries and clutter, such as an office or a bathroom?

In Section 7, we identified and discussed a number of technology gaps in automated as-built BIM creation capabilities. These gaps reveal areas where research on this problem should be concentrated. The main areas that need to be addressed include: 1) modeling of more complex structures than simple planes; 2) handling realistic environments with clutter and occlusion; 3) representing models using volumetric primitives rather than surface representations; 4) developing methods that are easily extensible to new environments; 5) creating reference testbeds that span the use cases for as-built BIMS; 6) representing non-ideal geometries that occur in real facilities; and 7) developing quantitative performance measures for tracking the progress of the field. The methods described in this article show promise for the future of automated as-built BIM creation, but as this list of technology gaps indicates, there is a long way to go before automated as-built BIM construction is considered a solved problem. As these technology gaps are closed, algorithms for automated as-built BIM creation will continue to migrate into commercial tools, and the benefits of this automation can be enjoyed by practitioners throughout the AEC/FM industry.

Acknowledgments

This research was supported, in part, by the National Institute of Standards and Technology (NIST) under grant 60NANB7D6156, by the General Services Administration (GSA) under grant GS00P07CYP0233, and by the National Science Foundation under Grant No. 0856558. Their support is gratefully acknowledged. Any opinions, findings and conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NIST or the National Science Foundation. Mention of trade names in this article does not imply endorsement by Carnegie Mellon University or NIST.

References

[1] N. Amenta, S. Choi, R. Kolluri, The power crust, unions of balls, and the medial axis transform, Computational Geometry: Theory and Applications 19 (2–3) (2001) 127–153.
[2] Revit Architecture — Autodesk, Inc., — http://usa.autodesk.com/adsk/servlet/index?id=3781831&siteID=123112.
[3] A.H. Barr, Superquadrics and angle-preserving transformations, Computer Graphics and Applications 1 (1) (1981) 11–23.
[4] U. Bauer, K. Polthier, Generating parametric models of tubes from laser scans, Computer-Aided Design 41 (10) (October 2009) 719–729.
[5] B.G. Baumgart, Winged Edge Polyhedron Representation, Stanford University, Stanford, CA CS-TR-72-320, 1972.
[6] F. Bernardini, H. Rushmeier, The 3D model acquisition pipeline, Computer Graphics Forum 21 (2) (June 2002) 149–172.
[7] P.J. Besl, Surfaces in Early Range Image Understanding, University of Michigan, PhD, 1986.
[8] P.J. Besl, R.C. Jain, Invariant surface characteristics for 3D object recognition in range images, Computer Vision, Graphics, and Image Processing (CVGIP) 33 (1) (1986) 33–80.
[9] I. Biederman, Recognition-by-components: a theory of human image understanding, Psychol. Rev. 94 (2) (April 1987) 115–147.
[10] T.O. Binford, Visual perception by computer, Proceedings of the IEEE Conference on Systems and Control, Miami, FL, 1971.
[11] J. Böhm, S. Becker, N. Haala, Model refinement by integrated processing of laser scanning and photogrammetry, Proceedings of 3D Virtual Reconstruction and Visualization of Complex Architectures (3D-Arch), Zurich, Switzerland, 2007.
[12] A. Borrmann, E. Rank, Specification and implementation of directional operators in a 3D spatial query language for building information models, Adv. Eng. Inform. 23 (1) (2009) 32–44.
[13] F. Bosche, C.T. Haas, Automated retrieval of 3D CAD model objects in construction range images, Automation in Construction 17 (4) (May 2008) 499–512.
[14] Industry Foundation Classes (IFC) — BuildingSmart, International Alliance for Interoperability — http://www.iai-tech.org/.
[15] R. Campbell, P. Flynn, A survey of free-form object representation and recognition techniques, Computer Vision and Image Understanding (CVIU) 81 (2) (2001) 166–210.
[16] H. Cantzler, Improving Architectural 3D Reconstruction by Constrained Modelling, College of Science and Engineering. School of Informatics, University of Edinburgh, PhD Edinburgh, 2003.

[17] C.S. Chua, R. Jarvis, Point signatures: a new representation for 3D object recognition, International Journal of Computer Vision 25 (1) (October 1997) 63–85.
[18] B. Curless, M. Levoy, A volumetric method for building complex models from range images, Proceedings of ACM SIGGRAPH, 1996, pp. 303–312.
[19] Leica Cyclone — Loyola Spatial Systems — http://leica.loyola.com/products/hds/cyclone.html.
[20] L. DeLuca, P. Veron, M. Florenzano, Reverse engineering of architectural buildings based on a hybrid modeling approach, Computers & Graphics 30 (2) (April 2006) 160–176.
[21] M. Eck, H. Hoppe, Automatic reconstruction of B-spline surfaces of arbitrary topological type, Proceedings of Computer graphics and interactive techniques, 1996, pp. 325–334.
[22] P. Faber, B. Fisher, How can we exploit typical architectural structures to improve model recovery? 3D Data Processing Visualization and Transmission (3DPVT, 2002, pp. 824–833.
[23] T. Fan, G. Medioni, R. Nevatia, Recognizing 3D objects using surface descriptions, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 11 (11) (November 1989) 1140–1157.
[24] M.A. Fischler, R.C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM vol. 24 (1981) 381–395.
[25] R. Fisher, Solving architectural modelling problems using knowledge, International Conference on 3-D Digital Imaging and Modeling (3DIM), 2003, pp. 343–351.
[26] A.W. Fitzgibbon, D.W. Eggert, R.B. Fisher, High-level model acquisition from range images, Computer-Aided Design 29 (4) (1997) 321–330.
[27] S. Fleishman, D. Cohen-Or, C. Silva, Robust moving least-squares fitting with sharp features, Proceedings of ACM SIGGRAPH, 2005.
[28] A. Frome, D.F. Huber, R. Kolluri, et al., Recognizing objects in range data using regional point descriptors, Proceedings of the European Conference on Computer Vision (ECCV), May, 2004.
[29] Y. Furukawa, B. Curless, S.M. Seitz, et al., Reconstructing building interiors from images, Proceedings of the International Conference on Computer Vision (ICCV), October, 2009, pp. 80–87.
[30] H.E. Goldberg, State of the AEC industry, Cadalyst 22 (5) (2005) 44–45.
[31] A. Golovinskiy, V.G. Kim, T. Funkhouser, Shape-based recognition of 3D point clouds in urban environments, Proceedings of the International Conference on Computer Vision (ICCV), September, 2009, pp. 2154–2161.
[32] G.H. Golub, C.F. van Loan, An analysis of the total least squares problem, SIAM Journal on Numerical Analysis 17 (6) (1980) 883–893.
[33] F. Goulette, Automatic CAD modeling of industrial pipes from range images. 3D Digital Imaging and Modeling (3DIM), 3D Digital Imaging and Modeling (3DIM), 1997, pp. 229–233.
[34] GSA, "GSA BIM Guide For 3D Imaging, version 1.0." vol. 3 http://www.gsa.gov/bim: U.S. General Services Administration (GSA), 2009.
[35] GSA, "GSA BIM Guide For Energy Performance, version 1.0." vol. 5 http://www.gsa.gov/bim: U.S. General Services Administration (GSA), 2009.
[36] H. Hajian, B. Becerik, A research outlook for real-time project information management by integrating advanced field data acquisition systems and building information modeling, 2009 ASCE International Workshop on Computing in Civil Engineering, Austin. Texas, June 24–27, 2009.
[37] J. Haymaker, J. Kunz, B. Suter, et al., Perspectors: Composable, Reusable Reasoning Modules to Automatically Construct a Geometric Engineering View from Other Geometric Engineering Views, Stanford University, Center for Integrated Facility Engineering, 2003.
[38] D. Hernandez, Qualitative Representation of Spatial Knowledge, Springer-Verlag, New York, Inc, 1994.
[39] E. Hoffman, Understanding and Specifying Laser Scanning Services, 2005.
[40] H. Hoppe, T. DeRose, T. Duchamp, et al., Piecewise smooth surface reconstruction, Proceedings of Computer graphics and interactive techniques, 1994, pp. 295–302.
[41] H. Hoppe, T. DeRose, T. Duchamp, et al., Surface reconstruction from unorganized points, Proceedings of ACM SIGGRAPH, July, 1992, pp. 71–78.
[42] B.K.P. Horn, K. Ikeuchi, The mechanical manipulation of randomly oriented parts, Scientific American. vol. 251 (1984) 100–109.
[43] D. Huber, M. Hebert, Fully automatic registration of multiple 3D data sets, Image and Vision Computing (IVC) 21 (7) (July 2003) 637–650.
[44] D. Huber, A. Kapuria, R. Donamukkala, et al., Parts-based 3D object classification, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June, 2004, pp. 82–89.
[45] J. Jia, Z. Qin, J. Lu, Stratified helix information of medial-axis-points matching for 3D model retrieval, Proceedings of the international workshop on multimedia information retrieval, 2007, pp. 169–176.
[46] A. Johnson, M. Hebert, Using spin images for efficient object recognition in cluttered 3D scenes, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 21 (5) (May 1999) 433–449.
[47] A. Johnson, R. Hoffman, J. Osborn, et al., A system for semi-automatic modeling of complex environments, Proceedings of the International Conference on 3D Digital Imaging and Modeling (3DIM), 1997, pp. 213–220.
[48] M. Kazhdan, T. Funkhouser, S. Rusinkiewicz, Rotation Invariant Spherical Harmonic Representation of 3D Shape Descriptors, Symposium on Geometry Processing, Aachen, Germany, June, 2003.
[49] A. Kemper, M. Wallrath, An analysis of geometric modeling in database systems, ACM Comput. Surv. 19 (1) (1987) 47–91.
[50] K. Khoshelham, Building Extraction from Multiple Data Sources: A Data Fusion Framework for Reconstruction of Generic Models, ISPRS Congress, Istanbul, Turkey, 12 to 23 July 2004 (2004).
[51] K. Klasing, D. Althoff, D. Wollherr, et al., Comparison of Surface Normal Estimation Methods for Range Sensing Applications, Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA), 2009.
[52] A. Kosaka, A. Kak, Fast Vision-guided Mobile Robot Navigation Using Model-based Reasoning And Prediction Of Uncertainties, Proceedings of Intelligent Robots and Systems (IROS), 1992, pp. 2177–2186.
[53] D.T. Kuan, R.J. Drazovich, Model-based Interpretation of Range Imagery, Proceedings of the AAAI, 1983.
[54] J.-F. Lalonde, R. Unnikrishnan, N. Vandapel, et al., Scale Selection for Classification of Point-sampled 3-D Surfaces, Fifth International Conference on 3-D Digital Imaging and Modeling (3DIM), June, 2005, pp. 285–292.
[55] I.-K. Lee, Curve reconstruction from unorganized points, Computer Aided Geometric Design 17 (2) (2000) 161–177.
[56] A. Leonardis, A. Jaklic, F. Solina, Superquadrics for segmenting and modeling range data, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 19 (11) (1997) 1289–1295.
[57] X. Li, C.-Y. Han, W.G. Wee, On surface reconstruction: a priority driven approach, Computer-Aided Design 41 (9) (2009) 626–640.
[58] W.-C. Lin, T.-W. Chen, CSG-based object recognition using range images, Proceedings of the International Conference on Pattern Recognition (ICPR), 1988, pp. 99–103.
[59] Lin Zhou, C. Kambhamettu, Extending superquadrics with exponent functions: modeling and reconstruction, Proceedings of Computer Vision and Pattern Recognition, 1999, pp. 73–78.
[60] W. Ma, J.P. Kruth, NURBS curve and surface fitting and interpolation for reverse engineering, The International Journal of Advanced Manufacturing Technology 14 (12) (1998) 918–927.
[61] B. Matei, Y. Shan, H. Sawhney, et al., Rapid object indexing using locality sensitive hashing and joint 3D-signature space estimation, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 28 (7) (July 2006) 1111–1126.
[62] G. Medioni, M.-S. Lee, C.-K. Tang, A Computational Framework for Segmentation and Grouping, Elsevier Science, 2000.
[63] A.S. Mian, M. Bennamoun, R.A. Owens, Automatic correspondence for 3D modeling: an extensive review, International Journal of Shape Modeling (IJSM) 11 (2) (2005) 253–291.
[64] A.S. Mian, M. Bennamoun, R.A. Owens, A novel representation and feature matching algorithm for automatic pairwise registration of range images, International Journal of Computer Vision 66 (1) (Jan 2006) 19–40.
[65] J.V. Miller, C.V. Stewart, MUSE: Robust Surface Fitting Using Unbiased Scale Estimates, Proceedings of Computer Vision and Pattern Recognition (CVPR), 1996.
[66] M.P. Nepal, S. Staub-French, J. Zhang, et al., Deriving construction features from an IFC model, Proceedings of the CSCE Annual Conference, Québec, QC, June 10–13, 2008 (2008).
[67] T.-H. Nguyen, A.A. Oloufa, K. Nassar, Algorithms for automated deduction of topological information, Automation in Construction 14 (1) (2005) 59–70.
[68] A. Nüchter, J. Hertzberg, Towards semantic maps for mobile robots, Journal of Robotics and Autonomous Systems (RAS) 56 (11) (November 2008) 915–926.
[69] R. Osada, T. Funkhouser, B. Chazelle, et al., Shape distributions, ACM Transactions on Graphics 21 (4) (October 2002) 807–832.
[70] I. Panushev, J. Brandt, 3D Imaging Pilot Projects: Three Case Studies, Harvard Design School, Boston, MA, Research Report, 2007 2007.
[71] S. Petitjean, A survey of methods for recovering quadrics in triangle meshes, ACM Comput. Surv. 34 (2) (2002) 211–262.
[72] P.J. Phillips, H. Moon, P.J. Rauss, et al., The FERET evaluation methodology for face recognition algorithms, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 22 (1) (October 2000) 1090–1104.
[73] Polyworks — Innovmetric Software, Inc. — www.innovmetric.com.
[74] S. Pu, G. Vosselman, Extracting windows from terrestrial laser scanning, ISPRS Workshop on Laser Scanning, 2007, pp. 320–325.
[75] S. Pu, G. Vosselman, Knowledge based reconstruction of building models from terrestrial laser scanning data, ISPRS Journal of Photogrammetry and Remote Sensing 64 (6) (November 2009) 575–584.
[76] QuantaCAD — Quantapoint, Inc. — http://www.quantapoint.com/capabilities/integratelaserscanning/quantacadlaserscanning/.
[77] F. Rottensteiner, Semi-automatic building reconstruction integrated in strict bundle block adjustment, IAPRS 2000, Amsterdam, 2000.
[78] S. Ruiz-Correa, L.G. Shapiro, M. Meila, et al., Symbolic signatures for deformable shapes, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 28 (1) (Jan 2006) 75–90.
[79] S. Rusinkiewicz, Estimating curvatures and their derivatives on triangle meshes, Proceedings of the Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT), 2004.
[80] S. Rusinkiewicz, M. Levoy, QSplat: a multiresolution point rendering system for large meshes, Proceedings of ACM SIGGRAPH, 2000, pp. 343–352.
[81] R.B. Rusu, Z.C. Martona, N. Blodowa, et al., Towards 3D point cloud based object maps for household environments, Robotics and Autonomous Systems, vol. 56, no. 11, November 2008, pp. 927–941.
[82] R. Sacchi, J.F. Poliakoff, P.D. Thomas, et al., Curvature estimation for segmentation of triangulated surfaces, Proceedings of 3-D Digital Imaging and Modeling (3DIM), 1999, pp. 536–543.
[83] D. Scharstein, R. Szeliski, A taxonomy and evaluation of dense two-frame stereo correspondence algorithms, International Journal of Computer Vision, vol. 47, no. 1—3, April–June 2002, pp. 7–42.
[84] E. Schwalbe, H.-G. Maas, F. Seidel, 3D building model generation from airborne laser scanner data using 2D GIS data and orthogonal point cloud projections, ISPRS Laser Scanning 2005, Enschede, the Netherlands, 2005.

[85] V. Shapiro, D. Vossler, Construction and optimization of CSG representations, Computer Aided Design, vol. 23, no. 1, Jan/Feb 1991, pp. 4–20.
[86] P. Shilane, P. Min, M. Kazhdan, et al., "The Princeton Shape Benchmark," in *Proceedings of Shape Modeling International*, Genova, Italy, June, 2004.
[87] J. Sinnott, T. Howard, A hybrid approach to the recovery of deformable superquadric models from 3D data, Proceedings of Computer Graphics International, 2001, pp. 131–138.
[88] R. Staiger, "Terrestrial Laser Scanning Technology, Systems and Applications", in 2nd FIG Regional Conference, Sheraton Marrakech Marrakech, Morocco, December 2–5 (2003) 2003.
[89] I. Stamos, Y. Gene, G. Wolberg, et al., 3D modeling using planar segments and mesh elements, Proceedings of 3D Data Processing, Visualization, and Transmission (3DPVT), 2006, pp. 599–606.
[90] I. Suveg, G. Vosselman, Reconstruction of 3D building models from aerial images and maps, ISPRS Journal of Photogrammetry and Remote Sensing 58 (3–4) (January 2004) 202–224.
[91] F. Tarsha-Kurdi, T. Landes, P. Grussenmeyer, Hough-transform and extended RANSAC algorithms for automatic detection of 3d building roof planes from lidar data, Proceedings of the ISPRS Workshop on Laser Scanning, 2007, pp. 407–412.
[92] S. Thrun, C. Martin, Y. Liu, et al., A real-time expectation–maximization algorithm for acquiring multi-planar maps of indoor environments with mobile robots, IEEE Transactions on Robotics 20 (3) (2004) 433–443.
[93] E. Trucco, R. Fisher, Experiments in curvature-based segmentation of range data, IEEE Trans. Pat. Anal. and Mach. Intel. 17 (2) (Feb 1995) 177–182.
[94] G. Turk, M. Levoy, Zippered polygon meshes from range images, Proceedings of ACM SIGGRAPH, July, 1994, pp. 311–318.
[95] V. Verma, R. Kumar, S. Hsu, 3D building detection and modeling from aerial LIDAR data, Computer Vision and Pattern Recognition, 2006, pp. 2213–2220.
[96] E.G.L. Walker, Frame-based geometric reasoning for construction and maintenance of 3D world models, School of Computer Science. Doctor of Philosophy Pittsburgh: Carnegie Mellon University, 1989, p. 145.
[97] K. Wu, M.D. Levine, Recovering parametric geons from multiview range data, Proceedings of Computer Vision and Pattern Recognition (CVPR), 1994, pp. 159–166.
[98] J.V. Wyngaerd, L.V. Gool, R. Kock, et al., Invariant-based registration of surface patches, Proceedings of the International Conference on Computer Vision (ICCV), September, 1999, pp. 301–306.
[99] S. You, J. Hu, U. Neumann, et al., Urban site modeling from LiDAR, Proceedings of the Workshop on Computer Graphics and Geometric Modeling (CGGM), May, 2003.
[100] K. Yue, D. Huber, B. Akinci, et al., The ASDMCon project: the challenge of detecting defects on construction sites, Proceedings of the Third International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT), June, 2006.