Adapting CAD models of complex engineering objects to measured point cloud data

Gábor Erdős[a], Takahiro Nakano (3)[b], József Váncza (1)[a,c]

[a] Institute for Computer Science and Control, Hungarian Academy of Sciences, Budapest, Hungary
[b] Hitachi, Ltd., Yokohama Research Laboratory, Yokohama, Japan
[c] Dept. of Manufacturing Science and Technology, Budapest University of Technology and Economics, Budapest, Hungary

Construction, maintenance and retrofitting of complex engineering objects like factories or plants calls for matching their model and actual state that are inevitably different. The paper presents a generic method for tailoring the computer aided design (CAD) model of such objects to their state given in terms of measured 3D, high resolution point clouds. The workflow includes efficient storage of massive measurement data, segmentation of a triangulated mesh-based CAD model into features, matching and adapting the features to the data. The method is demonstrated in a real-world setting, using the CAD model and point cloud data of an industrial plant.

Computer aided design (CAD), Object recognition, Point cloud

1. Introduction

This work is aimed at matching two different spatial geometrical representations of the same engineering object. Specifically, complex artefacts are studied whose *intended design* captured by a 3D computer aided design (CAD) model is not necessarily the same as their *actual* state that can be assessed by some dimensional measurement. Differences between model and reality are common for engineering objects with relatively long life-cycle, like equipment in production or power plants, whose original design is under unceasing change in time, from the very beginning of their construction. However, even deliberate changes are rarely documented by modifying the corresponding model, let alone the effects of accidental events [1]. Tailoring the model to reality time and again is though essential when monitoring the progress of *construction* projects and registering what has (or has not) been built according to schedule or specifications. *Operations and maintenance* (OM) can make more informed decisions by using a model that captures the up-to-date status of the object. When retrofitting or de-constructing complex industrial objects, a correct model of the work environment can facilitate both safety and efficiency. Finally, as presented by Bernard et al., technical heritage can be brought back in time by means of digital reconstruction methods [2].

Hence, the overall *objective* of this research was to develop appropriate methods for monitoring the state of complex constructed industrial objects by *optical measurement technology*, as well as matching this information with the 3D CAD model of the object under study. The comparison should primarily identify those elements of the reference model that can be recognized in the measurement data. Further on, in case of differences between reality and the model, elements and their relations in the model have to be adapted to their perceived status.

Both analysis of the state-of-the-art [3]-[6] and preliminary experiments led to the conclusion that in the actual problem domain where typically fine resolution measurements have to be taken of relatively large objects having complex structure, *3D scanning* using laser technology is superior to recording 2D visual information. Result of such scanning is a data set of points on the surface of the object, commonly referred to as *point cloud*.

While in various branches of engineering methods of recognizing 3D objects by means of optical measurement or computer tomography [7] methods are being intensively investigated, there are severe technology gaps when it comes to integrating sensor data of different sources [8], to working with complex geometry, and especially to handling objects with internal structure where measurement is burdened by clutter and occlusion [3]. In such cases, human involvement is still essential in the recognition or reconstruction process [1]-[5]. Recently, in the field of production engineering, Stark et al. have investigated segmentation, parts and structure identification methods in support of reverse engineering 3D assembly models from scanned data [9]. As an addition to the general recognition workflow, a contact graph of parts joined in an assembly is generated in the course of a semi-automated process. Recognition normally goes through the phases of registering point cloud(s), generating a triangulated surface mesh from tessellation of the points, noise filtering, subdividing the point data into smaller segments, and extracting volumetric entities or features that capture design intent in the reconstructed model [3][10]. Methods like curvature tensor-based region growing [11], random walks over topological neighborhood [12], or spin images [6] can be used for decomposing point clouds or meshes into segments that fit surfaces typical to engineered objects, such as planes, cylinders, or torus sections. Though, the main concern of this work is comparing the actual status of measurands to their reference model. Hence, segmentation can be driven by design intent embedded in the model, while for adapting model elements methods of dimensional metrology can be borrowed [7][8].

2. Problem statement

The measurement and model adaptation method had to meet a number of generic requirements. First, *neutral* representation should be used for the reference CAD model. Hence, CAD models will be given in *Standard Tessellation Language* (STL) format that is a triangular mesh representation of a 3D surface geometry. Finally, compared to actual recognition techniques used in the practice, increased accuracy and significant reduction of processing time are taken as key performance criteria.

The *input data* of the problem at hand are as follows:
- The CAD reference model of the object is given in terms of its STL representation that captures the object's polytope geometry as tessellated triangles.
- As measurement data, a point cloud in 3D is provided as a set of points characterized by their spatial coordinates. The points, even if measured from different locations, are all registered in the common space of the CAD reference model. The point cloud may contain several hundred million points.

The expected, automatically generated *results* of the optical recognition technique are the following:
- Well-defined, individual elements of the CAD reference model and their interconnections (like a subsystem of pipes).
- The *classification* of these elements according to the result of matching: whether they could be found within tolerance, found with changed geometry, partially found, partially found with changed geometry, or, after all, not found in the measurements.
- *Modification* of the parameters of those elements that have been found with changed geometry.

Performance of the recognition method is evaluated in terms of (1) the *ratio* of the correctly recognized elements, and its (2) required *total processing time*. Note that reverse engineering of those elements that are not included in the reference model but could be recognized in the measured point cloud is out of the scope of the above problem statement and is subject of future research.

3. Workflow of CAD model to point cloud matching

The problem statement implies conflicting challenges whose resolution calls for the balanced applications of principles well-known in production engineering, too. First, the charge of using a straightforward, robust and uniform representation for CAD modelling like STL is that the model does not contain any explicit structural information. At the same time, because of the large size of the data to be processed and the complexity of the calculations involved, it is essential to apply the principle of "divide and conquer" and *decompose* both the CAD model and the point cloud into smaller segments. Hence, there is a need of recognizing in the CAD model local *features*, together with their properties and relations. Decomposition should facilitate not only the matching of features to appropriate subsets of the measured points, but also the parallelization of such computations. This is the key to exploiting the potential of general purpose computation on graphics processing units (GPGPU).
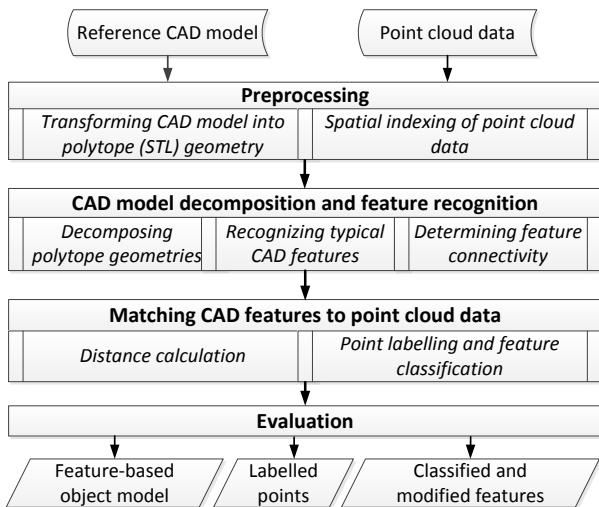

Figure 1. Workflow of the CAD model recognition process.

Finally, the large size of datasets requires calculations to be performed over point cloud data that are *indexed* according to the special features of the problem at hand. The recognition *workflow* consists of the phases shown in Figure 1. Pre-processing involves the standard transformation of a CAD model from a proprietary format into polytope (STL) geometry, as well as the efficient storage of massive measurement data. So as to support efficient queries, so-called *spatial indexing* [13] is used. These pre-processing steps are not elaborated here; instead, focus is set to further phases of the workflow in the sequel.

4. CAD model decomposition and feature recognition

4.1 Decomposing polytope geometries

The first step in feature recognition is to decompose the complete polytope CAD model into a set of subcomponents. The subcomponents are also polytope geometries but they contain less faces, moreover their bounding boxes are much smaller than that of the whole object model. If the original model is considered an assembly, than the subcomponents are the parts of this assembly. Even though the STL format in general does not store any topological information but only a "soup of triangles", fortunately, one may exploit that the STL representation of the complex object have been generated by a CAD system using *manifold geometric models.* In such models the volumes are well defined: (1) each edge belongs to two faces, (2) each face is surrounded by a loop of edges, (3) faces meet each other only in common edges and vertices, and finally, (4) there is material only on one side of a face, meaning that the faces' normal vectors are pointing always in the "outward" direction.

Hence, the CAD system implicitly defines the topologies of the manifold models as they are exported into mesh representation. These topologies can be recognized and extracted by making use of *face adjacency graphs* (FAG). *Nodes* of the face adjacency graph are faces given as triangles, while there is an *edge* between any two nodes that represent adjacent triangles. The assembly composed of parts is represented by a set of disconnected FAGs, while the FAG of any such part may contain connected components characteristic to some particular object. For instance, Figure 2 shows the mesh model and the corresponding FAG of a cuboid object.
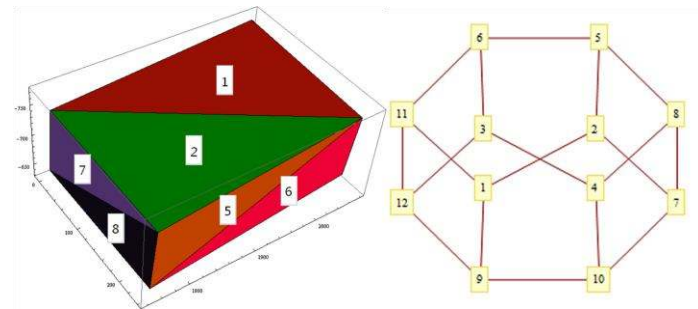

Figure 2. Face adjacency graph of a manifold polytope object (cuboid).

4.2 Recognizing typical CAD features

The tessellated triangles define the object in terms of its polytope geometry. The goal of *feature recognition* is to decompose this model into such smaller entities of specific local topological and geometrical characteristics that facilitate the matching of the model to the measured points. Since this process involves both extensive computations and manual modifications, such *features* have to have also accepted semantics in the application domain. In the case study the method was applied to recognizing a plant (for details see Sect. 6.2) consisting of pipes, bent pipe segments and containers. Consequently, the respective

features have been cylinders, torus sections and cuboids. (Note that Lübke et al. use similar features for separating measured points in the domain of micro deep-drawing [14]). The CAD model of an engineering object may also contain symbolic elements – like, in a plant model cone and full torus is used for representing a valve – but these have to be excluded from the matching process because they have no corresponding measurement data.

Features are defined in terms of some local properties of their polytope geometries. A *cylinder* feature contains two circular planar faces, where the vector connecting the center of the two circular faces is parallel to the opposing normals of the faces, and the radii of the two planar faces are equal. A *torus section* feature has circular planar faces, where the normal vectors of the circular faces are not parallel to each other, but the radii of the two planar faces are equal. A *cuboid* feature contains six planar faces, each given by four vertices. Further on, the six faces are defined with eight different vertices, and the adjacent edges are perpendicular to each other.

Definitions of features are expressed in feature recognition procedures that are applied to every FAG of the object under study. For cylinder features it is first checked whether the FAG contains any *planar* faces, i.e., connected triangles with opposite face normal within a pre-defined tolerance range. Next, *circularity* is checked: outer loops are calculated and a circle is fitted on the vertices. Whenever circles are found, by taking their radius and the center points of the planar faces, the conditions of the cylinder as defined above can be checked. Torus section feature is recognized in a similar way, but unlike to the cylinder its position and orientation is not fully defined by the center points of the two circular faces. In this case a *reference coordinate system* is calculated and assigned to the torus section feature. (For lack of space, details of this as well as of the cuboid recognition procedure are not given here.) Figure 3 presents the stages and results of the cylinder and torus section recognition processes.
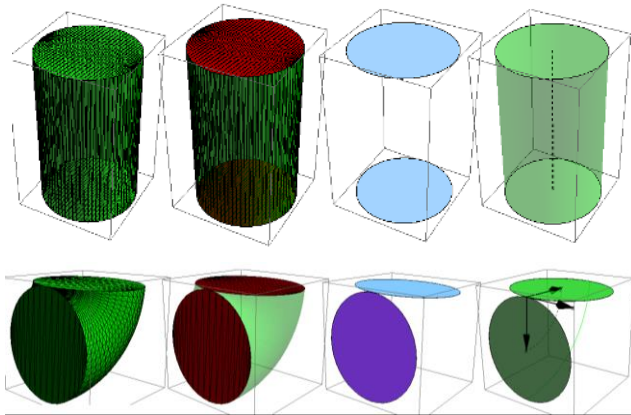

Figure 3. Recognition steps of cylinder (a) and torus section (b) features, with extracted parameters.

4.3 Determining feature connectivity

Local entities as they are features cannot be handled entirely in separation because they are interacting with each other. Interactions are specific to the domain and the feature types. For instance, in the plant recognition domain under particular study *connectivity* of the cylinder and torus section features is important information because it refers to a subsystem of bended pipes. Hence, taking only the potential target features, a *feature connectivity graph* is created where nodes are features and edges represent neighborhood relations. Two features are considered neighbors if the distance between their start and end points is within a given tolerance range. Figure 4 provides a closer look of a recognized connected subsystem of bended pipes.
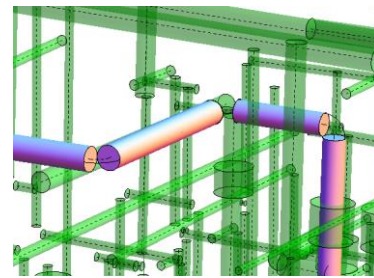

Figure 4. Connected pipe subsystem: details of recognized bended pipes.

5. Matching CAD features to point cloud data

The goal of *feature matching* is to decide whether the recognized CAD features exist or not in the measurements, and if they exist, what are their real parameters, positions and orientations. Feature matching is based on the assumption that the local, measured points in the vicinity of a CAD feature hold basically sufficient information for answering these questions. Hence, the feature matching algorithm proceeds by matching each feature with the subset of points that fall within the *offset bounding box* of the feature at hand.

5.1 Feature points based on distance calculation

The set of *selected points* of the bounding box query may contain so-called *feature points*, i.e., points related to the CAD feature, but further on also points that belong to other features or even to objects that are not included in the reference model. On the other hand, due to limited visibility, the set of scanned points might give only partial information about the surface of an object. However, given the kind of measurand – or, better to say, the type of CAD feature – the existence of this object can be deduced. Accordingly, a CAD feature is *recognized* if the majority of the selected points within its bounding box are feature points. Note that this definition applies for partially scanned features, too. The *degree of match* is defined as follows: there is given a feature f in an arbitrary position and orientation, a set of n selected points within the bounding box of f, and a tolerance ε. The degree of match is then characterized by the *number of points* with *distance* to the surface of f less than ε. This distance is calculated in a feature specific way: for a cylinder given with its center line and radius r, feature points are within the $r\pm\varepsilon$ range of the center line. For torus section the definition is similar, but the points should be within $r\pm\varepsilon$ range of the arc connecting the endpoints (see Figure 5). For cuboids, the degree of match is given as the number of points whose minimal distance from any of the faces of the cuboid are within the $\pm\varepsilon$ range.
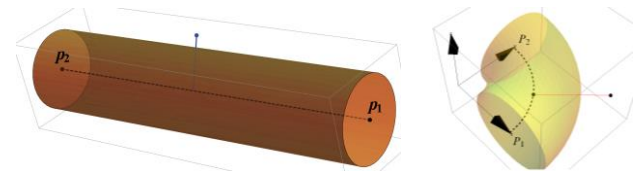

Figure 5. Distance calculation for cylinder (a) and torus section (b).

Next, the matching algorithm *optimizes* the position and orientation of the feature in such a way that it maximizes the degree of match. This is done by iterative search, with feature specific operators. For a cylinder, the start and end points of the center line are varied until the maximal number of points gets within tolerance. Adjoining torus sections can next be found in the feature connectivity graph, whose circular faces are adjusted to those of the cylinders, and again, local search looks for the most fitting radius parameter value. Figure 6 shows the reference (green) and optimized (orange) posture of a cylinder, together with the distance distribution calculated for 100000 points.
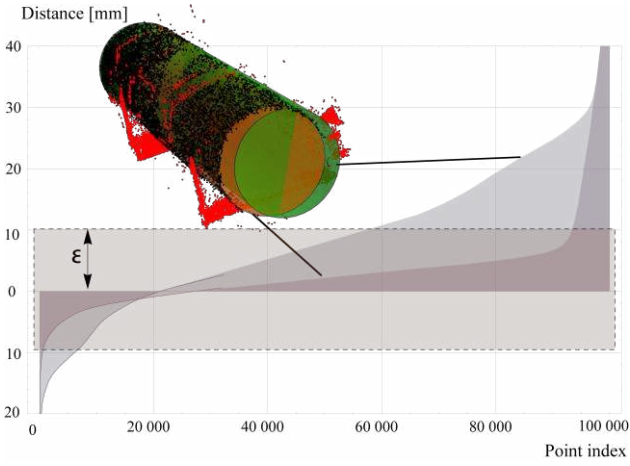
Figure 6. Matching and adapting cylindrical feature: Distance distribution for reference and optimized posture. Non-feature points are in red color.

5.2 Point labelling and feature classification

As a result of feature matching, within the local bounding box of any feature each point can be *labelled* as (1) feature point, (2) inner non-feature point, or (3) outer non-feature point. This labelling, in turn, is used to evaluate the result and to provide a final classification of features in light of the actual point cloud data. Specifically, result of matching needs an *engineering interpretation* because the CAD features may (or may not) match to the point cloud in different ways, depending on the amount, quality and distribution of data. Hence, the following classification has been introduced:

- *Found*: the point cloud available provides sufficient evidence for the existence of the feature in the real environment (see Figure 7a).
- *Found with changed geometry*: The point cloud provides sufficient evidence for the existence of the feature. However, there are considerably many inner points; hence the reference CAD model only approximates reality.
- *Partially found*: The point cloud gives some evidence for the existence of the feature, but due to some reasons (changes in the reference geometry, occlusion) this is weak. Such recognized features have relatively many outer points.
- *Partially found with changed geometry*: The point cloud data provides some evidence for the existence of the feature, some segments of it can even be identified, however, only with changed geometry. Features recognized this way have relatively many inner and outer points (see Figure 7b).
- *Not found*: The point cloud data provides no sufficient evidence for the existence of the feature because it cannot be found within the bounding volume using the point cloud.

So as to make the above classification scheme operational, a finer distinction for evaluating the quality of matching has to be made: supposing that matching of a feature resulted in N feature points, I inner non-feature points and O outer non-feature points, the *matching ratio R* is defined as $R=N/(N+I+O)$. On the other hand, the *matching ratio without inner non-feature points*, S is calculated as $S=N/(N+O)$. By using thresholds for the values of R and S, the above classification rules can be declared as presented in Table 1.

Table 1. Classification rules

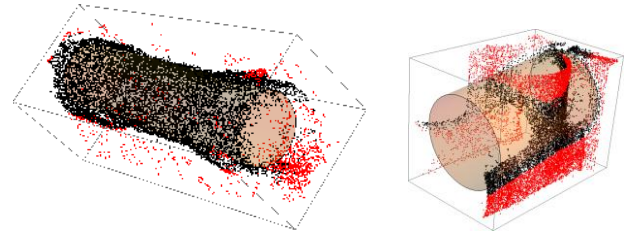| Classification	Condition
Found	$R > 80\%$
Found with changed geometry	$R \leq 80\% \wedge S > 80\%$
Partially found	$40\% < R \leq 80\%$
Partially found with changed geometry	$R \leq 40\% \wedge S > 40\%$
Not found	$N < 1000 \vee S \leq 40\%$

Figure 7. Examples of featured classified as *found* (a) and *partially found with changed geometry* (b). Non-feature points are in red color.

6. Experimental results

6.1 Implementation

The methods presented above have been implemented on top of an Oracle database system that was dedicated to store and handle point cloud data on a very large scale. So as to speed up bounding box queries, special octree-based spatial indexing scheme has been applied [13]. All the algorithms have been implemented by using *Mathematica* v8. The tests have been run on a virtual server machine with 6 core Intel Xeon X5650 2,67 GHz processor and 6 GB RAM.

6.2 Industrial case study

The workflow and the algorithms have been tested in a real-life setting, for matching the CAD model of a pure water equipment plant to laser scanned point cloud data. The industrial partner provided the CAD reference model in form of an STL file that contained 188616 vertices and 375540 faces. The 3D measured point cloud was collected in an area of ca. 6000 x 6000 x 4500 mm by a phase-based laser scanner from 25 different locations. The measured and registered dataset of the target plant consisted of ca. 250 million points and required storage space (as text files) of ca. 15 GB. Thanks to spatial indexing, the bounding box query – a basic procedure applied in several phases of the workflow – became extremely fast (1-2 sec/query in the whole dataset). A sample of measured point cloud is shown in Figure 8.
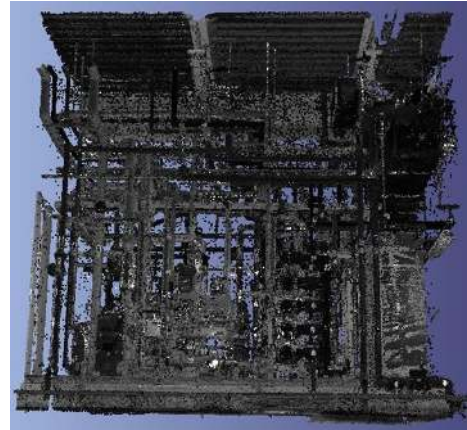

Figure 8. 3D measured point cloud of the pure water equipment.

In the feature recognition phase, with an accuracy tolerance ε set to 5 mm, altogether 463 cylinder, torus section, and cuboid features have been automatically identified (see also Table 2). A closer inspection has shown that all the non-recognized subcomponents of the CAD model had been full torus and cone geometries representing symbolic valves. The total processing time of feature recognition in the CAD model was 220 sec. After finding the features their connectivity graph has been built by exploiting information of connected feature pairs. Figure 9 depicts the feature connection graph of the whole plant where the longer chains represent continuous bended pipe segments.
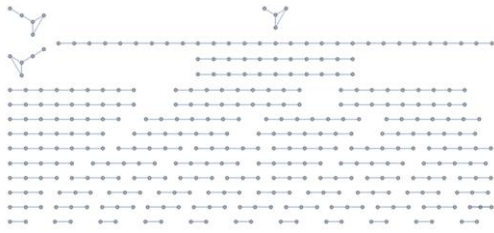
Figure 9. Feature connectivity graph of the whole plant.

The overall results of matching features to the point cloud are summarized in Table 2. All in all, less than 5% of the features have not been found. Individual analysis exposed some reasons of failed recognition: the original feature was removed in the meantime, some features were included or replaced by other ones not in the reference model, some were densely surrounded by points belonging to other features, or simply, due to obstructed visibility, there were too few measured points to identify them. As for cylindrical features, Figure 10 shows color-coded results of matching, with found (green), partially found (orange) and not found (red) pipe segments. The total time of matching cylinder features took 180 minutes, while torus section features were matched in 105 minutes. Cuboid features have been completely recognized, hence all of them are depicted in green color in Figure 11. The total time of matching cuboid features was 30 minutes.

Table 2. Summarized classification results for the main feature types.

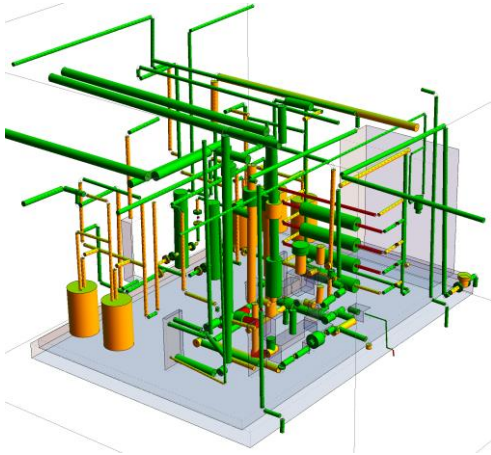|	cylindrical		torus section		cuboid	
Classification	**#**	**%**	**#**	**%**	**#**	**%**
Found	137	45	58	42	15	83
Found with chgd. geometry	46	15	7	5	3	17
Partially found	92	30	69	50	0	0
Part. found with chgd. geo.	13	4	0	0	0	0
Not found	19	6	4	3	0	0
Total	**307**	**100**	**138**	**100**	**18**	**100**

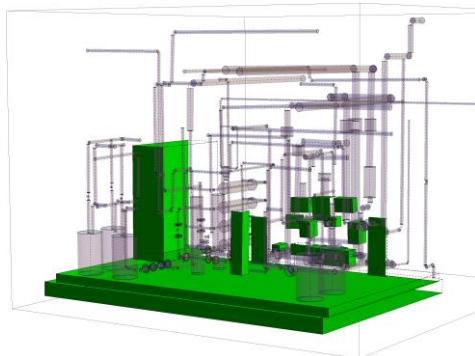**Figure 10.** Results of matching cylindrical features.

Figure 11. Results of matching cuboid features.

7. Conclusions and future work

The presented CAD model matching method takes widely supported input in form of mesh models, works also for partially measured objects, and for any type of features. The only precondition of its application is to have an appropriate distance function for assessing the degree of match between a feature and its respective point cloud data. Tolerance of matching can be adjusted, so the method is capable of handling noisy data. Computations in the most intensive phase of the workflow can be parallelized and implemented by using GPGPU technology.

As core of the workflow a novel feature recognition method has been developed that uses the face adjacency graph as its only input. The method is generic, neutral, and is able to work on very large graph structures as well. Since it cuts across the interfacing problem of different CAD systems, it has wide application potential. For instance, a precursor of this recognition method was also applied in a manufacturing process planning research for identifying volume primitives to be removed by distinct machining operations (see [15]).

Future work is aimed at improving the accuracy of registration by means of segmentation based methods [16], reducing processing time by parallelization, as well as identifying those elements of the real, complex object that are not modelled in the reference CAD model but have traces in the measured data. This reverse engineering task is accomplished by informed guesses. Once having a hypothetical model of a feature, the method presented above can be applied for assessing its existence.

Acknowledgement

References

[1] Goedert JD, Meadati P (2008) Integrating Construction Process Documentation into Building Information Modeling. *Journal of Construction Engineering and Management* 134(7):509–516.
[2] Bernard A, Laroche F, Ammar-Khodja S, Perry N (2007) Impact of New 3D Numerical Devices and Environments on Redesign and Valorisation of Mechanical Systems. *CIRP Annals - Manufacturing Technology* 56(1): 143–148.
[3] Tang P, Huber D, Akinci B, Lipman R, Lytle A (2010) Automatic Reconstruction of As-Built Building Information Models from Laser-Scanned Point Clouds: A Review of Related Techniques. *Automation in Construction* 19(7):829-843.
[4] Arayici Y, Hamilton A, Gamito P (2006) Modelling 3D Scanned Data to Visualise and Analyse the Built Environment for Regeneration. *Surveying and Built Environment* 17(2):7–28.
[5] Randall T (2011) Construction Engineering Requirements for Integrating Laser Scanning Technology and Building Information Modeling. *Journal of Construction Engineering and Management* 137(10):797–805.
[6] Date H, Kaneta Y, Hatsukaiwa A, Onosato M, Kanai S (2011) Object Recognition in Terrestrial Laser Scan Data Using Spin Images. *Computer-Aided Design and Applications* 9(2):187–197.
[7] Kruth JP, Bartscher M, Carmignato R, Schmitt R, De Chiffre J, Weckenmann A (2011) Computed Tomography for Dimensional Metrology. *CIRP Annals - Manufacturing Technology* 60(2):821–842.
[8] Weckenmann A, Jiang X, Sommer K-D, Neuschaefer-Rube U, Seewig J, Shaw L, Estler T (2009) Multisensor Data Fusion in Dimensional Metrology. *CIRP Annals - Manufacturing Technology* 58(2):701–721.
[9] Stark R, Grosser H, Müller P (2013) Product Analysis Automation for Digital MRO Based on Intelligent 3D Data Acquisition. *CIRP Annals - Manufacturing Technology* 62(1): 123–126.
[10] Wang J, Gu D, Yu Z, Tan C, Zhou L (2012) A Framework for 3D Model Reconstruction in Reverse Engineering. *Computers & Industrial Engineering* 63:1189–1200.
[11] Lavoué G, Dupont F, Baskurt A (2005) A New CAD Mesh Segmentation Method Based on Curvature Tensor Analysis. *Computer-Aided Design* 37(10):975–987.
[12] Lai Y-K, Hu S-M, Martin RR, Rosin PL (2009) Rapid and Effective Segmentation of 3D Models Using Random Walks. *Computer Aided Geometric Design* 26(6):665–679.
[13] Schön B, Mosa ASM, Laefer DF, Bertolotto M (2013) Octree-Based Indexing for 3D Pointclouds within an Oracle Spatial DBMS. *Computers & Geosciences* 51:430–438.
[14] Lübke K, Sun Z, Goch G (2012) Three-dimensional Holistic Approximation of Measured Points Combined with an Automatic Separation Algorithm. *CIRP Annals - Manufacturing Technology* 61(1):499–502.
[15] Nonaka, Y, Erdős G, Kis T, Kovács A, Monostori L, Nakano T, Váncza J (2013) Generating Alternative Process Plans for Complex Parts. *CIRP Annals - Manufacturing Technology* 62(1):453–458.
[16] Shaw L, Weckenmann A (2011) Automatic Registration Method for Hybrid Optical Coordinate Measuring Technology. *CIRP Annals - Manufacturing Technology* 60:539–542.