2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Building Semantic Object Maps from Sparse and Noisy 3D Data

Martin Günther, Thomas Wiemann, Sven Albrecht and Joachim Hertzberg

Abstract— **We present an approach to create a semantic map of an indoor environment, based on a series of 3D point clouds captured by a mobile robot using a Kinect camera. The proposed system reconstructs the surfaces in the point clouds, detects different types of furniture and estimates their poses. The result is a consistent mesh representation of the environment enriched by CAD models corresponding to the detected pieces of furniture. We evaluate our approach on two datasets totaling over 800 frames directly on each individual frame.**

I. Introduction

Building 3D maps of indoor environments by mobile robots has received increasing interest since the launch of inexpensive 3D sensors such as the Microsoft Kinect. Several successful approaches exist that generate 3D point-cloud maps (e. g., [1], [2]) or mesh representations (e. g., [3]) based on RGB-D data. Nevertheless, automatically providing additional *semantic* information to the maps, such as location and type of furniture present, is still an unsolved problem. This kind of semantic information however is necessary for many advanced tasks of an autonomous robot, such as object search or place recognition. Also, it has advantages for the map building process itself: If the class and location of objects in the map are known, CAD models can be used to complete missing sensor data, or loops can be closed based on semantic in addition to geometric information.

This paper presents an approach to semantic mapping that:

1) reconstructs the surfaces from noisy 3D data, captured from a single frame of a Kinect camera, and creates a triangle mesh;
2) recognizes furniture objects in the point clouds;
3) and finally adjusts their poses using ICP, and augments the created map with their corresponding CAD models.

In our previous work [4], [5], we have shown preliminary results using 3D laser scanner data obtained by manually placing the laser scanner in several positions in an office scene. In this paper, we extend the previous work to frame-based semantic analysis of Kinect data. The motivation for the frame-based approach is that it allows for an online analysis of the perceived data. In the future, this can be utilized to explicitly gather new data in an attention based fashion and to close the loop between data interpretation and gathering. Using state of the art SLAM algorithms, these annotated point clouds can be easily used to maintain a consistent semantic map of the complete environment. The system is evaluated extensively on two different dataset consisting of a total of 810 point clouds containing six different classes of furniture.

All authors are with the Institute of Computer Science, University of Osnabrück, Albrechtstr. 28, 49076 Osnabrück, Germany. Joachim Hertzberg has a second affiliation with the DFKI Robotics Innovation Center, Osnabrück Branch, 49076 Osnabrück, Germany. This work is supported by the RACE project, grant agreement no. 287752, funded by the EC Seventh Framework Programme theme FP7-ICT-2011-7.

{martin.guenther, thomas.wiemann, sven.albrecht, joachim.hertzberg}@uni-osnabrueck.de

II. Related Work

Several approaches exist that build geometric maps from 3D point clouds. Nüchter et al. [6] present a 3D SLAM approach that can build a consistent map from arbitrary 3D point clouds. Endres et al. [1], Henry et al. [2] and May et al. [7] propose 3D SLAM approaches exploiting the specific characteristics of RGB-D cameras to improve the registration quality. Biswas and Veloso [8] also developed a 3D mapping system geared towards RGB-D cameras, but instead of a point cloud map, they create a map of the planar regions in the environment. The KinectFusion algorithm by Newcombe et al. [3] instead creates a mesh from RGB-D data in real time, but is limited to a constrained volume.

In the field of semantic mapping, several authors have proposed algorithms that label point clouds with semantic information. For instance, Rusu et al. [9] detect several types of furniture and approximate them as cuboids. Nüchter and Hertzberg [10] classify coarse structures (walls, floors, doors) using a semantic network and detect smaller objects using a trained classifier. Mason and Marthi [11] autonomously build a semantic object map over a long time span, focusing on small and medium-sized objects instead of furniture. Neumann and Möller [12] investigate the use of description logics for high-level scene interpretation tasks. Similar to our approach, Pangercic et al. [13] build a semantic map based on furniture CAD models.

CAD models have been used for object recognition before. For mass-produced objects, ranging from furniture over household appliances to tableware, CAD models exist and are widely available – either as the exact model directly from the manufacturer or as a similar model via sources like the Google 3D Warehouse. The first approaches in this direction started in the mid 90's using vision based sensors [14] or a combination of a laser projector, a stereo camera and several additional cameras [15]. These approaches have in common that they try to recognize objects at a known position, i. e., the object in question is already centered in the obtained sensor data and no additional objects or occlusions are present in the sensor data. A more recent approach on a larger scale is presented in [16], where several matched 3D laser scans of a construction site are compared to a model in order to track progress and detect divergences between model and actual

construction site. A prerequisite for this work is a correct model in advance, i. e., it is known what the sensors are supposed to measure and subsequently a quantitative analysis concerning the differences between expected measurements and received measurements is performed. In contrast to these approaches we neither demand a complete model of the perceived scene nor do we require the exact poses of candidates for object recognition in advance, but employ general domain knowledge to generate object hypotheses and use CAD models to refine these. From this characterization, our approach is similar to the work of [17], [18], [19]. The major difference to [17] is that we perform the actual CAD matching with the 3D environmental data instead of 2D image data. In [18], the authors use CAD models to create synthetic point clouds of several types of furniture and extract geometric features which are used to build a vocabulary of objects via machine learning techniques. After a probabilistic Hough voting step to generate likely object hypotheses, they apply a RANSAC approach to fit the objects into the scene and confirm / refute their hypotheses. They also work on individual frames (like our system does). Lastly, the work in [20] should be recognized, which reports promising results in object labeling using full RGB-D information (color and depth). However their work focuses on streams of registered point clouds and thus the available data is usually much more complete than from a single frame.

III. Model Based Object Recognition

The system consists of three steps: (1) surface reconstruction transforms the input point cloud into a triangle mesh and extracts planar regions; (2) the planar regions are classified, furniture objects detected, and an initial pose estimate based on the planar regions is calculated; (3) the final pose is computed using ICP, and the corresponding CAD model placed in the scene. Fig. 5 exemplifies these steps.

A. Surface Reconstruction

The plane extraction for object recognition is done on a mesh representation of the surfaces captured with the Kinect camera using the Las Vegas Surface Reconstruction Toolkit (LVR) [21], [22]. LVR provides an open source C++ library with implementations of several algorithms for polygonal map generation. The surface reconstruction process mainly consists of two steps: initial mesh generation using Marching Cubes followed by a post-processing pipeline of several mesh optimization and segmentation algorithms.

Mesh generation is done using an optimized Marching Cubes implementation that utilizes Hoppe's distance function [23] to estimate an isosurface representation of the point cloud data. To generate this isosurface, surface normals for the data points have to be estimated. This is done using an adaptive RANSAC-based approach that is optimized for sparse data sets containing Kinect specific discretization effects and noise [22].

To approximate a triangle mesh representation to the isosurface defined by the points and normals, a modified Marching Cubes algorithm is used. This initial mesh is enhanced by several mesh optimization algorithms. To remove artifacts based on outliers in the point cloud data, small clusters of connected triangles are automatically removed. Furthermore, LVR delivers a hole filling procedure to close holes in the mesh that result from occlusions. After initial mesh generation, hole filling and outlier removal, connected planar patches in the mesh are extracted. The planar segmentation is done via a region growing approach, which has to check each triangle only once (as described in [22]).

The output of this algorithm is a set of planar clusters that can be used to generate the model hypotheses for the object recognition process. For this step, the actual size of an extracted plane is needed. In contrast to point clouds, the exact area can be easily computed in a mesh representation by summing up the areas of the created triangles. In real-life application scenarios, the interesting surfaces of furniture will usually be populated with objects on top of them, especially in table top scenarios. For example, take a laid breakfast table where the table top contains dishes, bowls and various other objects that are needed for a proper breakfast. These objects will certainly create occlusions and holes in the reconstruction of the table top plane. As long as our region growing algorithm can find connected patches of the surface between the present objects, we will get a representation of that area, but the estimated area will be significantly reduced due to the occlusions. To restore the initial plane, we extract the contours of the holes in the plane and the outer contour. This can easily be done by collecting the border edges in a half edge mesh representation. To get an optimized contour representation, we fuse edges on the same line via contour tracing, using the `psimpl` library [24]. Once all contours of a plane are collected, we sort them topologically to detect the outer contour of the planar region. This contour is then triangulated using the OpenGL tesselator. This way, all holes within the plane are closed and we get a realistic approximation of the area of an occluded planar surface (cf. Fig. 2).

This approach works fine as long as the outer contour is not interrupted by shadows of present objects. In this case, the shadow might break the outline and create a bay in the contour which in turn will decrease the estimated surface. Alternatively, one could use a convex hull approach to estimate a surface, but by doing so the surface of non-convex planes – like the L-shaped desk in the office dataset (Sec. IV) – would be overestimated. Our approach can be used for arbitrary shapes. Another problem occurs when a surface is too populated. In this case, the region growing procedure will not be able to find a connected remaining surface and the estimated plane will be split. An approach to solve this problem is to detect close patches which satisfy similar plane equations. In these cases, their areas can be summed up. While these optimizations make the area estimation more stable, the classification step (next subsection) still needs to be robust to variations from the true area size. An analysis to evaluate the robustness of our approach is presented in Section IV.

B. Planar Region Classification

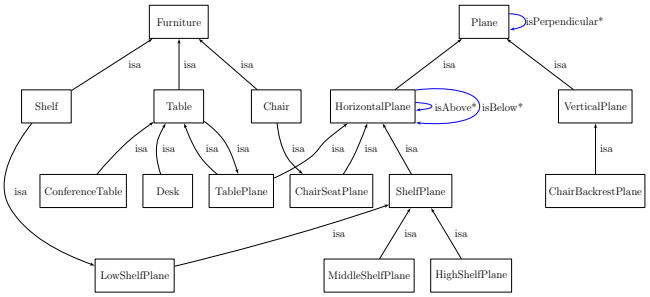

Fig. 1. Parts of the OWL-DL ontology used for classification: classes (black) and properties (blue).

Once all planar regions have been extracted in the previous step, those regions corresponding to pieces of furniture have to be classified. Here, we make use of the fact that most pieces of furniture are comprised of planar structures that have a certain size, orientation, height above ground and spatial relation to each other. These features (and combinations of features) are expressed in an OWL-DL ontology in combination with SWRL rules.

The Web Ontology Language (OWL) is the standard proposed by the W3C consortium as the knowledge representation formalism for the Semantic Web. One of its three sub-languages, OWL-DL, corresponds to a Description Logic, a subset of First-Order Logic that provides many expressive language features while guaranteeing decidability. It has been extended by SWRL, the Semantic Web Rule Language, which allows to write Horn-like rules in combination with an OWL-DL knowledge base and includes so-called built-ins for arithmetic comparisons and calculations. We decided to use OWL-DL as the knowledge representation format for this paper for several reasons: OWL-DL ontologies can be easily re-used and linked with other sources of domain knowledge from the Semantic Web, they easily scale to arbitrarily large knowledge bases, and fast reasoning support is available. In our implementation, we use the open-source OWL-DL reasoner Pellet, which provides full support for OWL-DL ontologies using SWRL rules.

The class hierarchy of the ontology we use here is shown in Figure 1. The basic classes are Furniture (the parent class of all recognized furniture objects) and Plane (the planar regions of which Furniture objects are comprised). A set of SWRL rules is applied to the extracted planar regions to assign them classes in the Plane sub-hierarchy; for example, the lower plane of a shelf can be characterized by the following SWRL rule:

```
LowShelfPlane(?p) ← HorizontalPlane(?p)
   ∧ hasSize(?p, ?s) ∧ swrlb:greaterThan(?s, 0.01)
   ∧ swrlb:lessThan(?s, 0.5) ∧ hasPosY(?p, ?h)
   ∧ swrlb:greaterThan(?h, 0.08)
   ∧ swrlb:lessThan(?h, 0.18)
```

These rules are not exclusive, so one planar region can receive multiple labels (e. g. MiddleShelfPlane and ChairSeatPlane). The definitions of the classes in the Furniture sub-hierarchy refer to these labels; e. g., the fact that a Shelf consists of three planes on top of each other can be stated as:

```
Shelf ≡ LowShelfPlane and
   (isBelow some (MiddleShelfPlane and
      (isBelow some HighShelfPlane)))
```

Likewise, chairs are defined by a seat and a backrest, both with certain sizes, orientations, heights above ground and which are perpendicular to each other.

For each detected object, initial position and orientation are estimated. The position is always the centroid of their main plane. Since chairs have two perpendicular planes (the backrest and the seat), a unique orientation can be calculated by the difference vector between the centroids of those planes. For tables and shelves, the PCA of their main plane is calculated to define the orientation.

C. Final Pose Adjustment and Model Replacement

The initial pose estimate calculated in the previous step is potentially inaccurate, since it wholly depends an abstraction of the recognized objects as a set of planar regions. To improve the pose estimate, we perform ICP matching of the appropriate (sampled) CAD model with the original point cloud data. This process is described in detail in [5].

IV. Results

We performed two experiments to evaluate the robustness and accuracy of our recognition system. First, we investigated the effectiveness of our hole filling procedure separately to see whether it is capable of estimating the true surface area of a table under the influence of increasing amounts of clutter. Second, we evaluated the detection accuracy of our complete system on two series of point clouds captured by a mobile robot.

A. Robustness Against Occlusion

In real life applications, furniture is usually used to store objects, so an obvious problem for our detection procedure is that the surfaces relevant for recognition may be partially occluded. To evaluate the robustness of the surface extraction procedure against occlusions, we gradually added typical objects like books, cups and bottles to a table surface and tried to segment the table top. The results of this experiment are shown in Fig. 2 and Table I. The estimated area remains close to the ground truth area even for significant amounts of clutter; only when the outer contour is disrupted, the area estimation breaks down.

B. Complete System

To evaluate our recognition system, we captured two series of point clouds from a Kinect camera mounted on a mobile robot (see Fig. 3). In the first scenario, the robot was teleoperated around an office while continuously capturing point cloud data at 2.2 Hz, resulting in a total of 431 point clouds. The office contained 13 recognizable objects from 5 classes (1 desk, 1 conference table, 1 office chair, 5 conference chairs and 5 book shelves). For the second dataset, the robot

TABLE I

RECONSTRUCTED AREAS IN THE TABLE TOP EXPERIMENT UNDER INCREASING AMOUNTS OF CLUTTER (RECONSTRUCTED TABLE TOP AREA IN M^2 AND AS PERCENTAGE OF GROUND TRUTH). FIRST ROW: ONLY REGION GROWING. SECOND ROW: REGION GROWING + CONTOUR TRIANGULATION.

	0 objects	2 objects	3 objects	4 objects	5 objects	6 objects	7 objects	12 objects
Region Growing	1.50 93%	1.47 92%	1.47 92%	1.40 87%	1.35 84%	1.26 79%	1.17 73%	0.95 59%
Contour Triangulation	1.50 93%	1.50 93%	1.49 93%	1.52 95%	1.52 95%	1.52 95%	1.50 93%	1.20 75%

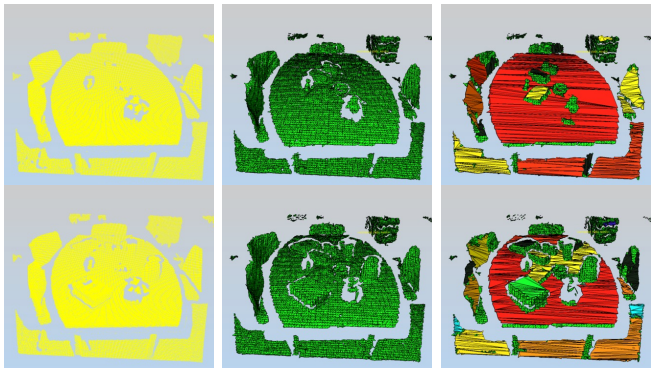

Fig. 2. Segmentation results for a table top setup. First column: the captured point clouds; second column: initially created triangle mesh; third column: segmentation results. Triangles that were not classified as belonging to a planar patch are rendered in green. In each step more objects were added. In the last line a shadow disrupted the outer contour and the segmentation broke the table top plane into two clusters.

platform was driven through a seminar room, capturing a total of 379 point clouds. The objects present in this dataset are 12 seminar tables and 20 chairs. One challenging aspect of this dataset is that there is a high level of occlusion.

For both datasets, we registered the point clouds into a consistent full-scene point cloud, using the SLAM6D toolkit [6]. The full-scene point clouds were used to generate the ground truth poses for each piece of furniture by hand. These poses are used to evaluate the results of our classification and the ICP refinement step. Note that the registration information is only used in the evaluation process; the recognition system itself does not require the input point clouds to be registered.

Ground truth data for each frame was generated by manually labeling each frame with the information which of the objects occur in that frame. The ground truth poses of each object were estimated by manually placing each CAD model into a scene consisting of the fully registered datasets (see Figure 4). In combination with the camera trajectory obtained from the registration process, the ground truth object poses in each frame can be computed. A detection is considered "true positive" if its distance to the nearest true object pose of the same class was below a threshold, depending on the CAD model's size (15 cm for chairs, 25 cm for shelves, 45 cm for tables). If multiple detections fell into that range, only the nearest was counted as a true positive, and the others as false positives.

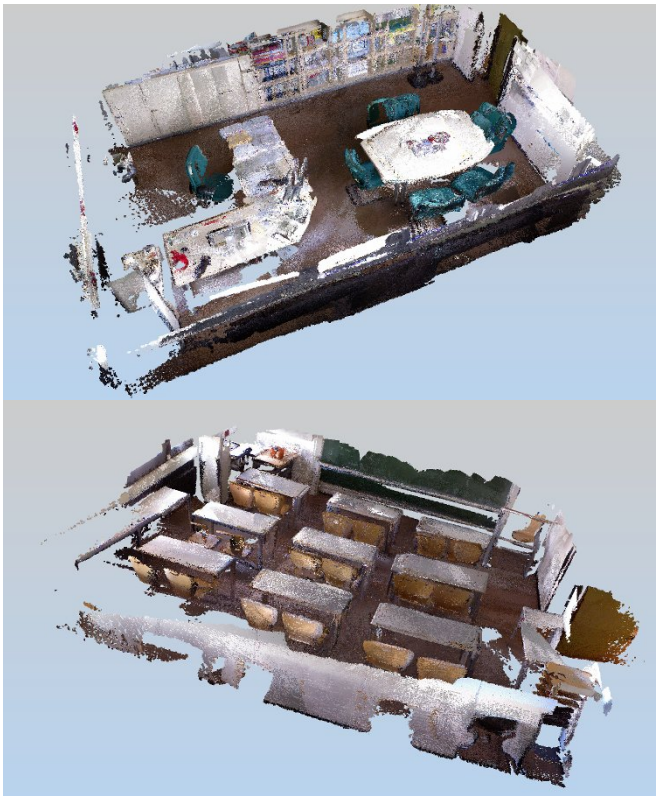

Fig. 4. Overview of fully registered frames. Top: Office dataset (431 frames); bottom: seminar room dataset (379 frames). Note that point color information is displayed for reader convenience, our approach does not require it.

Fig. 3. The Kurt robot used for our experiments.

These high thresholds were chosen due to the fact that each point cloud is subject to considerable sensor noise, especially near the limit of the depth camera's range. Even if all individual point clouds are registered perfectly, there will be noisy "shadow points" around each object. Our ground truth poses are located in the middle of the noisy point cloud resembling the object in question.

The detection results for both datasets are shown in Table II. In addition to the detection results, the translation and rotation error of the initial guess (based on PCA for tables and shelves, and based on the vector from backrest to seat for chairs) and the translation and rotation error after ICP pose correction are shown. Figure 5 depicts some exemplary object detections for single frames while Figure 4 provides an overview of the two complete registered datasets.

The detection rates for the single point clouds are acceptable and show that our approach is not only robust enough to deal with the noise present in low cost 3D sensors, but also copes with occlusion and partial visibility, typical for sensors with a small opening angle. Unexpectedly, the final ICP pose correction did not improve the average initial guess on single frames for the first dataset. We attribute this to the fact that we matched a complete CAD model to a partially occluded object view; possible solutions are outlined in the next section. In the second dataset however, ICP clearly improved the rotation of the chairs compared to the initial guess created by the SWRL rules.

The results also show varying detection success for the different classes. Shelves have one of the lowest detection rates and highest final pose error in our experiments. This is unsurprising: All shelves in our data set were completely filled with books, so only a small portion of the actual shelf was visible. This also explains why the final pose correction performed worse on shelves compared to the other object classes. In addition, we currently do not handle aggregates of objects: If two shelf segments or two tables are positioned with no gap between them, the planar classification will combine them into one potential object, with the possible location at the center of the combined plane. This usually leads to one false positive (for the non-existent object at the center location) and several false negatives for actual objects creating the aggregated plane. Another problematic object is the big L-shaped desk: The desk is so big that only a small part of it is visible in most single point clouds.

V. Summary and Future Work

We have presented a semantic mapping system that creates a triangle mesh of an office environment, detects several classes of furniture, and replaces them by their corresponding CAD models, based on Kinect point cloud data captured using a mobile robot. The system was evaluated both on two datasets containing 810 single point clouds. Our system achieved a detection rate of 46.0 % for the office dataset, containing one particularly large object and several seamlessly connected instances of shelves and of 79.4 % on the seminar room with less variation of the classified furniture.

In future work, the most needed improvement is a matching criterion that decides how well a CAD model was matched in the point cloud. Such a criterion could be used to reject false hypotheses, and to disambiguate between similar models.

The performance of the final ICP pose correction could be improved by explicitly taking occlusion into account. Instead of trying to match a full CAD model to the point cloud, a pre-assignment filter similar to [7] could take only those CAD model points into account which are expected to have a corresponding point in the sensor data, and vice versa.

Furthermore, we need to improve the generation of the object hypotheses, especially in scenes where we detect planes that are larger than expected due to gap-free placement of several pieces of furniture. We also intend to implement an active data interpretation / gathering loop that identifies and actively explores regions with potentially valuable but missing data. On top of this it could prove worthwhile to incorporate the color information provided by RGB-D cameras when merging planar regions.

References

[1] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard, "An evaluation of the RGB-D SLAM system," in *Proc. ICRA-2012*, St. Paul, Minnesota, 2012.

[2] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: Using kinect-style depth cameras for dense 3D modeling of indoor environments," *IJRR*, vol. 31, no. 5, pp. 647–663, April 2012.

[3] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. W. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," in *Proc. ISMAR*, Basel, Switzerland, October 26-29 2011, pp. 127–136.

[4] M. Günther, T. Wiemann, S. Albrecht, and J. Hertzberg, "Model-based object recognition from 3D laser data," in *Proc. KI-2011*, 2011, pp. 99–110.

[5] S. Albrecht, T. Wiemann, M. Günther, and J. Hertzberg, "Matching CAD object models in semantic mapping," in *Proc. ICRA 2011 workshop: Semantic Perception, Mapping and Exploration, SPME '11*, Shanghai, China, 2011.

[6] A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann, "6D SLAM – 3D mapping outdoor environments," *J. Field Robot.*, vol. 24, no. 8/9, pp. 699–722, 2007.

[7] S. May, R. Koch, R. Scherlipp, and A. Nüchter, "Robust registration of narrow-field-of-view range images," in *Proc. SYROCO '12*, 2012.

[8] J. Biswas and M. Veloso, "Planar polygon extraction and merging from depth images," in *Proc. IROS-2012*, Vilamoura, Portugal, 2012.

[9] R. B. Rusu, Z. C. Marton, N. Blodow, M. E. Dolha, and M. Beetz, "Towards 3D point cloud based object maps for household environments," *Robot. Auton. Syst.*, vol. 56, no. 11, pp. 927–941, 2008.

[10] A. Nüchter and J. Hertzberg, "Towards semantic maps for mobile robots," *Robot. Auton. Syst.*, vol. 56, no. 11, pp. 915–926, 2008.

[11] J. Mason and B. Marthi, "An object-based semantic world model for long-term change detection and semantic querying," in *Proc. IROS-2012*, Vilamoura, Portugal, 2012, pp. 3851–3858.

[12] B. Neumann and R. Möller, "On scene interpretation with description logics," *Image Vision Comput.*, vol. 26, no. 1, pp. 82–101, 2008.

[13] D. Pangercic, B. Pitzer, M. Tenorth, and M. Beetz, "Semantic object maps for robotic housework - representation, acquisition and use," in *Proc. IROS-2012*, Vilamoura, Portugal, 2012, pp. 4644–4651.

[14] J. Majumdar and A. G. Seethalakshmy, "A CAD model based system for object recognition," *J. Intell. Robotics Syst.*, vol. 18, no. 4, pp. 351–365, Apr. 1997.

[15] C. Brenner, J. Böhm, and J. Gühring, "CAD-based object recognition for a sensor/actor measurement robot," *IAPRS, HAKODATE*, vol. 32, pp. 209–216, 1998.

[16] F. Bosché, "Automated recognition of 3D CAD model objects in laser scans and calculation of as-built dimensions for dimensional compliance control in construction," *Adv. Eng. Inform.*, vol. 24, no. 1, pp. 107–118, Jan. 2010.

[17] U. Klank, D. Pangercic, R. B. Rusu, and M. Beetz, "Real-time CAD model matching for mobile manipulation and grasping," in *9th IEEE-RAS Intl. Conf. Humanoid Robots*, Paris, 2009.

[18] O. M. Mozos, Z. C. Marton, and M. Beetz, "Furniture Models Learned from the WWW – Using Web Catalogs to Locate and Categorize Unknown Furniture Pieces in 3D Laser Scans," *Robotics & Automation Magazine*, vol. 18, no. 2, pp. 22–32, June 2011.

TABLE II
DETECTION RATES

	true positives	false positives	false negatives	initial transl. error	initial rotation error	final transl. error	final rotation error	precision	recall	F_1 score
(a) office dataset:										
Shelf	82	53	237	16.2 cm	5.1°	60.1 cm	27.2°	60.7 %	25.7 %	36.1 %
OfficeChair	8	19	6	5.8 cm	61.0°	6.0 cm	10.8°	29.6 %	57.1 %	39.0 %
ConfChair	100	190	114	9.0 cm	51.9°	11.0 cm	56.8°	34.5 %	46.7 %	39.7 %
Desk	10	11	29	31.6 cm	125.6°	53.7 cm	118.8°	47.6 %	25.6 %	33.3 %
ConfTable	81	0	0	8.5 cm	8.8°	9.4 cm	6.2°	100.0 %	100.0 %	100.0 %
total	281	273	386	11.7 cm	28.7°	26.2 cm	34.5°	50.7 %	42.1 %	46.0 %
(b) seminar room dataset:										
Chair	358	26	257	7.3 cm	40.6°	8.3 cm	17.9°	93.2 %	58.2 %	71.7 %
SeminarTable	522	153	21	9.3 cm	4.7°	9.2 cm	3.7°	77.3 %	96.1 %	85.7 %
total	880	179	278	8.5 cm	19.3°	8.8 cm	9.5°	83.1 %	76.0 %	79.4 %

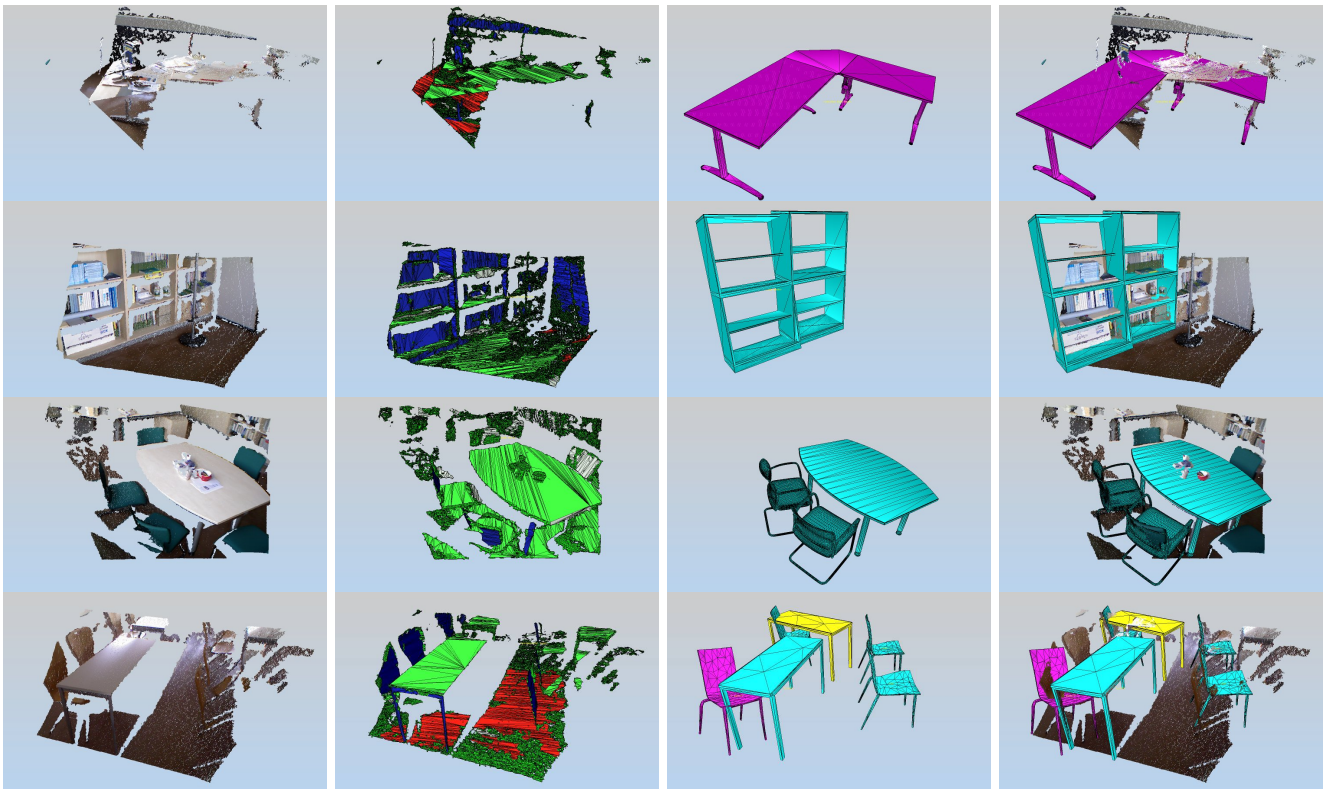

Fig. 5. Results of each step in the processing pipeline for single frames. Columns, from left to right: (a) the original point cloud from a single Kinect frame; (b) the reconstructed triangle mesh; (c) sampled CAD models after ICP pose correction; (d) the point cloud overlaid with the recognized CAD models. Top row: error in pose estimation due to only partial visibility of the desk; next two rows: correct placement of the recognized objects; bottom row: several good matches and one false positive. The color of the models indicates the quality of the recognition: cyan indicates a true positive, where the pose fits well with the actual pose, magenta is a true positive where the final pose is not well aligned and yellow shows a false positive. The bottom row shows data from the seminar room dataset, while the other rows depict data from the office dataset.

[19] W. Wohlkinger, A. Aldoma, R. B. Rusu, and M. Vincze, "3DNet: Large-scale object class recognition from CAD models," in *Proc. ICRA-2012*, St. Paul, Minnesota, 2012, pp. 5384–5391.

[20] K. Lai, L. Bo, X. Ren, and D. Fox, "Detection-based object labeling in 3D scenes," in *Proc. ICRA-2012*, 2012.

[21] T. Wiemann, A. Nüchter, K. Lingemann, S. Stiene, and J. Hertzberg, "Automatic construction of polygonal maps from point cloud data," in *Proc. 8th IEEE Intl. Workshop Safety, Security, and Rescue Robotics (SSRR-2010)*, Bremen, 2010.

[22] T. Wiemann, K. Lingemann, A. Nüchter, and J. Hertzberg, "A toolkit for automatic generation of polygonal maps – Las Vegas Reconstruction," in *Proc. ROBOTIK-12*, 2012, pp. 446–415.

[23] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," *Comp. Graph.*, vol. 26, no. 2, pp. 71–78, 1992.

[24] E. de Koning, "Polyline Simplification Library (PSIMPL)," 2010, http://psimpl.sourceforge.net/.