Matching CAD Object Models in Semantic Mapping

Sven Albrecht, Thomas Wiemann, Martin Günther, Joachim Hertzberg

Abstract— **In this paper we present an approach to introduce semantics into a SLAM-generated 3D point cloud map from 3D laser scans of an office environment. For semantic classification we propose to first reconstruct surface planes in the point cloud. Using an OWL-DL ontology, we automatically analyze relations between surface planes in a cluster to generate hypotheses for present objects. To check these hypotheses we surface sample appropriate CAD models and use standard ICP to fit the scan data with the model of the hypothesized object. The final result is a hybrid semantic map, in which all identified objects have been replaced by their corresponding CAD models.**

I. Introduction

Robotic mapping techniques developed tremendously over recent years. Map accuracy and resolution have improved, and fast and reliable algorithms for solving the SLAM problem in 6D are available [1], [2], [3]. These methods focus on solving the problem of integrating the collected data, typically 3D laser scans, into a global coordinate system. Such maps are very successful for robot navigation, but for more complex tasks like context-based action planning a more meaningful semantic scene interpretation is needed.

A semantic map goes beyond pure geometrical information by labeling objects found in the map as instances of specific classes, thus linking them with semantic background knowledge. This enables the robot to reason about its environment, e. g., room types (a room with a dishwasher is probably a kitchen) or the function of objects (a kitchen table has other functions than a conference table). Possible applications of this kind of semantic information include human-robot interaction and goal-directed manipulation of the environment, for example in service or rescue robots.

The challenge that arises is to automatically extract this inherent information from the raw 3D point cloud data. Our goal is to produce what we call "hybrid semantic maps" – i. e., an environmental representation where instances of known objects are replaced with precomputed polygonal models in the original point clouds. Such maps provide several benefits for robotic applications: Polygonal representations are much more compact than point clouds but still contain full geometric information. The replacement of original point cloud data with precomputed models is feasible, since CAD models of virtually every kind of object are nowadays obtainable, e. g., from manufacturers, facility management or from the Internet (Google's 3D Warehouse, IKEA etc.).

After such models have been instantiated, the hybrid map contains more valuable information than the map from the original scan data. Besides for the applications mentioned above, the additional knowledge about the real geometry of the identified objects can be used to improve the geometric quality of the auto-generated 3D maps. Possible applications include filling up missing sensor data or using the pose information of the identified objects for loop closing.

This paper describes a novel approach to automatically generate hybrid semantic maps of indoor environments from 3D point clouds. The goal is to achieve a top-down segmentation of the scanned scene: First, large connected planar areas are detected and classified in terms of walls, floors and ceilings. To the remaining planes, domain knowledge about indoor environments is applied to detect possible positions of furniture. In a third step, the detected furniture is replaced with CAD models.

II. State of the Art

State of the art in robotic mapping is the use of 2D grid maps for self-localization. These maps are constructed based on 2D or 3D laser scans. To produce consistent maps, the SLAM problem has to be solved. For 2D SLAM, probabilistic methods like FastSLAM [4], GraphSLAM [5] or EM [6] are state of the art. In the case of 3D point clouds, the SLAM problem is commonly solved by 6D scan matching [1], [3], [7]. To further refine the registration quality of the gained point clouds, a 3D version of Lu and Milios's technique [2] can be applied as a post-processing step [8]. All these approaches produce accurate metrical maps, but do not encode information about the types of objects that are present in the map.

On the other side of the spectrum, there are approaches for anchoring semantic information, provided by a conceptual hierarchy, in spatial maps [9], [10]. However the focus of these approaches is on place labeling rather than object recognition and accurate 3D pose estimation, and the environment is represented by a 2D map. A complete semantic mapping framework based on 3D laser data is presented in [11]; in contrast to our work, the model fitting part uses cuboids of variable dimensions instead of CAD models. The work in [12] uses CAD models to train an object detection system used to label objects in urban 3D laser scans. A robot manipulation system that integrates knowledge processing mechanisms with semantic perception routines, including CAD model matching, is presented in [13].

An idea similar in spirit to our approach is proposed in [14] where matching of CAD models is utilized to allow for manipulation tasks such as grasping in a household environment. Contrary to our approach, the matching itself is performed within 2D image data, instead of the 3D environment representation.

Our procedure is able to detect instances of available CAD models of furniture using ICP matching in 3D point clouds.

Initial poses for these objects are estimated by applying domain knowledge from a given ontology. With our work we further close the gap between the 3D mapping techniques preserving high details of the environment and the potentially rich context information available in semantic maps.

III. CAD Model Anchoring

As the initial situation for our approach we assume a successfully registered consistent 3D point cloud, obtained as for example described in [1]. Our procedure to instantiate CAD models in registered 3D scenes can be summarized as follows: First, surface reconstruction is applied to the point cloud (see Sec. III-A), which helps to find flat surfaces in the environment. Next, initial poses for possible locations of models in our ontology are estimated by analyzing the geometrical properties of the labeled planes that were generated in the first step (III-B). These pose estimations are corrected using ICP by subsampling a CAD model of the assumed object (III-C), and fitting it into its region (III-D).

A. Environment Surface Reconstruction

Surface reconstruction from point cloud data is a complex problem. The most common approach is to create a polygonal mesh representation using Marching Cubes based methods [15] or to fit mathematically defined surfaces like planes or splines to the given data [16]. Mesh-based approximations are suitable to hold high-quality representations of arbitrary surfaces, but automatically generated models commonly show more surface patches than needed to describe the scanned objects. Hence additional optimization procedures have to be applied [17], [18].

Fitting planes or splines to a given set of data points is mathematically easy using least squares methods, resulting in compact representations. The main problem is to determine the most suitable surface description for a given set of points. Using the Hough transformation is a possible approach to solve this problem [19], but it is computationally expensive compared to mesh-based methods.

For our purposes we decided to use an optimized Marching Cubes implementation that uses a region-growing algorithm to extract connected planar surfaces in the model and to represent them as polygons [18]. The outcome of our procedure is a three-dimensional polygon mesh, where connected planar regions are represented as single polygons. Fig. 1 shows an example. Once such a surface representation has been computed, it is possible to classify the extracted polygons by analyzing their orientations and locations towards each other [20], [18].

B. Initial Pose Estimation

Semantic knowledge about identified objects is stored using an OWL-DL ontology in combination with SWRL rules (Fig. 2). Description logics were chosen as the knowledge representation format, because DL ontologies can be easily re-used and linked with other sources of domain knowledge, and fast reasoning support is available.

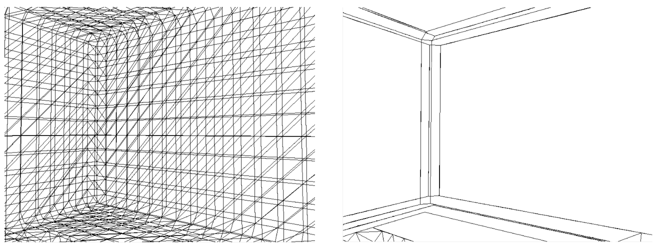


Fig. 1: An example of our polygonalization algorithm. The left image (a) shows the output of the marching cubes implementation: A room corner is represented by a mesh of nearly coplanar or orthogonal triangles. The right image (b) shows the same scene after plane detection.

An OWL-DL reasoner is used to generate hypotheses of possible object locations and initial pose estimation, based on the planes extracted in the previous section. Each plane is added as an individual to the ontology, along with its orientation (horizontal/vertical, based on the normal), its height above ground (based on the centroid), its bounding box and its area.

The definitions of furniture classes in the ontology contain a set of conditions that are used to classify the planes into possible furniture instances. For example, most standard desks have a height of approximately 70 cm. So all horizontal planes that have about this height are valid candidates for table tops. To distinguish different types of tables, the surface area is used. Similar considerations apply to office chairs: A chair has a ground parallel plane to sit on (at a height of around 40 cm) and another perpendicular plane near it (the backrest). Fig. 3 presents, as an example, the ontology representation of a shelf.

For each object instance returned by the OWL-DL reasoner, we calculate axis-parallel bounding boxes and center points of the constituting planes. The center point of one predefined plane (e. g., the table top) is used to anchor the position. Information about the orientation depends on the geometry of the expected models. The intrinsic orientation has to be identified and encoded according to the model class. For some objects this orientation is easy to identify, e. g., chairs where the normal of the back rest defines the orientation of the whole object. For other objects like tables, we apply a Principal Component Analysis (PCA) to the points that lie within the plane that defines the intrinsic orientation. This method delivers two new orthonormal basis vectors that approximate the orientation within the global coordinate system. For successful matching, all used models have to be pre-processed to be in a center-point-based local coordinate system that reflects the assumptions described above.

C. Model Surface Sampling

The ICP algorithm [21] used for Model Fitting (see Sec. III-D) needs to establish point-to-point correspondences

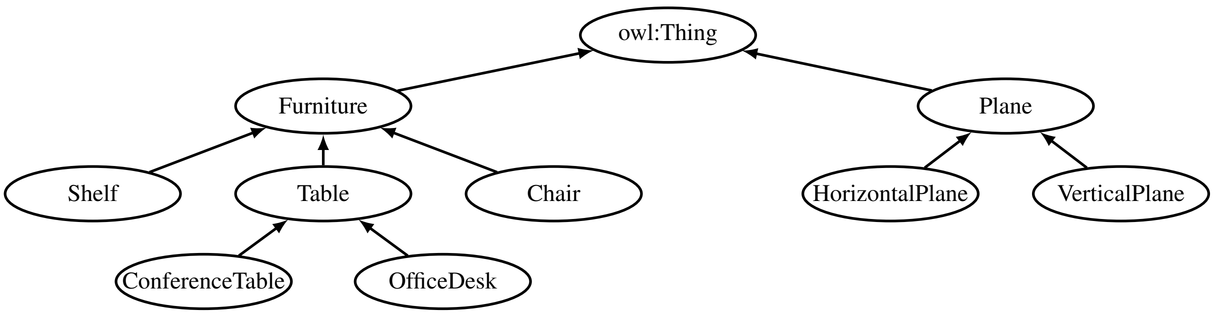


Fig. 2: The relevant parts of the ontology's class hierarchy.

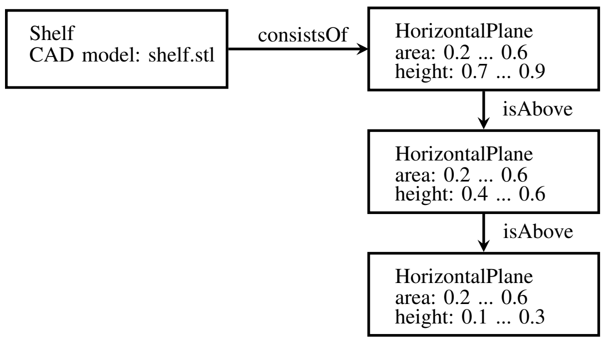


Fig. 3: A fragment of the ontology used for initial pose estimation to represent a shelf.

between given sets of data and model points. In our scenario the data points are given by a registered 3D point cloud, obtained from several laser scans. Our model data is provided as a CAD model. The correspondences used in ICP are defined via point distances, i. e., for each point in the data set the closest point on the surface of the model is required. Although it is possible to calculate the correspondences between the data set and the polygonal CAD models exactly using ray tracing, for practical reasons, we decided to sample the surfaces of the models. The main benefit is that existing point cloud matching software can be used for fitting. Another advantage is that we can re-use these precomputed representations, including internal data structures like kd trees that are needed for efficient matching, for all instances of the model. The kd tree allows for efficient nearest neighbor search for a data point and needs to be built only once when starting the ICP algorithm. Another benefit of this approach is that even for very complex 3D models, the registration process can be reduced to standard ICP since the problem is reduced to registering two rigid 3D point clouds.

The downside of the sampling is that instead of the actually closest point of the model, we only get an approximation by the surface sampling. Dense sampling may result in a better approximation of the underlying CAD model, but will result in a more expensive kd tree nearest neighbor search. On the other hand a coarse sampling will result in only a rough approximation of the CAD model and therefore negatively affect the quality of the ICP registration process. Using a sampling with a point density slightly higher the data point density of the registered point cloud proved to deliver acceptable results.

For creating a surface sampling of a given CAD model several approaches are possible, for example random sampling. To achieve a point distribution in the sampling that emulates the scan point density we use another approach. In the standard CAD exchange formats, models are commonly defined as a set of triangular faces. The task of surface sampling is essentially the task to sample a triangle in three-dimensional euclidean space. For a given desired sample point distance, a triangle is sampled in the following fashion.

We assume the triangle to feature three sides of different length; for isosceles or equilateral triangles we may choose sides arbitrarily. We label the vectors pointing to the vertices of the triangle so that $\mathbf{a}$ adjoins the shortest with the longest side and $\mathbf{b}$ the longest with the medium side. The remaining vertex is labeled $\mathbf{c}$ (see Fig. 4). To sample the surface of the triangle we use two generator vectors $\mathbf{g}_1$ and $\mathbf{g}_2$ which are defined as

$$\mathbf{g}_1 = \frac{\mathbf{b} - \mathbf{a}}{\|\mathbf{b} - \mathbf{a}\|} d \tag{1}$$

$$\mathbf{g}_2 = \frac{\mathbf{c} - \mathbf{a}}{\|\mathbf{c} - \mathbf{a}\|} d \tag{2}$$

where d defines the desired maximal point distance for the sampling. These vectors point from $\mathbf{a}$ to $\mathbf{b}$ and from $\mathbf{a}$ to $\mathbf{c}$, respectively, and have a length of d. Employing $\mathbf{g}_1$ and $\mathbf{g}_2$, the triangle is sampled as described in Algorithm 1; Fig. 4 depicts an exemplary result of this sampling algorithm.

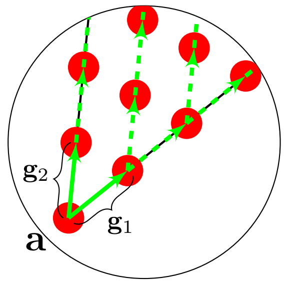
D. *Model Fitting*

Once an initial pose is automatically estimated for a specific instance of an object and a model surface sampling is constructed, the matching process can commence. This is done by placing the surface sampling of the corresponding CAD model at the pose estimate and subsequently running ICP on the sampled model and the registered scene. For sake of efficiency we only use points that are within the region of the initial pose estimate. Since the model points are rotated and translated during ICP we extend the considered region within the point cloud to a larger area than the bounding box of the model instance.

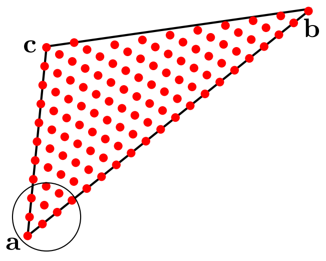
One major problem is how to evaluate the final pose provided by ICP. While ICP guarantees to converge in a local minimum, this does not mean that a sampled CAD

Algorithm 1: Triangle Sampling Algorithm

```
input  : a triangle $T$ and a desired sample distance $d$
output: a point sample $\mathcal{P}$ of triangle $T$
1  $\mathcal{P} \leftarrow \emptyset$
2  $\{\mathbf{a}, \mathbf{b}, \mathbf{c}\} \leftarrow$ order_sides($T$)
3  $\mathbf{g}_1 \leftarrow$ normalize_vec $(\mathbf{b} - \mathbf{a}) * d$  // see Eq. (1)
4  $\mathbf{g}_2 \leftarrow$ normalize_vec $(\mathbf{c} - \mathbf{a}) * d$  // see Eq. (2)
5  $\mathbf{p} \leftarrow \mathbf{a}$
6  while $\mathbf{p}$ inside_of $(T)$ do
7      $\mathbf{s} \leftarrow \mathbf{p}$
8      while $\mathbf{s}$ inside_of $(T)$ do
9          $\mathcal{P} \leftarrow \mathcal{P} \cup \mathbf{s}$
10         $\mathbf{s} \leftarrow \mathbf{s} + \mathbf{g}_2$
11     end
12     $\mathbf{p} \leftarrow \mathbf{p} + \mathbf{g}_1$
13 end
```



(a)



(b)

Fig. 4: Exemplary sampling of a triangle. Sample points are marked by red dots. On the left (a) the solid green arrows visualize the initial generator vectors $\mathbf{g}_1$ and $\mathbf{g}_2$ used to create sample points, starting from vector $\mathbf{a}$, while the dashed arrows indicate the further process of the sampling (see Algorithm 1). The right-hand image (b) depicts the completely sampled triangle.

model is registered correctly into a scene or that an instance of the CAD model is present in the scene at all. While we do not claim to have solved this problem, we developed a heuristic to determine the quality of the final pose: During ICP, correspondences between data and model points are established by searching for the nearest model point for any given data point, which is nearer than some predefined threshold. For our heuristic we change the perspective: We look at how closely the data resembles the original model. To this end we partition our model points into cubes, thus gaining a more coarse resolution of the model. Then we check for how many of these cubes corresponding data points (within the maximal defined distance) can be found. If the number of associated data points is below a threshold, we assume that this part of the model was not present in the data points. Afterwards we look at the ratio of cubes not present in the data to the number of total cubes for the model. If this ratio is satisfactory, our heuristic assumes the model to be correctly matched. This involves some parameter tuning and is one point of future work.

TABLE I: Estimated orientations for two table models via PCA compared to ground truth.

Ground Truth	12.0°	25.0°	55.0°	90.0°	125.0°	160.0°
Conference Table	8.2°	22.1°	51.4°	86.0°	121.0°	157.0°
Office Desk	4.0°	28.1°	46.7°	82.0°	118.0°	153.0°

IV. Experimental Results

In this section we will present an example for automatically generating a hybrid semantic map: The match of two office table CAD models into the scanned point cloud. The first part will apply and evaluate the techniques for initial pose estimation described above. The second part presents the resulting hybrid map.

A. *Initial pose estimation*

The input to our reasoner is a set of planes that were extracted for the input data. Fig. 5 shows the input point set for our experiments together with a surface reconstruction. The found planar structures in the scene are rendered in a red to blue gradient, all other surfaces are green. The basic characteristics of these patches (centroid, normal, bounding box, area) are used by the reasoner to identify possibly present models.

As one can see, the large connected surfaces on the floor are recognized, as well as smaller structures like the tabletops (gray) or the backrests of the chairs around the conference table (red, blue, light green). After feeding the extracted planes into the reasoner, two possible present objects were detected: The conference table on the right and the desk on the left. The main remaining problem is to determine the model's orientation. To solve this, we use a PCA implementation by Martagh [22] on the vertices of the table top reconstruction. To analyze the stability of this approach, we rotated a reference model of the conference table to different predefined angles to get ground truth and compared the original rotation angles with the PCA estimation. The results are shown in Table I. Although the estimated poses derived from bounding box show several degrees difference from ground truth, ICP is able to correct these estimations as shown in the next section.

B. *Model Replacement*

After the object hypotheses and pose estimations are generated, we subsample the corresponding CAD models as described in Sec. III-C. Fig. 6 shows an example sampling. This synthetic point cloud is then used to refine the initial pose estimation. Fig. 7 shows the results of the matching process for the tables that were detected in the given scene. The pictures on the left clearly show that the ICP process significantly improves the estimated poses. For the conference table we get an almost perfect fit. The fit for the office desk is not as good as the one for the conference table. Here we have an offset to the right of about two centimeters. This is due to registration errors in the used point cloud and differences

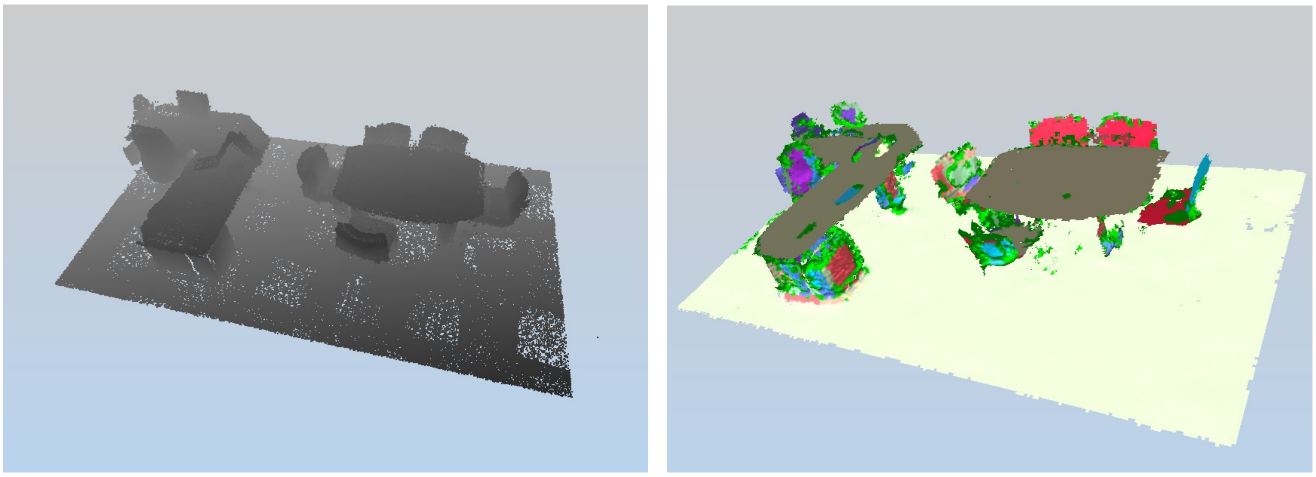

(a) (b)

Fig. 5: The used input point cloud (a) and the automatically extracted planes (b). The detected planes are colored in a red to blue gradient based on a running number. All surfaces that were not classified as belonging to a plane are rendered in green. Walls and ceiling in the original data set were manually removed to create a suitable perspective.

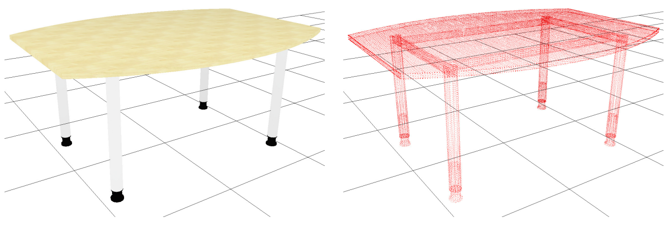

(a) (b)

Fig. 6: Exemplary sampling of a CAD Model: On the left (a) a CAD of one of our conference tables is shown, (b) depicts the result of our sampling algorithm. CAD model courtesy of Assmann Büromöbel GmbH & Co. KG

between the CAD model and the real world object. The real object shows clearances that are not considered in the model.

V. Conclusion and Future Work

This paper has sketched a new approach to anchor object instances of known classes in point cloud data using domain knowledge to generate hybrid semantic maps. The usability of our method was demonstrated in a practical example where instances of two different tables were found in a given point and replaced with CAD models. The results achieved until now and presented in this paper reveal some further questions. We are currently working on extending our domain-knowledge representation. Combining the ontology with first-order probabilistic reasoning promises to be more robust towards inaccurate or incomplete sensor data.

A second point to improve is a better automated evaluation of the final pose of the model fitting. Currently the only automatically generated measure of quality is the output of the ICP algorithm, i. e., the average point-to-point distance between the data and the model set. Clearly this error function does not solely depend on the final estimated pose, but also on other factors like the density of the surface sampling. Imagine the same 3D CAD model sampled with different densities and placed in the same 6D pose into a scene. The model data set with the higher point density will generally produce a lower point-to-point distance error – if the desired sampling distance is set accordingly, one can ensure that the coarsely sampled model points are a subset of the dense surface sampling. Although ICP will inherently terminate in a local minimum, this is not enough to ensure that the final pose generated by ICP does in fact correspond to the correct one for a given object in a scene. Therefore a different measure, which will yield information about the quality of the final pose, independent of the density of the surface sampling, is highly desirable.

Another direction of research is to create a feedback loop from the gained knowledge into the robot's behavior, for instance, to actively collect data based on domain knowledge. A conference table, for example, usually has several chairs around it. So it would be reasonable for a robot to search for more, if it has already detected the table and one single chair, to collect the expected information.

References

[1] D. Borrmann, J. Elseberg, K. Lingemann, A. Nüchter, and J. Hertzberg, "Globally consistent 3D mapping with scan matching," *Robot. Auton. Syst.*, vol. 65, no. 2, pp. 130–142, 2008.

[2] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *Auton. Robot.*, vol. 4, pp. 333–349, April 1997.

[3] M. Magnusson, T. Duckett, and A. J. Lilienthal, "Scan registration for autonomous mining vehicles using 3D-NDT," *J. Field Robot.*, vol. 24, no. 10, pp. 803–827, October 2007.

[4] D. Hähnel, D. Fox, W. Burgard, and S. Thrun, "A highly efficient FastSLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements," in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Las Vegas, USA, 2003.

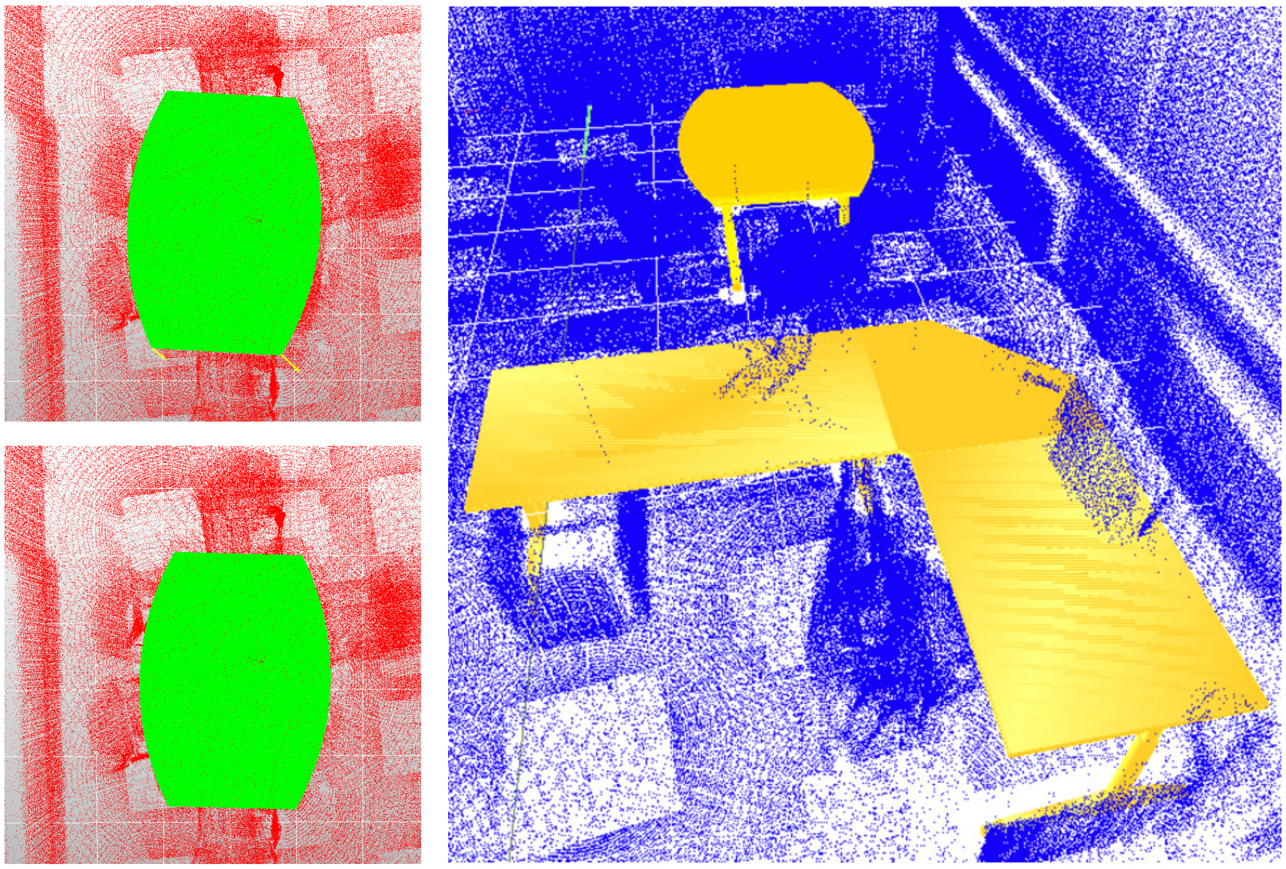


Fig. 7: Results of the ICP model matching process for the two table models. The left column shows the pose of the conference table before and after matching. The offset of the initial pose estimation from the final pose is indicated by the yellow arrows. The picture on the right shows the CAD models of both detected tables rendered in the original point cloud.

[5] S. Thrun and M. Montemerlo, "The GraphSLAM algorithm with applications to large-scale mapping of urban structures," *Int. J. Robot. Res.*, vol. 25, no. 5/6, pp. 403–430, 2005.

[6] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, September 2005.

[7] A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann, "6D SLAM – 3D mapping outdoor environments," *J. Field Robot., Special Issue on Quantitative Performance Evaluation of Robotic and Intelligent Systems*, vol. 24, no. 8/9, pp. 699–722, August/September 2007.

[8] J. Sprickerhof, A. Nüchter, K. Lingemann, and J. Hertzberg, "An explicit loop closing technique for 6D Slam," in *Proc. European Conf. on Mobile Robotics (ECMR '09)*, September 2009, pp. 229–234.

[9] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J. Fernández-Madrigal, and J. González, "Multi-hierarchical semantic maps for mobile robotics," in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Edmonton, CA, 2005, pp. 3492–3497.

[10] H. Zender, Ó. M. Mozos, P. Jensfelt, G.-J. M. Kruijff, and W. Burgard, "Conceptual spatial representations for indoor mobile robots," *Robot. Auton. Syst.*, vol. 56, no. 6, pp. 493–502, 2008.

[11] 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., Special Issue on Semantic Knowledge in Robotics*, vol. 56, no. 11, pp. 927–941, 2008.

[12] K. Lai and D. Fox, "Object recognition in 3D point clouds using web data and domain adaptation," *Int. J. Robot. Res.*, vol. 29, no. 8, pp. 1019–1037, July 2010.

[13] D. Pangercic, M. Tenorth, D. Jain, and M. Beetz, "Combining perception and knowledge processing for everyday manipulation," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, Taipei, Taiwan, October 18-22 2010.

[14] 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. on Humanoid Robots*, Paris, France, December 7-10 2009.

[15] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," in *Proc. ACM SIGGRAPH*, 1987.

[16] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Computing and rendering point set surfaces," *IEEE T. Vis. Comput. Gr.*, vol. 9, no. 1, pp. 3–15, January 2003.

[17] 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.

[18] 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 on Safety, Security, and Rescue Robotics (SSRR-2010)*, Bremen, Germany, July 2010.

[19] D. Borrmann, J. Elseberg, and A. Nüchter, "A data structure for the 3D hough transform for plane detection," in *Proc. 7th IFAC Symposium on Intelligent Autonomous Vehicles (IAV 2010)*, Lecce, Italy, September 2010.

[20] A. Nüchter and J. Hertzberg, "Towards semantic maps for mobile robots," *Robot. Auton. Syst., Special Issue on Semantic Knowledge in Robotics*, vol. 56, no. 11, pp. 915–926, 2008.

[21] P. Besl and N. McKay, "A method for registration of 3–D shapes," *IEEE T. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, February 1992.

[22] F. Martagh and A. Heck, *Multivariate Data Analysis*. Kluwer Academic, 1987.