Scene Perception and Recognition in industrial environments for Human-Robot Interaction

Nikhil Somani[†], Emmanuel Dean-León[‡], Caixia Cai[†] and Alois Knoll[†] [⋆]

Authors Affiliation: †Technische Universität München, Fakultät für Informatik.
‡Cyber-Physical Systems, fortiss - An-Institut der Technischen Universität München
Address: †Boltzmannstrae 3, 85748 Garching bei München (Germany). ‡Guerickestr. 25 80805 München (Germany)
Email: †{somani,caica,knoll}@in.tum.de, ‡{dean}@fortiss.org

Abstract. In this paper, a scene perception and recognition module aimed at use in typical industrial scenarios is presented. The major contribution of this work lies in a 3D object detection, recognition and pose estimation module, which can be trained using CAD models and works for noisy data, partial views and in cluttered scenes. This algorithm was qualitatively and quantitatively compared with other state-of-art algorithms. Scene perception and recognition is an important aspect in the design of intelligent robotic systems which can adapt to unstructured and rapidly changing environments. This work has been used and evaluated in several experiments and demonstration scenarios for autonomous process plan execution, human-robot interaction and co-operation.

1 Introduction

Scene perception and recognition, in the very general sense of the term, is the process of gathering information about the environment using sensors and processing this data to generate information which is useful in carrying out some task or process. The perception problem in the industrial robotics context involves detecting and recognizing various objects and actors in the scene. The objects in the scene consist of workpieces relevant to the task and obstacles. The actors involved are humans, and the robot itself. The major contribution of this work is an object detection, recognition and pose estimation module, which uses 3D point cloud data obtained from low-cost depth sensors like the Kinect.

Industrial robotics, which was hitherto mostly used in structured environments, is currently witnessing a phase where a lot of effort is directed towards applications of standard industrial robots in scenarios that are rather unstructured and rapidly changing. Hence, the scene perception and recognition module

[⋆] The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n 287787 in the project SMErobotics, the European Robotics Initiative for Strengthening the Competitiveness of SMEs in Manufacturing by integrating aspects of cognitive systems.

has now become an important component in intelligent robotic systems. This module provides information about the working environment which is used by reasoning modules and intelligent control algorithms to create an adaptive system. In these systems, the process plans are often written at a semantic level which is abstracted from the execution and scenario specific information. The perception module is the key for bridging this gap. On one hand, the perception module provides information which is used by the reasoning engines to provide an abstraction of the world and learn tasks at this abstract level by human demonstration. On the other hand, the perception module provides scenario specific information which is used by the low-level execution and control modules for plan execution.

Object detection, recognition and pose estimation using 3D point clouds is a well researched topic. The popular approaches for this task can be broadly classified as: local color keypoint [1], [2], local shape keypoint [3], global descriptors [4], [5], geometric [6], primitive shape graph [7], [8]. Each of these approaches have their own advantages and disadvantages. For example, color based methods would not work on texture-free objects. Shape based methods can not distinguish between objects having identical shape but different texture. Global descriptors such as VFH [4] require a tedious training phase where all required object views need to be generated using a pan-tilt unit. Besides, its performance decreases in case of occlusions and partial views. The advantage of these methods, however, lies in their computational speed. Some other methods such as [7], [9], [10] provide robustness to occlusions, partial views and noisy data. However, these methods are rather slow and not suitable for real-time applications in large scenes. In this paper, an extension to the Object Recognition RANSAC (ORR) [9], [10] method has been proposed, where the effort has been directed towards a solution which enhances its robustness to noisy sensor data and also increases its speed. Another object recognition and pose estimation algorithm has been proposed, which is complementary to the PSORR method with respect to the target object geometries.

To distinguish objects having identical geometry but different color, the point cloud is segmented using color information and then used for object detection. There are several popular approaches for Point cloud segmentation such as Conditional Euclidean Clustering [11], Region Growing [12], and graph-cuts based segmentation methods [13], [14], [15], [16]. In this paper, a combination of multi-label graph-cuts based optimization [16] and Conditional Euclidean Clustering [11] is used for color-based segmentation of point clouds.

2 Object Recognition and Pose Estimation

2.1 Shape Based Object Recognition from CAD models

There are two complementary approaches presented here. One is an extension of the ORR method [9], [10] called Primitive Shape Object Recognition Ransac (PSORR), and the other is based on Primitive Shape Graph (PSG) matching. The results obtained are qualitatively similar for both approaches. The PSORR

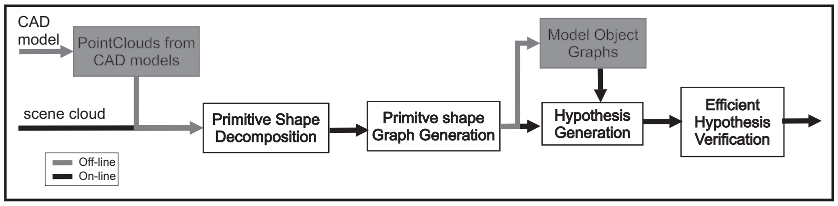
CAD model
PointClouds from CAD models
scene cloud
Primitive Shape Decomposition
Primitve shape Graph Generation
Model Object Graphs
Hypothesis Generation
Efficient Hypothesis Verification
Off-line
On-line

Fig. 1. Pipeline for Shape based perception.

method is more suitable for handling arbitrary object geometries and objects having few primitive shapes while the PSG method is more suitable for large models which decompose into a large number of stable primitive shapes. The pipeline for this module is shown in Fig. 1.

2.1.1 Primitive Shape Decomposition

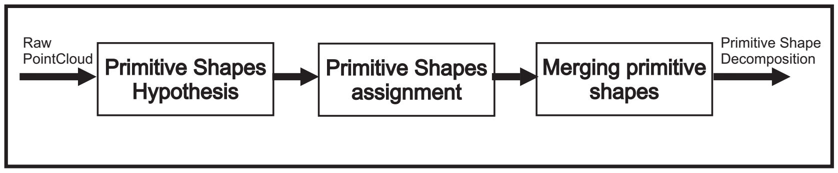
Raw PointCloud
Primitive Shapes Hypothesis
Primitive Shapes assignment
Merging primitive shapes
Primitive Shape Decomposition

Fig. 2. Pipeline for Primitive Shape Decomposition.

The pipeline for this step is shown in Fig. 2. This step is very important for the algorithm because the hypothesis generation and pose estimation step are based on this decomposition. The hypothesis verification step, which is a major bottleneck in most algorithms such as ORR, can also be significantly simplified and sped-up using this decomposition.

The point cloud P is represented as a set of primitive shapes s_i containing points $p_i \subseteq P$ such that $\cup p_i \subseteq P$. The primitive shapes s_i could be planes, cylinders, etc. An example of such a decomposition is shown in Fig. 3, where the original scene cloud is shown in Fig. 3 (a) and its decomposition into primitive shapes is shown in Fig. 3 (b).

Primitive Shape Hypothesis

Hypothesis for primitive shapes are generated by randomly sampling points in the point cloud. Once the hypotheses have been generated, each point in the cloud is checked to determine whether it satisfies the hypotheses. The method used for generating a hypothesis and determining its inliers depends on the type of primitive shape.

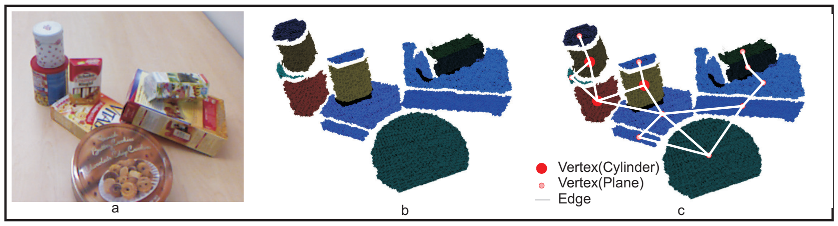

Fig. 3. Primitive Shape Decomposition example : (a) original Point Cloud (b) result of Primitive Shape Decomposition (c) Primitive Shape Graph representation.

- **Planes:** A plane hypothesis can be generated using a single point (X_0) with its normal direction $(\hat{n})$. To test if a point X lies on the plane $(X - X_0).\hat{n} = 0$, the distance of the point from the plane $|(X - X_0).\hat{n}|$ is used.
- **Cylinders:** A cylinder hypothesis can be generated using 2 points (X_0, X_1) with their normal directions $(\hat{n_0}, \hat{n_1})$. The principal axis of the cylinder is selected as the minimum distance line between the normal directions $\hat{n_0}$ and $\hat{n_1}$. The radius r is the distance of either point to this line. To test if a point X lies on the cylinder, the distance of the point from the cylinder's axis is used.

Primitive Shape Assignment

The hypotheses associated with each point in the cloud can be considered as labels for point. There may be multiple labels associated with each point and the labeling may be spatially incoherent. To resolve such issues and generate a smooth labeling, a multi-label optimization using graph-cuts is performed. In this setting, the nodes in the graph comprise all possible assignment of labels to the points. The data term indicating the likelihood of a label assignment to a point is inversely proportional to the distance of the point from the primitive shape. The smoothness term penalizes neighboring points having different labels and the penalty is inversely proportional to the distance between the neighboring vertices. Label swap energies are used for neighboring primitive shapes in a way that only neighboring primitive shapes labels can be swapped. This convex energy functional is then solved using the α - expansion, β -swap algorithms [13], [14], [15], [16] which give the label assignment for each point in the cloud, such that the total energy is minimized.

Merging Primitive Shapes

Each primitive shape has a $fitness\_score$ associated with it which indicates how well the primitive matches the point clouds. It is based on the minimum descriptor length(MDL) approach [17]. The fitness score of a primitive shape is defined as :

$$fitness\_score = \frac{inliers}{total\_points} + K * descriptor\_length \tag{1}$$

where, the first fraction represents the inlier ratio, i.e., the ratio of points which satisfy the primitive shape (*inliers*) to the total number of points in the input cloud (*total_points*), *descriptor_length* represents the complexity of the primitive shape (e.g. the number of values required to represent the shape). The constant K determines the relative weighting of the two factors. Higher values of K will support under-segmentation resulting in bigger, less accurate primitives, while low values will hamper robustness against over-segmentation, causing fewer merges and resulting in fragmented, over-fitted primitives.

The merging strategy is based on a greedy approach where pairs of primitive shapes are selected and merged if the combined primitive shape has a better fitness score than the individual primitive shapes. This continues till there are no more primitive shapes which can be merged.

2.1.2 Primitive Shape Graph(PSG) Representation

The primitive shapes detected in the previous step are now used to create a graphical representation of the point cloud. In this graph $G = (V, E)$, each primitive shape is a node $v \in V$ and neighboring primitive shapes are connected by an edge $e \in E$. An example of such a graph is shown in Fig. 3 (c).

2.1.3 Hypothesis Generation

PSORR method

An oriented point pair (u, v) contains two points along with their normal directions: $u = (p_u, n_u)$ and $v = (p_v, n_v)$. A feature vector $f(u, v)$ is computed from this point pair, as shown in Eq. 2.

$$f(u,v) = \begin{pmatrix} \|p_u - p_v\| \\ \angle\,(n_u, n_v) \\ \angle\,(n_u, p_v - p_u) \\ \angle\,(n_v, p_u - p_v) \end{pmatrix}, \tag{2}$$

The central idea in the ORR method is to obtain such oriented point pairs from both the scene and model point clouds and match them using their feature vectors. For efficient matching of oriented point pairs, a Hash Table is generated containing the feature vectors from the model point cloud. The keys for this table are the three angles in Eq. 2. Each Hash Cell contains a list of models ($M_i \in M$) and the associated feature vectors. Given an oriented point pair in the scene cloud, this Hash Table is used to find matching point pairs in the model cloud. Each feature vector f has an associate homogeneous transformation matrix F associated with it, see Eq. 3.

$$F_{uv} = \begin{pmatrix} \frac{p_{uv} \times n_{uv}}{\|p_{uv} \times n_{uv}\|} & \frac{p_{uv}}{\|p_{uv}\|} & \frac{p_{uv} \times n_{uv} \times p_{uv}}{\|p_{uv} \times n_{uv} \times p_{uv}\|} & \frac{p_u + p_v}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{3}$$

where $p_{uv} = p_v - p_u$ and $n_{uv} = n_u + n_v$. Hence, for each match f_{wx} in the hash table corresponding to f_{uv} in the scene, a transformation estimate can be obtained, see Eq. 4. This transformation estimate (T_i) forms a hypothesis $h_i = \{T_i, M_i\} \in H$ for the model (M_i) in the scene.

$$T = F_{wx} F_{uv}^{-1} \tag{4}$$

The raw point clouds are generally noisy, especially the normal directions. The original ORR method is sensitive to noise in the normal directions and hence, randomly selecting points to generate the feature vectors requires more hypothesis until a good oriented point pair is found. In the PSORR method, every node representing a plane in the scene PSG is considered as an oriented point (u) with the centroid of the plane as the point (p_u) and the normal direction as the orientation (n_u). The normal directions for these oriented points are very stable because they are computed considering hundreds of points lying on the plane. Therefore, we can use these centroids instead of the whole cloud to compute and match features, which leads to a significantly less number of hypotheses.

The centroid for the scene cloud primitives might not match the model centroids in case of partial views. Hence, for the model cloud, the point pairs are generated by randomly sampling points from every pair of distinct primitive shape clouds.

PSG Matching for hypothesis generation

In cases where the PSG is rather large and the individual primitive shapes are small, the speedups obtained by the PSORR method are not significant due to the additional cost of primitive shape decomposition. In this case, another approach is used where the scene PSG is matched with model PSG's and used to recognize the object and estimate its pose. Given both model and scene PSG's, the problem of object recognition becomes equivalent to constrained sub-graph matching, which is an NP-complete problem. However, the nature of the constraints on these graphs provide good heuristic solutions.

Some special cliques in this graph are minimal representations for object pose estimation, e.g. a clique of 3 intersecting planes, or a plane intersecting with a cylinder. A feature vector is computed for each of these cliques which can be used for matching. For a clique of 3 planes, the angles between the pairs of planes constitutes the feature vector. For a plane and cylinder intersection clique, the cylinder radius along with the angle between the plane normal and the cylinder axis direction constitutes the feature vector.

The clique matches between the scene and model point clouds generates full hypotheses $h_i \in H$, i.e., it gives the model (M_i) as well as the pose (T_i). Each of these hypotheses gives a set of partial matches for the scene and model graph vertices. Since they are full hypotheses, a fitness score can be computed for each of them which indicates the accuracy of the hypothesis.

The graph matching problem is identical to a vertex labeling problem. For each vertex V_s in the scene graph G_s, a match with a vertex V_m in the model graph G_m can be considered as a label. Hence, this problem can be posed as a multi-label optimization problem, where the *scene graph nodes* are the **nodes** and the *model graph nodes* are the **labels**.

This multi-label optimization problem is formulated as a Quadratic Pseudo-Boolean Optimization (QPBO) [18], [19] problem. In this setting, each vertex consists of a node and its possible label. Thus, the maximum number of nodes in this graph can be $|V_s| \times |V_m|$. Since the node matches are obtained in pairs or cliques, the co-occurring node labels are considered as neighbors in this graph. The weights for these vertices are obtained from the fitness scores of the hypotheses. By solving this optimization problem, we get the optimal match between the model and scene graphs. This acts like a filtering step which ensures that conflicting hypotheses are removed.

2.1.4 Efficient Hypothesis Verification

Hypothesis verification consists of transforming the model point cloud according to the transformation estimate and calculating how much of it matches with the scene point cloud. Since we use a primitive shape decomposition of the scene and model clouds, the hypothesis verification step can be simplified. The idea is to utilize this primitive shape decomposition and use it to speed up the point cloud matching step.

Since the model and scene clouds are decomposed into primitive shapes and represented as PSG's, matching these point clouds is equivalent to matching all the primitive shapes in their PSG's. A Minimum Volume Bounding Box (MVBB) [20] is computed for each of these primitive shapes. Matching these primitive shapes can then be approximated by finding the intersection of their MVBB's. The i-th MVBB comprises 8 vertices $v^i_{1,..,8}$, which are connected by 12 edges $l^i_{1,..,12}$ and forms 6 faces $f^i_{1,..,6}$. To find the intersecting volume between MVBB's i and j, the points p^i at which the lines which form the edges of MVBB i intersect the faces of MVBB j are computed. Similarly, p^j are computed. Vertices v^i of the first MVBB which lie inside the MVBB j and vertices v^j of the second which lie inside the MVBB i are also computed. The intersection volume is then the volume of the convex hull formed by the set of points $\left(p^i \cup p^j \cup v^i \cup v^j\right)$.

The fitness score for this match is the ratio of the total intersection volume to the sum volumes of the primitive shapes in the model point cloud. This score is an approximation of the actual match but the speed-ups achieved by this approximation are more significant compared to the error due to approximation.

Fig. 4 shows examples of results obtained using the PSORR algorithm. Fig. 4 (a) shows the case when a partial view of the object is present in the scene. Fig. 4 (b) shows the case where a very low resolution full view of the object is

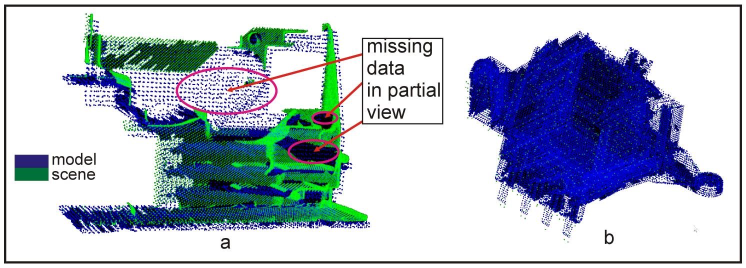

Fig. 4. Example of object recognition and pose estimation using PSORR algorithm: (a) scene cloud containing partial view of object (b) scene cloud containing sparse full view of object.

present in the scene. In both cases, the algorithm is able to recognize the object and estimate the pose accurately.

2.2 Combining shape and color information

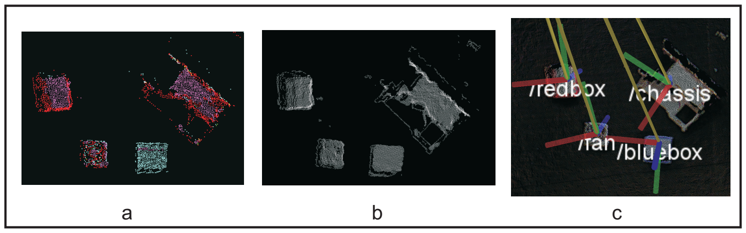

Fig. 5. Example of object recognition using a combination color and shape information: (a) Color Based segmentation (b) Detected Object Clusters (c) Final result of Object Recognition using shape and color information.

A combination of multi-label graph-cuts based optimization [16] and Conditional Euclidean Clustering [11] is used for color-based segmentation of point clouds. Fig. 5 shows an example of object recognition using a combination of color and shape information, where the point cloud is first segmented using color information. Each of these segmented objects is then recognized using the PSORR method described in Sect. 2.1.3. Fig. 5 (a) shows the color based segmentation, Fig. 5 (b) shows the clustered objects and Fig. 5 (c) shows the final recognized objects along with their poses.

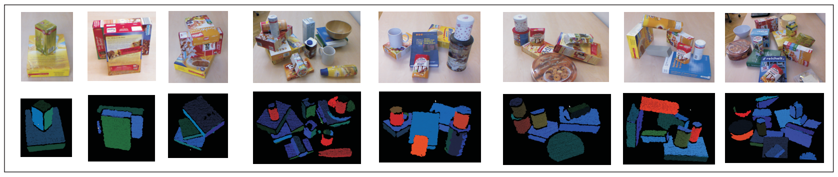

Fig. 6. Primitive Shape Detection results. Cylinders are shown in red and planes are shown in blue-green.

3 Evaluation and Performance Analysis

The Object Segmentation Database [21] was used to evaluate parts of this work. Fig. 6 shows the results from primitive shape decomposition of scene clouds taken from the Open Shape Database.

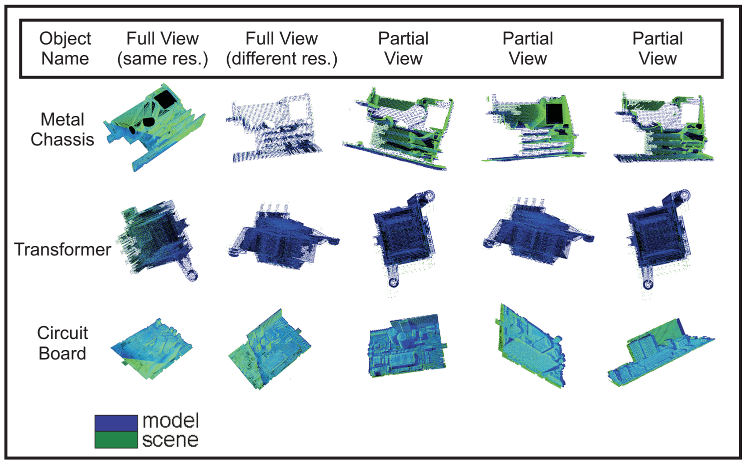

Fig. 7. Shape Based Object Recognition results.

Fig. 7 illustrates the results obtained for the PSORR algorithm (Sect. 2.1.3) over industrial workpieces using partial and full views at different resolutions.

Table 1 provides a comparison of the ORR and PSORR methods in terms of the number of hypotheses generated and the hypothesis verification time for each of the hypotheses. It can be observed that the PSORR method generates fewer hypotheses and has a much faster hypothesis verification phase.

4 Applications

The object recognition and pose estimation algorithm presented in this paper was evaluated on a HRI application in a realistic industrial setting. Such environ-

Table 1. Comparison of ORR and PSORR recognition algorithms

Object	**Algorithm**	**Number of Hypotheses**	**Hypothesis verification time**
Metal Chassis	ORR	2000	100ms
Metal Chassis	PSORR	100	1ms
Transformer	ORR	1000	50ms
Transformer	PSORR	30	1ms
Circuit Board	ORR	2000	100ms
Circuit Board	PSORR	30	1ms

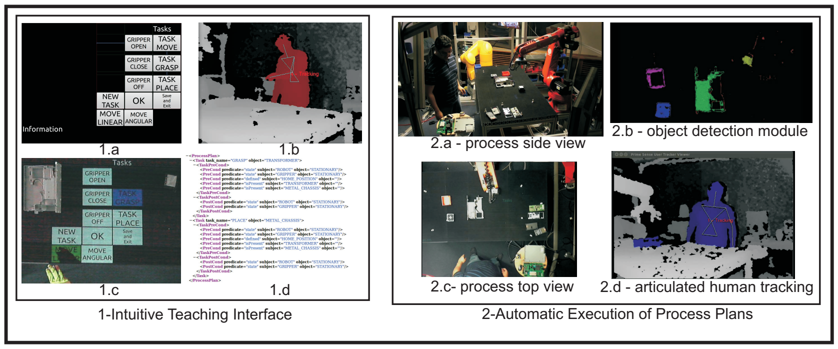

Fig. 8. 1. Intuitive Teaching Application : (a) A snapshot of the GUI used in the application (b) the Human tracking module providing the hand positions (c) The projected GUI controlled using hand gestures (d) Process plan taught using the application. 2. Automatic Execution of Process Plans : (a & c) Process Plan views (b) Object Recgonition and Pose estimation (d) Articulated Human Tracker

ments are typically unstructured and objects are often occluded by the human. Noisy point cloud data was obtained from the low-cost depth sensor (Microsoft kinect) used in the experiments. Also, accurate object poses are required for precise pick-and-place tasks, due to mechanical limitations of the 2-fingered gripper. Given these constraints, an accurate algorithm which can handle occlusions, partial views and sensor noise is essential for such scenarios.

In HRI scenarios, the separation of problem and solution spaces is a popular concept and the perception module is a key component linking these spaces. This separation enables the robot system to converse with the human about objects and their semantic properties rather than numeric values and parameters, which makes the HRI experience more intuitive for the human. Further details about this HRI setup and the associated concepts are beyond the scope of this paper.

4.1 Intuitive Interface for Teaching Process Plan

A mixed reality interface is designed for teaching process plans to the robot using intuitive physical human-robot interaction. The human can grasp the robot by its end-effector and take it to the desired position and orientation. Some of the results from this application are illustrated in Fig. 8 (1), where a GUI projected on the working table is controlled using hand gestures to record the taught robot poses. The perception module detects the objects present in the scene and a reasoning module associates objects with the taught poses to automatically generate a semantic description of this process plan in STRIPS [22] format.

4.2 Automatic Plan Execution

This application is aimed at automatic execution of semantic process plans in industrial scenarios. The perception module plays a key role in bridging the gap between the semantic level process plan and the real-world numeric parameters required for execution by providing positions and orientations of workpieces during execution. The object recognition and pose estimation approach used in this application is described in Sect. 2.2. The human can also point to objects on the table which will be considered as obstacles for the robot. Fig. 8 (2) shows snapshots from this application.

A video illustrating results for the algorithms presented in this paper and its use in the applications mentioned above can be found at :
http://youtu.be/6pjlpJa0C8Y.

5 Conclusion and Future Work

The main contribution of this work has been the development of a shape based object detection and recognition module which can handle sensor noise, occlusions and partial views. This module can be trained from CAD models or scanned 3D objects. In the current implementation, planes and cylinders were used for primitive shape decomposition of point clouds. This could be easily extended for other shape primitives such as torus, spheres or other conics. The primitive shape merging phase supports primitives in general as long as a fitness score and model complexity can be defined.

References

1. Scovanner, P., Ali, S., Shah, M.: A 3-dimensional sift descriptor and its application to action recognition. In: Proceedings of the 15th international conference on Multimedia. MULTIMEDIA '07, New York, NY, USA, ACM (2007) 357–360
2. Sipiran, I., Bustos, B.: Harris 3d: a robust extension of the harris operator for interest point detection on 3d meshes. Vis. Comput. **27** (2011) 963–976
3. Zhong, Y.: Intrinsic shape signatures: A shape descriptor for 3d object recognition. In: Computer Vision Workshops (ICCV Workshops), 2009. (2009) 689–696

4. Rusu, R.B., Bradski, G., Thibaux, R., Hsu, J.: Fast 3d recognition and pose using the viewpoint feature histogram. In: Intelligent Robots and Systems (IROS), 2010, IEEE (2010) 2155–2162
5. Rusu, R.B., Blodow, N., Beetz, M.: Fast point feature histograms (fpfh) for 3d registration. In: Robotics and Automation (ICRA)., IEEE (2009) 3212–3217
6. Hu, G.: 3-d object matching in the hough space. In: Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on. Volume 3. (1995) 2718–2723 vol.3
7. Schnabel, R., Wessel, R., Wahl, R., Klein, R.: Shape recognition in 3d point-clouds. In Skala, V., ed.: The 16-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2008, UNION Agency-Science Press (2008)
8. Schnabel, R., Wahl, R., Klein, R.: Efficient ransac for point-cloud shape detection. Computer Graphics Forum **26** (2007) 214–226
9. Papazov, C., Haddadin, S., Parusel, S., Krieger, K., Burschka, D.: Rigid 3D geometry matching for grasping of known objects in cluttered scenes. International Journal of Robotic Research **31** (2012) 538–553
10. Papazov, C., Burschka, D.: An efficient ransac for 3d object recognition in noisy and occluded scenes. Proceedings of the 10th Asian conference on Computer vision - Volume Part I (2011) 135–148
11. Hastie, T., Tibshirani, R., Friedman, J.: 14.3.12 Hierarchical clustering The Elements of Statistical Learning. 2nd ed. edn. New York: Springer, ISBN 0-387-84857-6 (2009)
12. Gonzalez, R.C., Woods, R.: Digital Image Processing. 2nd edn. Prentice Hall, New Jersey (2002)
13. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. **23** (2001) 1222–1239
14. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision. Pattern Analysis and Machine Intelligence, IEEE Transactions on **26** (2004) 1124–1137
15. Delong, A., Osokin, A., Isack, H., Boykov, Y.: Fast approximate energy minimization with label costs. In: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. (2010) 2173–2180
16. Delong, A., Osokin, A., Isack, H.N., Boykov, Y.: Fast approximate energy minimization with label costs. Int. J. Comput. Vision **96** (2012) 1–27
17. Leonardis, A., Gupta, A., Bajcsy, R.: Segmentation of range images as the search for geometric parametric models. Int. J. Comput. Vision **14** (1995) 253–277
18. Boros, E., Hammer, P.L.: Pseudo-boolean optimization. Discrete Appl. Math. **123** (2002) 155–225
19. Rother, C., Kolmogorov, V., Lempitsky, V., Szummer, M.: Optimizing binary mrfs via extended roof duality. In: Computer Vision and Pattern Recognition, 2007. CVPR '07. (2007) 1–8
20. Barequet, G., Har-Peled, S.: Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. J. Algorithms **38** (2001) 91–109
21. Richtsfeld, A., Morwald, T., Prankl, J., Zillich, M., Vincze, M.: Segmentation of unknown objects in indoor environments. In: Intelligent Robots and Systems (IROS), 2012. (2012) 4791–4796
22. Fikes, R.E., Nilsson, N.J.: Strips: A new approach to the application of theorem proving to problem solving. Technical Report 43R, AI Center, SRI International, 333 Ravenswood Ave, Menlo Park, CA 94025 (1971) SRI Project 8259.