Field of view dependent registration of point clouds and incremental extraction of table-tops using time-of-flight cameras

Georg Arbeiter, Martin Hägele and Alexander Verl

Abstract— **Perception of the environment is crucial for many robot applications. Thus, geometric and semantic mapping using point cloud sensors is subject to many research activities. In this paper, an approach to incrementally register point clouds from time-of-flight cameras and create feature maps is presented. Frustum culling ICP, key frame based plane segmentation and aggregation of a feature map are presented and evaluated on the service robot Care-O-bot® 3. The focus is on table-top extraction for tele-operated robots.**

Index Terms— **Registration, frustum culling, table-top extraction, tele-operation**

I. INTRODUCTION

Geometric and semantic mapping are very important techniques for robotics as they enable a robot to perceive and interact with the environment. Recently, it has been subject to many research activities. Most approaches use laser scanners or 3-D cameras as input, build point maps while or after robot movement and try to extract geometric shapes from the point map. Geometric properties and relationships can then be used to generate semantic information about the environment.

In the field of tele-operated robots, requirements are different from those for autonomous robots. First, real-time map update plays an important role. The human remote user has to be provided with continuous map updates as fast as possible. Otherwise, controlling the robot in an intuitive way becomes almost impossible. For automomous robots it is most of the times sufficient to have a map update at a planned point of time. Second, the map representation has to be usable for visualization in tele-operation. The tele-operator has to get a visual impression of the environment that he or she can understand easily. Assistant functions like highlighting regions of interest in the GUI can improve robot control greatly. For instance, objects for manipulation are often located on table-tops. Therefore, table-top segmentation can be used for identification of regions of interest for object detection.

In this paper, we present an approach for incremental table-top extraction during tele-operated robot movement. The focus is on computation time efficient registration of point clouds and extraction of planar surfaces. To reduce the mean computational load we try to limit the amount of sensor data that has to be processed. Therefore, we use modifications of well-known algorithms that mainly work on current sensor data and only process selected key frames. Also, we use laser range finder based 2D localization in order to ease the mapping problem. As our robot is equipped with a time-of-flight camera, sensor data filtering also is an important part of our architecture. We will show that our approach reduces computation time compared to state-of-the-art methods while it is able to keep a consistent map representation.

The remainder of this paper is structured as follows: an overview of related work is given in Section II. Section III presents our architecture for incremental extraction whereas Section IV shows our filter efforts for time-of-flight sensors. Section V describes the key frame selection and point cloud registration. The table-top extraction is shown in Section VI, followed by feature map and merging in Section VII. Evaluation of our method is given in Section VIII. Finally, conclusions and outlook are presented in Section IX.

II. RELATED WORK

Segmentation and detection of horizontal surfaces has been presented in some recent publications. For example, Rusu et al. used a tilting laser scanner to acquire point clouds in [1] and performed planar segmentation using RANSAC in order to find table surfaces. The detected tables were used as a base for model fitting and surface reconstruction of objects residing on the tables. Further work by Rusu et al. shows semantic object labeling of planar surface structures in kitchen environments like cupboards, tables and drawers [2]. Geometric features like width or height of the planes are used for classification. Another recent approach to semantic mapping of surfaces is presented in [3]. Trevor et al. show an extension to their feature based mapping technique that is able to identify and label horizontal planar surfaces during SLAM. They rely on Rusu's work for planar segmentation and are able to add table locations to their map graph. However, all those approaches use the point cloud of a whole area as input data. Thus, table segmentation can only be done after scanning larger parts of the environment. In contrast, we propose an incremental method for table segmentation. Also, all those methods use tilting laser scanners in order to generate point clouds. Those are significantly different w.r.t. accuracy, frame rate and field of view. Thus, they cannot be applied without modification to time-of-flight camera data.

Aggregation of 3-D maps using time-of-flight cameras also has been subject to various research activities. First of all, measurements errors of time-of-flight cameras are much higher than those from laser range finders. This leads to the need of filter algorithms that are able to deal with the error effects. Some effective filtering methods were proposed in [4], namely amplitude filter and filtering of jump edges. In

The authors are with the Institute for Manufacturing Engineering and Automation, Fraunhofer IPA, 70569 Stuttgart, Germany <first name>.<last name>@ipa.fraunhofer.de

addition, statistical outlier removal [5] is a powerful means to reduce the amount of spurious measurements in the point cloud.

Meanwhile, the Iterative Closest Point algorithm (ICP) [6] is a common means for point cloud registration. It has been used in various variants for building 3-D maps, e.g. in [7], [8], [9], [10]. One interesting alternative – if the robot position estimate is fairly accurate – is frustum ICP, as proposed in [11]. The authors use knowledge about the field of view of a sensor to achieve a pre-segmentation of the input data for registration. This keeps the time needed for registration constant with growing map size.

III. SYSTEM ARCHITECTURE

As the focus of the table-top extraction system is on application in tele-operation, incremental construction of a feature map and fast map update has to be assured. The architecture which is used for achieving this is shown in figure 1. Raw data from a time-of-flight camera is used as input. First, the sensor data has to be processed in a filter cascade in order to improve the point cloud quality. Several filter methods are applied in a carefully chosen order. To reduce the amount of redundant sensor information, only key frames are further processed. The key frames are selected w.r.t. to the robot motion so that they have a certain overlap. In the next step, field-of-view dependent registration of key frames takes place. This is done to reduce the amount of data and therefore the time needed for registration. The registered point clouds are stored in the point map and used for subsequent registration steps.

The registered key frame is passed to the object extraction component that performs planar decomposition and generates a convex hull for each table-top surface in the current view. This enables a fast extraction at a limited point cloud size. As the same table-top can be extracted from different key frames, the convex hulls have to be merged. This is done by a polygon clipping algorithm. The merged table-tops are finally stored in the object map.

This architecture allows an incremental update of both the point map and the feature map. In contrast, working on the full dataset would slow down the system over time as more and more data is added to the point map and both registration and extraction would require an increasing computational effort. The problem of keeping the feature map up-to-date is overcome by a simple merging method.

IV. TOF DATA FILTER

In order to improve the quality of the sensor data, it has to be filtered. The main error types are presented in [12]. In a first step, jump edges are filtered. These occur at sudden transitions between surfaces in the environment, e.g. between an object on a table and the background. Jump edges can be filtered by comparing geometric relations between neighboring pixels. Because other filter methods like amplitude filtering remove points from the point cloud and therefore destroy relevant neighbor relations, the jump edge

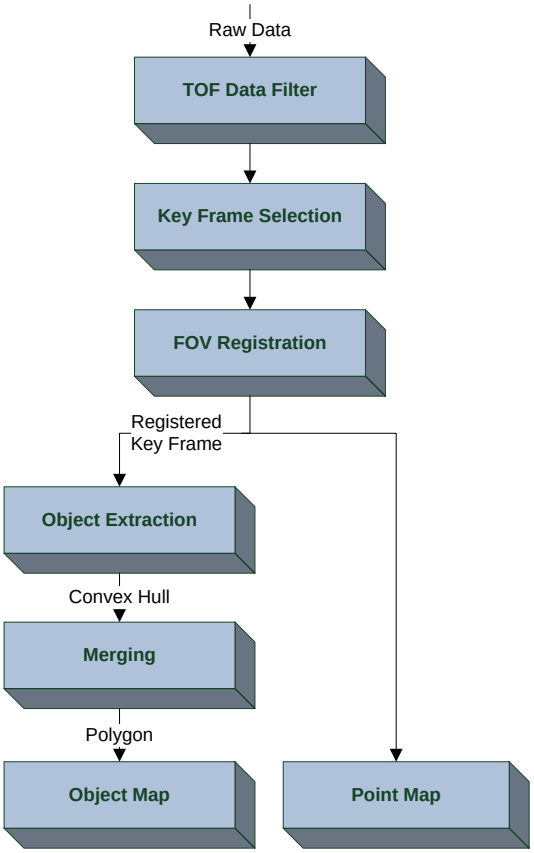

Fig. 1. System architecture

filtering has to be done first. Calculation of the angle between neighboring points is done according to [4], i.e.

$$\xi_i = \max \arcsin \left(\frac{\|p_{i,n}\|}{\|p_{i,n} - p_i\|} \sin \varphi \right) \tag{1}$$

with the angle ξ_i between the vector p_i and the difference vector between p_i and one of the eight neighboring points $p_{i,n}$. φ depicts the apex angle between two neighboring pixels. If $\xi_i > \xi_{th}$, p_i is considered to be part of a jump edge and neglected.

In a second step, errors caused by low illumination or measurements outside the non-ambiguity range can be handled by amplitude thresholding: Sensor values with a low amplitude $I_n < I_{th}$ usually indicate a low confidence and can be removed. Some time-of-flight sensors provide a confidence map that hold a probability of confidence for each pixel. The confidence value can provide additional information on how the respective pixel should be treated.

To reduce Gaussian noise and spurious pixels remaining after previous steps, statistical outlier removal is an effective algorithm. As proposed in [4], [5], for all points in the point cloud, the mean distance to neighboring points is calculated. Afterwards, thresholding w.r.t. the deviation of the distance can take place and thus, points with a higher distance from their neighbors than the mean are filtered.

All those filter mechanisms are well evaluated in literature. However, the filter sequence can have a huge influence on the

quality of the resulting point cloud. We propose to perform jump edge filtering first because the geometric structure of the point cloud should be unchanged for that. The amplitude filter should be applied next as it is a reliable indicator of measurement errors. The statistical outlier removal is applied at last as it does not account for the physical principle of errors and just removes spurious pixels that remain after amplitude and jump edge fitering. Furthermore, the two initial filters are not able to filter all erroneous measurements, but at least reduce the density of the point cloud in erroneous regions. This increases the effectiveness of the statistical outlier filter in those regions.

V. KEY FRAMES AND REGISTRATION

The filtered point clouds are registered in the next step. Although we rely on laser range finder based localization, registration of point clouds is still necessary as the localization is inaccurate especially when the robot does rotational movement. However, only selected key frames are processed in registration. The selection process uses information about the robot motion, i.e. the position change of the platform since registration of the last key frame, to determine new key frame candidates. The parameters of the selection process mainly depend on the sensor properties (opening angle, maximum range) and the average distance of environment structures. Subsequent key frames must have a certain overlap in order to guarantee successful registration. Registration of key frames only reduces the mean computational load over time in the way that it avoids multiple registration of redundant sensor data.

ICP is a well established method for point cloud registration. However, when registering to the full map, the time needed for one iteration grows over time. Therefore, we use an ICP variant proposed in [13] called frustum ICP. The idea behind is, that if the robot position is known at least approximately, the current sensor data can be registered to the subset of the map residing in the current field of view of the sensor.

Calculation of the field of view is done once for each sensor. The method used models the measurement cone as a frustum (see Figure 2). Five clipping planes are defined regarding the sensor properties like horizontal and vertical opening angle and maximum range. For each clipping plane the normal vector

$$n_i = v_l \times v_m \tag{2}$$

is calculated. The normal vectors are used to determine whether a point is inside or outside the frustum by testing

$$(x_p - x_0)n_{i,x} + (y_p - y_0)n_{i,y} + (z_p - z_0)n_{i,z} < 0 \tag{3}$$

The field-of-view frustum has to be transformed from F_{cam} to F_W in each time step according to the current robot position estimate.

VI. TABLE-TOP EXTRACTION

Our table-top extraction method is similar to those already proposed in literature like [1]. The difference is, that it is applied to each key frame and not to the full point map.

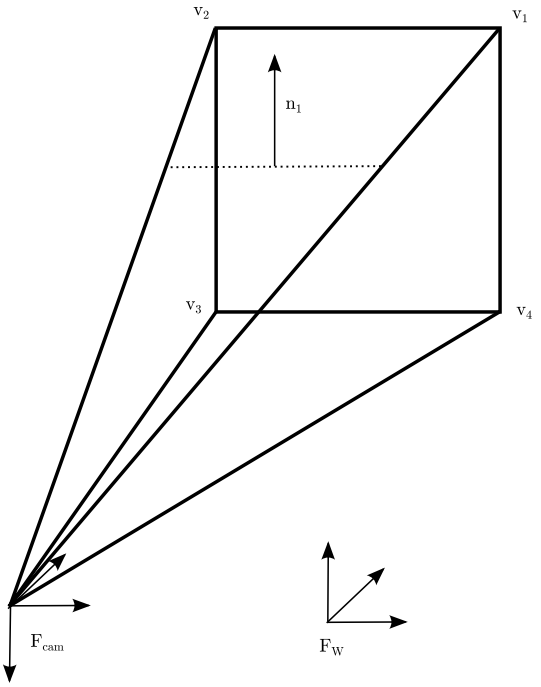

Fig. 2. Field of view of a time-of-flight camera modeled as a frustum.

First, Eucledian clustering is applied to the key frame to identify connected parts of the point cloud. For each cluster of at least a minimum size, the point normals are estimated to enhance the next step, plane segmentation. We use RANSAC to fit a plane model to each cluster in order to identify the dominant plane.

For each segmented plane certain rules are evaluated in order to calculate the confidence that it is indeed a table-top. Evaluation consists of two steps:

1) Each plane with a number of inliers below a threshold is neglected as we only want to process planes of a certain size
2) The plane has to be horizontal and at a certain height interval

If these conditions are fulfilled, the plane is considered part of a table-top.

The next step is then to calculate the convex hull for the inliers of the plane. This is done according to [**?**] and the output is a 2D hull consisting of some tenths of points. Each convex hull is passed to the merger for incorporation in the feature map.

VII. MERGING AND FEATURE MAP

The merger works on the feature maps and tries to incorporate the convex hulls coming from the table-top extractor in the feature map. The convex hulls can be considered polygons. Therefore, a polygon clipping algorithm can be applied to merge incoming hulls to those already residing in the map.

We use an algorithm derived from the separating axis theorem [**?**]. It says that for two convex shapes in 2D space there exists a line onto which the projections of

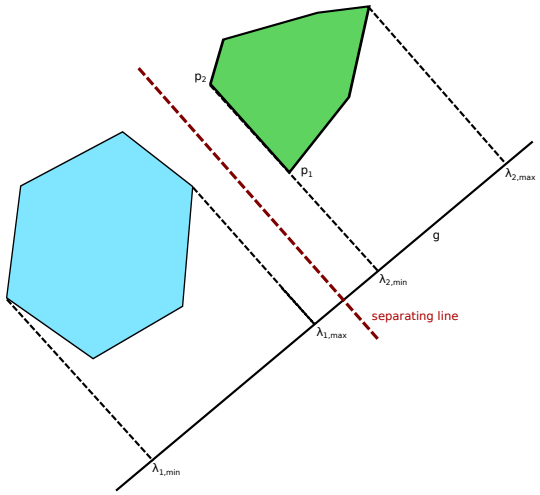

Fig. 3. Polygon clipping

their points are separate if and only if they are not intersecting. This means, we have to check for each side of the two polygons whether the projections of all points of the two shapes on a line perpendicular to the side are separate (see Fig. 3). The detailed sequence of calculation is:

1) For two adjacent points of a polygon p_n and p_{n+1} the linear equation of the perpendicular line is formed by

$$\vec{x} = \vec{p}_n + \lambda \vec{g} \tag{4}$$

with

$$\vec{g} = \begin{pmatrix} -(p_{n+1,y} - p_{n,y}) \\ p_{n+1,x} - p_{n,x} \end{pmatrix} \tag{5}$$

2) The projection s_i of each point q_i of the polygons on the line is calculated by

$$(\vec{s}_i - \vec{q}_i) \circ \vec{g} = 0 \tag{6}$$

3) s_i is part of the line and can be expressed by equation (4). Together with equation (6), the line parameter for each projected point can be calculated by

$$\lambda_i = \frac{\vec{s}_i \circ \vec{g} - \vec{p}_n \circ g}{g_x^2 + g_y^2} \tag{7}$$

4) In order to decide whether the current perpendicular line is a separating line, the condition

$$[\lambda_{1,min}, \lambda_{1,max}] \cap [\lambda_{2,min}, \lambda_{2,max}] = \emptyset \tag{8}$$

with the line parameters λ_1, λ_2 corresponding with the points of polygon 1 and 2 respectively, has to hold.

If there is at least one separating line, the two polygons are considered to be non-intersecting. Otherwise, the two convex hull polygons have to be merged. A simple method is to concatenate the two input hulls and calculate a new convex hull for the concatenation. In the case of intersecting polygons, the respective feature in the map is updated. If merging with all features in the map fails, a new feature is added.

VIII. RESULTS

The incremental table-top extraction method is evaluated on the service robot Care-O-bot® 3 [14] in the Fraunhofer IPA robot lab. Comparison of computation time between our method and state-of-the-art was done. Quality of the table extraction was tested empirically.

A. Robot Platform

The Care-O-bot® 3 platform (see Fig. 4) is a service robot product vision developed by Fraunhofer IPA. It consists of an omni-directional base with three laser range finders used for localization and navigation. For manipulation tasks, it is equipped with a 7 degree-of-freedom light-weight arm and a three-finger gripper. On the agile head, two color cameras and a SwissRanger™ SR4000 time-of-flight camera is mounted. The SR4000 is used as sensor for the method described in this paper.

Fig. 4. Care-O-bot® 3, used for evaluation

B. Data Sets

The dataset for evaluation was recorded in the IPA kitchen environment. The robot was driven manually on a circle-like trajectory around a table. Sensor and localization data was recorded untriggered during movement. On the whole, the dataset covers about 52 s of robot operation. The scene consists of a small table with three objects on top, a second table with computer monitors on top in the background and also areas that are out of range of the time-of-flight camera. This yields noisy measurements. Fig. 5 shows a picture of the kitchen environment.

C. Evaluation

First of all, the filters were evaluated. The quality of the filtered point cloud was rated empirically by visualizing the data. The filters were adjusted in order to filter a least all

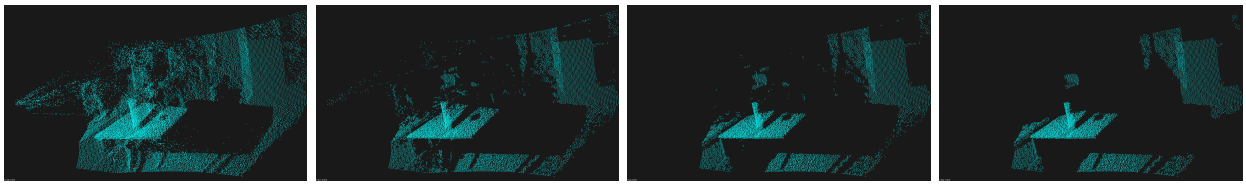

Fig. 6. Sensor data filtering: raw data, jump edge filter, amplitude filter, statistical outlier removal

Fig. 5. Fraunhofer IPA kitchen environment

erroneous measurements. As most parts of the scene are seen from multiple views during movement, rather strong filtering is the better choice.

Fig. 6 shows the unfiltered point cloud and the filtered clouds after jump edge, amplitude and statistical outlier filtering. As can be seen, jump edge and amplitude filtering already remove most of the noise. The statistical outlier removal only discards single remaining pixels in this case. Thus, the influence of the point cloud quality is rather low. This leads to the conclusion that this kind of filtering is obsolete for most of the images. Well-working filter parameters are an amplitude threshold of 4000, an angle threshold of 170° for jump edges and a *mean K* of 50 with a standard deviation multiplier of 1 for the statistical outlier removal.

Keyframe selection only passes a new frame to registration if a certain robot pose change occured. There should be at least 50% of overlap between consecutive key frames to ensure correct registration. For our dataset, we identified proper parameters to be 5° for rotational movements and 0.3 m for translation.

In the next evaluation step, standard ICP is compared to frustum culling ICP regarding computation time. In order to get meaningful results, we measured the time for the first iteration. Otherwise, the number of iterations has a huge influence on the total registration time which makes it hard to get comparable values. Fig. 7 shows the computation time for the first iteration for 15 consecutive key frames. While the computation time using standard ICP is growing over time due to the growing map size, the computation time for frustum ICP stays constant after an initial increase because it takes some frames until the collected data covers the whole frustum. However, for our rather small dataset of 15 key frames, the impact on the whole registration time is not very distinct. Yet it will grow for larger data sets. ICP was able to converge in the correct minimum for all key frames in the scene, regardless whether we used the standard method or frustum culling. Hence, there is no difference in quality between both methods.

Table-top extraction is evaluated by comparing the computation times between using the full point map in each extraction step and only the aligned key frame with successive merging. Fig. 8 depicts the computation time over key frames for our dataset. It can be seen that the time needed for table-top extraction grows boundlessly when using the full map whereas it is nearly constant when only using the key frame. Merging is very cheap as the convex hulls consist of few points only and the number of features in the map is low for a typical household scene.

The resulting point map and feature map can be seen in Fig.

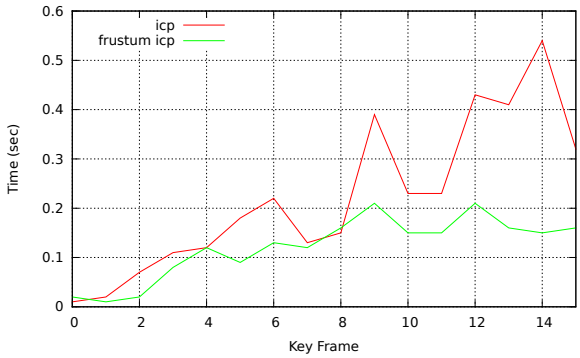

Fig. 7. Comparison of the computation time for the first iteration between standard ICP and frustum ICP.

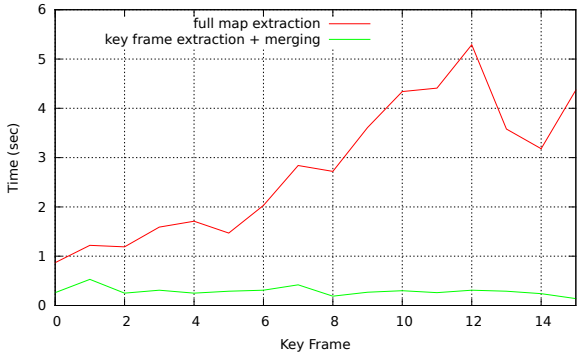

Fig. 8. Comparison of the computation time for table-top extraction on the full cloud and on the key frame only with successive merging.

9 and Fig. 10. Fig. 9 shows the maps after key frame 2. The table-top is uncomplete. Fig. 10 depicts the final maps with a completely merged table-top.

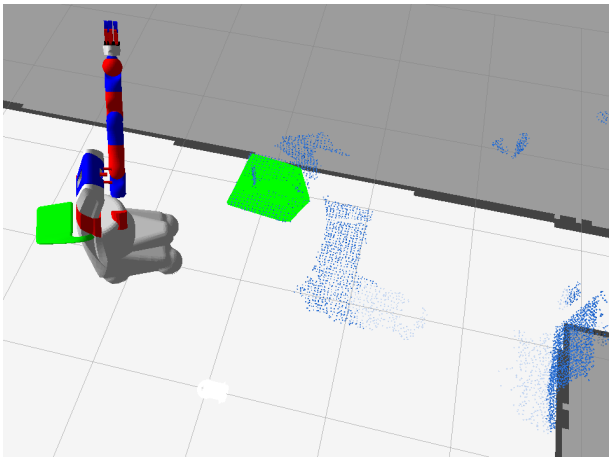

Fig. 9. Point and feature map after key frame 2.

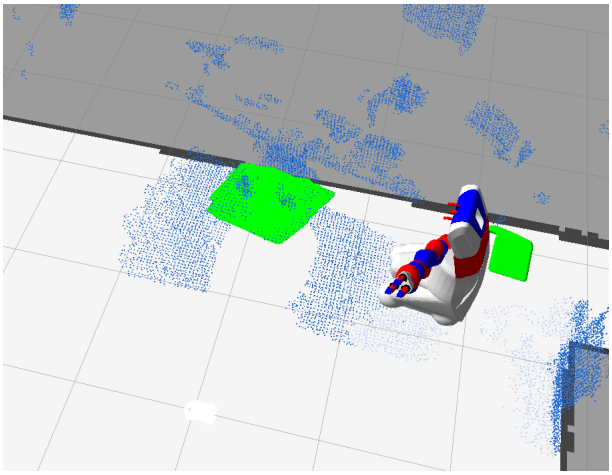

Fig. 10. Final point and feature map after key frame 15.

IX. CONCLUSIONS AND FUTURE WORK

We presented an approach for incremental registration and table-top extraction on a tele-operated robot. The focus was on reduced computation time. This was achieved by limiting the amount of data the algorithms work on. For registration, the frustum culling variant was used to enable working on data in the current field of view only. Table-top extraction was performed on key frames only to reduce computation effort. However, resulting table-tops from different frames had to be merged in a feature map, which is computationally cheap.

Future work will focus on improved feature maps. For example, different geometric shapes identifying walls, shelves, doors could be extracted and incorporated in the feature map. Parts of the point cloud that cannot be described by simple geometric shapes could at least be clustered or processed by surface mesh generation. The long-term goal is to describe most parts of the environment by geometric features. The main benefit from this is a serious reduction of the amount of data to be transferred to a remote user and easy visualization.

X. ACKNOWLEDGEMENTS

This research was partly funded from the EU FP7-ICT-247772 Multi-Role Shadow Robotic System for Independent Living and from ARTEMIS Joint Undertaking as part of the project R3-COP and from the German Federal Ministry of Education and Research (BMBF) under grant no. 01IS10004C.

REFERENCES

[1] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz, "Close-range scene segmentation and reconstruction of 3D point cloud maps for mobile manipulation in domestic environments," in *Proceedings of the 2009 IEEE/RSJ international conference on Intelligent robots and systems*, p. 1–6, 2009.

[2] R. B. Rusu, Z. C. Marton, N. Blodow, A. Holzbach, and M. Beetz, "Model-based and learned semantic object labeling in 3D point cloud maps of kitchen environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), St. Louis, MO, USA*, 2009.

[3] A. Trevor, J. R. III, C. Nieto-Granda, and H. Christensen, "Tables, counters, and shelves: Semantic mapping of surfaces in 3D," in *IROS Workshop on Semantic Mapping*, 2010.

[4] S. May, D. Droeschel, D. Holz, C. Wiesen, and S. Fuchs, "3D pose estimation and mapping with Time-of-Flight cameras," in *International Conference on Intelligent Robots and Systems (IROS), 3D Mapping workshop, Nice, France*, 2008.

[5] R. B. Rusu, N. Blodow, Z. Marton, A. Soos, and M. Beetz, "Towards 3D object maps for autonomous household robots," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, p. 3191–3198, 2007.

[6] P. Besl and H. McKay, "A method for registration of 3-D shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 239–256, Feb. 1992.

[7] R. B. Rusu, N. Blodow, and Z. C. Marton, "Aligning point cloud views using persistent feature histograms," in *Proceedings of the 21st IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2008.

[8] A. E. Johnson and S. B. Kang, "Registration and integration of textured 3-D data," in *3dim*, p. 234, 1997.

[9] S. Druon, M. Aldon, and A. Crosnier, "Color constrained ICP for registration of large unstructured 3D color data sets," in *2006 IEEE International Conference on Information Acquisition*, (Veihai, China), pp. 249–255, 2006.

[10] T. Masuda, K. Sakaue, and N. Yokoya, "Registration and integration of multiple range images for 3-D model construction," in *Proceedings of 13th International Conference on Pattern Recognition*, (Vienna, Austria), pp. 879–883, 1996.

[11] S. May, S. Fuchs, D. Droeschel, D. Holz, and A. Nüchter, "Robust 3D-mapping with time-of-flight cameras," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009.

[12] S. Fuchs and G. Hirzinger, "Extrinsic and depth calibration of ToF-cameras," in *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, p. 1–6, 2008.

[13] J. Luck, C. Little, and W. Hoff, "Registration of range data using a hybrid simulated annealing and iterative closest point algorithm," in *IEEE International Conference on Robotics and Automation*, vol. 4, p. 3739–3744, 2000.

[14] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Trans. Math. Softw.*, vol. 22, pp. 469–483, December 1996.

[15] E. Golshtein and N. Tretyakov, *Modified Lagrangians and monotone maps in optimization*. New York: Wiley, 1996.

[16] C. Parlitz, M. Hägele, P. Klein, J. Seifert, and K. Dautenhahn, "Care-o-bot 3 - rationale for human-robot interaction design," in *Proceedings of 39th International Symposium on Robotics (ISR), Seul, Korea*, 2008.