The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Robust On-line Model-based Object Detection from Range Images

Bastian Steder Giorgio Grisetti Mark Van Loock Wolfram Burgard

***Abstract*— A mobile robot that accomplishes high level tasks needs to be able to classify the objects in the environment and to determine their location. In this paper, we address the problem of online object detection in 3D laser range data. The object classes are represented by 3D point-clouds that can be obtained from a set of range scans. Our method relies on the extraction of point features from range images that are computed from the point-clouds. Compared to techniques that directly operate on a full 3D representation of the environment, our approach requires less computation time while retaining the robustness of full 3D matching. Experiments demonstrate that the proposed approach is even able to deal with partially occluded scenes and to fulfill the runtime requirements of online applications.**

***Index Terms*— Object detection, point clouds, range images**

I. Introduction

Service robots that offer various services to their users need interfaces that facilitate high-level interactions with the user and the environment. To achieve this goal, one fundamental issue is the ability to classify and localize the objects under discourse or relevant for the current task. For example, to execute high level commands such as “please bring a chair” or “please wait at the table” the robot needs to have the ability to identify the corresponding objects in the scene. Whereas the ability to robustly identify objects is obviously required for service robot applications, even other tasks including navigation may take advantage from such a capability. For example, in the domain of simultaneous localization and mapping the ability to recognize objects can be combined with the prior about their dynamic characteristics. This would improve the robustness of the existing SLAM systems in non-static environments and additionally would allow them to produce high level models of their environment.

In the literature, the problem of identifying objects in 3D point clouds has been intensively studied. Most of the existing techniques approach the problem by seeking for a transformation that aligns the model and the data. In principle, this task could be solved by one of the many 3D scan-matching algorithms. However, this is generally not practical because of two reasons. First, in object recognition an initial guess of the position of the object is not known in advance. This would require a scan matching algorithm to perform an expensive search in the full space of potential transformations. Second, to operate properly most of the scan matchers require a good overlapping of the model and the data. Due to the nature of the sensor, in a single 3D scan typically at most 50% of an object is visible and in presence of occlusions this percentage decreases. Therefore, one seeks for alternative and more effective solutions.

B. Steder, G. Grisetti and W. Burgard are with the Dept. of Computer Science of the University of Freiburg. Mark Van Loock is with Toyota Motor Europe/NV/SA, Production Engineering - Advanced Technologies, Zaventem, Belgium.{steder,grisetti,burgard}@informatik.uni-freiburg.de,Mark.Van.Loock@toyota-europe.com

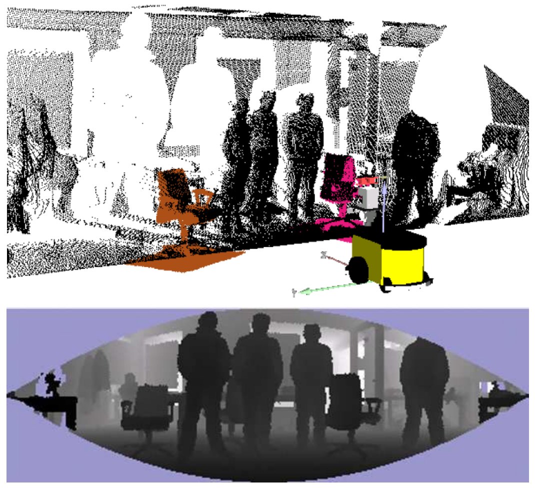

Fig. 1. Three-dimensional point cloud recorded with a mobile robot (top) and the corresponding range image (bottom). In the point cloud we also highlight two chairs identified with our approach.

In this paper, we present a fast and robust technique for detecting multiple objects in complex scenes. Our approach relies on the analysis of range images obtained from raw 3D laser data and is based on the extraction of point features from the range images. Robust feature matching is performed using a variant of the GOODSAC [9] algorithm. Furthermore, we propose a validation strategy based on range images, which allows us to robustly reject false positives. Compared to existing approaches, one of the major advantages of our method is that it is designed for online application. It directly operates on range images extracted from single 3D scans and thus is highly efficient. Furthermore, it does not require the robot to change its viewpoint.

The paper is organized as follows. After discussing related work, we will introduce our feature extraction and matching procedure in Section III-A. Then, Section IV will discuss the relationship of our approach to the popular spherical spin images [10]. Finally, in Section V we will present experiments illustrating the recognition rate of our approach and analyzing the dependency on the distance of the object.

II. Related Work

Many approaches for object detection operating on raw 3D data attempt to find the best alignment by determining correspondences between regions of the current scan and regions of the model. Typically, these regions are represented by features that compactly describe areas in the data so that comparisons can be carried out efficiently.

This concept is extensively applied in computer vision, where object recognition based on feature extraction is a popular research topic. E.g., Lowe [7] used SIFT features to compare an image with a database of reference images. Clusters of matching features that agree on a possible object pose are extracted. The probability to be a true match is computed based on how well the matched features fit and on the number of probable false matches.

Our approach operates on 3D laser data. Compared to cameras these sensors provide accurate range information and are less sensible to the lightening conditions, but they are affected by a slow acquisition time. The increased accuracy of 3D lasers allows us to determine more accurate positions of the objects in the scene.

Gelfand *et al.* [3] present an approach to global registration based on "integral volume descriptors", which are one-dimensional descriptors whose values depend on the volume enclosed by the local surface around a point. Compared to our approach described in this paper, this technique is mainly targeted towards finding the best alignment of two shapes and not to test if an identical shape exists.

Johnson *et al.* [6] proposed the *spin-images* for object detection in 3D data. A spin image is a 2D representation of the surface surrounding a 3D point. This technique has been often reported to provide good matching results. In their approach they compute a spin-image for every point in the model and every point in the scene. Correa *et al.* [10] proposed a variant of spin-images (spherical spin images) that simplifies the comparison to nearest neighbor search by using the linear correlation coefficient as the equivalence classes of spin-images. To efficiently perform the comparison of features they compress the descriptor.

Triebel *et al.* [12] use spin-images as features for an associative Markov network (AMN). The parameters of this AMN are learned from a manually labeled training data set. The main difference between this work and our approach is that we identify complete instances of objects from their partial views observable in a scene, rather than labeling the single points according to classes they resemble. Mian *et al.* [8] use the so-called *tensor descriptors* which rely either on an accurate estimate of the surface normals or on the mesh structure of the data in general.

Stiene *et al.* [11] present an object detection approach based on silhouettes extracted from range images. Their features are based on a fast Eigen-CSS method and a supervised learning algorithm. Similar to this approach, our method also works on range images created from 3D laser range scans. However, in our method the extraction of features is not restricted to the contour of the objects. It also uses feature points that have a higher resistance against partial occlusion.

III. Object Detection based on Range Images

Our object database consists of a set of object models, which are given as point clouds obtained from real 3D data. From these data points, we first calculate a set of range images and from those a set of features. The overall object detection procedure works as follows: When a new 3D scan is acquired, we compute the corresponding range image for this scan (see Section III-A). We then extract a set of point features from this image. Subsequently, we compare the features in the scene with the features of the models. From a set of corresponding features we determine a set of potential alignments, as described in Section III-B. To this end we use a GOODSAC-like procedure and we rank the solutions based on the score function discussed in Section III-C. The final step seeks to validate each candidate solution based on the overlap between the range image of the scene and the range image of the candidate model. This is described in detail in Section III-D.

A. Feature Extraction and Matching from Range Images

Range images are visual representations of 3D scenes. The grey-value of every pixel in the image is proportional to the distance of the closest object in the direction of the ray corresponding to the pixel. Range images can be calculated efficiently from a 3D-scan by implementing a z-buffer [2]. An example of such a range image can be seen in Figure 1. For online object detection, considering all points in a range image would be too computationally demanding. Therefore we extract a set of *interest points* from the range images. We utilize the Harris Detector [4] for this purpose.

For each of these points p_i we compute a descriptor vector f_i which captures the structure of the object in the neighborhood of p_i. These descriptors are computed as follows:

- Let $\mathcal{N}(p_i)$ be a set of 3D points of the scene whose distance from p_i is below a given threshold.
- We compute the normal $\mathbf{n}_i$ of the planar approximation of the set $\mathcal{N}(p_i)$.
- We select a point v_i along the line which passes through p_i and is oriented according to $\mathbf{n}_i$.
- We set the observer position at v_i and its viewing direction at $-\mathbf{n}_i$.
- We finally compute a descriptor vector f_i for the point p_i by generating a range image which contains all the points in $\mathcal{N}(p_i)$, according to the computed observer position. To make the feature vector scale invariant we subtract from each entry of f_i the mean of the values in the range image.

The procedure described above is illustrated in Figure 2. Note that the selection of the viewpoint v_i along the normal vector of the surface resolves two of the three degrees of freedom in the observer orientation, since the angle of the range image along the normal axis is not specified. In our current implementation, we resolve this degree of freedom by orienting the image patch according to the z-axis in the world, i.e., the x-axis of the image patch will always be orthogonal to the z-axis in the world. In this way, our feature descriptor is not invariant to changes in the roll and the pitch of an object. However, since our goal is the detection of objects in human-made environments, like chairs or tables, this restriction makes sense and increases the robustness of the approach. However, in our experiments we found that our approach is still able to detect objects under mild violations of this assumption (variations of $\pm 15°$ in roll and pitch).

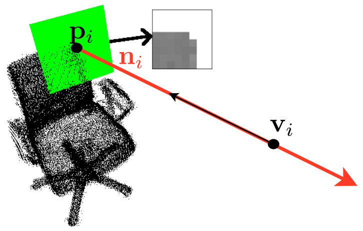

Fig. 2. Example for the extraction of the feature descriptor. In the point cloud of this chair, a square marks the projection plane used for the calculation of the range image patch. The normal of the patch corresponds to the plane approximation of the points in the area. The square is then divided into a grid (the pixels) of a fixed size and the distances of the surrounding points projected onto the plane determine the value for every grid cell.

As mentioned above, the object models are given as sets of features extracted from range images of the objects. These range images are sampled from point clouds of the objects obtained by scanning the object from different viewpoints. An example of this procedure is illustrated in Figure 3. From each of these range images we then extract the features of the model in the same way as we do with the scene.

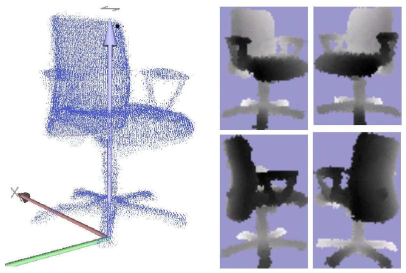

Fig. 3. Point cloud model of a chair (left) and four range images sampled from different perspectives (right)

The feature matching is done by treating the small range-image patches as vectors and searching for the closest neighbors in the feature space. For each feature f_i^s in the current scene and each feature f_j^m in the model, we compute the Euclidean distance $d(f_i^s, f_j^m)$ between their descriptor vectors. Based on this distance we create the set of potential correspondences $C = \{c_{ij}\}$ which are given by all feature pairs whose distances are below a given threshold $d_{\max}$. To efficiently perform this comparison process, we store all feature description vectors in a *k*d-tree. To find the neighbors in the feature space, we apply the best-bins-first technique proposed by Lowe [7] to handle the high dimensionality of the vectors.

B. Computing the Transformations

Since the range images contain the entire information of a 3D scan, it is possible to align a range image of the model with that of the current scene. Furthermore, if we know at least three correct correspondences between features in the scene and in the model, we can compute the 3D transformation between them (see Horn *et al.* [5]). To reject false feature matches, we use a method similar to GOODSAC [9]. This procedure is a deterministic variant of RANSAC [1], which also takes the quality of the matches (in our case the Euclidean distance between the descriptors) into account. This results in a performance increase, as shown in [9].

For simplicity of notation, we will refer to the elements of C as c, neglecting the indices i and j. Then, this procedure works as follows: A possible object pose, i.e., a transformation between the point cloud of the current scene and the point cloud of the model, is determined by three correspondences. Accordingly, the number of possible transformations is proportional to $|C|^3$. We limit the computational requirements of this approach by sorting the correspondences based on their descriptor distances and by considering only the best n correspondences. Let $C' = \{c'_1, \ldots, c'_n\}$ be the resulting ordered set.

Triples of correspondences are then selected from C' in the order defined by the loop structure of Algorithm 1. This generates the sequence (c'_1, c'_2, c'_3), (c'_1, c'_2, c'_4), (c'_1, c'_3, c'_4), (c'_2, c'_3, c'_4), $(c'_1, c'_2, c'_5), \ldots$. In this way, we obtain the best correspondences (according to the descriptor distance). Furthermore, the search will not get stuck if one of the correspondences with a low descriptor distance is a false match.

Algorithm 1 Nested loops for the selection of triples of correspondences.

```
FOR ($k = 3$; $k \leq |C|$; $k = k + 1$)
  FOR ($j = 2$; $j \leq k$; $j = j + 1$)
    FOR ($i = 1$; $i < j$; $i = i + 1$)
      $triple = (c_i, c_j, c_k)$
      ...
```

The current triple of correspondences $c'_a, c'_b, c'_c \in C'$ is then used to compute a candidate transformation T_{abc}. This candidate transformation now has to be validated. To this end, we apply the score function described in the following section.

C. Assigning a Score to the Candidate Matches

To compute the score of a transformation, we sample different views of the object and select a fixed number n (in our implementation, $n = 100$) of uniformly distributed 3D points from the resulting range images. In the remainder of this document, we will refer to these points as *validation points*. Note that this procedure is performed during model creation and does not need to be repeated during the matching process.

For each candidate transformation T_{abc} we consider the validation points sampled from the range image whose viewpoint is closest to the transformation. We then map the selected validation points into the scene by applying T_{abc}. For each transformed validation point of the model we consider the corresponding pixel in the scene.

If a candidate transformation is correct, the depth values of the original points in the range image of the scene $D^s = \{d_1^s, \ldots, d_n^s\}$ and the depth values of the transformed validation points $D^v = \{d_1^v, \ldots, d_n^v\}$ should be the same. Therefore, we choose the following scoring function for a single validation point:

$$s(d_i^v, d_i^s) = \begin{cases} d_i^v - d_i^s < -\varepsilon : & 0.0 \\ |d_i^v - d_i^s| < \varepsilon : & \frac{|d_i^v - d_i^s|}{\varepsilon} \\ d_i^v - d_i^s > \varepsilon : & -p \end{cases} \quad (1)$$

Here, ε is the maximal error value where the validation point can still be considered to be at the right place. If $d_i^v - d_i^s > \varepsilon$, the point in the scene is behind a point in the model and we give a negative reward of $-p$ to this situation (in our implementation $p = 10$). In other words we give a penalty of $-p$ to the points in the scene which should have been occluded by the matched object. If $d_i^v - d_i^s < -\varepsilon$ something was in front of the object and blocked the view. The score for the complete set of validation points has a value between 0.0 and 1.0 and is defined as

$$S(D^s, D^v) = \frac{\max(0.0, \sum_i s(d_i^v, d_i^s))}{n} \quad (2)$$

Once the score of the current transformation is computed, we repeat this procedure for the subsequent three correspondences. In our implementation, we stop the search when a maximum number of trials has been reached. Since our approach only requires three correspondences to calculate a potential alignment, it can handle partial occlusions. Our algorithm returns a possible position of an object in the scene for every transformation with a score above a certain threshold γ.

We further reduce the computational cost of this approach by discarding the triples of correspondences which seem to be invalid in advance. For example, we found it very effective to check if the metric distances between the three feature points in the scene are similar to the distances of the feature points in the model. Additionally, we do not use triples of features that lie on a line and use only features with a relative distance above a certain threshold.

D. Rejection of False Positives

Typically, the procedure described above returns many instances of an object at very similar locations. This results from different triples of correspondences, which all originate from the same object in the scene. Therefore, we prune the solutions by keeping only the ones with the highest score for all objects that are found in a local area. Then, we validate each of these solutions based on two criteria: the absence of collisions between neighboring solutions and the similarity of range images. If two nearby solutions are conflicting, the corresponding models transformed according to the computed transformations will collide. This collision check can be performed efficiently by using pre-computed low-resolution point clouds of the models stored in a *k*d-tree. If collisions are found, the solution with the lower score is rejected.

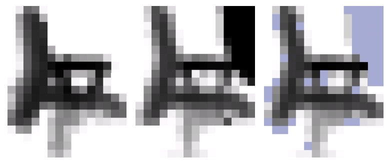
Fig. 4. Range image as it should look like according to the model (left) and actual part of the scene range image (middle). The black part on the top right of the image is clutter in front of the object. The right image shows the overlay of the other two images. The parts that are too different are marked in blue.

To reject remaining false positives, we calculate a low-resolution range image of the model given the virtual camera position described by the transformation. If the match is valid, the result should be similar to the corresponding part of the range image of the scene. We compare these two sub-images and determine another score value based on their similarity. This scheme is based on the same principle as scoring based on the validation points explained in Section III-C. The difference here is that instead of using previously sampled range images, we compute one from the viewpoint defined by the transformation. The score is then calculated by pixel-to-pixel comparison. This step is more expensive, since we have to compute a new range image from the model. However, we found that this leads to more reliable score values. Note that the computational overhead is minor, since this last validation step is performed only on a few potential object poses. Figure 4 illustrates the procedure described above.

IV. Optimal Parameters and Comparison with Spin Images

Our system relies on several parameters, the most important ones are related to the feature extraction: the resolution of the image (i.e., the size of the description vector), the size of the area around the point where the feature was extracted (the *support*), and the maximal descriptor distance $d_{\max}$ used to determine if a feature pair is a possible correspondence. To learn the optimal values for these parameters, we used a set of 30 3D scans with hand-labeled object positions and evaluated the object recognition system for different combinations of the parameters. For the evaluation of the parameters, we used a very low threshold γ to reject false matches ($\gamma = 0.3$ - see Section III-C). Having a small γ increases the number of the solutions reported by the first step of the algorithm. We furthermore disabled the rejection of false positives described in Section III-D. This allows us to measure the discriminative power of the features themselves.

For every combination of parameters we computed a score based on the number of true positives, false positives, and false negatives found in the test scenes. Since the false positive rejection described in Section III-D is very effective, we are mainly interested in the true positives and in the false negatives. False positives mainly affect the runtime. To take this into account, we introduced the following score function

$$\zeta = \frac{(\#\text{true positives})^2}{(\#\text{false negatives})^2 + (\#\text{false positives})}. \quad (3)$$

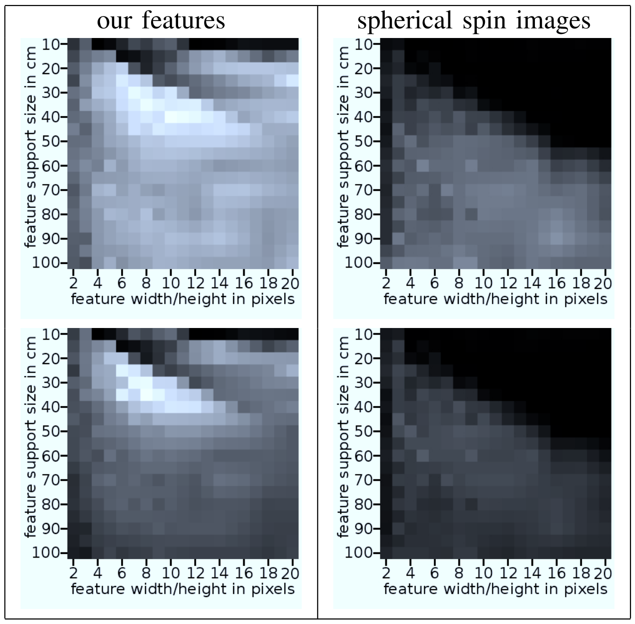

Fig. 5. Score according to Eq. 3 of different parameter sets when performing a search. Bright values correspond to high scores. The top row shows the results for different parameters when using our features (left) and the spherical spin images (right). The bottom row shows the score normalized by the runtime.

For each set of parameters we also measured the time required for the search.

We furthermore compared our features with the so called spherical spin images [10], a variant of the original spin images by Johnson [6], that simplifies the comparison between the description vectors to nearest neighbor. The runtime of this approach is similar to the one of our method. Our object detection method is independent of the specific features used and it only relies on the Euclidean distance between feature descriptors. In this experiment we simply replaced our features with the spin images. To learn the correlation between the spin image parameters and the performance of the system, we used the same procedure as for our features.

Figure 5 summarizes the result of these trials. It visualizes the score values for different combinations of the feature resolution and the support radius. The score used for the visualization was chosen as the maximum over different values for the descriptor distance threshold $d_{\max}$. Since we are interested in an object recognition system that can run online, we also took the runtime for the object search into account.

The influence of the parameters on the runtime is typically as follows. An increased descriptor distance threshold increases the runtime, since it leads to more candidate transformations. An increase of the support size makes the feature extraction phase more expensive, since a larger radius has to be considered for every feature. Increasing the descriptor size makes the feature more descriptive and therefore reduces the number of candidate transformations, leading to a reduced runtime. This effect is stronger than the runtime increase caused by the more expensive matching phase.

The bottom row of Figure 5 depicts the score values normalized by the runtime. By selecting the maximum value from the image, we can determine the optimal parameters according to our score function ζ. The optimal values for our features regarding score and runtime are descriptors of size 8x8 with a support width of 35 cm. For the spin images they are 5x5 and 60 cm. Over all, the spin images reached significantly lower maximal scores in this context, which confirms our choice of features.

V. Experimental Results

The approach described above has been implemented and tested using real world data recorded with a wheeled robot equipped with a SICK LMS laser range scanner mounted on a pan-tilt unit. The resolution of the range images extracted from the scans (100,000-150,000 points) was 0.4° per pixel, resulting in 450x225 pixels. Our platform requires about 15 seconds to acquire a full 3D scan. The models of the objects were created by joining 3D scans obtained from different view-points into a single point-cloud.

We tested our system with seven classes of objects: a cardboard box, a fan, a monitor, two types of chairs, a small pioneer robot and a table. For all the experiments, the average time to analyze a 3D scan and identify the objects was between 1.5 and 3.0 seconds on a 2.4 GHz Dual-core Pentium PC. The typical time requirements of the individual components were as follows: calculation of the range images: 100-120ms, extraction of the interest points (800-1300 points per scene): 150-250ms, calculation of the feature descriptors: 300-500ms, matching of the features: 250-450ms, search of the candidate transformations: 100-300ms scoring the candidate transformations using the validation points: 15-150ms; collision detection: 1-100ms; rejection of the false positives: 500-1400ms.

In the remainder of this section we analyze the performance of our system in a typical office environment. Subsequently we measure the variation of the success rate as a function of the distance to the corresponding object.

A. Success Rate in Cluttered Environments

In this experiment we measured the success rate of our approach in a cluttered office environment. The recognition procedure was tested on 20 3D scans. Each object was present exactly 10 times although sometimes only partly covered by the scan. A typical result of the object search can be seen in Fig. 6. The result of all scans is summarized in Table I. Whereas the rows represent the classes of objects which can be detected by our system, the columns represent the objects in the scene. Each entry contains the number of times an object in the scene (column) has been identified by the system as the object in the row. Since our system can detect multiple instances of objects, neither the row nor the columns sum up to the number of examples. The last column contains how often an object was recognized at places where no known object was actually present (in the clutter of the surrounding).

On average, an object contained in the scene was correctly matched in 84% of the cases. For the 20 scans and 7 objects,

TABLE I

STATISTICS ON THE DETECTION PERFORMANCE (B=BOX, F=FAN, M=MONITOR, O=OFFICE CHAIR, T=TABLE, W=WOODEN CHAIR, AND C=CLUTTER).

	B	F	M	O	P	T	W	C
B	8	0	3	0	0	0	0	4
F	0	7	0	0	0	0	0	0
M	2	2	10	1	0	0	1	4
O	0	1	0	8	0	0	2	0
P	0	0	0	0	10	0	0	3
T	0	0	0	0	0	7	0	1
W	0	2	0	3	0	0	9	0

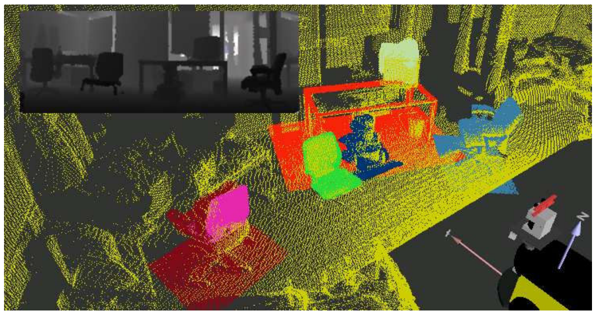

Fig. 6. Example of a 3D scan and the corresponding range image. The objects detected are marked in the scan. One instance of the monitor, the pioneer robot and the table were identified correctly. The office chair was present two times, which was also found correctly by the system. The wooden chair was also found correctly, but a second instance was wrongly matched on the office chair.

there was a total number of 29 false positives which means that the false positive rate is 20%. It turned out that this is often due to the high similarity of the considered objects. For example, the two chairs led to several failures since their shapes are similar. The box and the monitor also had a similar size and form which made them hard to distinguish. Additionally, several other objects in the scenes were box-like which led to a higher number of false positives.

B. Performance Depending on the Distance to Objects

In principle, our approach does not depend on the distance to the objects in the scene. However, due to the limited angular resolution of the scanner, the performance of our approach degrades with the distance to the observed object. To analyze this, we measured the performance of our algorithm as a function of the distance. We considered the same object classes as in the previous experiment. For every object, we took five scans from different viewpoints at a fixed distance from the object. Then we increased the distance and acquired five new scans. This procedure was repeated, until none of the objects was correctly identified anymore. The results of this experiment are illustrated in Fig. 7. It turned out that for distances under 5m without occlusions the success rate is about 89%. For distances larger 5m the success rate decreased and beyond 12m no successful detection was possible. Note that these values depend strongly on the size of the objects and the resolution of the sensor at hand. For example, at 12m the size of one of the chairs in the range image is only about 10x10 pixels. This leads to the situation,

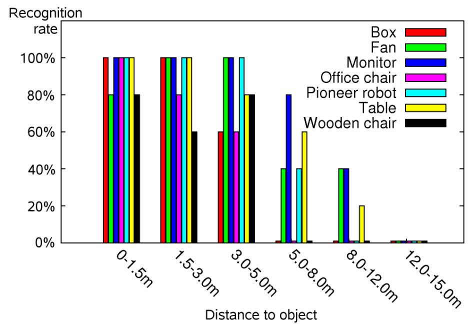

Fig. 7. Recognition rate depending on the distance to the object. For every distance step, five scans from different perspectives were taken.

in which an 8x8 pixels large feature descriptor is extrapolated from less than 3x3 pixels of the source image.

VI. CONCLUSIONS

In this paper we presented an efficient approach to object recognition in range scans. Our approach operates on features extracted from range images calculated from the range scans. It is highly efficient and requires only a few seconds to identify all objects in a given scene. The approach has been implemented and tested on data acquired with a laser range scanner installed on a pan/tilt unit. The experiments demonstrate that our approach can robustly identify the objects even if they are only partly visible. Our approach furthermore showed superior performance compared to spin images.

REFERENCES

[1] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.
[2] J. D. Foley, A. Van Dam, K Feiner, J.F. Hughes, and Phillips R.L. *Introduction to Computer Graphics*. Addison-Wesley, 1993.
[3] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann. Robust global registration. In *Proc. of the third Eurographics symposium on Geometry processing*, page 197, 2005.
[4] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.
[5] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America. A*, 4(4):629–642, Apr 1987.
[6] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(5):433–449, 1999.
[7] D.G. Lowe. Object recognition from local scale-invariant features. In *Proc. of the Int. Conf. on Computer Vision (ICCV)*, 1999.
[8] A. S. Mian, M. Bennamoun, and R. A. Owens. 3d recognition and segmentation of objects in cluttered scenes. In *WACV-MOTION '05: Proceedings of the Seventh IEEE Workshops on Application of Computer Vision*, pages 8–13, 2005.
[9] E. Michaelsen, W. von Hansen, M. Kirchhof, J. Meidow, and U. Stilla. Estimating the essential matrix: GOODSAC versus RANSAC. 2006.
[10] S. Ruiz-Correa, L. G. Shapiro, and M. Meila. A new signature-based method for efficient 3-d object recognition. In *CVPR (1)*, pages 769–776, 2001.
[11] S. Stiene, K. Lingemann, A. Nüchter, and J. Hertzberg. Contour-based object detection in range images. In *Proc. of the Third International Symposium on 3D Data Processing, Visualization, and Transmission*, pages 168–175, 2006.
[12] R. Triebel, R. Schmidt, O. M. Mozos, and W. Burgard. Instance-based AMN classification for improved object recognition in 2D and 3D laser raneg data. In *"Proc. of the International Joint Conference on Artificial Intelligence(IJCAI)"*, 2007.