# 3D Object Classification for Mobile Robots in Home-Environments Using Web-Data

Walter Wohlkinger, Markus Vincze
Vision4Robotics Group, Automation and Control Insitute
Vienna University of Technology, Austria
{ww, vm} @acin.tuwien.ac.at

***Abstract*—Building knowledge for robots can be tedious, especially if focused on object class recognition in home environments where hundreds of everyday-objects - some with a huge intra class variability - can be found. Object recognition and especially object class recognition is a key capability in home-robotics. Achieving deployable results from state-of-.the-art algorithms is not yet achievable when the number of classes increases and near real-time is the goal. Hence, we propose to exploit contextual knowledge by using sensor and hardware constraints from the robotics and home domains and show how to use the internet as a source for obtaining the required data for building a fast, vision based object categorization system for robotics. In this paper, we give an overview of the available constraints and advantages of using a robot to set priors for object classification and propose a system which covers automated model acquisition from the web, domain simulation, descriptor generation, 3D data processing from dense stereo and classification for a - not too far - robot scenario in an internet-connected home-environment. In this work we show that this system is capable of being used in home robotics in a fast and robust way for recognition of object classes commonly found in such environments, including but not limited to chairs and mugs. We also discuss challenges and missing pieces in the framework and useful extensions.**

***Index Terms*—dense stereo, 3D, disparity, support planes, cognitive, object class recognition, real-time, web, home-environment, mobile robot, shape matching, retrieval**

## I. Introduction

Object class recognition, especially in indoor environments, is an important task for mobile robotics applications as it paves the way for robots to operate successfully in home environments. Human machine interaction, robot object interaction, robot navigation and localization and mapping can greatly benefit from a system which is able to categorize objects in a home environment.

This paper addresses the problem of vision-based categorization of objects on a robot platform in home environments. We are particularly interested in categorizing types of furniture found in home environments. This is a challenging problem due to poorly-textured scenes with many wiry objects like chairs, occlusions and clutter and piecewise planar surfaces.

Laser scanners are widely used in robotics but most have insufficient resolution and power to detect narrow objects such as leg of chairs for example. The single scanline of the laser at a specific height makes its use for detecting furniture like tables challenging, as many legs are displaced towards the center of the object and horizontal layers may therefore function as invisible obstacles. The use of rotating lasers for full 3D capturing is not applicable in our scenario as the research goal is for low-cost passive sensors for use in home-environments.

To overcome the disadvantages of the environment and target categories, we decided to use a pure-vision based system for the task, namely dense stereo, and develop the system operation from the view of an embodied agent situated in the home environment. We start from the requirement to devise a framework that can be easily extended to new object classes. To this end we utilise the internet as source to obtain models for new objects. This has been attempted for appearance of objects in [11] where they showed a rather slow approach used laser scans and domain adaption. Alternatively, we propose to use only the perfect 3D data and transform it into sensor simulated data to cope with the typical problems of real applications such as only one view of an object (2.5D) with self-occlusion, incomplete models, aliasing effects and realistic noise levels froms tereo data. We then use these data to calculate object classifiers extending the 3D Harmonics descriptor [3] with the constraints from the robotics domain and match it against the database to find the nearest class to the object.

The paper is organized as follows. After discussing related work, Section III introduces the hardware configuration and discusses the corollary constraints and challenges. This is followed by the stage of object model collection and descriptor preparation in Section IV. The 3D data acquisition, filtering, support plane extraction and segmentation is presented in Section V to obtain the elementary parts for the recognition step. Section VI describes the descriptors and the matching for the object classification task. Finally, experimental results demonstrate our approach on a challenging dataset in Section VII and we discuss the open problems and propose ways to solve them. This leads to a short conclusion and an outlook on future improvements is presented in Section VIII.

## II. Related Work

Recent results on the Pascal visual object classes challenge 2009[1] show the state of the art in 2D object class recognition. In the classification challenge, where the goal is to predict

The research leading to these results has received funding from the European Community's Sixth Framework Programme (FP6/2003-2006) under grant agreement no FP6-2006-IST-6-045350 (robots@home).

[1]http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2009/

whether at least one object of a given class is present in an image, the top performer reaches 88.1% average precision (AP) on class aeroplane and 59.5% AP on chairs. In the detection challenge, where the goal is to predict the bounding boxes of all objects of a given class in an image, the results are far below the classification challenge. The best methods gets 47.8% on the aeroplane class and 15.0% on chairs. These results are produced using only imagery from flickr[2] as training input.

At the other side of the spectrum there is the work on 3D model retrieval in large databases. One state of the art search engine for 3D models is presented in [3] where they support 2D and 3D shape queries, are able to achieve more the 50% AP on living room chairs with their 3D Harmonics descriptor and are able to search the whole database in less than one second. The data they are working on is purely synthetic and the models are in high resolution with fine details and no noise.

Between these two extrema, there is the work on 2.5D imagery mostly taken from depth sensors like laser scanners, time-of-flight cameras or even stereo cameras. Object classification utilizing object part detection with a time-of-flight range camera was done in [5]. They use shape factors for describing components of a chair, where the object parts are tracked in their system as a single view contains not enough information. This approach was only shown on chairs, makes assumptions on the pose of the chair and is hard to apply to a wider variety of classes. [6] utilized a 3D laser scanner on a mobile robot for object classification using an Adaboost cascade of classifiers composed of several simple classifiers on 3D range and reflectance data. Recognition of objects in 3D LIDAR point clouds of an entire city was done by [8] using spin images [7] and other shape and contextual features.

Most of the systems using real world data make assumptions on the location of the objects, i.e., ground floor detection or detection of other support planes. The work most similar to our work is [10] where a mobile robot equipped with a tilting laser scanner extracts support planes, on which objects are assumed and a subsequent object recognition and classification step is performed on 2D high resolution images taken at these locations.

We also have to mention the Sharp 3D SHREC shape retrival contest [3] where the goal is to retrieve similar objects from a database given a range scan. The best method, which uses a SIFT-based approach on a grid spanned over the model, achieves about 50% AP on these high resolution, laser scanned objects. This contest is - to our knowledge - the thematically closest dataset available to test 2.5D object classification on.

## III. Robot Setup

### A. Hardware & Software

For our experiments we used our robot which consists of a MobileRobots Pioneer P3-DX[4] with an aluminum rack for mounting the stereo camera and the laptop. For navigation purpose only, we also use a hokuyo laser scanner. The stereo camera consists of two monochrome and one color usb camera modules UI-1226LE[5]. Stereo calculation is done with the two monochrome cameras with a baseline of 12cm, the centered color camera is used to overlay appearance if needed. Because the stereo camera is the main sensor modality of the robot, it is important to maximize the field of view: we use a 2.5mm lens, which gives us a usable resolution of 600x300 pixels after rectification. The stereo engine is a GPU implementation of the census transform[1] which has the advantage of freeing the CPU cores for 3D data processing in addition to image acquisition and robot control. As main computer a MacBookPro is used and software modules are linked together with the robot operating system ROS[6] from WillowGarage[7].

### B. Priors & World Knowledge

We now give an overview of constraints that we exploit as priors in our robotic object classification system.

*1) Scale:* Pure image based recognition and classification algorithms do not have access to the scale of objects in the image. Therefore they have to use multi-scale approaches that increase the computational load tremendously. Exploiting a calibrated stereo camera, the scale ambiguity is non-present and algorithmic complexity is reduced such as shown recently in [9]. In addition to the knowledge of the scale of data, we also know the dimensions of the object classes. The minimum and maximum dimension can be used in filtering and segmentation stages as additional domain knowledge. Finally, even if the dimension for an object class is not known, a robot system provides the possibility to interact with the user to learn this information or to explore the environment to obtain it from trials.

*2) Orientation:* Knowing the sensor-to-robot geometry and assuming regular robot movement regarding to the robot specifications, we can infer the orientation and location of the ground plane to within a certain accuracy. We use this information in the ground plane extraction step and constrain the ground plane to ”touch” the wheels of the robot. Not doing so indicates that the camera can not see the floor because the robot is in front of a table (which is a common situation in home robotics where the robot searches for objects on table and counters). Furthermore, for locating objects, it is a valid assumption that objects reside on almost horizontal regions, which significantly reduces the search space.

*3) Room and Object Class:* Another priory is to exploit contextual knowledge, for example, about the class of objects to search for. Rooms are likely to contain room-specific objects. As shown in [2] this information can be extracted automatically from the web and proves valuable in object search.

*4) Datasources:* A mobile robot equipped with sensors has the option to take images from different views. Additionally, different image characteristics can be exploited: monochrome,

[2]www.flickr.com
[3]http://www.itl.nist.gov/iad/vug/sharp/contest/2010/RangeScans/
[4]http://www.mobilerobots.com
[5]http://www.ids-imaging.de
[6]http://www.ros.org
[7]www.willowgarage.com

color, and 3D data. Finally, we noticed an advantageous side effect with using cheap wide angle lenses without an IR filter: this results in visually inferior imagery but black objects appear in shades of grey and 3D data can be calculated. For the "real" appearances of the objects, we need to use the centered color camera module.

## IV. Web-based Model Aquisition

Data collection for learning new object classes is a time consuming and sometimes tedious work if one tries to get a large number of classes and especially exhausting if the data is not standard imagery but 3D data which has to be created with special hardware. Therefore we propose another approach and tap the huge amount of freely available 3D models from the web.

### A. Web-Download

The input into our model acquisition system is the name of the new object class. With this keyword we search for 3D models on Google Warehouse[8]. Because some object classes have a huge intraclass variability, a large amount of training data is required. Classes such as "dining chair" have a high intraclass variability and need many exemplar models whereas classes such as "apple" only need one or two exemplars to be useful. Empirically we found that an upper bound of 60 models per class is a good choice, where this upper bound is not reached if there are not enough models available. As the 3D models from Google Warehouse come in a proprietary format, we first use Google Sketchup for automatic conversion into the open STL format.

### B. Domain Simulation

The next step is to adapt the models to the domain. The models from the web are full 3D models whereas the sensor input from the stereo cameras is 2.5D (data from one view of the object including self occlusion and possibly other occlusions, noise and erroneous data). In addition to that, the models from the web are so called "polygon soups": a loose bunch of triangles with no ordering, with holes, and artefacts. Models have no scale, no orientation, and the level of detail can range from very low (hundreds of triangles) to very high (hundred thousands of triangles). The data from the robot on the other side consists only of (noisy) data points.

To use the models from the web, we generate synthetic 2.5D models by rendering and sampling the 3D models from views around the model. We use 45 degree steps in elevation and azimuth which results in 24 views for a model (Figure 1). These 24 views are sufficiently dense for the type of descriptor used to interpolate between views. To discard details and therefore improve generalization of the models, we downscale the models by rendering the models in 100x100 pixel images which gives about 5000 data points for a model which fills the rendering window to 50%. Figure 2 shows the 2.5D partial point clouds created for one model from the chair and mug classes. The most useful information in these figures is the fact that the same object can have a completely different shape and therefore a different representation when seen from different viewpoints. Because the orientation of the 3D model downloaded is unknown, we have to render views equally spaced around the model, despite our prior from the real world where chairs for example are mostly seen standing on their legs. To obtain a generic method that extends to arbitrary objects we do not include special priors for this case. Finally, for every of the 24 views of the model a 3D descriptor is calculated and stored into a database (please see Section VI).

[8]http://sketchup.google.com/3dwarehouse/

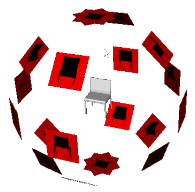

Fig. 1. Model rendering from 24 views at 45 steps in elevation and azimuth.

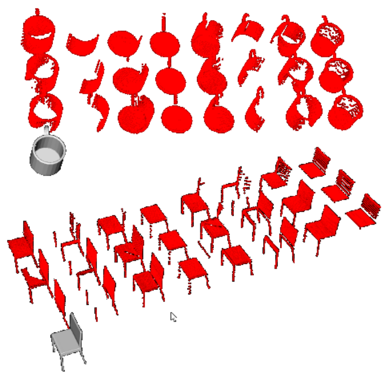

Fig. 2. The resulting 24 2.5D point-clouds generated from the web-downloaded 3D model of a mug and a chair.

### C. Discussion

This web-based model learning is an easy method for teaching the robot new objects and object classes and provides a user-friendly interface where only the name has to be fed into the system, which can easily be done by audio or keyboard input. The system can handle large intraclass variability, is extensible and scalable and does not need user interaction despite the new class names.

## V. Semantic Data Partitioning

An overview of the steps in the processing chain is given in Figure 3 and explained in the following sections. Our GPU-based dense stereo system delivers disparity maps from which we calculate the initial 3D point cloud. The left rectified image and the corresponding disparity map are shown in Figure 4.

### A. Filtering

Because data from the stereo system includes outliers we have to employ outlier-filtering of the 3D data prior to further processing. To eliminate sparse points we use a box filter where

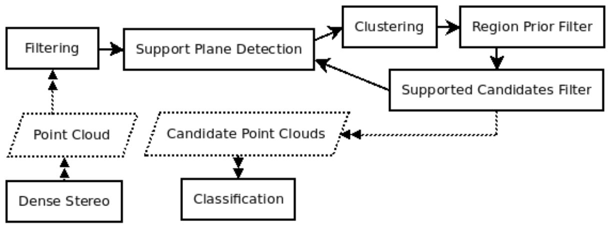

Fig. 3. Processing steps for extraction of candidate point clouds.

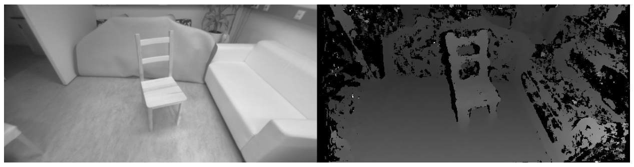

Fig. 4. Left rectified input image and corresponding disparity map.

the size of the filter is twice the mean point distance in the point cloud. The filter significantly reduces sparse points on the object boundaries and wrong matches coming from the stereo engine which can be seen in Figure 6.

## B. Support Plane Detection

In the first iteration of the data partitioning the ground floor is the basic and elementary support plane. We apply a RANSAC-based plane fitting procedure exploiting the known robot geometry, i.e., the camera pose wrt. the wheels, and we use the two points where the wheels touch the ground as two of the three points in the plane candidate search. This gives more robust results in cases where the ground plane is not the dominant plane in the scene. Figure 5 shows the initial 3D point cloud with the floor detected and coloured in red.

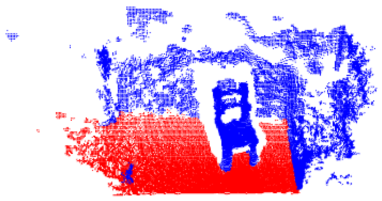

Fig. 5. Floor detection result.

## C. Clustering

The next step is to obtain groups of data points that may indicate individual objects. To this end we use clustering based on flood filling on the remaining points above the ground plane. The size of the filter kernel depends on the size of the smallest object we are looking for and the minimal allowed distance between two objects. As we already have the object classes we can extract these parameters from the database. For the experiments, the kernel diameter is set to 5cm. Results of this procedure can be seen in Figure 6 where the clusters are encoded with different colours.

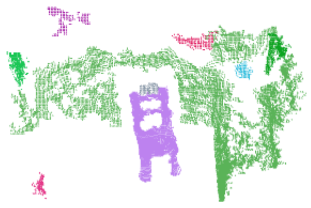

Fig. 6. Point cloud clusters.

## D. Region Prior Filter

One prior is the assumption that objects are located only on support planes. Hence, an efficient way of reducing the candidate clusters for object classification is to reject all clusters outside the support region. To obtain the boundary for the support plane, the 2D convex hull is calculated and the projected point clusters are tested against the plane polygon using the odd-even test. The support plane boundary is shown in red in Figure 7 with the projected clusters in green. Only clusters inside this support plane are valid candidates for the next processing step.

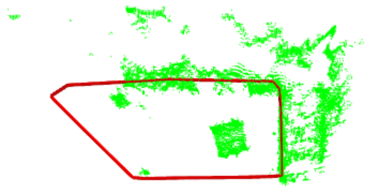

Fig. 7. Support plane boundary.

## E. Supported Candidates Filter

The ”object on support plane” assumption also implies that the candidate objects are actually ”on” the support plane. We therefore only process pointclusters which are attached to their support plane. The resulting object candidates for the classification produced by the data quisition chain are shown in red in Figure 8

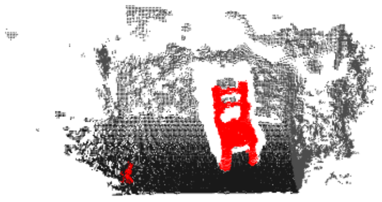

Fig. 8. Candidate objects for classification.

## F. Play it again, Sam

Up to this point, the only support plane extracted from the data is the ground floor. For finding objects on further support planes – such as objects on tables and counters – the RANSAC-based support plane detection and the associated clustering, region prior and candidate filtering stages are performed again on the clusters. The object candidates are not modified for the classification stage, so objects residing on chairs are included in the chair descriptor in the first place and are detected separately in the second stage. Altering point clouds – i.e., removing parts or objects – prior to their classification is not feasible, because the object class is not know yet.

## VI. Classification

The goal of classification is to find the correct class label for a given data cluster. This can also be seen as finding the most similar object to the query data and assigning the label of the most similar match. Finding similar objects - especially 3D models in large databases - with efficient and robust algorithms has attracted a good amount of researchers over the last years. Following the proposed robotics approach to object classification, we want to stress the following properties and differences to classical approaches [].

### A. 3D Descriptor Properties

The biggest challenge in 3D shape matching is the fact that objects should be considered to be the same if they differ by a similarity transformation. To explicitly search over the whole space of transformations is impracticable because efficiency is a key property a retrieval algorithm should have. Hence, the similarity metric must implicitly provide the similarity at the optimal alignment of the two models.

*1) Normalization:* This can be achieved in a normalization step prior to the similarity calculation for each model where translation, rotation and scale are normalized, i.e., a canonical frame is computed.

Normalization for translation is usually done by translation of the center of mass of the model into the coordinate origin. Normalization for scale can be done by scaling the points by the maximum distance from the origin or by the average distance of the points to the center of mass which provides more robustness when the data has outliers.

Normalization for rotation is less robust than normalization for translation and scale and heavily depends on the object and object classes as illustrated in [3]. Normalization for rotation is often done by using the principal axes calculated from the eigenvectors of the covariance matrix for rotating the object into a common frame. Its already problematic on 3D data, extracting principal components from 2.5D data will give incorrect results due to the missing one-view data.

*2) Invariance:* To avoid the imperfections of a prior normalization step, the alternative is to design the descriptor in a transformation invariant fashion. This means that the descriptors produce the best similarity measure for any transformation. Because of all the difficulties with normalization, we propose to apply an invariant descriptor.

*3) Spherical Harmonics Descriptor:* The spherical harmoncs descriptor[4] is a affine invariant descriptor which is calculated from a 64x64x64 voxel grid. The calculation of the descriptor from a coarse voxel grid gives us the needed generalization for object class recognition. The descriptor is represented and stored as a 2D histogram in the database which enables us to efficiently calculate the k-nearest neighbours using the Euclidean distance between the query and all entries in the database.

### B. Bounding Boxes

Using the real world size of objects, we can apply a prefiltering stage for sorting out object classes which are way to big or small to be the searched object class. This not only increases the performance in terms of precision/recall but also increases the computational performance as only a small part of the database has to be matched against the query, which can be a huge speed-up when the number of classes increases.

## VII. Results

Tests are shown on datasets collected in our lab on chairs and on a table scene.

The chair classes in our system are ”dining chair”, ”office chair” and ”arm chair”. For chairs, the system is able to produce classification results useful for robotic purposes. Our test-database consists of 119 point clouds where 47 chairs are present. The segmentation stage was able to successfully extract 30 of the 47 chairs from the scene. A representative sample of testscenes with chairs in arbitrary position and distractor objects can be seen in Figure 10. Figure 9 shows the precision recall curve for the diningchair class. The system only detected 13 instances of dining chairs correctly. The reason for this is that some of the chairs differ a lot from the chairs we downloaded from the web and that some of the chairs were classified as office- or armchair as these classes share common similarities. Figure 9 shows the precision recall curve when testing against the metaclass of chairs, consisting of dining-, office- and armchairs. All tests are performed using the boundingbox size of the point clouds for a coarse prefiltering of possible classes which increases the performance of the system as the point clouds alone provide - depending on the object class - too little information to produce robust and reliable results. The decrease in performance is only marginal for chairs as to be seen in Figure 9 as they provide enough structural shape to differ from other object classes. This is no longer true for primitive shaped objects like bottles or mugs as the only distinction between a round paper basket and a mug - when seen from a certain viewpoint - is their size, despite their common location. This can be seen in Figure 11 where there is just not enought data for distinguishing between some small objects. Nevertheless most of the mugs on the table were correctly identified.

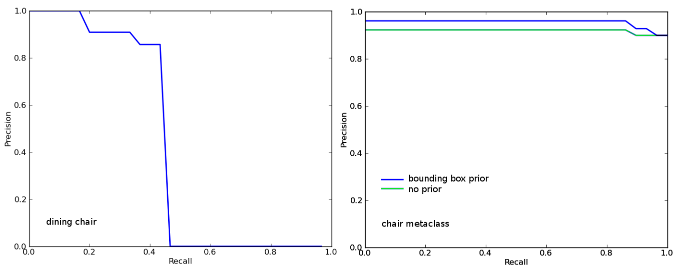

Fig. 9. Left:Precision recall curve for the diningchair class. Right:Precision recall curve for the metaclass cair. The blue curve shows the performance of the spherical harmonics despcriptor with a boundingbox prior, green without.

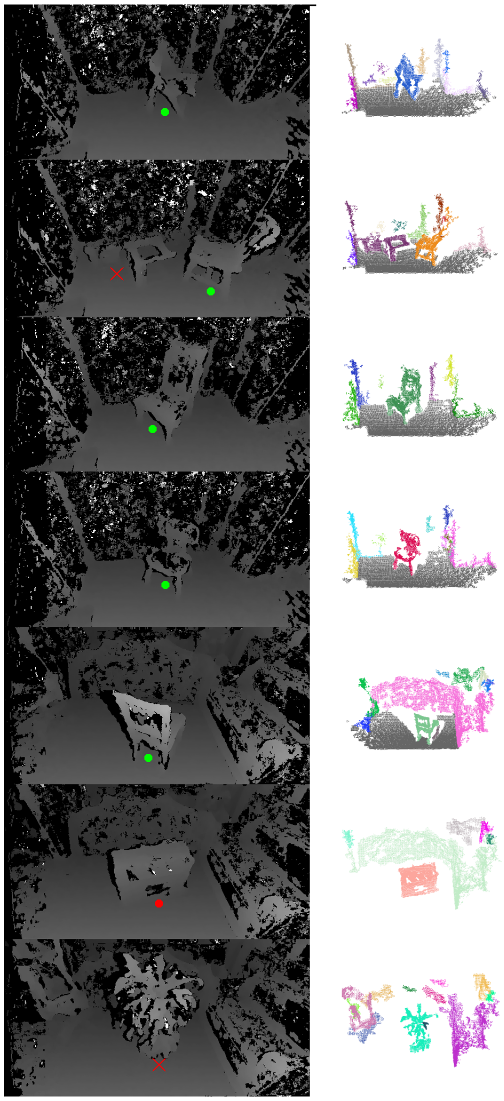

Fig. 10. Testscenes with chairs and distractor objects. Green dots indicate correct class, red dot indicate false positive and red cross indicates not detected as belonging to class chair. Image 2:Chair lying on the ground could not be classified due to unsufficient segmentation; Image 6 the box was wrongly classified as belonging to the class chair.

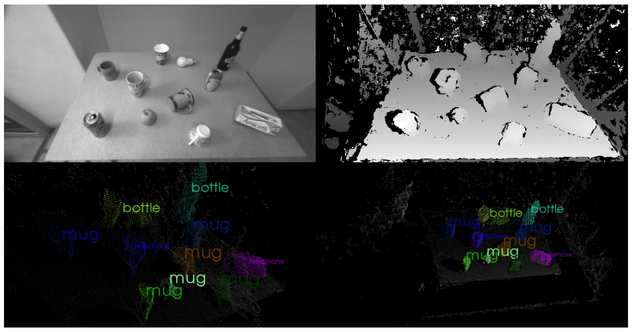

Fig. 11. A tablescene with instances of classes mugs, cans, bottles, apple and light bulb.

The presented system heavily depends on good and complete data. Without sufficient data the support planes can not be extracted and the clustering of the object will produce multiple fragments which renders the classification with global desciptors useless.

## VIII. Conclusion & Future Work

We have shown a robot system which is able to categorize objects in a home environment with vision only sensors. The set-up uses a stereo-camera as only sensor and uses the dense 3D data as primary features. The system can perform scene segmentation and object class recognition on a wide variety of classes including furniture like chairs and objects often found in office and home-environments like mugs and bottles. We have shown that this approach is feasible to work under real conditions and gives some promising results.

The future system improvements are twofold: On the one side we will port the algorithm from the existing Python implementation to C++ to gain a speed-up which allows us to perform object class recognition in shorter intervals and enables a more smooth robot-user interaction. On the other side, to overcome the deficiencies of using a local stereo algorithm, we plan to switch to a global dense stereo system with more dense and boundary preserving 3D data, which enables us to detect our support planes in a more stable manner. We also hope to use the improved data for door detection. To better disambiguate better between the classes, the spatial arrangement of these objects with respect to the map or to other objects will be incorporated in future work.

## References

[1] M. Weber, M. Hummenberger, W. Kubinger, ”A Very Fast Census-Based Stereo Matching Implementation on a Graphics Processing Unit”, *12th International Conference on Computer Vision Workshops, Kyoto, 2009*

[2] P. Viswanathan, D. Meger, T. Southey, J. J. Little, and A. K. Mackworth, ”Automated Spatial-Semantic Modeling with Applications to Place Labeling and Informed Search”, *Canadian Robot Vision, Kelowna, Canada, 2009*

[3] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin and D. Jacobs, ”A Search Engine for 3D Models”, *ACM Transactions on Graphics, January 2003*

[4] M. Kazhdan, T. Funkhouser, S. Rusinkiewicz, ”Rotation Invariant Spherical Harmonic Representation of 3D Shape Descriptors”, *Symposium on Geometry Processing, Germany, June 2003*

[5] S. Gaechter, A. Harati, R. Siegwart, ”Incremental Object Part Detection toward Object Classification in a Sequence of Noisy Range Images”, *International Conference on Robotics and Automation, Pasadena, 2008*

[6] object class A. Nuechter, H. Surmann, J. Hertzberg, ”Automatic Classification of Objects in 3D Laser Range Scans”, *Intelligent Autonomous Systems, Amsterdam, 2004*

[7] A. Johnson, M. Hebert, ”Using spin images for efficient object recognition in cluttered 3D scenes”, *IEEE Transactions on PAMI, 1999*

[8] A. Golovinskiy, V.G. Kim, T. Funkhouser, ”Shape-based Recognition of 3D Point Clouds in Urban Environments”, *International Conference on Computer Vision , September, 2009*

[9] S. Helmer, D. Lowe, ”Using Stereo for Object Recognition”, *International Conference on Robotics and Automation, Anchorage, May, 2010*

[10] D. Meger et al., ”Curious George: An Integrated Visual Search Platform”,*Canadian Robot Vision Conference, 2010.*

[11] K. Lai, D. Fox, ”3D Laser Scan Classification Using Web Data and Domain Adaption”, *Robotics Science and Systems, 2009*