SEVENTH FRAMEWORK PROGRAMME

ICT - Information and Communication Technologies

HOBBIT
The Mutual Care Robot

Collaborative project

Deliverable 6.2

Safe and autonomous robot navigation in AAL-homes (publication)

Organisation name of lead partner: **TU WIEN**

Proposal/Contract no.: **288146**

Title: HOBBIT - The Mutual Care Robot

Acronym: HOBBIT

Grant Agreement Number: 288146

Deliverable	**D6.2 Safe and autonomous robot navigation in AAL-homes (publication)**
Revision	
Associated WP	WP6 Robot Navigation and Manipulation
Associated Task	T6.1: Map building and self-localisation T6.2: Mobility in unstructured environments T6.3: Fine positioning T6.4: Object detection T6.5: Integration of "Fetch & Carry"
Due Date	31.1.2014
Date Delivered	31.1.2014
Lead Partner	TU WIEN
Partners Involved	TU WIEN, FORTH
Authors	TU WIEN (Paloma de la Puente, David Fischinger, Daniel Wolf, Markus Bajones, Markus Vincze), FORTH (Antonis Argyros)
Dissemination Level	PU
Abstract	The purpose of this Deliverable is to report on the development of the navigation system for upcoming use in the HOBBIT PT2 mobile platform. The main focus hereby lies on the visual perception for map building and self-localisation (T6.1) and for mobility in unstructured environments and accurate positioning (T6.2 and T6.3). We report the findings on advanced navigation, fine positioning and object detection as used in PT2 are summarized in a scientific paper. We also report on progress in T6.4 to obtain object detection and T6.5 for the integrated Fetch & Carry scenario.

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 288146.

Table of Contents

List of Figures

1 Executive Summary

The purpose of this Deliverable – D6.2 Safe and autonomous robot navigation in AAL-homes (publication) – is to present the advancements regarding navigation, fine positioning and object detection as it will be used in PT2 in the form of a paper. To relate the work performed to the work outlined for WP6 in the DoW, we give a summary of the intended tasks and how they have been achieved and are presented in the paper.

The tasks the PT2 HOBBIT robot needs to fulfil are map building, self-localisation and safe navigation as well as detecting objects within search, fetch and carry scenarios. The perception system shall be suitable for domestic environments, which are in general less structured and controlled than for instance industrial or hospital settings. To establish the link to the DoW, we repeat here the descriptions of the relevant Tasks in an abbreviated form to stress the key points related to this Deliverable. The focus of the Deliverable is on the individual components from Tasks 6.1 to 6.4. The upcoming deliverable D6.3 will them focus on the integration towards the Fetch & Carry scenario of Task 6.5:

Task T6.1: Map building and self-localisation
The purpose of this Task concerns the development of map building, update methods and self-localisation. Of particular interest to HOBBIT users is to implement a system that is reliable yet cost effective. To this end the present methods of laser mapping and localisation shall be replaced by novel RGB-D sensors such as the Kinect. Hence, we need to compare existing tools for allowing this extension and develop what is missing to improve performance towards the first prototypes. The idea is to start with the navigation methods available in ROS – Robot Operating System – and then add the functionalities required to approach reliable mapping and localisation.

Task T6.2: Mobility in unstructured environments
The focus of this Task is to make the step from an industrial, office, and museum-like environment, to the home environment. Today lasers only see one plane, while in cluttered home settings all height information up to the height of the robot needs to be regarded to enable safe navigation. Hence, we need to design the perception system such that it can cope with mapping, localisation as well as safe navigation in home settings.

Task T6.3: Fine positioning
Because of the physical design of the HOBBIT robot (by WP7) it is crucial to guarantee proper fine positioning for all operations concerning object manipulation. The resulting solutions will be also the basis for the developments in WP5 and WP7 (recharge procedure). To achieve this, we will improve and refine the mechanisms delivered by Task T6.2. This includes accurate localisation relative to objects (see Task T6.4) for pick up from the floor or grasping from various locations on tables or sofas.

Task T6.4: Object detection
The users want the platform to search for objects as well as to grasp and bring them. Hence, we need to design the perception system such that this functionality can be provided. We build on experience with object class detection using RGB-D sensors as

proposed for the tasks above. Further along the project we then need to learn and detect the newly learned objects in search and grasp tasks.

Task T6.5: Integration of "Fetch & Carry"
This Task signifies the fusion of the single mechanisms developed in Tasks T6.1 to T6.4 to provide the functionality for the "Fetch & Carry" operations.

To solve the technical issues described in Tasks 6.1 to 6.3, a combined map building, localisation, navigation and fine positioning method is needed. It is important that all these aspects play together in an adequate way. This will also be relevant for Task 6.4, where the platform needs to position the robot arm within a certain range of the object for the pick-up operation. Finally, this is also an important requirement to fulfil the integrated Fetch & Carry tasks in Task 6.5. Specifically, the technical function provided here will provide solutions for the user requirements, where the scenario numbers refer to the table of scenarios in D1.6:

- Call HOBBIT (scenario I),
- Patrolling as part of Emergency (scenarios II),
- Guiding the user in the safety scenario (scenario III),
- Moving to an object as part of the Pick-up and clear floor commands (scenarios IV and V),
- Searching for an object as part of the Bring me command (scenario VII),
- Following the user as part of the Transport command (scenario XI), and
- Moving to the recharging station (scenario XII).

For achieving a combined approach to map building, localisation, navigation and fine positioning we built on known ROS modules, analysed existing solutions, and developed an approach that overcome existing problems.
We present these developments in a paper to document the scientific advance also to the robotics community. The paper is included in this deliverable as Appendix.
In the following we highlight how the developments contribute to solving the issues raised for in the respect tasks of WP6. Section 2 summarises work towards mapping and localisation; Section 3 covers navigation, which is in PT2 highly improved such that Section 4 about Fine positioning is not separate but integrated into the general approach.
Section 5 presents work on object detection and results of object search using the PT1 platform. Section 6 presents first work towards the "Fetch & Carry" scenario as the main integration scenario of the above functions. It will be used in different aspects to provide the functionality for the scenarios outlined in D1.6.

2 Map Building and Self Localisation (Task 6.1)

Regarding map building and localisation, the tasks for PT2 will be to build the map, to mark places that are of interest to the user, and to then provide means for autonomous navigation between any of these places.

For map building and localisation we use the bottom RGB-D camera of the two-camera set-up as presented in D6.1.

For PT2 operation in a new user's apartment, the procedure will work as follows. The robot will be shown around the environment so that an accurate and complete map can be built. This task will be carried out in tele-operation mode by the facilitator. A metric map is built by using the ROS package of the Simultaneous Localization and Mapping algorithm GMapping[1]. When building the map, it is important that there are no dynamic obstacles moving around the apartment and that clutter is removed so that a stable representation of the environment is achieved. Visual feedback to the facilitator from the mapping process allows to obtain a satisfactory final map. New maps from larger environments including corridors - a key challenge in robotic mapping[2][3] - were built for test purposes and the results were good for localisation. This was tested on the two prototypes of the Hobbit robot available at TUW. Fig. 1 shows examples of the mapping results.

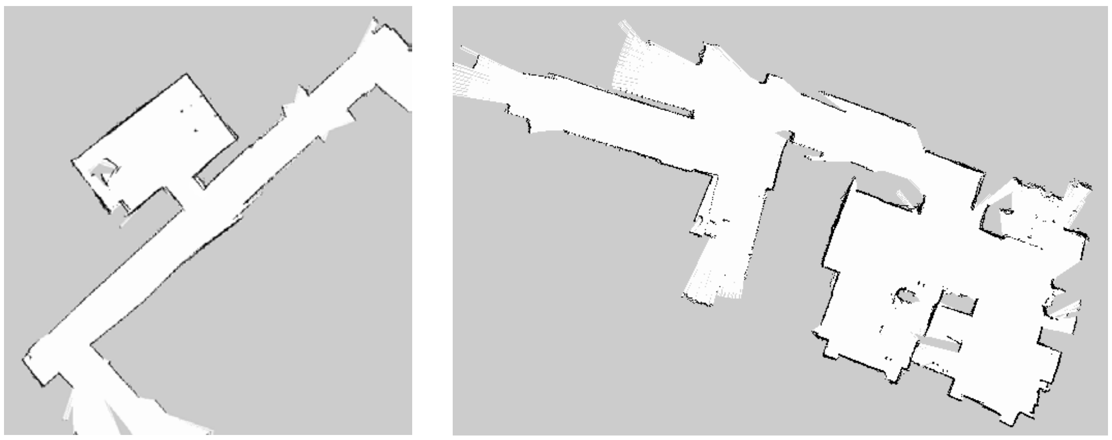

Figure 1: New maps built with the two Hobbit prototypes PT1 at TUW, including long corridors.

Once the metric map is built and saved, it is possible to open it with a tool editor (based on the Qt cross-platform application framework[4]) developed so as to add room labels. An example and more details are given in the paper in the Appendix. The result is the association of places to rooms, which is obtained automatically, and is now available for navigation.

[1] G. Grisetti, C. Stachniss, and W. Burgard: Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters, IEEE Transactions on Robotics, Volume 23, pages 34-46, 2007

[2] P. de la Puente. Probabilistic mapping with mobile robots in structured environments. PhD dissertation. Universidad Politécnica de Madrid. ETSI Industriales. December 2012.

[3] http://answers.ros.org/question/46996/

[4] M. Dalheimer (January 2002). "Programming with Qt" (2nd ed.). O'Reilly Media. ISBN 978-0-596-00064-6

As a next step, the facilitator will add to the map places from where the robot will look for the user. A default searching position is required for every room. This default position will be preferably located at the centre of the room and it must allow for 360° rotations of Hobbit. Other searching positions complying with this condition may be added as well.

Finally, the facilitator will also specify places for searching objects. Search locations must ensure that the robot covers in a search procedure all surfaces where objects could be placed and that these surfaces are within the optimal search range where objects are neither too small nor too large to be recognised. For an example of a search see Section 6.

With these results we achieve robust map building and localisation. Besides porting the methods to PT2, the next step is to show that the methods also work in user homes, which is planned to verify during pilot tests.

3 Mobility in Unstructured Environments (Task T6.2)

The PT2 platform will have to autonomously navigate in homes. Autonomous navigation performs the motion between any of the specified places as outlined in the previous Section. Depending on the scenario, the place information is provided either directly by the user (the robot asks first for a room and then a place in this room) or may be produced by the system, e.g., exploiting the pre-defined search places or automatically accessing all the defined places in a room.

Regarding the low level navigation, we have to cope with a non-holonomic platform. The consequence is that each method needs to be accurately adjusted to the specific kinematic drive properties of the very platform. This can be achieved by tuning the respective navigation parameters and it requires thoroughly testing the resulting platform motions. Typical problems that are encountered with standard methods are related to the fact that a non-holonomic platform has a limited driving and curve range and difficulties for finding good parameter compromising performance in the existing solutions, especially for navigation in narrow spaces. We are not the first ones to notice these problems[5][6] yet standard solutions do not yet exist.

The paper in the Appendix summarises what has been done to find an optimal set of parameters for navigation in homes.

Along the work, a serious bug of the default global planner provided by ROS was discovered. The consequence is that sometimes there are global plans passing through walls, obstacles and unknown space in the map. We found out that other people also had problems with this issue and that Hydro, the most recent version of ROS, should resolve this problem. Before migrating the whole system to Hydro - which is planned for the end of the year (WP 7) - several adjustments were made to reduce this problem. For the solution a compromise is needed so that it is possible to allow for a limited localization error while making sure that no static obstacles are ever ignored in the path planning. The short range blind area of the Kinect sensor makes the problem more noticeable. Avoiding planning through unknown

[5] http://answers.ros.org/questions/

[6] http://code.ros.org/lurker/message/20100904.071241.569a5c7f.en.html

areas was simpler by means of a given parameter and it should prevent the problems to occur and this will be tested next. Figures 2 shows two cases in which the global path provided by the navfn algorithm available in ROS suffers from this kind of problem.

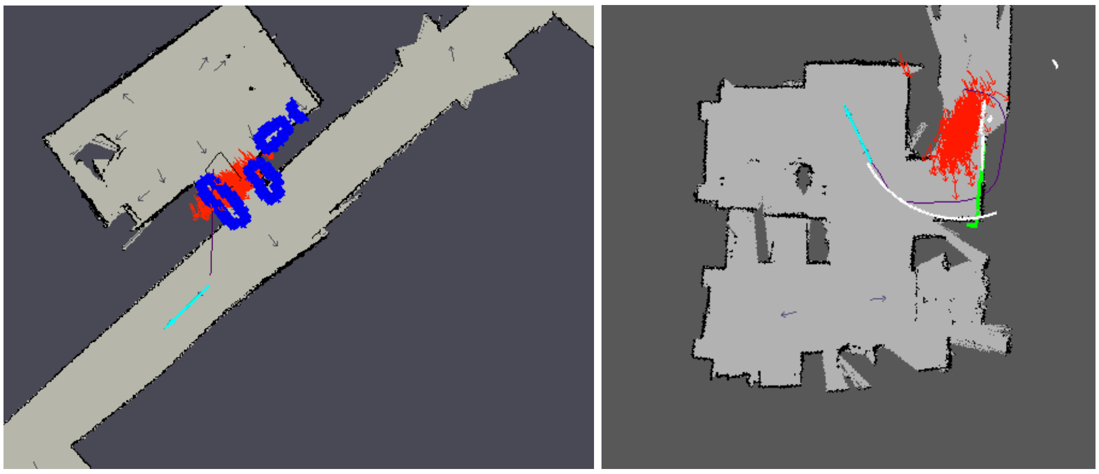
Figure 2: Global paths through walls and unknown space observed in our tests in two different environments.

Navigation can now be used and tested in further scenarios. It is possible to retrieve the paths and traveling distances to all the places from the path planner, without the robot needing to actually go there. This service was successfully used to implement an efficient search procedure for the *locate user* functionality presented in the scenario description in D1.6. The robot is now able to autonomously look for the user from all the search positions defined for each room, employing the distance criterion and previous knowledge about the last room where the user was detected if available. This functionality was implemented by means of the ROS SMACH architecture for creating complex robot behaviour[7] (WP 3). Error cases like the path planning failing to obtain a path and even better navigation in narrow spaces will be future work before full integration into the PT2 platform.

The results regarding navigation indicate that the challenges in user homes should be overcome. Next we will port methods to PT2 and then show that the methods also work in user homes, which is planned to verify during pilot tests.

4 Fine Positioning (Task T6.3)

For PT1 we had to specifically consider small platform motions, since the basic navigation framework did not work accurately enough. Also previous experience pointed this way, such that in the DoW, this task was added.
However, with the navigation method in place as described in the previous Section, the robot reaches all places within its map with sufficient accurate such that no specific fine positioning measures are necessary any longer. The work planned for this task was effectively used to obtain a more generic and complete accurate navigation method rather than an additional fine positioning method.

[7] http://wiki.ros.org/smach

5 Object Detection (Task T6.4)

Object detection is one of the core abilities for a robot in an object search and deliver scenario. Challenges of finding everyday life objects are the enormous number of different types and conditions in home environment. In deliverable D1.6 we summarized object and environmental parameter which make robust and reliable object search feasible. In the following paragraphs we explain newly developed and adapted approaches for object learning, recognition and grasping proposed for the PT2.

In the spirit of the MUC concept, learning of objects is interactive. The PT2 robot navigates to a location close to the user, the robot grasps the turntable located at its body (see Figure 3) and presents the turntable to the user in a position such that it can be reached conveniently. Then HOBBIT asks the user to place an object on the turntable and starts learning the object. We designed a squared turntable which enables robust and accurate camera pose tracking (see Figure 6.6, left). Hence, any kind of object regardless of its texture or shape can be learned.

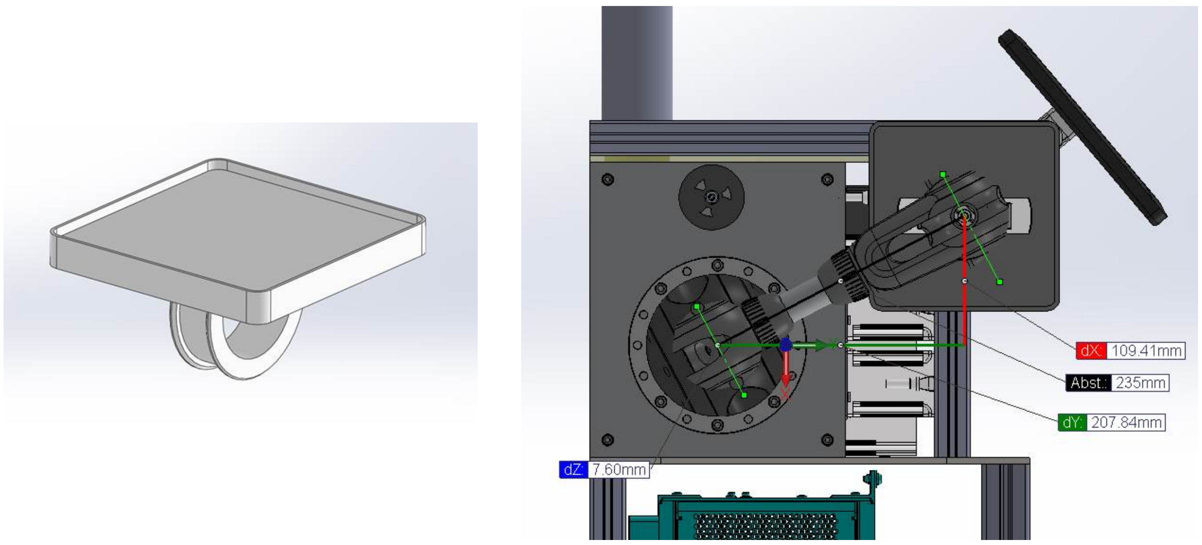

Figure 3: Squared turn table for object learning (left) and the robot grasping the turntable located at the top right of its body (right).

Given the positions on the turntable, object learning operates as follows. First RGB-D images are captured and the camera pose is tracked with respect to the region of interest (ROI) covering the object and the squared turn table. Two algorithms, namely an image keypoint based pose tracking pipeline and an Iterative Closest Point (ICP) approach are implemented to estimate the camera motion. Both algorithms are state of the art and allow robust camera pose tracking. The decision which one will finally be used is postponed and depends on the robustness and real time performance. Additionally, we implemented the non-linear pose optimization proposed by Fantoni et al.[8] which compensates the drift. A final filtering step

[8] Fantoni, S. and Castellani, U. and Fusiello, A., "Accurate and Automatic Alignment of Range Surfaces," 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012

using a weighted voxel grid inspired by KinectFusion[9] is used to sub-sample and smooth the reconstructed object point cloud. Results of the first tests are shown in Figure 4.

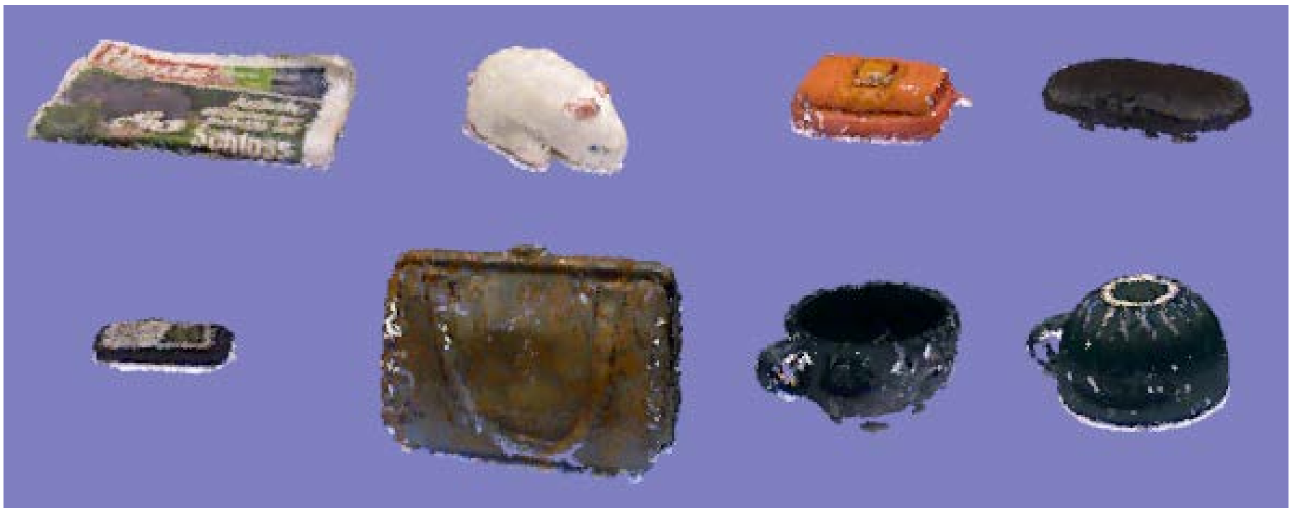

Figure 4: Reconstructed objects used for learning object detection models.

For object detection we integrated an effective algorithm developed in our previous work[10]. The proposed method is based on a combination of different recognition pipelines, each exploiting the data in a diverse manner and generating object hypotheses that are ultimately fused together in a Hypothesis Verification stage[11] that globally enforces geometric consistency between model hypotheses and the scene. Such a scheme boosts the overall recognition performance as it enhances the strength of the different recognition pipelines (Figure 5) while diminishing the impact of their specific weaknesses. Specifically, the currently implemented pipelines take advantage of the multi-modality of the data:

- A semi-global 3D descriptor representing an extension of the OUR-CVFH approach[12] based on the colour, shape and object size cues. Regarding the segmentation stage required by the semi-global pipeline, we use the strategy recently proposed in our previous work[13].

[9] Izadi, Shahram and Kim, David and Hilliges, Otmar and Molyneaux, David and Newcombe, Richard and Kohli, Pushmeet and Shotton, Jamie and Hodges, Steve and Freeman, Dustin and Davison, Andrew and Fitzgibbon, Andrew, “KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera,” in 24th Annual ACM Symposium on User Interface Software and Technology, Santa Barbara, California, USA, 2011.

[10] A. Aldoma Buchaca, F. Tombari, J. Prankl, A. Richtsfeld, L. di Stefano, M. Vincze, "Multimodal Cue Integration through Hypotheses Verification for RGB-D Object Recognition and 6DOF Pose Estimation," in IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany; 2013.

[11] A. Aldoma, F. Tombari, L. Di Stefano, and M. Vincze, “A global hypothesis verification method for 3d object recognition,” in European Conference on Computer Vision (ECCV), 2012.

[12] A. Aldoma, F. Tombari, R. Rusu, and M. Vincze, “Our-cvfh: Oriented, unique and repeatable clustered viewpoint feature histogram for object recognition and 6dof pose estimation,” in Joint DAGM-OAGM Pattern Recognition Symposium, 2012.

[13] A. Richtsfeld, T. Mörwald, J. Prankl, M. Zillich, and M. Vincze, “Segmentation of Unknown Objects in Indoor Environments,” in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2012.

- A 2D local descriptor (SIFT[14]) which is able to generate object hypotheses with associated 6 DOF pose by back-projection of the 2D keypoint locations into the 3D space.
- A 3D local descriptor (SHOT[15]) aimed at establishing correspondences between model and scene surface patches.

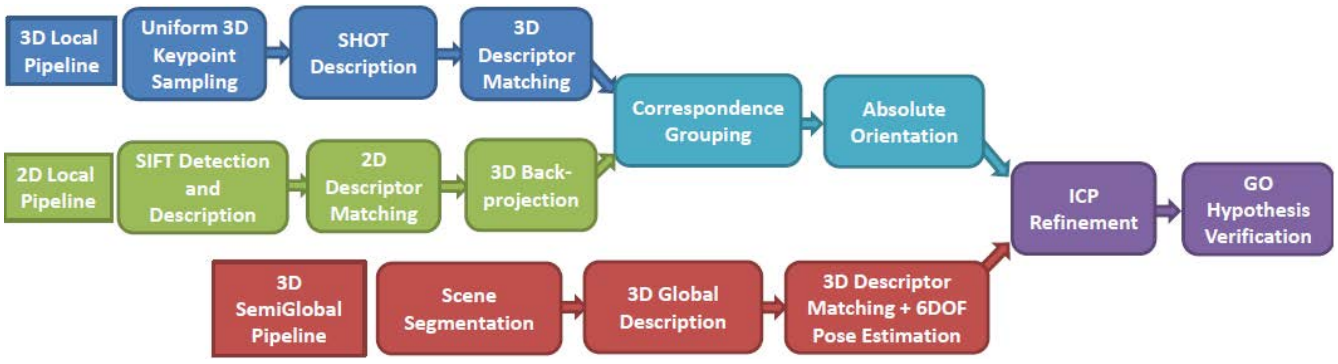

Figure 5: Object detection pipeline including the different stages with complementary features.

Figure 5 sketches the proposed algorithm by showing the various stages therein and the way the three different pipelines are merged together, ending up into a final Hypothesis Verification stage which is in common with all pipelines. As usual for recognition systems, our system consists of a training stage, where models of the objects to be recognized are learned based on the images the corresponding camera poses and the reconstructed objects (described in the previous paragraph). Finally, the identification and pose estimation of objects in the scene is done online and provides the 3D location for grasping the object.

For the actual grasp action we decided to integrate an arm with six degrees of freedom, in contrast to the 5 degrees-of-freedom IGUS arm used in PT1. The additional degree of freedom makes path planning more flexible and a dynamic calculation more applicable. A model of the new arm was built for simulation with 3D CAD design software and the kinematic capabilities and path planning with obstacle avoidance were tested in the robotic simulation environment OpenRAVE. The new arm is tested separately and will be available on the robot in March. To improve the grasping method itself, we continued to use the PT1 platform (Figure 6).

The existing method for grasp point detection (Height Accumulated Features - HAF) was improved and extended to cope with certain situations where objects were placed on top of bigger objects (like boxes) and near to the objects boarder of these bigger objects. These improvements were published at ICRA 2013. Furthermore, the results demonstrate the capability of the approach to grasp objects in cluttered environments (e.g. in a pile of objects).

Further work was done to calculate an optimal gripper opening width as a pre-step for grasping. This enhancement enables grasping of an object even if the object is placed in

[14] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” International Journal of Computer Vision (IJCV), vol. 60, no. 2, pp. 91–110, 2004.

[15] F. Tombari, S. Salti, and L. Di Stefano, “Unique signatures of Histograms for local surface description,” in Proc. 11th European Conference on Computer Vision (ECCV 10), 2010.

between other objects. An example scenario with a video tape in a shelf is depicted in Figure 6.

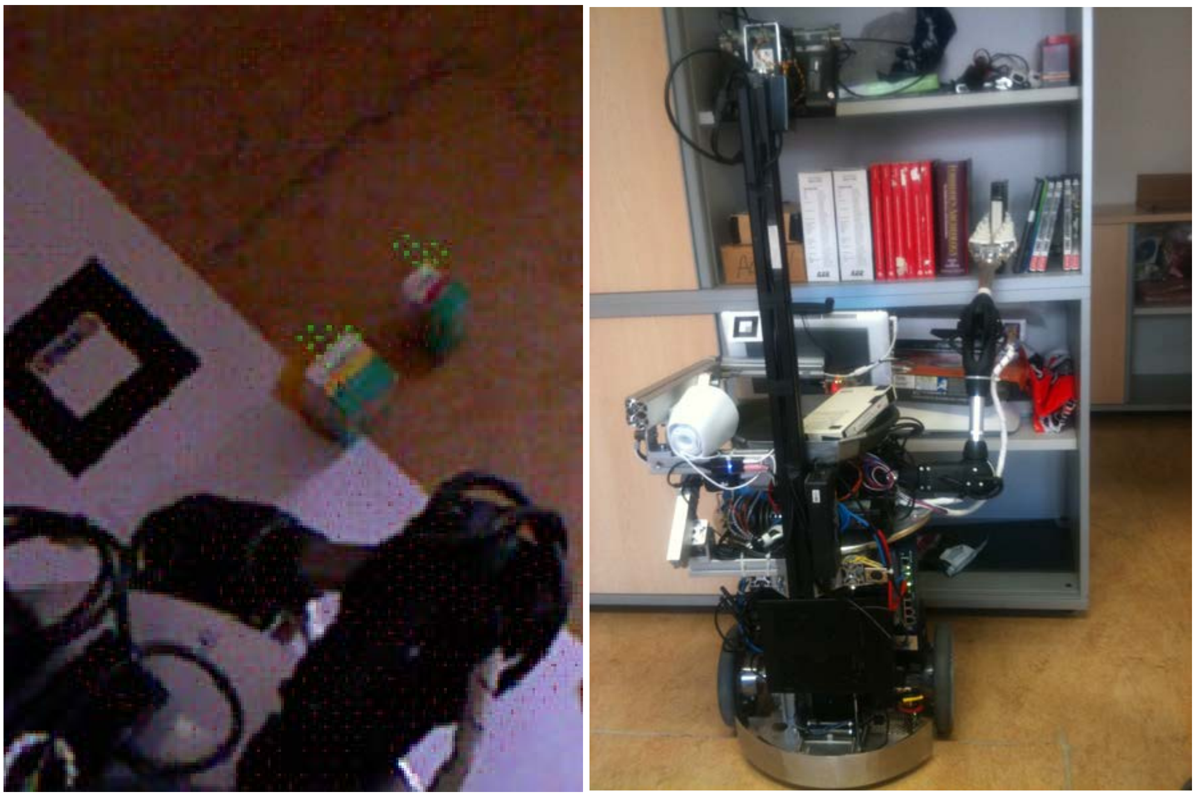
Figure 6: View from camera in robot head and PT1 robot arm taking on object.

During user trials for PT1 after each grasp was executed the robot checked if the grasp was successful. The procedure was time consuming, especially if a grasp was unsuccessful because the robot had to start grasping from the default position of the arm. Since one important point of criticism from user side was that Hobbit executes his tasks rather slow, we worked on optimising this procedure using a kind of visual servoing approach. We want to detect as soon as possible if a grasp was successful or not. A first approach detects edges of the gripper and compares the resulting edges with a reference of the gripper. The underlying concept is that the (edges of the) gripper gets deformed when the gripper is closed and an object is inside. Regrettably, the resolution of the Kinect camera is not sufficient to get significant views of a grasping position near the floor. So this approach could only be implemented with better hardware or perception method. A direct picture match with the SIFT algorithm (Scale Invariant Feature Transform) did not result in the correct outcome either, because there are too few interest points on the gripper. An additional implemented approach uses a CAD-model of the gripper and detects it with a pre-implemented algorithm called 3D-Net of the PCL (PointCloudLibrary, http://www.pointclouds.org). As the depth information of the gripper is not satisfying the needs of the algorithm – only a few depth points of the gripper can be detected – the gripper was covered with balloon-plastic to increase the visible gripper surface. Although this improved the results, it was still not satisfying our robustness requirements.

Our favourable approach we are currently working on is based on a pattern matching algorithm with AR-markers. A pattern was designed and mounted on both sides of the manipulator fingers. The gripper is closed if an object is detected in a promising grasp position relative to the gripper. If the gripper closes completely the pattern can be matched as the cut region get connected and complete the pattern. As the pattern gets matched it is obvious, that there is no object in the gripper. The grasping approach failed. If the pattern can't be matched there has to be an object between the two parts of the gripper. Moving the robot arm up, trying to take the object away, and again starting the pattern matching verifies if the grasp was stable. The precondition for this solution is the top-view of the pattern, which is mounted at the gripper. The matching works with a rotated gripper in the range of approximately –45 … +30 degrees, measured from the top-view. The pattern cannot be matched if the gripper is rotated to the lateral view. Therefore we use a predefined arm positions with minimal arm movement necessary from the actual grasp position for checking the success of executed grasps.

Next work is to port the robot arm planning methods to the new 6DOF arm and then start testing. The object detection method can be used as it is.

6 Integration "Fetch & Carry" (Task 6.5)

Work has been started towards integration for the search task. An efficient search procedure, based on the optimization of a cost function, has been worked out. As a prerequisite, several "search locations" per room have to be defined in the map during the initialization phase. To that end, a special annotation tool has been developed (Section 2).

If an object has to be searched for, the cost function is evaluated for every search location. The locations are then sorted according to their corresponding cost, which yields the optimized search procedure for the object. The cost function takes several aspects into account, such that a good trade-off between the probability of the object being found at a search location and the time it takes to get there can be found. While the different locations are searched by HOBBIT one-by-one, the probabilities of the object being there are permanently updated depending on if the object has been found or not.

Moreover, a penalty term is added to locations which are in the same room as the user, as we assume that the object is most likely located in a different room which should therefore be searched first. If a room cannot be reached because the path is blocked, the costs for all search locations in that room are increased such that these locations are considered last during the search procedure.

As an additional approach to reduce the execution time for the fetch & carry procedure, we are working on an efficient real-time semantic segmentation algorithm. The purpose of the algorithm is to generate a segmentation of the scene, visible by the head Kinect, into semantically meaningful parts, like floor, wall, table, cabinet... Using the additional knowledge of the semantic segmentation result, the search procedure for objects can be further automated and optimized, without the need of specifically labelled "search locations". Instead, the robot figures out itself, where possible object locations might be (objects are most likely located on tables, shelves...) and where it has to navigate to in order to be able to detect them. The knowledge could be exploited even further, to generate complete "semantic maps" of the environment, allowing the user to send the robot to automatically detected places like "the table in the living room". Example of the current capabilities of the algorithm can be seen in Figures 8 and 9.

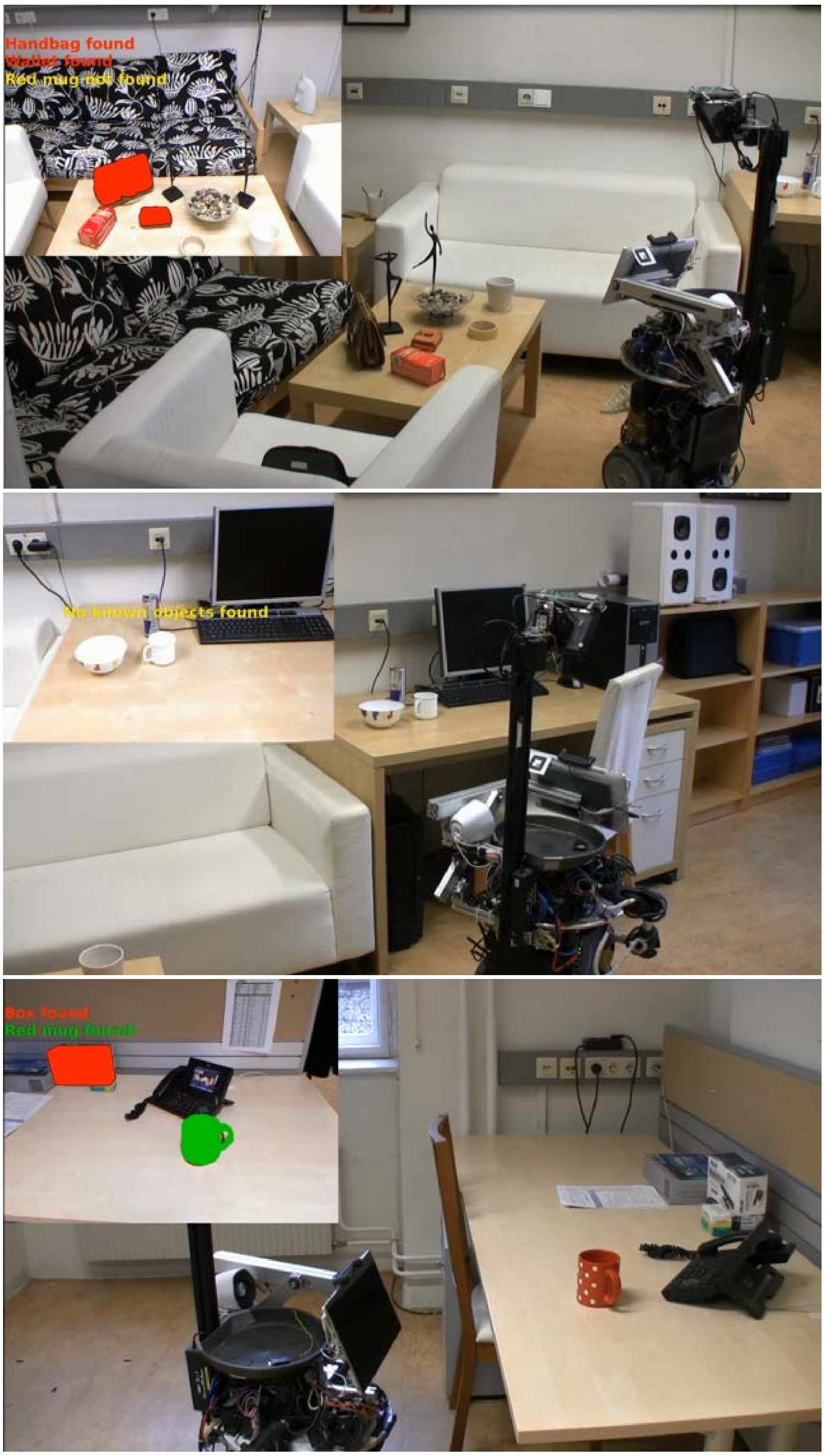

Figure 7: Example of a search procedure. The task is to search for the red mug. When entering the room, two other locations are closer and searched first. Other known objects are found and their location is stored for future search operations.

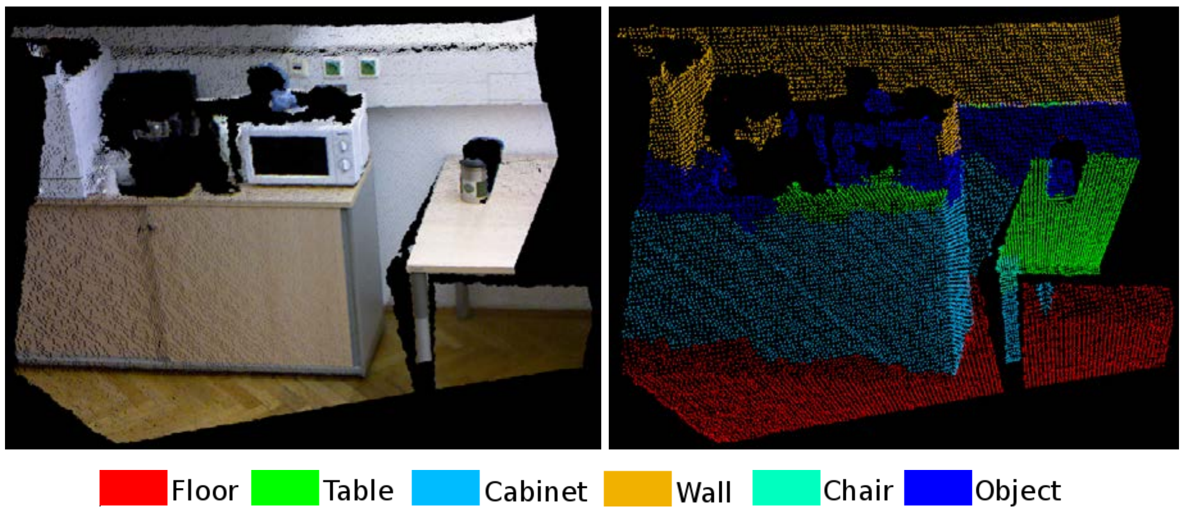

Figure 8: Example result of the semantic segmentation algorithm. Note that even the small area in front of the microwave is labelled as "table" and consequently recognized as a potential object location.

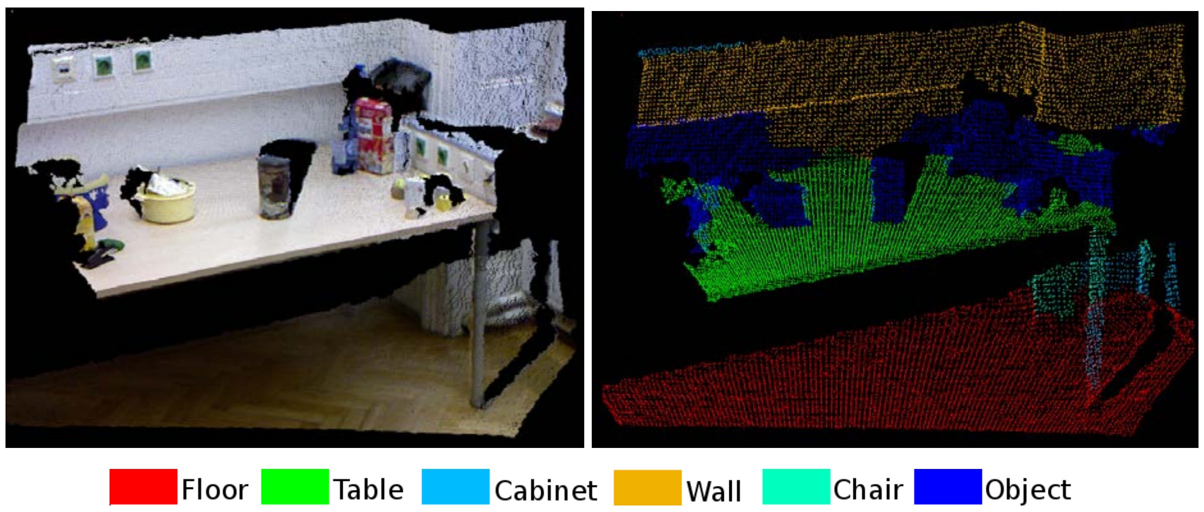

Figure 9: Another example result of the semantic segmentation algorithm. Only local geometry features are used so far, further improvements are part of current work.

At the moment, the method is only using local geometry features to classify the points returned by the top Kinect camera, colour information will be included in the next step. Finally, we will add a probability model for the spatial relations between class labels, what is expected to further enhance the labelling accuracy. Thus, from training data the algorithm will learn relations like objects are more likely to be located on tables or cabinets that are close to walls and so on.

The next steps are to thoroughly test this method and analyse if it can be integrated into the search procedure. The search procedure itself needs to be ported to PT2 and then tested in lab and pilot tests.

7 Conclusion

The aim of this Deliverable was to describe the progress with respect to the components of WP6 for future integration into the PT2 HOBBIT robot. The main technical advance regarding map building and navigation (Tasks 6.1 to 6.3) is presented as a publication. The intention is to submit this paper to IROS 2014. We also highlighted key points and related the achieved results with the description of work. In summary the objective of replacing the laser sensor(s) with RGB-D sensors has been achieved and we retain robust mapping, localisation, obstacle avoidance, and navigation capabilities.
Furthermore, we also report work in Tasks 6.4 and 6.5. This work has the goal to obtain a practical system for object detection and object search for use in the PT2 HOBBIT robot. We show that we achieve object learning with a method that is easier to handle for the user than in PT1. We further outline a method to use the hierarchical place and room structure to search for learned objects and to store objects if found during the search.

The next step is to transfer the developments from PT1 to the new PT2 platforms. This work started already by discussing interfaces between the ROS methods and the Metralabs platform. Furthermore, we will test the existing localisation and navigation methods at Metralabs for potential integration or combination. Results will be reported in D6.3.

8 Appendix

[1] P. de la Puente, M. Bajones, P. Einramhof, D. Wolf, D. Fischinger, M. Vincze: RGB-D Sensor Setup for Multiple Tasks of Home Robots and Experimental Results; to be submitted to IROS 2014.

RGB-D Sensor Setup for Multiple Tasks of Home Robots and Experimental Results

P. de la Puente, M. Bajones, P. Einramhof, D. Wolf, D. Fischinger, M. Vincze[1]

Abstract— **While navigation based on 2D laser data is well understood, the application of robots at home environments requires seeing more than a slice of the world. RGB-D cameras have been used to perceive the full scenes and solutions exist consuming extensive computing power. We propose a setup with two RGB-D cameras that covers the need for conflicting requirements regarding localization, obstacle avoidance, object search and recognition, and gesture recognition. We show that this setup provides sufficient data to enable safe navigation at homes and we present how ROS modules need to be configured to use virtual RGB-D scans instead of laser data for operation in real-time (10Hz). Finally, we present first results of exploiting this versatile setup for a home service robot that picks up things from the floor to prevent potential falls of its future users.**

I. INTRODUCTION

Service robots are envisioned to support humans in a variety of everyday activites, such as cleaning, fetching and carrying objects, or monitoring and assisting older adults [1]–[3]. Therefore robots have to enter domestic environments and need to be equipped with a sensor set-up that allows close and safe interaction with the user. Besides sensors mounted on the robot itself, ambient assisted living environments (AAL) can add information and allow for more complex and locally distributed tasks.

Common to these different applications is that they require the robot to navigate in a cluttered 3D environment, to detect users and their gestures, and to recognise objects. While technical solutions exist for each of these core use cases, there are only few robot systems that integrate all functionalities and the solutions that exist are rather costly, e.g., PR2 or Care-o-bot [3].

Up to now only service robots with very limited functionality have entered private households, such as vacuum cleaners and entertainment robots. One reason for this is definitly the cost factor. With the intention to enter a home market, which presents a high cost-saving potential, it is important to study affordability. While cost factors can be reduced at all fronts, we inspect more closely the sensor system needed for a versatile home robot. The intention is to propose an affordable sensor setup that provides data for all use cases with a minimal configuration. The inherent conflicts to resolve can be summarised as follows:

1) Most relevant navigation problems are considered practically solved using rather expensive 2D laser scanners with 180 or more degrees of field of view (FOV) mounted low on the robot front [4], [5]. There exist good datasets, e.g., [6] and open source solutions in ROS. However, obstacle avoidance is limited to one height and a shoe or table edge would go undetected, both likely to be found at home. A second laser or camera looking downwards is required to deal with these cases.
2) Detecting the user and objects both require RGB or RGB-D images and a camera position higher on the robot. While it is good to look rather straight ahead for users, objects on tables or on the floor are better seen from above than from a degenerated side view.

In this paper we propose a sensor setup based on two RGB-D cameras (Fig. 1): camera 1 is mounted on the robot front similar to lasers and camera 2 is mounted on a robot head that has at least two positions to look straight forward or to look down. Camera 1 obtains data for the purpose of seeing as far as possible for localization, while camera 2 either looks straight for user and gesture detection or looks down to find obstacles. The search for objects can use both viewing angles. Alternatively to the pivoting camera 2, two cameras could be used. Since the head can also be used as indication to the user what the robot attends to, we prefer the proposed version.

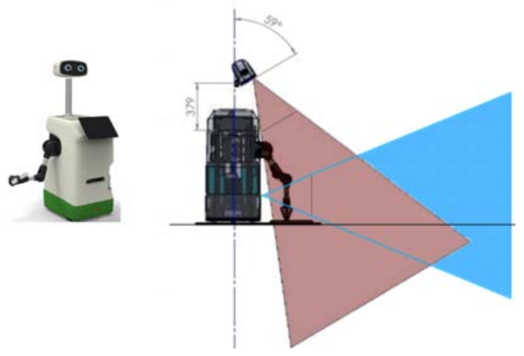

Fig. 1. Final design of the Hobbit robot, with two RGB-D sensors. The top RGB-D sensor in the head of the robot can be tilted for object detection, obstacle avoidance, object grasping and object learning tasks.

One of the most important problems in current robotics is the transition from theory to working systems, using and improving over state of the art available open source tools. The contribution of this paper is to show why the proposed setup with two RGB-D cameras is sufficient to fulfill the core home tasks. In particular we show how standard navigation methods designed for laser data need to be adapted to cope with the lower accuracy, shorter range, and, most critical, the

[1]The authors are with the ACIN Institute of Automation and Control. Technical University of Vienna. authors@acin.tuwien.ac.es.
This work was partially funded by the European Commission, project HOBBIT (FP7-288146).

small FOV of RGB-D cameras. Furthermore, depending on the specific robot construction and place to mount the sensors, there may be a blind spot in front of the robot. We show how to overcome these issues by dynamically adapting the planning frequency and merging local information with the global path. And we show how to cope with possibly reduced localization accuracy and still obtain safe robot paths. If the paper is accepted, we will make the code publicly available in ROS. We then show how navigation is integrated into the service robot tasks based on the ROS SMACH architecture for creating complex robot behaviour [7]. And finally we present the results of operation in different environments.

The paper proceeds as follows. The next section reviews related work. Section II lists and discusses the partially conflicting requirements on the sensor setup for home robots. Section III presents in more detail proposed sensor setup (Fig. 1). Section IV presents the solutions for navigating based on this camera setup and Section V embeds navigation in the robots behaviour. Finally, Section VI evaluates the RGB-D setup in different home-like settings.

A. Related Work

Safe and reliable autonomous navigation in home environments remains an open topic in mobile robotics. The standard solution for 2D mapping, localization and navigation uses one laser scanner with 180 degrees field of view mounted horizontally in front of the robot looking forward. This 2D approach has proved an effective solution for mapping and localization in most indoor environments [x, y], but it is limited with respect to obstacle avoidance when the environment contains obstacles of different heights. In that case, other solutions must be found [8], [9].

Another option is to rely on RGB-D cameras. The large interest is supported by a recent proposal for a Kinect Navigation Challenge in the robotics community. The initiative was launched jointly by Microsoft and Adept Mobile Robots. The RGB-D sensor characteristics add new challenges such as lower sensor accuracy, small FOV, and handling large image data. Also, a first dataset and benchmark to evaluate RGB-D based SLAM has recently been published [10].

Several recent works have attempted to show whether RGB-D sensors, in particular the Kinect Sensor by Microsoft, could replace laser scanners [11], [12] and be used for mobile robotics navigation [13]–[17]. Interesting technical comparisons mainly focusing on the specifications and performance of the sensors were presented in [11], [14]. The standard settings of state of the art implementations were used without any explanations or adaptations. No suggestions to improve the results were proposed or tested, even when the outcome was not deemed satisfactory. Complete navigation tasks with state of the art implementations were not addressed. Novel methods and algorithms especially designed for RGBD sensors were also developed [16], [17]. An interesting approach applies a wall extraction method for localization and an incremental path-finding algorithm that avoids full re-planning for obstacle avoidance, specifically mentioning some particular challenges of real world home environments [16]. Convincing results were obtained with this approach, although the validation was limited to a single navigation task from one predefined point to another. Another interesting solution is based on plane detection filtering and matching with 2D lines for localization, while all the projected points are used for obstacle avoidance by obtaining open path lengths for different angular directions [17]. The results of this localization method were more accurate and robust than those of approaches simulating the readings of a laser scanner. Long run trials of the complete navigation system were succesfully performed. Other researchers have analyzed the possibility of using a Kinect sensor for obstacle avoidance [18], particularly pointing out and addressing problems related to the existence of a blind detection area. The reliability for detecting thin obstacles was also evaluated.

Regarding a solution for navigation based on RGB-D data, we summarise that full 3D methods are computationally expensive and not suitable to be combined with parallel tasks for affordable home robots. Furthermore, standard path planning algorithms are not designed or implemented to work with 3D data. The rest of the paper presents practical solutions for a working system based on RGB-D sensor data using available open source implementations in ROS. We believe there is merit in reporting the lessons learned and extensions made in the development of the whole system.

II. REQUIREMENTS FOR HOME SERVICE ROBOTS

The challenge of service robots for homes is that a large spectrum of functions is desired [19]. Extensive studies enumerate plenty of tasks [20] also including the use of state-of-the-art AAL features [21], [22]. Key functions range from natural interfaces including language and gestures, visual perception to find and manipulate objects of all sorts, safe navigation independent of clutter expected at home and connection to existing AAL devices, to detecting emergency situations. With these tasks in mind, we focus on the requirements which define the sensor configuration.

- **Call robot.** The robot has to be able to come to a given place when called by the user. This task requires localization and obstacle avoidance for safe navigation.
- **Find user.** The robot has to be able to look for the user. This task requires localization, obstacle avoidance for safe navigation and person detection capabilities.
- **Bring object.** The robot has to be able to bring objects to the user. Besides localization and safe navigation, this task implies being able to detect and pick up objects.
- **Multimodal interaction.** The robot should support multimodal human robot interaction (HRI). This task requires gesture recognition capabilities.

The basic tasks subsume further functions such as following and guiding a person, patrolling and security checks, or continuously checking the floor for potential obstacles that could cause a fall. From these tasks to the user, we extract technical requirements for the perception system of the robot:

- **2D localization and mapping** Ideally the sensor(s) observes large, planar, static structures that are at a

maximum distance from the robot, like walls or heavy furniture. A full 3D reconstruction may be possible but time consuming. The purpose is also served by using available implementations for 2D data requiring only horizontal depth data.
- **Obstacle avoidance for safe navigation** All height ranges up to the height of the robot need to be observed to detect obstacles such as table edges. For safety reasons it must be assured that the traversable floor area is detected and that there are no stairs leading downwards.
- **Object detection and pick up** Users studies [20], [23] indicate that objects on the floor are most important. Convenience asks for objects at medium high, while in special cases grasping objects from higher than the typical head height might be required. Starting with the critical cases, it is required to cover heights from the floor up to $90cm$ (kitchen counters). This includes tables of all heights as well as lower shelves. Object detection requires RGB-D data. The sensor should look down at tables, which allows for table plane detection to simplify object segmentation. Furthermore, side views of objects may be degenerated and render recognition difficult.
- **Person detection and gesture recognition** RGB-D cameras are becoming the standard for these functions, e.g., [24], [25] and there are source implementations available in ROS. The mounting height can range from 0.6m to 1.8m, and the optical axis should be approximately parallel to the ground plane so as to detect standing/sitting persons as well as their gestures.

These requirements obviously bring about conflicts in the decision making for the sensor setup, including aspects such as where to place the sensors in general, at which height and with which orientation. One solution could be to have one static RGB-D sensor for each required capability, which is hardly feasible because of space, connectivity and computing reasons. Although RGB-D sensors are cheap, it is highly preferrable to have a minimum setup configuration.

III. PROPOSED SENSOR SETUP

In view of the requirements presented above, the selected solution was to mount two RGB-D sensors on the robot (Fig. 1).

1) Bottom Camera, fixed: a ground-parallel RGB-D bottom camera at a height of about 35 cm is used for mapping and localization. This height was selected because it makes it easier to detect walls and static furniture despite the presence of chairs and tables. At the same time, it allows for the detection of low static elements such as low shelves or sofas, which can improve the localization behavior in wide rooms. The ASUS Xtion Pro Live RGB-D sensor was selected because of its slightly larger FOV (58°H x 45°V vs 57°H x 43°V of the Kinect).
2) Pan/tilt Top Camera: an RGB-D camera is mounted on a pan-tilt unit at a height of about 120 cm. This camera is used for object detection, human-robot interaction and obstacle avoidance. Although the pan-tilt unit allows for continuous variations of the pan and tilt angles, we only make use of a set of predefined pan-tilt angle combinations. In an initial setup, two fixed angles up and down proved to be sufficient. However, users perceive the head as more natural, if the camera directly looks at them. When looking forward, with the optical axis parallel to the ground, the depth data are used for human detection and tracking and for detecting and grasping objects on table tops. When looking forward and down (tilt angle about 60°), the depth data are used for close-range obstacle detection during navigation. Additionally, when looking down and left, forward or right, the depth data are used for detecting objects on the floor. Since the robot has an arm on its right side, looking down and right is used for object grasping from the floor and when learning new objects. The Kinect was chosen in this case because its VGA colour images have better quality: the Kinect uses a Bayer pattern to encode the colour information while the ASUS uses YUV422.

IV. ADAPTING ROS NAVIGATION TO RGB-D CAMERAS

The intention of providing a generic camera setup is combined with the goal to provide a generic setup for navigation. We build on existing solutions in ROS [26] and the available *navigation stack* including:

- Mapping. The `slam_gmapping` node is a ROS wrapper of the GMapping algorithm [27]. It creates occupancy grid maps from laser and odometry data collected by a mobile robot. A map server utility for saving and accessing previously obtained maps is also provided.
- Localization. The `amcl` ROS node is an implementation of the adaptive (or KLD-sampling) Monte Carlo localization [28] approach, which uses a particle filter to track the pose of a robot against a known occupancy grid map.
- Autonomous navigation. The `move_base` ROS node makes use of a global and a local planner to drive the robot towards a given goal. The available global planner is `navfn`, which operates on a costmap to find a minimum cost plan from a start point to an end point in a grid, applying Dijkstra's algorithm [29]. The available local planner is a `base_local_planner`, which provides implementations of the Trajectory Rollout [30] and Dynamic Window [31] approaches to local robot navigation on a plane.

The `navigation_experimental` stack provides other algorithms, but they are not in a mature enough state to be used reliably, some specific functions are missing and the documentation is scarce.

A. Data Preprocessing

The first problem to overcome towards using the 2D navigation tools available in ROS comes to properly converting the provided depth data matrices to the expected input format. The `pointcloud_to_laserscan` package can be used for that purpose, but we implemented our own nodes for the conversion in order to have more flexibility in the selection of distance measurements and the segmentation of obstacles.

The 640x480 individual 2.5D data computed from the depth images of the ground-parallel bottom RGB-D camera are initially reduced to 640 individual virtual laser beams. To do so, the range is obtained by estimating the vertical structure for each of the 640 columns using a slice of the 2.5D data above and below the plane spanned by the cameras optical axis and the central row of the depth image. Provided that the RGB-D camera had produced valid depth information within such a slice, the maximum distance within each column is taken as a measurement for the virtual 2D laser scan. The reason for taking the maximum distance is that walls, the most adequate features for localization, are the boundaries of indoor environments. The angle information for each column is taken from a lookup table generated at system start-up from the known geometry of the RGB-D camera. In our tests we used a slice of 5cm around the virtual 2D scan plane. To be compatible with ROS, the 640 measurements are re-sampled into a scan with equal angle increments (0.5 intervals were used).

To detect obstacles in front of the HOBBIT platform we use the data from the tilted top camera and apply a segmentation algorithm for the conversion into a virtual scan. We apply an approach which is based on v-disparities [32] that we initially developed for segmentation with stereo cameras, but it works without change on RGB-D camera disparity images too. Fig. 2 shows an example of the results. Details can be found in [33]. It is possible to ignore a rectangular area of the images so as not to create obstacles that correspond to the lower part of the robot base.

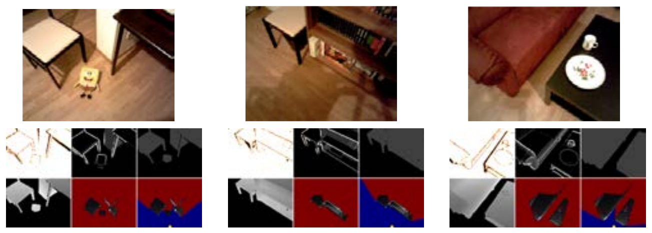
Fig. 2. Top: RGB scenes. Bottom: preprocessing for obstacle avoidance. For each scene, from left to right and top to bottom: confidence map, relative gradient values of the lower resolution disparity image with respect to the vertical gradient of the line corresponding to the floor, points outside a disparity value band around the floor disparity are labeled as obstacle points, projection onto the floor plane, projection labeled, virtual 2D laser scan obtained by raytracing the labeled grid.

B. Reasoning about the Parameter Configuration

The default parameter values used in the ROS navigation stack are specifically provided for laser based systems. As previously mentioned, RGB-D data present very different characteristics, so a proper selection of parameters is required.

The proposed sensor configuration allows the blind area in front of the robot to be reduced, but not completely avoided (see Fig. 1). This issue, together with the reduced FOV for obstacle avoidance, the shorter maximum range and the fact that the designed mobile platform is a non-holonomic platform, are the most important points to consider in order to find good parameter values to make the system work well with the provided input data. This task is definitely not easy -not even with standard laser based systems- and many researchers and ROS users have pointed it out before.

To begin with, by making the robot consider its orientation when following the global plan (`heading_scoring` parameter set to `true`), it rotates in the first place and does not get so much separated from the global path when starting to follow it. Depending on the followed trajectory, this may or may not be helpful for avoiding collisions with small undetected obstacles, considering the blind zone of the sensors. Fig. 3 shows two examples of this situation. In general, considering longer trajectories, it is very unlikely that an obstacle lying closer to the direct route from one point to another is not detected. Given the way the local planner works, and the reduced accuracy of localization, we found this mode of operation safer, especially in narrow spaces.

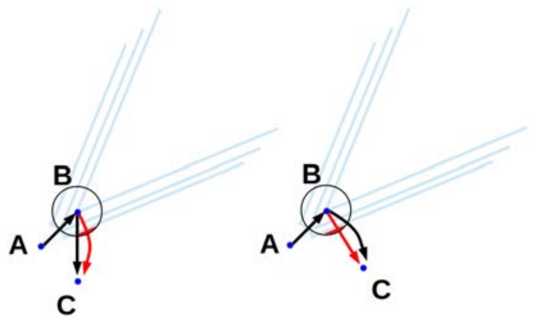

Fig. 3. In place rotations before moving towards a goal can reduce the risk of collision in some cases, but not in others. The black circles represent the blind zone and the blue lines represent the limits of previously observed areas. The robot goes from A to B and then from B to C. The small red areas are not detected by the sensors with the given trajectory, but they could be covered by longer trajectories.

The weights were adjusted so that the global path is followed accurately enough while still trying to reduce the distance to the goal. Furthermore, the planning frequency was increased because the plan needs to be modified when a new obstacle is encountered, and there is less time, since the maximum range of RGB-D sensors is not very large. A compromise has to be found so as to avoid oscillations with paths going through alternative sides of an obstacle.

The effects of modifying the simulation time for the local planner were also analysed. The main problem with the approach of the local planner is that it does not allow for combinations of motion primitives towards longer term planning. If the simulation time is too short, the robot may choose a command action that is the best for that instant but may require higher manoeuvrability later. If the simulation time is too large, single translational and rotational velocity values for all the simulation period are not the best option, especially if some obstacles are not detected. Finding a satisfactory value for this simulation time helped solve some problems with oscillations in the motion without getting undesired effects within different motion primitives.

C. Solving the problems related to the blind area.

Even with a suitable configuration of parameters and settings, the ROS implementation does not allow the static and obstacle maps to be properly used for global and local path planning if there is a blind detection area. Real obstacles in the static map can be erased and a selective memory is required to deal with obstacles not represented in the static map which get too close.

In the first place, the ray tracing method applied to clear up the static map and hence achieve better performance when the robot is slightly mislocalized cannot be used in a straight forward way. Due to the blind area in front of the robot, real obstacles would be erased. This is not a problem if the obstacles are thick external walls and the global planner is configured not to allow navigation thorough unknown areas, but it is an issue when dealing with partition walls and other kind of obstacles inside the environment. On the other hand, if the static map is not cleared at all, localization errors can result in the planner producing undesired paths, as shown in Fig. 4 left, where the path goes closer to the left wall than expected. The proposed alternative comes to adding a new cost map layer plug-in, in the Hydro version of ROS. The cells in the static map which are inside the blind area or beyond the limits defined by the maximum range of the Kinect should be copied to the new layer. Ray tracing must be applied to the intermediate area only. This way, the resulting paths are closer to the desired solution and only deviate in the blind area, as shown in Fig. 4 right.

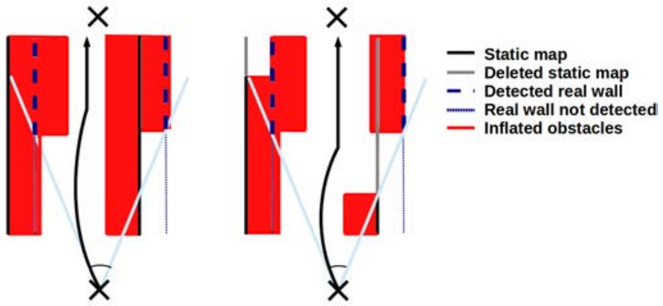

Fig. 4. Planning under slightly wrong localization circumstances. Left: using both the static map and the detected obstacles. Right: applying the proposed approach the static map outside the blind zone is deleted by ray tracing and not considered by the planner, only the dashed lines corresponding to the real world are taken into account. Note that there is more free space for planning.

Besides keeping the static map within the blind zone for safety reasons to avoid collisions when rotating, for instance, it is also important to remember previously observed obstacles close to the robot. With the standard system, when the robot approaches an obstacle it suddenly disappears and the path planner updates can make the robot crash. If all the obstacles are remembered, dynamic elements no longer present make global planning harder or even not possible, without reason. The proposed idea is again to modify the new layer so that at every iteration the obstacles outside a given window are erased by means of the `resetMapOutsideWindow` function, while the window is copied from the obstacles layer to the new layer and the `observation_persistence` parameter is increased so that the obstacles inside this window are remembered for a while. This idea is somewhat similar to the short term memory concept developed in [18]. In our setup, the RGB-D sensor is used for localization, which hence is less reliable and may cause some obstacles to get slightly enlarged sometimes. A good choice of parameters is again very important.

D. Rooms and places

The 2D navigation system was extended to use the concept of rooms, and places inside the rooms. This extension is not related to the sensor modality and can be used by any system. It facilitates the tasks to be carried out in the home environment. Other researchers also pointed out the importance of using intuitive map and place representations [34], [35].

Once a metric map is built and saved, it is possible to open it with a tool editor developed so as to add room labels (see Fig. 5 for an example). The tool is based on the Qt cross-platform application framework and it processes maps in the ROS format (pairs of .pgm and .yaml files), but it is independent of ROS and does not require ROS. The corners of a room are indicated by mouse-clicking on the desired points, then the room must be saved and an adequate name given by the user must be entered into a dialogue box. If the user is not satisfied with the room shape or the given name, it is possible to delete the room and add it again, at any point of the room labelling process. The geometry of the rooms does not have to be very precise, what is important is that it contains all the places of interest that the user wants to specify in a subsequent stage of the initialization phase. One advantage of this manual process is that spatial ambiguity is not a problem, since it is the user who decides how to partition the environment. When the result is acceptable, it can be saved to an xml-file to be used later. Using this approach, the room annotation functionality can be incorporated into the mapping process in a very easy and convenient manner.

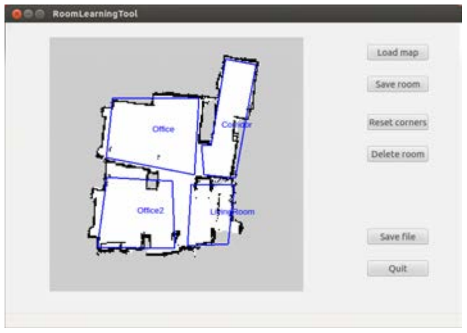

Fig. 5. Room labelling example.

After the approximate geometry and the names of the rooms in the environment have been saved, places of interest can be added to each room. To this end, the robot is tele-operated to the selected places, while *amcl* localization is performed within the map created by GMapping. The robot is stopped at the places of interest and a place label is given (published). The system automatically recognizes the current room and then the place name, along with its x-y position coordinates, is stored in the list of available places for that room. The result is a hierarchy of rooms and places inside each room. The recognition of the current room is based on the crossing number algorithm to detect whether a point lies inside a generic polygon. Our implementation was inspired by the original article by M. Shimrat [36] and by the pseudo code provided by D. Eppstein [37], where we resolved inconsistencies and improved the method to make it more generic. Consequently, the association of places to rooms operates automatically. This functionality is available by means of two ROS nodes, one for getting the current room name and another one for adding the places to the corresponding room in the xml file.

Navigation to the desired places is now possible. A parser node uses the xml file previously created and converts places names to poses to be used by the `move_base` node for autonomous navigation. Using all these new functionalities is extremely easy.

V. IMPLEMENTATION OF HOME ROBOTIC TASKS IN AMBIENT ASSISTED ENVIRONMENTS

The functionalities presented in Section II work as follows.

Navigation to places on demand. The user can select a room and the name of a place, and the robot navigates to that place. Wireless *call buttons* in fixed positions of the AAL environment will be used to define the target place. The user can call the robot to a place. Pressing a call button just sends a message with the place name corresponding to the id number of that button. This activates autonomous navigation to that place.

Locate user. The robot can autonomously look for the user from all the search positions defined for each room, employing a distance criterion and previous knowledge about the last room where the user was detected if available. This information can be obtained from presence and activity sensors placed in different rooms of the AAL environment. The functionality was implemented by means of the ROS SMACH architecture [7]. The state machine for this task is shown in Fig. 6. Please note that the head camera is looking forward when in the ROTATION state (user search) and downwards otherwise. The user detection approach is described in [25].

Bring object. The user can ask the robot to bring a previously learned object. The predefined list of search positions is updated with likelihood values based on previous findings. The top Kinect is used for object recognition and grasping. The description of the algorithms used for this purpose is beyond the scope of this paper, but their main

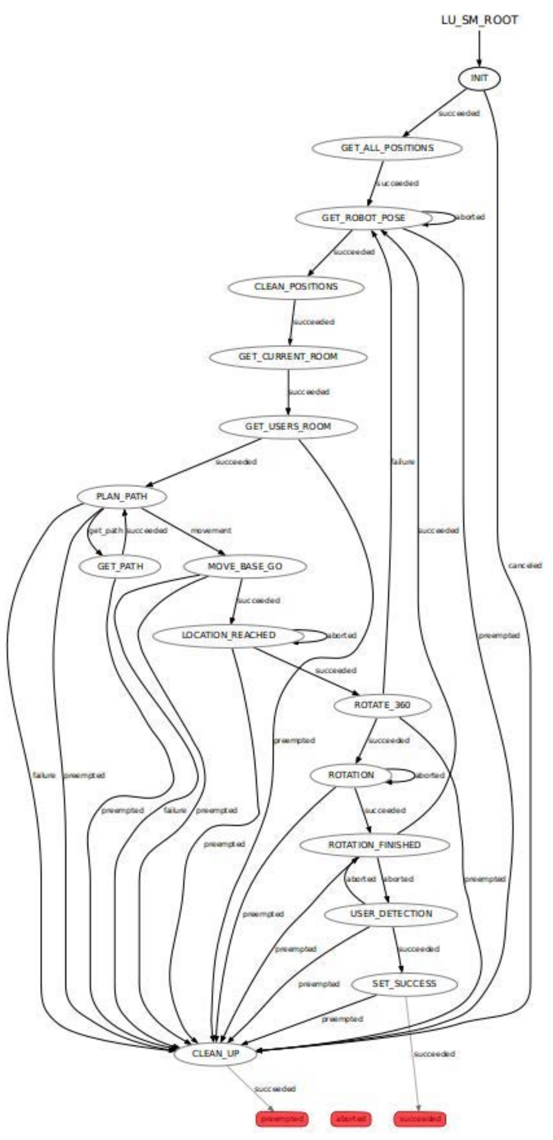

Fig. 6. State machine for the *Locate User* functionality

ideas and evaluations were already published [23], [38]. The implementation of this functionality is based on SMACH too.

Multimodal HRI. The gesture recognition techniques used by the robot are based on the methods presented in [25]. Other ways to communicate with the robot are based on speech recognition and touch screen commands.

VI. RESULTS

As a result of the proposed sensor setup the following a priori specifications and requirements were derived. The open space of a room should be smaller than five meters. The user should not modify the main furniture layout. However, other objects such as chairs and smaller obstacles can be moved. There will not be any long corridors. The obstacles that can be detected by the robot are those objects larger than 500 mm^2 and higher than at least 20 mm with a mate or opaque surface. Obstacles of this kind can be detected as long as they are located at a height between the floor level and the height of the top camera. Some examples are protruding table corners, chairs, stools, cardboard boxes lying on the floor, etc. If at a given point, after several attempts, a new obstacle is blocking the robot path, the robot will ask the user to remove it.

We measured that the average error of the virtual laser scan with respect to a Hokuyo URG-04LX laser sensor is below 1.25cm within a range of 4m. We concluded that the localization is satisfactory if at least 30% of the mapped environment can be observed. Localization with the laser scanner needs only 5% in direct comparison. The algorithm was set to update the 500 particles after the robot has moved at least 0.25m or turned at least 0.2 radians.

The results from our tests to find better parameter values are presented in Table I. Table II contains a more detailed analysis of how changing the parameters `sim_time` and `path_dis_bias` affect navigation through a narrow doorway with quite a straight forward trajectory (see Fig. 7 left). The *Time per run* is computed as the average value for the tests in which no extra rotations were performed. A good initial localization estimate was given at the beginning of every run. Of course, it is hard to isolate the effects of one parameter alone, but by providing similar starting and ending conditions and keeping the other parameters the same, we tried to limit the influence of other factors.

TABLE I

PARAMETER VALUES

Parameter	Value
`heading_scoring`	true
`path_distance_bias`	0.6
`goal_distance_bias`	0.8
`planner_frequency`	5
`sim_time`	2
`observation_persistence`	5

TABLE II

ANALYSIS OF PARAMETERS

Parameter	`sim_time`				`path_dis_bias`	
Value	1	1.5	2	2.5	0.6	0.8
Success rate	9/10	9/10	10/10	10/10	10/10	10/10
Number of rotations	1.2	0.1	0.4	0.2	0.4	0.9
Time per run (s)	42	27	27	37	27	40

The proposed setup was tested in a couple of home-like environments with a previous prototype of the Hobbit robot (Fig.7). The robot safely navigated to predefined places in most cases, even if some of them were quite close to obstacles such as a table or an armchair. A summary of the results is presented in III. These results are comparable to those obtained with other more specific methods [18], but longer term tests will be required. Also, we could see that the localization results both in terms of error and uncertainty can be notably improved by means of in place rotations (Fig. 8), which should facilitate recovery, as proposed in ROS. The Locate User functionality was also succesfully tested, although the user detection needs to be better adjusted to the current settings.

Fig. 9 shows how using the top RGB-D camera for obstacle avoidance makes navigation safer. In Fig. 10 the path for avoiding a close-by obstacle can be observed. It was followed by the robot without risk of collision. Fig. 11 shows the robot picking up an object from the floor.

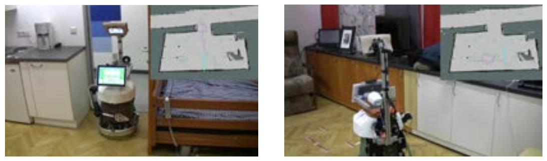

Fig. 7. A previous prototype of the Hobbit robot (with the same sensor configuration as described here) navigating between predefined places.

TABLE III

NAVIGATION BETWEEN PLACES. SUMMARY OF RESULTS.

Test	Succes rate
Navigation between places not going through a door	20/20
Navigation between places going through a door	16/20

VII. CONCLUSIONS AND FUTURE WORK

This paper presented an affordable system for domestic robotic tasks including navigation, using RGB-D sensors and the ROS framework. Improvements over existing solutions and new specific functionalities were proposed and tested.

The configuration of the sensors allows the blind detection area in front of the robot to be reduced. We focus on avoiding and detecting obstacles lying on the floor, which are the most common obstacles in indoor environments and the ones that cause a higher risk of falling down. Obstacles like the narrow stick used in [18] are detected only when the robot gets closer, but we think that this is a minor drawback. There will be no such obstacles in the given environments.

In future work, we will test if our set up can be easily transferred to a different platform without much adaptation work. More improvements will be required in order to make navigation in narrow spaces more reliable. Further experiments will be conducted in very challenging longer term pilot studies and afterwards in real apartments. New recovery behaviors may need to be implemented. We also plan to incorporate semantic mapping capabilities into the system, so that relevant places in each room can be recognized and added autonomously.

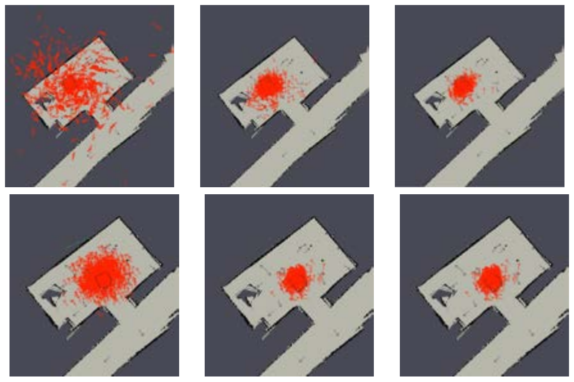

Fig. 8. Localization improvements after rotation. Top: initially acceptable pose estimate with high uncertainty. Bottom: initially incorrect pose (note that the laser data do not match the actual wall, there is an important orientation error) with intermediate uncertainty. Left: before rotation. Middle: after one 360° rotation. Right: after two consecutive 360° rotations. The improvements are qualitatively substantial.

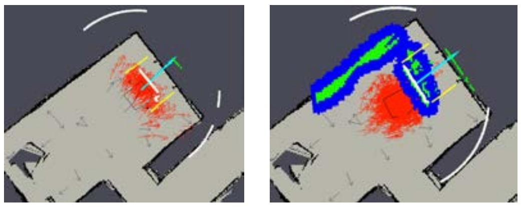

Fig. 9. Left: using only the bottom RGB-D sensor (green measurements) resulted in the robot colliding with a table and losing localization. Right: when the top RGB-D sensor (white measurements), was used for obstacle avoidance the robot stopped and would not go through the table. The thin yellow lines were added for visualization purposes, to show where the real table approximately is.

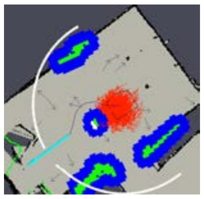

Fig. 10. Avoidance of a close-by obstacle remembered for a while.

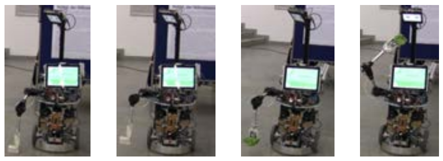

Fig. 11. The robot picking up objects from the floor.

References

[1] T. Taipalus and K. Kosuge, "Development of service robot for fetching objects in home environment," in *Proc. of EEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, 2005, pp. 451–456.

[2] S. Ekvall, D. Kragic, and P. Jensfelt, "Object detection and mapping for service robot tasks," *Robotica*, vol. 25, no. 2, pp. 175–187, 2007.

[3] C. Schroeter, S. Mueller, M. Volkhardt, E. Einhorn, C. Huijnen, H. van den Heuvel, A. van Berlo, A. Bley, and H.-M. Gross, "Realization and user evaluation of a companion robot for people with mild cognitive impairments," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2013, pp. 1153–1159.

[4] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[5] M. Hentschel and B. Wagner, "An adaptive memory model for long-term navigation of autonomous mobile robots." *Journal of Robotics*, 2011.

[6] A. Howard and N. Roy. The Robotics Data Set Repository (Radish). [Online]. Available: http://radish.sourceforge.net

[7] Smach. [Online]. Available: http://wiki.ros.org/smach

[8] C. Pocol, S. Nedevschi, and M. Obojski, "Obstacle detection for mobile robots, using dense stereo reconstruction," in *IEEE International Conf. on Intelligent Computer Communication and Processing*, 2007.

[9] P. de la Puente, D. Rodriguez-Losada, A. Valero, and F. Matia, "3D feature based mapping towards mobile robots' enhanced performance in rescue missions," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009, pp. 1138–1143.

[10] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[11] S. Zug, F. Penzlin, A. Dietrich, T. T. Nguyen, and S. Albert, "Are laser scanners replaceable by Kinect sensors in robotic applications?" in *Proc. of IEEE International Symposium on Robotic and Sensors Environments(ROSE)*. IEEE, 2012, pp. 144–149.

[12] R. El-laithy, J. Huang, and M. Yeh, "Study on the use of Microsoft Kinect for robotics applications," in *Proc. of IEEE Position Location and Navigation Symposium (PLANS)*, 2012, pp. 1280–1288.

[13] H. W. Keat and L. S. Ming, "An investigation of the use of Kinect sensor for indoor navigation," in *IEEE Region 10 Conference*, 2012.

[14] A. Oliver, S. Kang, B. C. Wünsche, and B. MacDonald, "Using the Kinect as a navigation sensor for mobile robotics," in *Proc. of the 27th Conf. on Image and Vision Computing New Zealand*. ACM, 2012.

[15] D. Correa, D. Sciotti, M. Prado, D. Sales, D. Wolf, and F. Osorio, "Mobile robots navigation in indoor environments using Kinect sensor," in *Second Brazilian Conf. on Critical Embedded Systems (CBSEC)*, 2012.

[16] J. Cunha, E. Pedrosa, C. Cruz, A. J. Neves, and N. Lau, "Using a depth camera for indoor robot localization and navigation," in *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, 2011.

[17] J. Biswas and M. Veloso, "Depth camera based indoor mobile robot localization and navigation," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2012.

[18] J. González-Jiménez, J. Ruiz-Sarmiento, and C. Galindo, "Improving 2D reactive navigators with Kinect," in *10th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 2013.

[19] Robocup@home. [Online]. Available: http://www.robocupathome.org/

[20] J. Beer, C. Smarr, T. Chen, A. Prakash, T. Mitzner, C. Kemp, and W. Rogers, "The domesticated robot: Design guidelines for assisting older adults to age in place," in *7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2012.

[21] G. Bugmann and S. N. Copleston, "What can a personal robot do for you?" in *Towards Autonomous Robotic Systems(TAROS)*, 2011.

[22] P. Mayer and P. Panek, "A social assistive robot in an intelligent environment." *Biomedical Engineering*, 2013.

[23] D. Fischinger, M. Vincze, and Y. Jiang, "Learning grasps for unknown objects in cluttered scenes," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[24] A. Ramey, V. Gonzalez, and M. Salichs, "Integration of a low-cost RGB-D sensor in a social robot for gesture recognition," in *ACM/IEEE International Conf. on Human-Robot Interaction (HRI)*, 2011.

[25] K. Papoutsakis, P. Padeleris, A. Ntelidakis, S. Stefanou, X. Zabulis, D. Kosmopoulos, and A. A. Argyros, "Developing visual competencies for socially assistive robots: The hobbit approach," in *Proc. of the 6th International Conference on PErvasive Technologies Related to Assistive Environments (PETRA)*. New York, NY, USA: ACM, 2013.

[26] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[27] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, p. 2007, 2007.

[28] D. Fox, W. Burgard, and F. D. Sebastian, "Monte Carlo Localization: Efficient position estimation for mobile robots," in *In Proc. of the National Conference on Artificial Intelligence (AAAI)*, 1999.

[29] E. W. Dijkstra, "A note on two problems in connexion with graphs," *NUMERISCHE MATHEMATIK*, vol. 1, no. 1, pp. 269–271, 1959.

[30] B. P. Gerkey and K. Konolige, "Planning and control in unstructured terrain," in *In Workshop on Path Planning on Costmaps, Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[31] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," vol. 4, no. 1, 1997.

[32] R. Labayrade and D. Aubert, "Real time obstacle detection in stereovision on non flat road geometry through v-disparity representation."

[33] P. Einramhof and M. Vincze, "Stereo-based real-time scene segmentation for a home robot," in *IEEE ELMAR Conference*, 2010.

[34] A. J. B. Trevor, A. Cosgun, J. Kumar, and H. I. Christensen, "Interactive map labeling for service robots," in *Workshop on Active Semantic Perception at IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Algarve, Portugal, 2012.

[35] J. González-Jiménez, C. Galindo, F. Melendez-Fernandez, and J. R. Ruiz-Sarmiento, "Building and exploiting maps in a telepresence robotic application," in *10th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 2013.

[36] M. Shimrat, "Algorithm 112: Position of point relative to polygon," *Communications of the ACM*, vol. 5, no. 8, pp. 434–, Aug. 1962.

[37] D. Eppstein. Computational geometry. [Online]. Available: http://www.ics.uci.edu/ eppstein/161/960307.html

[38] D. Fischinger and M. Vincze, "Shape based learning for grasping novel objects in cluttered scenes," in *Proc. of the 10th International IFAC Symposium on Robot Control (SYROCO)*, 2012.