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 navigation at homes and we present how ROS modules can 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 the robot should be able to navigate in a cluttered 3D environment, to detect users and their gestures, and to recognize 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 [4] or Care-o-bot [5], as pointed out in [3]. The CompanionAble robot is more affordable, but it still uses expensive sensors.

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 wider field of view (FOV) mounted low on the robot front [6], [7]. There exist good datasets for mapping and localization, e.g. [8], and open source solutions in ROS. However, if obstacle avoidance is limited to one height, 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 similarly 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 an indication to the user of what the robot is paying attention to, we prefer the proposed version.

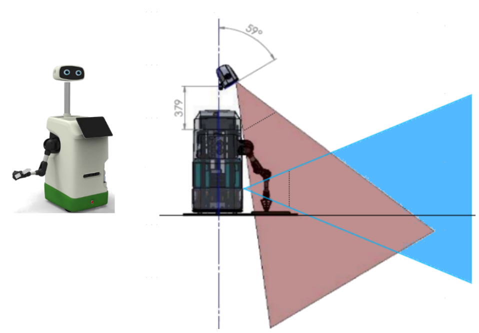

Fig. 1. Final design of our robot, Hobbit, with two RGB-D sensors. The top RGB-D sensor in the head of the robot can be tilted for object detection, obstacle avoidance, object grasping and object learning tasks.

One of the main problems in current robotics is the transition from theory to working systems, using and improving over state-of-the-art tools. The contribution of this paper is related to this, showing why the proposed setup with two RGB-D cameras can fulfill the core home tasks and hence is a step forward to more affordable service robots. In particu-

[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).

lar, we focus on navigation with the proposed setup and show how standard methods designed for laser data need to be adapted to cope with the lower accuracy, shorter range, and the small FOV of RGB-D sensors. Furthermore, depending on the specific robot 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 how to cope with possibly reduced localization accuracy and still be able to obtain safe robot paths. Details about object and user detection can be found elsewhere [9]–[12]. We then show how navigation is integrated for the service robot tasks based on the ROS SMACH architecture for creating complex robot behaviour [13]. Finally we present 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 the proposed sensor setup. 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. Narrow areas are especially challenging. The standard solution for 2D mapping, localization and navigation uses one laser scanner with 180° FOV 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, 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 [14], [15].

Another option is to rely on RGB-D cameras. The RGB-D sensor characteristics add new challenges. The large interest is supported by the recent proposal for a Kinect Navigation Challenge. The initiative was launched jointly by Microsoft and Adept Mobile Robots. Also, a first dataset and benchmark to evaluate RGB-D based SLAM has been recently published [16].

Several recent works have attempted to show whether RGB-D sensors, in particular the Kinect Sensor by Microsoft, could replace laser scanners [17], [18] and be used for mobile robotics navigation [19]–[23]. Interesting technical comparisons mainly focusing on the specifications and performance of the sensors were presented in [17], [20]. 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 RGB-D sensors were also developed [22], [23]. 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 [22]. 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 [23]. 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 [24], 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 full 3D methods, we summarise that they 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.

II. REQUIREMENTS FOR HOME SERVICE ROBOTS

The challenge of service robots for homes is that a large spectrum of functions is desired [25]. Extensive studies enumerate plenty of tasks [26] also including the use of state-of-the-art AAL features [27], [28]. With these tasks in mind, we focus on the requirements which define the sensor configuration.

- **Call robot.** The robot should 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 should be able to look for the user. This task requires localization, obstacle avoidance for safe navigation and person detection capabilities.
- **Bring object.** The robot should 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). For this task, gesture recognition capabilities are desired so that the user can communicate with the robot even if the robot is out of reach and speaking is not a good option, as long as the user is in the field of view.

The basic tasks subsume further functions. From the user-relevant tasks, we extract technical requirements for the perception system of the robot:

- **2D localization and mapping.** Ideally the sensor(s) should observe large, planar, static structures that are at a maximum distance from the robot. A full 3D reconstruction may be possible but time consuming. Using available implementations for 2D data requiring only horizontal depth data is another option.

- **Obstacle avoidance for safe navigation.** The environment up to the height of the robot should be observed to detect obstacles such as table edges. For safety reasons, the traversable floor area must be detected.
- **Object detection and pick up.** User studies [11], [26] indicate that objects on the floor are of utmost importance. Convenience asks for objects at medium height, while in particular cases grasping objects from higher than the typical head height might be needed. Starting with the critical cases, the requirement is to cover heights from the floor up to 90 cm (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., [12], [29] and there are open 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 about 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, in our initial setup two fixed angles up and down proved to be sufficient. 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.

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 [30] and the available *navigation stack* including:

- Mapping. The `slam_gmapping` node is a ROS wrapper of the GMapping algorithm [31]. 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 [32] 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 [33]. The available local planner is a `base_local_planner`, which provides implementations of the Trajectory Rollout [34] and Dynamic Window [35] 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 and accuracy 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 5 cm 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 robot 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 [36], initially developed for segmentation with stereo cameras. Fig. 2 shows an example of results. Details can be found in [37]. It is possible to ignore a rectangular area of the images so as not to create obstacles that correspond to lower parts 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 and virtual 2D laser scan obtained by raytracing the labeled grid.

B. Reasoning about the parameter configuration and solving problems related to the blind area

The default parameter values used in the ROS navigation stack are specifically provided for laser based systems. Since RGB-D data present very different characteristics, 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 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 [38].

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.

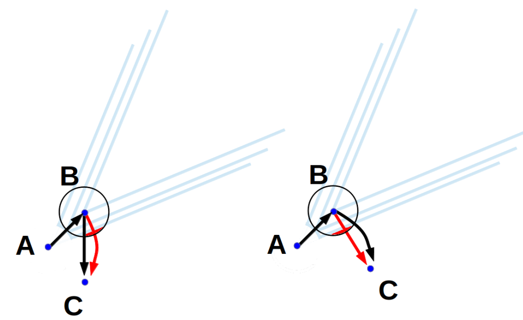

Fig. 3. In place rotations before moving towards a goal can reduce the risk of collision in some cases, but not in others. The black circles represent the blind zone and the blue lines represent the limits of previously observed areas. The robot goes from A to B and then from B to C. The small red areas 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 analyzed. 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 solved problems with oscillations in the motion without getting undesired effects within different motion primitives.

Even with a suitable configuration of parameters and settings, the ROS implementation presents problems 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 could not be used in a straight forward way. Due to the blind area in front of the robot, real obstacles would be erased. This should not be a problem if the obstacles are thick external walls and the global planner is configured not to allow navigation through unknown areas, but it is an issue when dealing with partition walls and other kinds 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 sensor should be copied to the new layer. Ray tracing must be applied to the intermediate area only. If the localization errors are too large, problems can still be found, but this way, the resulting paths are closer to the desired solution and only deviate in the blind area, as shown in Fig. 4 right.

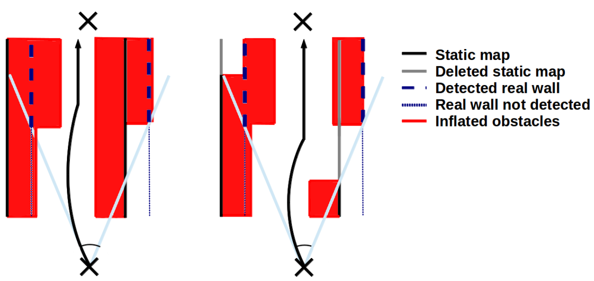

Fig. 4. Planning under slightly wrong localization circumstances. Left: using both the static map and the detected obstacles. Right: applying the proposed modification the static map outside the blind zone is deleted by ray tracing and not considered by the planner, only the dashed lines corresponding 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 for long, dynamic elements no longer present can make global planning harder or even not possible. 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 corresponding to the blind area is preserved. The size of this window has to be precisely defined, otherwise there is also a risk that a dynamic obstacle coming too close may keep the robot blocked. Regarding the local planner, the `observation_persistence` parameter is increased so that the obstacles are remembered for a while. This is somewhat similar to the short-term memory concept developed in [24]. 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. Similar ideas were tested with ROS Fuerte.

C. 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 [39], [40].

Once a metric map is built and saved, it is possible to open it with a tool editor developed in order to add room labels (see Fig. 5 for an example). The tool is based on the Qt 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.

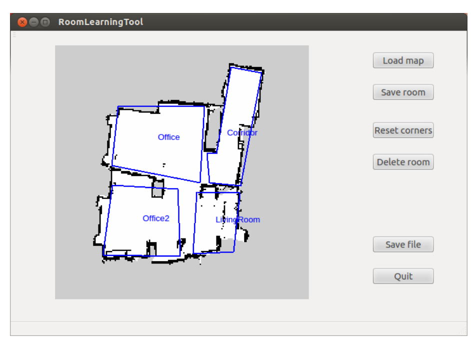

Fig. 5. Room labelling example.

After the defined rooms 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 localization is performed within the map. 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 [41] and by the pseudo code provided by D. Eppstein [42], where we resolved inconsistencies and made the method 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. A GUI version was also developed and it allows a place type for the defined place to be selected.

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. All these new functionalities are extremely easy to integrate and use.

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. 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 will 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 [13]. 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 initial user detection approach was described in [12].

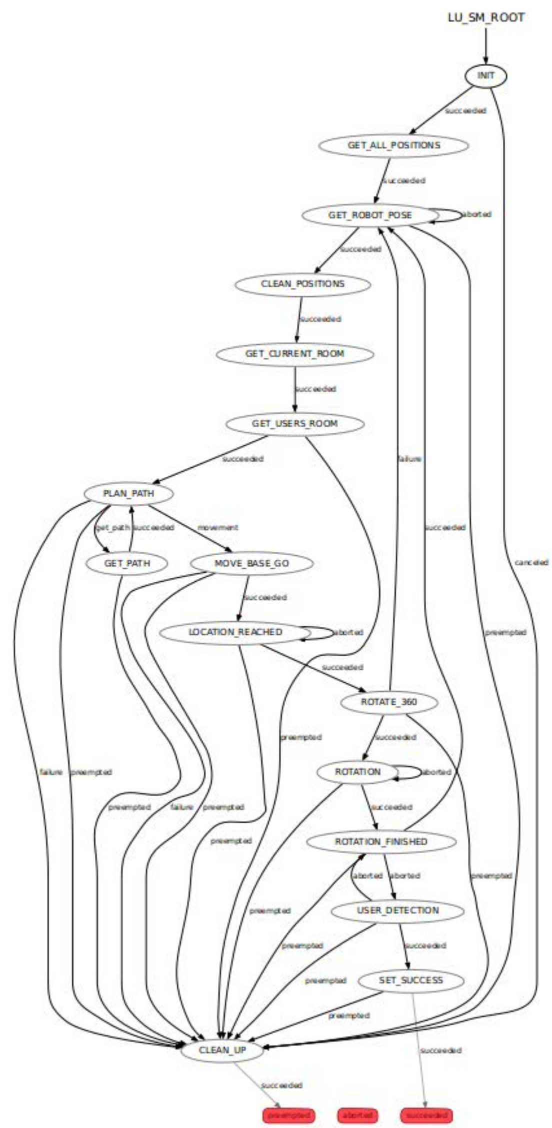

Fig. 6. State machine for the *Locate User* functionality

Bring object. The user can ask the robot to bring a previously learnt object. The predefined list of search positions is updated with likelihood values based on previous findings. The top sensor 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 ideas and evaluations were already published [10], [11]. 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 [12]. Other ways to communicate with the robot are based on speech recognition and touch screen commands.

VI. RESULTS

A priori specifications and requirements were derived from a combination of the application needs and the limitations resulting from the low cost system setup. The open space of a room should be smaller than five meters. Access to narrow corridors is limited, 90 cm are required for a rotation. 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 1.25 cm within a range of 4 m. The algorithm was set to update the 500 particles after the robot has moved at least 0.25 m 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` affects 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 for the same position was given at the beginning of every run. 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). New tests and adaptations are currently being carried out with the final prototype robot. 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 Table III. These results are comparable to those obtained with other more specific methods [24], but longer term tests will be required. 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 succesfully tested, although the user detection needs to be better adjusted to the current settings. The Bring Object functionality has been partially tested, but the grasping is not completely integrated into a whole scenario and the object recognition procedure is also being better adjusted.

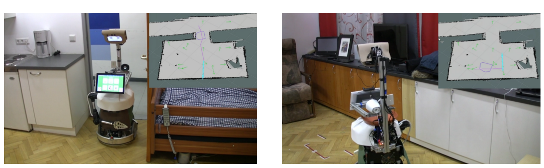

Fig. 7. A previous prototype of the Hobbit robot (with the same sensor configuration as described here) navigating between predefined places.

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

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.

VII. CONCLUSIONS AND FUTURE WORK

This paper presented an affordable system for domestic robotic tasks including navigation, using RGB-D sensors and

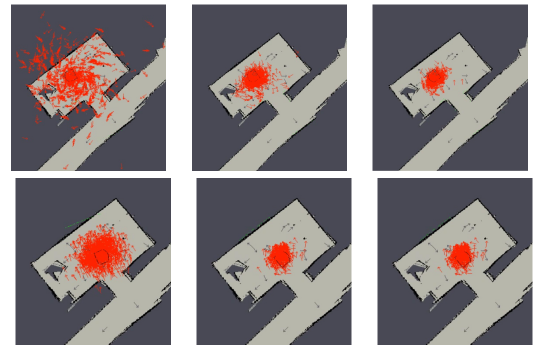

Fig. 8. Localization improvements after rotation. Top: initially acceptable pose estimate with high uncertainty. Bottom: initially incorrect pose (note that the laser data do not match the actual wall, there is an important orientation error) with intermediate uncertainty. Left: before rotation. Middle: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

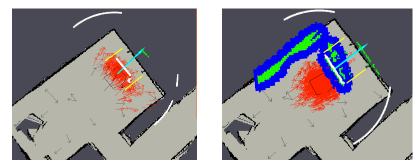

Fig. 9. Left: using only the bottom RGB-D sensor (green measurements) resulted in the robot colliding with a table and losing localization. Right: when the top RGB-D sensor (white measurements), was used for obstacle avoidance the robot stopped and would not go through the table. The thin yellow lines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

the ROS framework. Improvements over existing solutions and new specific functionalities were proposed and tested. We reported the lessons learned and extensions made in the development of the whole system.

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 [24] are detected only when the robot gets closer, but we think that this is a minor drawback. It is hard to find such obstacles in domestic environments.

Despite the improvements introduced in order to adapt ROS navigation to be used with RGB-D sensors, we have finally decided to use Mira [43] navigation instead of ROS navigation for the project. The navigation parameters were already very well tuned for the new robotic platform and the planning algorithms behavior is more robust.

In future work, we will continue the adaptation and testing work with ROS Hydro and the new platform. Migration to the new version of the ROS openni2 driver requires some changes in the virtual laser for obstacle avoidance. More improvements will be required in order to make navigation in narrow spaces more reliable. Further experiments will be conducted and new recovery behaviors may be needed. We also plan to incorporate semantic mapping capabilities into the system, so that relevant places in each room can be recognized and added autonomously.

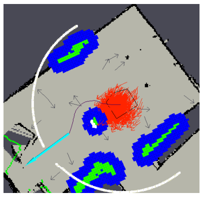

Fig. 10. Avoidance of a remembered close-by obstacle.

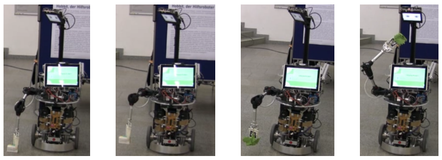

Fig. 11. The robot picking up objects from the floor.

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] J. Bohren, R. Rusu, E. Jones, E. Marder-Eppstein, C. Pantofaru, M. Wise, L. Mosenlechner, W. Meeussen, and S. Holzer, "Towards autonomous robotic butlers: Lessons learned with the pr2," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[5] U. Reiser, T. Jacobs, G. Arbeiter, C. Parlitz, and K. Dautenhahn, "Care-o-bot3 vision of a robot butler," in *Your Virtual Butler*. Springer Berlin Heidelberg, 2013.

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

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

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

[9] D. Fischinger and M. Vincze, "Empty the basket - a shape based learning approach for grasping piles of unknown objects," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

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

[11] 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.

[12] 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.

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

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

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

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

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

[26] 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.

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

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

[29] 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.

[30] 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.

[31] 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.

[32] 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.

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

[34] 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.

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

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

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

[38] Ros answers. [Online]. Available: http://answers.ros.org/questions/

[39] 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)*, 2012.

[40] 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.

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

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

[43] E. Einhorn, T. Langner, R. Stricker, C. Martin, and H. Gross, "Mira - middleware for robotic applications," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.