RoboCup 2011 - homer@UniKoblenz (Germany)

Susanne Thierfelder, David Gossow, Dominik Grüntjens, Carmen Navarro Luzón, Sebastian Nowack, Nico Merten, Susanne Friedmann, Lydia Rebecca Weiland, Daniel Mies, Julian Giesen, Marcel Häselich, Dagmar Lang, Viktor Seib, Dietrich Paulus

Active Vision Group
University of Koblenz-Landau
Universitätsstr. 1
56070 Koblenz, Germany
susanne.thierfelder@uni-koblenz.de
http://robots.uni-koblenz.de/

Abstract. This paper describes the robot hardware and software used by team homer-@UniKoblenz of the University of Koblenz, Germany, for the participation at RoboCup@-Home 2011 in Istanbul. A special focus is put on novel scientific achievements and newly developed features with respect to last year's competition. For navigation and mapping we use well-established SLAM techniques based on particle filter and grid maps. Object recognition is achieved by clustering of local invariant features. Three-dimensional scans of the environment are acquired using a RGB-D sensor mounted on a pan-tilt unit. In order to manipulate objects at different heights, a 2 DOF gripper and a 6 DOF industrial-grade robotic arm are employed. We initiated the integration of the open-source operating system ROS to provide reusability for other teams. For the improvement of user-interaction we developed a generic face model that is synchronized to speech.

1 Introduction

Our team homer@UniKoblenz has already participated successfully as finalists at the RoboCup@-Home World Championship in Suzhou, China (2008), Graz, Austria (2009) and in Singapur (2010), where it was honored with the RoboCup@Home Innovation Award. Besides RoboCup@Home we compete in the RoboCup Rescue league with our robot *Robbie*, where our team won the Interleague Mapping Challenge award (2010) and became German Champion in Rescue Autonomy for the 4th time (2010). In 2011 we will attend the RoboCup@Home World Championship in Istanbul with our robots *Lisa* and *GiGo*.

In the next section our team will be presented. Section 3 describes the hardware used for our master robot Lisa and slave GiGo. In Section 4 our software architecture, autonomous navigation and human-robot interaction will be discussed. Finally, Section 5 will summarize this paper.

2 About our Team

The members of team *homer@UniKoblenz* are students from the University of Koblenz-Landau, Germany. They develop and improve our robots in practical courses.

2.1 Team Members and their Contributions

Team *homer@UniKoblenz* consists of the following members and their contributions

– Susanne Thierfelder: team leader *homer@UniKoblenz*, scientific supervisor
– Viktor Seib: documentation, scientific supervisor
– David Gossow: project integration, scientific supervisor
– Dominik Grüntjens: computer graphics, scientific supervisor
– Carmen Navarro Luzón: team leadership assistant, programming

Fig. 1. Our robots GiGo (left) and Lisa (right)

- Sebastian Nowack: technical chief designer, programming
- Nico Merten: quality ensurance, programming
- Susanne Friedmann: hardware assistant, programming
- Lydia Rebecca Weiland: media, programming
- Daniel Mies: public relations, programming
- Julian Giesen: robot face, programming

Our team is supported by the Center of Excellence of the Chamber of Crafts in Koblenz[1] (see section 3.1).

2.2 Focus of Research

Our interests in research are grid-based 2D mapping, localization and navigation, system architecture for autonomous systems, real-time visualization of sensor data and system states, sensor fusion for person tracking, object, face and gesture recognition as well as object manipulation using a manipulator.

3 Hardware

3.1 Robot Platforms

Master Robot We use a MobileRobots Pioneer3-AT as a platform. It provides sonar and odometry sensors and is equipped with four air-filled tires having a diameter of 21.5 cm. They can be controlled individually, allowing the robot to turn on the spot while maintaining a high degree of stability. Attached to the platform is a 2 DOF gripper, which is used to pick up objects from the floor. On top of the P3-AT, we have installed a prototype framework, which was designed and built by the Center of Excellence of the Chamber of Crafts in Koblenz and is able to carry additional sensors and a notebook running the control software.

[1] Center of Excellence of the Chamber of Crafts in Koblenz `http://www.hwk-kompetenzzentrum.de`

Cleaning and Disposal Robot (New in 2011) The slave robot uses the commercial vacuum robot Roomba SE from iRobot[2] as platform [SGVP10]. It is equipped with a notebook and a Hokuyo URG-04-LX laser range finder for self localization. The master robot controls the slave via a wireless LAN connection with the same control software running on both systems. A bin is mounted on top of the platform that allows the robot not only to clean the floor, but also to carry items that are inserted by the master robot.

3.2 Sensors and additional Hardware

Notebooks The software of the master robot runs on a LG P310 Camini 8400 notebook equipped with a 2,26 GHz Intel Processor and 3 GB of RAM using Ubuntu Linux 10.04 as operating system. The slave robot is controlled by a HP Mini 210-1019EG netbook with a 1,6 GHz Intel Atom and 1 GB RAM running Ubuntu Linux 10.04 as well.

SICK LMS100 laser range finder The SICK LMS100 is mounted at the bottom and generates 270° scans. It has an adjustable angular resolution, while its maximal measured distance is 20 m. It is used for mapping, localization and people tracking.

Hokuyo URG-04-LX laser range finder The Hokuyo laser range finder generates 240° scans that measure the distance of the closest objects. It has an angular resolution of 0.36° and a maximal measured distance of 5.6 m. The Hokuyo URG-04-LX is mounted on the slave robot in order to localize itself and forward this information to the master robot.

DirectedPerception PTU-D46 pan-tilt unit The DirectedPerception PTU-D46 is mounted on top of the robot's neck. It is able to rotate 159° in each direction and to tilt from +31° to −47° out of a horizontal position. The angular resolution is 0.012857°. Further sensores are attached on top of this unit.

Philips 1300NC color camera A color camera with 1.3 megapixels is mounted on the robot platform for object recognition. Is is used to capture images of objects reachable by the 2 DOF gripper.

Neuronics Katana 400HD The Katana 400HD is a 6 DOF industrial-grade robot arm. It is attached to our robot's body plate and used to manipulate objects on tables and other furniture of similar height. With an accuracy of 1 mm and a length of 90 cm, it enables us to perform delicate manipulation tasks on light-weight objects. The end effector is a standard pincher gripper and is safe for interaction with humans. Furthermore, it allows within its workspace to hand over items from the 2 DOF gripper or the slave robot's bin.

Microsoft Kinect (New in 2011) The Microsoft Kinect ist attached to the pan-tilt unit. It contains the PrimeSensor[3] that projects IR light in a specific pattern onto the environment and produces a depth image of the scene. The sensor provides depth and colour images of size 640x480 pixels at 30Hz. The depth sensor operation range lies between 0.8 m and 3.5 m at a horizontal field of view of 58° and a vertical field of view of 45°. It also has two microphones and a tilt motor for sensor adjustment. However, the main robot is equipped with another microphone for speech recognition. Its tilt motor is not used because of the pan-tilt unit that the Kinect is mounted on.

[2] iRobot Corporation `http://www.irobot.com/`
[3] PrimeSense Corporation `http://www.primesense.com`

4 Technology and Scientific Contribution

4.1 System Core Architecture

Our R16 software is based on a generic core that handles the forwarding of messages in the system. Messages are sent and received by modules and exchanged via a dispatcher. Modules use workers as a small set of program code which mainly provide computing functionalities. Devices provide access to hardware components. The system can be configured dynamically by a central registry that contains various profiles that store the required modules and configuration settings for a certain task.

(New in 2011) The R16 software architecture has an interface for using the open-source meta-operating system ROS[4]. It allows to use all services provided by ROS and meanwhile keep the current system core architecture of R16. The main idea is to use both systems in order to keep on the one hand the robust and extensive implementations of R16 and on the other hand to develop new skills in ROS. However, in the long-term a complete changeover to ROS is intended to provide easier compatibility by using a widely spread platform and to offer open-source implementations to the community (please refer to section 4.2 and 4.6).

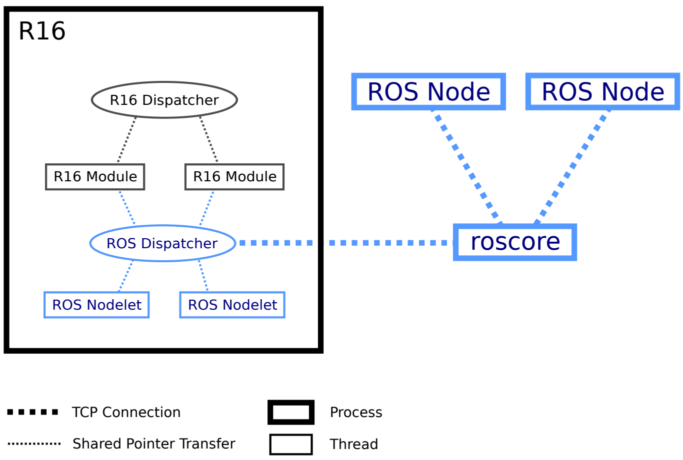

Fig. 2. Schematic description of the communication between R16 and ROS.

Figure 2 gives a schematic overview of the communication between the R16 software architecture and ROS. The changes in R16 include an additional ROS dispatcher besides the R16 dispatcher that manages the messages transfered by ROS nodelets within R16. Nodelets in R16 can subscribe and publish topics via shared pointer transfer within R16 and in the meantime also receive and publish the common R16 messages. They act as a linkage between the two systems. Beyond R16 the common ROS functionalities can be used which is depicted here as ROS Nodes communicating with roscore.

4.2 Graphical Interface

The R16 framework offers a GUI that can be run directly on the robot or on a different computer via WLAN. The user interface is realized using Qt4 and OpenGL. This feature has shaped up as a very important tool for understanding and improving the complex algorithms needed for a fully autonomous robot.

[4] Documentation of ROS `http://www.ros.org/`

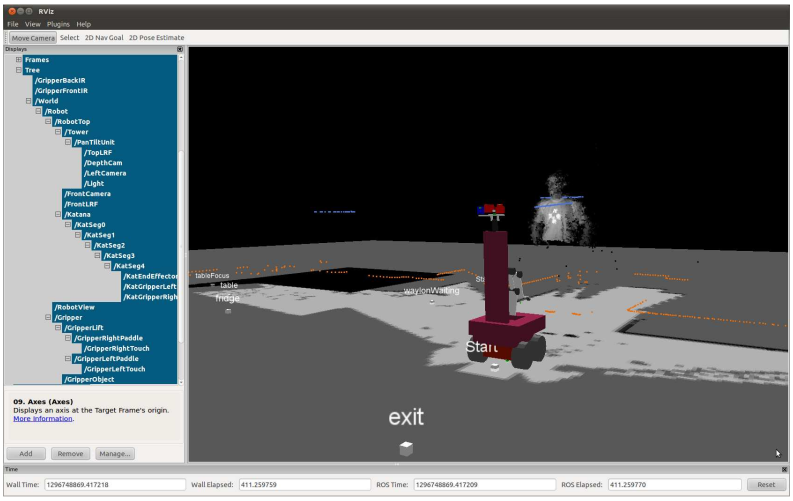

Fig. 3. The rviz visualization tool showing various visualizations of sensor data. The application can be used for real-time surveillance of the robot as well as for playing back recorded sensor log files of R16 and ROS. In the @Home league, this is mainly used by the developers for testing and evaluation.

(New in 2011) The integration of ROS enables us to use the rviz visualisation tool[5]. A screenshot of our robot with various sensor data in the 3D visualization environment is depicted in Figure 3. It allows us for example to monitor the robot's transformation tree, visualize laser data and the points of interest in the current map in a 3D scene.

4.3 Simultaneous Localization and Mapping

To enable users without technical knowledge to use the robot and to ease the setup procedure, it has to be able to create a map without the help of a human. For this purpose, our robot continuously generates and updates a 2D map of its environment based on laser scans. Figure 6 shows an example of such a map.

Fig. 4. Real-time maps of the RoboCup 2008 (left) and 2009 (right) @Home arena.

[5] Documentation of ROS package rviz `http://www.ros.org/wiki/rviz`

4.4 Navigation in Dynamic Environments

In real-life situations, the approach described above is not sufficient for navigating through an everyday environment, as due to the movement of persons and other dynamic obstacles, an occupancy map that only changes slowly in time does not provide sufficient information.

Thus, our navigation system, which is based on Zelinsky's path transform (see [Zel88,Zel91]), always merges the current laser range scan as a frontier into the occupancy map. A once calculated path is then checked against obstacles in small intervals during navigation, which can be done at very little computational expense. If an object blocks the path for a given interval, the path is re-calculated. This approach allows the robot to efficently navigate in highly dynamic enviroments.

4.5 Autonomous Exploration

Several tasks in the @Home league, like "Go Get It!", require the robot to autonomously explore its environment. For this purpose, we are using an exploration algorithm combining Yamauchi's frontier based exploration [Yam97] with Zelinsky's path transform. The path transform is extended in a way that instead of calculating the cost of a path to a certain target cell, the cost of a path that leads to a close frontier is calculated (see Figure 5). More details can be found in [WP07].

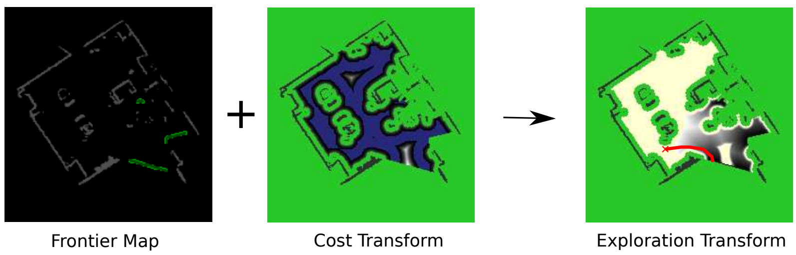

Fig. 5. Illustration of the exploration transform algorithm described in [WP07]. The left image shows the extracted frontiers. The blue area in the middle image contains accessible areas. The red line in the right image represents the path of the robot. The dark values indicate a short distance to the next safe frontier, white cells indicate a long way.

4.6 Human-Robot Interface

The robot is equipped with speakers and a microphone, which enables communication via speech interface. In addition, it has a small screen that is used to display facial expressions and state information. On the software side, we decided to use two open source libraries: pocketsphinx[6] for speech recognition and festival[7] for speech output.

(New in 2011) We have designed a concept of a talking head that is synchronized to speech via mouth movements. For this new feature we are using the open source libraries festival for speech synthesis and speech output and Ogre3D[8] for visualisation. Furthermore, the face is modelled with Blender[9] and exported via the Ogre Mesh Exporter[10] for the use with Ogre. To include the talking head into the R16 software, we have created a ROS nodelet for this application. For this purpose the talking head should be released as an open source package for ROS, so it is possible to create a unique face with Blender and apply it for any roboter.

[6] Speech recognition system pocketsphinx `http://www.speech.cs.cmu.edu/pocketsphinx/`
[7] Speech synthesis system festival `http://www.cstr.ed.ac.uk/projects/festival/`
[8] Open-source graphics rendering engine Ogre3D `http://www.ogre3d.org/`
[9] Free open source 3D content creation suite Blender `http://www.blender.org/`
[10] Ogre Mesh Exporter `http://www.ogre3d.org/tikiwiki/Blender+Exporter`

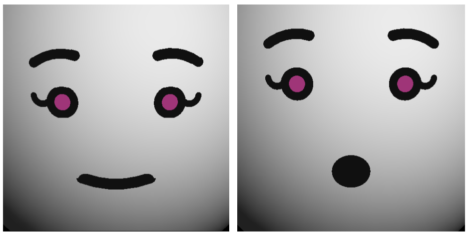

Fig. 6. Animated face of our home robot

4.7 Gesture Recognition

Gestures are recognized from the depth data of our RGB-D camera. During a gesture, the user has to stand in front of the robot, facing it. First, hands are detected as local minima in the depth image. Their motions are tracked and low-frequency fourier features extracted from their geometrical paths. These are then matched against the training set by an approximate nearest-neighbour search.

4.8 Object and Face Recognition

The object recognition algorithm we use is based on Speeded Up Robust Features (SURF) [BTVG06], which are local scale-invariant features of gray images.

First, features are matched between the trained image and the current camera image based on their euclidean distance. A threshold on the ratio of the two nearest neighbours is used to filter unlikely matches. Then, matches are clustered in hough space using a four dimensional histogram using their position, scale and rotation. This way, sets of consistent matches are obtained. The result is further optimized by calculating a homography between the matched images and discarding outliers. Our system was evaluated in [DTPG10] and shown as suitable for fast training and robust object recognition.

4.9 Person Tracking

(New in 2011) For the perception of people we use the depth data provided by the PrimeSense sensor of the Microsoft Kinect. The driver for this sensor is OpenNI via the openni_kinect stack of ROS[11]. The integrated PrimeSense Nite middleware can be used for skeleton tracking[12]. On this way tracking of sitting people is also possible if they are calibrated before.

For people tracking without calibration we use the lower laser range finder to detect legs by looking for pairs of arc segments in the range data as in [XCRN05].

4.10 Object manipulation

Our system detects planes in the acquired 3D point cloud using RANSAC. This information is used to find euclidean point clusters on top of planes which fit into the gripper and are thus regarded as candidates for grasping.

If a specific object has to be grasped, one of the color cameras is adjusted to face the object cluster. The object detection algorithm is executed on the region of interest defined by the object cluster's bounding box. Depending on the position of the object, camera and manipulation device are selected. These are either the bottom camera with the 2 DOF gripper or the top camera combined with the 6 DOF robotic arm.

The movement planning for our robotic arm is performed using an approach operating directly in working space. Chaining motion primitives, our path planner builds a graph from the starting

[11] Documentation of ROS package for OpenNI driver `http://www.ros.org/wiki/openni_kinect`

[12] Documentation of ROS package nite for skeleton tracking `http://www.ros.org/wiki/nite`

position to the goal. The planning can be optimized towards specific objectives. These are performing a smooth path or keeping a maximum distance from obstacles using heuristic and cost functions [CCL10].

5 Conclusion

In this paper, we have given an overview of the approach used by team homer@UniKoblenz for the RoboCup@Home competition. We presented a combination of out-of-the box hardware and sensors and a custom-built robot framework. We explained the fundamentals of our message-based robot architecture and its further enhancements, the use of well-established techniques like SLAM based on a particle filter and a grid map. Furthermore, we explained our approach for object recognition using matching and clustering of local invariant features and the ability to detect and manipulate objects with a 2 DOF gripper and 6 DOF robotic arm. Based on the existing system from last years competition, effort was put into enhancing the detection of people and the human-robot interaction abilities of our robot. 3D scans of the environment are aquired using a new RGB-D sensor. A slave robot was developed which is controlled by our main robot.

References

[BTVG06] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded up robust features. *ECCV*, pages 404–417, 2006.

[CCL10] Benjamin Cohen, Sachin Chitta, and Maxim Likhachev. Search-based planning for manipulation with motion primitives. Anchorage, Alaska, 2010.

[DTPG10] Peter Decker, Susanne Thierfelder, Dietrich Paulus, and Marcin Grzegorzek. Dense Statistic Versus Sparse Feature-Based Approach for 3D Object Recognition. In *10th International Conference on Pattern Recognition and Image Analysis: New Information Technologies*, volume 1, pages 181–184, Moscow, 12 2010. Springer MAIK Nauka/Interperiodica.

[SGVP10] Viktor Seib, David Gossow, Sebastian Vetter, and Dietrich Paulus. Hierarchical multi-robot coordination. In *RoboCup 2010: Robot Soccer World Cup XIV*, 2010.

[WP07] Stephan Wirth and Johannes Pellenz. Exploration transform: A stable exploring algorithm for robots in rescue environments. *Workshop on Safety, Security, and Rescue Robotics*, pages 1–5, 2007.

[XCRN05] M. Xavier, J.and Pacheco, D. Castro, A. Ruano, and U. Nunes. Fast line, arc/circle and leg detection from laser scan data in a player driver. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005.

[Yam97] B. Yamauchi. A frontier-based approach for autonomous exploration. *1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, page 146 ff., 1997.

[Zel88] Alexander Zelinsky. Robot navigation with learning. *Australian Computer Journal*, 20(2):85–93, 1988.

[Zel91] Alexander Zelinsky. *Environment Exploration and Path Planning Algorithms for a Mobile Robot using Sonar.* PhD thesis, Wollongong University, Australia, 1991.

Document date: 2011-05-18