Tech United Eindhoven @Home SPL 2017 Team Description Paper

M.F.B. van der Burgh, J.J.M. Lunenburg, R.P.W. Appeldoorn, R.W.J. Wijnands, T.T.G. Clephas, M.J.J. Baeten, L.L.A.M. van Beek, R.A. Ottervanger, H.W.A.M van Rooy, J. Scholtes and M.J.G. van de Molengraft

Eindhoven University of Technology,
Den Dolech 2, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
`http://www.techunited.nl, techunited@tue.nl, https://github.com/tue-robotics`

Abstract. This paper describes the research interest and technical approach of the Tech United team towards using the Toyota HSR system for the RoboCup @HOME standard league. Tech United will use an advanced world modelling representation system called the Environment Descriptor (open source) that allows straight forward implementation of localization, navigation, exploration, object detection & recognition, object manipulation and robot-robot cooperation skills. This will yield a fast development to create added functionality for the HSR system and to be used by the open source community. Recently developments are improved object detection via deep learning methods, a generic GUI for different user levels and improved natural language interpretation.

1 Introduction

Tech United Eindhoven is the RoboCup student team of Eindhoven University of Technology that (since 2005) successfully competes in the robot soccer Middle Size League (MSL) and later (2011) joined the ambitious @Home League. The Tech United @Home team is the vice champion of RoboCup 2016 in Leipzig and the reigning European Champion of the 2016 RoboCup European Open. The robot soccer mid-size Tech United team has an even greater track record with 3 world championship titles. See the Tech United website for more results.
This proposal will explain our technical development plans with the Toyota HSR system.

2 Description of the approach planned to be implemented on the robot

This section will explain the methods and software that will be used to solve the RoboCup@Home challenges with the Toyota HSR system.

2.1 Environment Descriptor (ED)

The TUe Environment Descriptor (ED) is a Robot Operating System (ROS) based 3D geometric, object-based world representation system for robots. In itself ED is database system that structures multi-modal sensor information and represents this in an object-based world representation that can be utilized for robot localisation, navigation, manipulation and interaction functions. See Figure 1 for a schematic overview of ED. Tech United uses ED in its AMIGO and SERGIO robots that participate (and collaborate) in the @Home league. In previous years, developments have been focussed towards making ED platform independent. As a results ED had been used

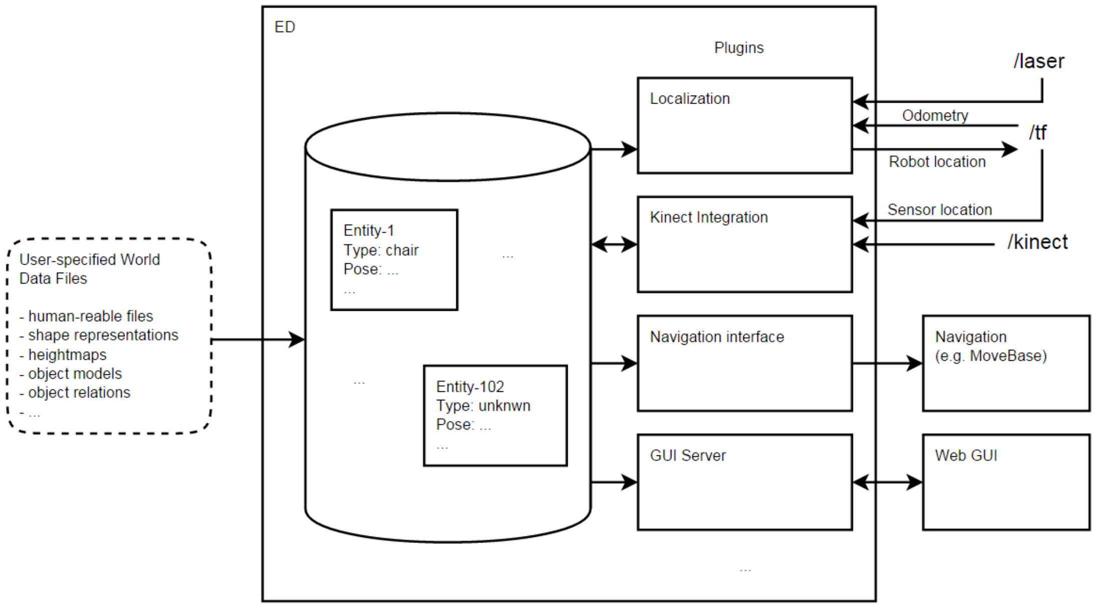


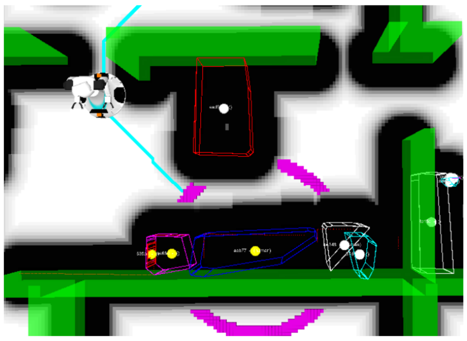
Fig. 1: schematic overview of TUe Environment Descriptor.

on the PR2 system, Turtlebot and Dr. Robot systems (X80). ED is one re-usable environment description that can be used for a multitude of needed functionalities. Instead of having different environment representations for localization Adaptive Monte Carlo Localization (AMCL), navigation (MoveBase), manipulation (MoveIt!), interaction, etc.. An improvement in this single, central world model will reflect in the performances of the separate robot capabilities. It omits updating and synchronization of multiple world models. At the moment different ED modules exist which enable robots to localize themselves, update positions of known objects based on recent sensor data, segment and store newly encountered objects and visualize all this through a web-based GUI.

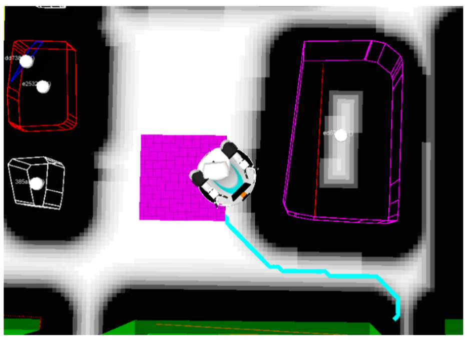
2.2 Localization, Navigation and Exploration

The ED-localization plugin implements AMCL based on a 2D render from its world model.

With use of the ed_navigation plugin, an occupancy grid is derived from the world model and published as a nav_msgs/OccupancyGrid. This grid can be used by a motion planner to perform searches in the configuration space of the robot.
With the use of the cb_base_navigation ROS package. The robots are able to deal with end goal constraints. With use of a ROS service, provided by the ed_navigation plugin, an end goal constraint can be constructed w.r.t. a specific world model entity described by ED. This enables the robot to not only navigate to poses but also to areas or entities in the scene, as illustrated by Figure 2. Somewhat modified versions of the local and global ROS planners available within move_base are used.
The configurations for move_base node has now been created for the AMIGO and SERGIO robots, the HSR node will be added. The node integrates local and global path planning and local control. Its actionlib interface has become in fact standard for sending 2D navigation goals to robots in ROS.



(a) Circle constraint, $x^2 + y^2 > a^2$ and $x^2 + y^2 < b^2$, frame *kitchenblock*

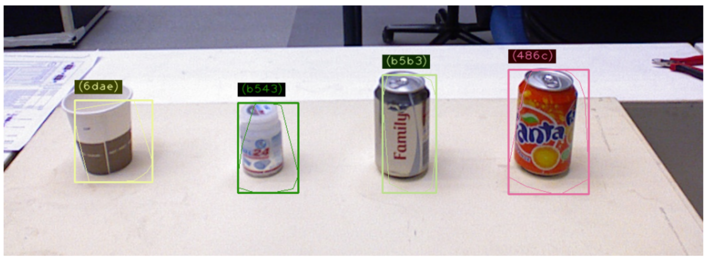


(b) Rectangular constraint, $x > a$ and $x < b$ and $y > c$ and $y < d$, frame *livingroom*

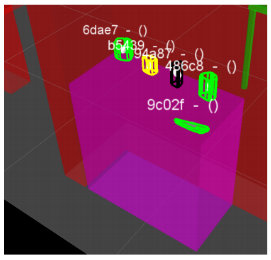
Fig. 2: Navigation position constraints w.r.t. other entities in the environment

2.3 Object detection and recognition

Detection & Segmentation ED enables integrating sensor data with use of the plugins present in the ed_sensor_integration package. Two different plugins do exist: 1. laser_plugin: Enables tracking of 2D laser clusters. This plugin can be used to track dynamic obstacles such as humans. 2. kinect_plugin: Enables world model updates with use of Kinect data. This plugin exposes several ROS services that realize different functionalities: a. Segment: Service that segment sensor data that is not associated with other world model entities. Segmentation areas can be specified per entity in the scene. This allows to segment object 'on-top-of' or 'in' a cabinet. b. FitModel: Service that fits the specified model in the sensor data of the Kinect. This allows updating semi-static obstacles such as tables and chairs.
The ed_sensor_integration plugins enable updating and creating entities. However, new entities are classified as unknown entities.



(a) Cup, Pills, Diet cok, Fanta - Kinect view



(b) World representation

Fig. 3

Recognition using Deep Learning In order the classify or train unknown entities, the ed_perception plugin exposes ROS Services to classify the entities in the world model. The ed_perception module interfaces with various image_recognition nodes that apply state of the art image classification techniques based on Convolution Neural Networks (CNN) 4. Object recognition is done using Tensorflow: retraining the top-layer of a Inception V3 neural network. The top layers are retrained on a custom dataset using a soft-max top-layer that maps the image representation on a specified set of labels.
Face detection and recognition is done using Openface based on Torch. Openface is an existing state-of-the-art face recognition library. We implemented a ROS node that enables the use of

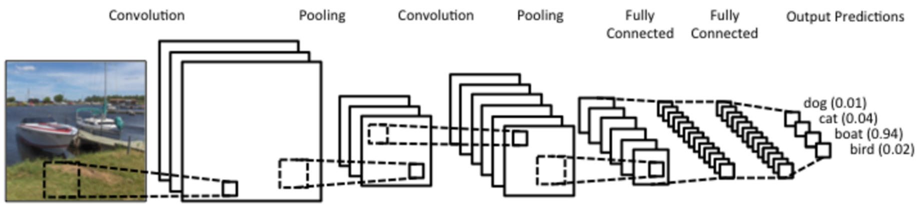


Fig. 4

these advanced technologies within the ROS network.
In order to create a new training set for specific objects, the ed_perception and the image_recognition packages contains several tools for segmenting and annotating object. Also tools for retraining neural networks are included.
Our image recognition ROS packages can be found at GitHub with tutorials and documentation.

2.4 Object grasping, moving and placing

As for manipulating objects, the architecture is only focused on grasping. The input is the specific target entity in the world model ED. The output is the grasp motion, i.e. joint positions for all joints in the kinematic chain over time. Figure 5 shows the grasping pipeline. A python executive

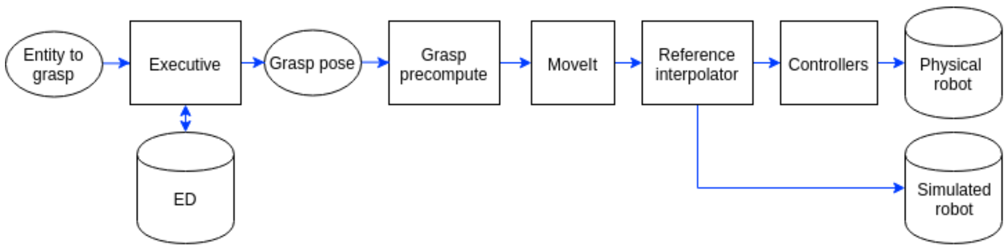


Fig. 5: Grasping pipeline.

queries the current pose of the entity from ED. The resulting grasp pose goes to the grasp precompute component which makes sure that we approach the object in a proper way. MoveIt will produces joint trajectories over time with use of the current configuration, the URDF model and the final configuration. Note that MoveIt currently does not take any information from ED into account. Finally, the trajectories are sent to the reference interpolator which sends the trajectories either to the controllers or the simulated robot.

2.5 Reasoning

The reasoning layer of the AMIGO ROS based software consists of a set of finite state machines (robot_smach_states) that build upon the robot's skill layer (robot_skills). These state machines are useful if you want the robot to execute some complex plan, where all possible states and state transitions can be described explicitly. The implementation is done with use of the open-source SMACH package.

2.6 Human-Robot Interface

In order to interact with the robot aside of speech, a web-based Graphical User Interface (GUI) has been designed. The interface has been made with HTML5 and is hosted on the robot itself.

This allows multiple users on different platforms (*e.g.* Android, iOS) to access functionality of the robot. The interface is implemented in JavaScript with AngularJS and it offers a graphical interface to the Robot API which exposes all the functionality of the robot. Figure 6 gives an overview of the connections between these components.

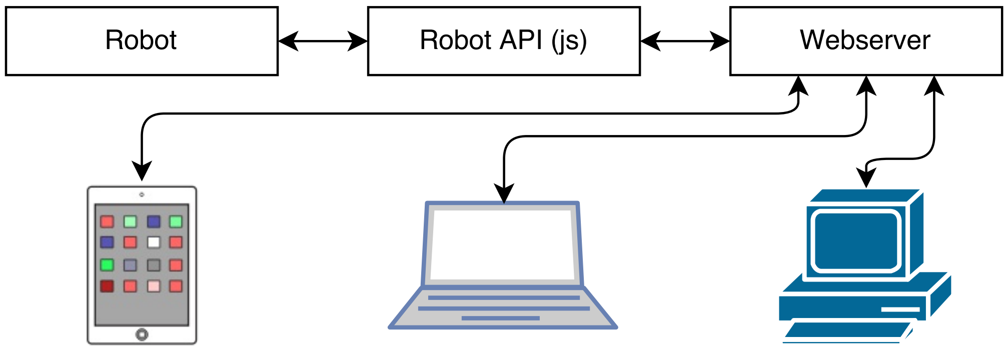


Fig. 6: Overview of the WebGUI architecture. The robot's functionalities are exposed with the Robot API that is implemented in JavaScript. A webserver that is hosting the GUI connects this Robot API to a graphical interface that is offered to multiple clients on different platforms.

3 List of externally available components that are planned to be implemented

An overview of the software used by the Tech United Eindhoven @Home robots can be found in Table 1. All our software is developed open-source at GitHub.

Table 1: Software overview of the robots.

Operating system	Ubuntu 14.04 LTS Server
Middleware	ROS Indigo
Low-level software	Orocos Real-Time Toolkit
World model	Environment Descriptor (ED), custom
Localization	Monte Carlo using Environment Descriptor (ED), custom
SLAM	Gmapping: `http://wiki.ros.org/gmapping`
Navigation	Global: custom A* planner
	Local: modified ROS DWA
Arm navigation	Custom implementation using MoveIt and Orocos KDL
Object recognition	Tensorflow ROS
People detection	Custom implementation using contour matching
Face detection & recognition	Openface ROS
Speech recognition	Dragonfly + Windows Speech Recognition
Speech synthesis	Philips Text-to-Speech
Task executors	SMACH `http://wiki.ros.org/smach`

3.1 Focus of research and scientific contributions

The TUe Control Systems Technology (CST) group has an internationally recognized reputation. CST targets areas in precision machines, robotics, biomedical, agriculture and automotive engineering. New controller and observer synthesis methods are being developed and applied to create innovative intelligent systems. The recent scientific visitation of the group resulted in the highest achievable score (excellent).
Specific fields of robotics research are: world modelling, context awareness, motion planning, whole-body motion control, active perception, sensor fusion, distributed control (swarms).
CST was coordinator of the world wide acclaimed FP7 project RoboEarth, proposing a system design for the internet of robots. The Tech United team directly benefits from currently running robotics research projects: R5COP (resilient and reconfigurable robot SW/HW), EurEyeCase (robot assisted eye-surgery), ROPOD (warehouse robots) and AUTOPILOT (autonomous driving).
Our recent research efforts have focused on i) fitting furniture objects to update our object-oriented world model, thereby improving localization, navigation and object segmentation ii) developing a WebGUI to provide the user with a platform-independent way to interact with the robot, iii) natural language interpretation to ease the specification of written commands for the robot and to make speech recognition more robust and iv) improved image recognition using neural network training via tensorflow_ros.
Our recent scientific contributions can be found on the Tech United website

3.2 Re-usability of the system for other research groups

Tech United takes great pride in creating and maintaining open-source software and hardware to accelerate innovation. Tech United initiated the Robotic Open Platform website, to share hardware designs. In terms of software, all our repositories are available on GitHub. All packages are equipped with documentation and tutorials. Recently, our wrapper for openface was forked by the Toyota Research Institute. Moreover, Tech United and its scientific staff have the capacity to co-develop (+10 people), maintain and assist with questions.

3.3 Applicability of the approach in the real world

For us, science doesn't stop at software simulation. At TUe, our test infrastructure contains real life environments. Here, different room types can be simulated (i.e. home, care institution) and real life products are used. In addition, Tech United carries out many demonstrations per year at different indoor locations throughout the Netherlands. This ensures that our concepts keep being tested. At this point our system are able to operate in dynamic environments used by humans.

3.4 Community Outreach and Media

The Tech United team carries out many promotional activities to promote technology and innovation with children. These activities are carried out by separate teams of student assistants. Tech United often visits primary and secondary schools, public events, trade fairs and have regular TV performances. In 2015 and 2016 together, 100+ demos were given and an estimated 50k were reached through live interaction. Tech United has also got a very active (website, and interacts on many social media mediums: Facebook, YouTube, Twitter and Flickr. Our robotics videos are often shared on the IEEE video Friday website.