Project Acronym:	ACAT
Project Type:	STREP
Project Title:	Learning and Execution of Action Categories
Contract Number:	600578
Starting Date:	01-03-2013
Ending Date:	30-04-2016

Deliverable Number:	D6.3
Deliverable Title:	Repositories of software, data bases and benchmarks.
Type (Internal, Restricted, Public):	PU
Authors :	L. Bodenhagen, D. Nyga, B. Ridge, D. Vitkutė-Adžgauskienė, M. Tamošiūnaitė, N. Krüger
Contributing Partners:	SDU, UoB, JSI, UGOE

Contractual Date of Delivery to the EC:	29-02-2016
Actual Date of Delivery to the EC:	02-03-2016

Contents

Executive Summary

With this deliverable we provide an overview over the different software components and benchmarks that have been developed during the ACAT project and are made accessible to the public. In total 10 software components and 3 benchmarks with 18 distinct performance indicators are available now.

1 Introduction

In the following we provide a brief overview on the individual software components and benchmarks that have been developed during the ACAT project. The software components can be categorized mainly in three different types: 1) tools for handling ADTs, typically based on scripts (section 2), 2) tools for processing instructions and deriving plans for robot actions, typically accessible by web-interfaces (section 3 and 4), and 3) software related to the execution of robot actions, typically libraries (sections 5 and 6). Software components that are not made available to the public, e.g. internal repositories, are not listed in this deliverable.

2 Tools for ADT handling

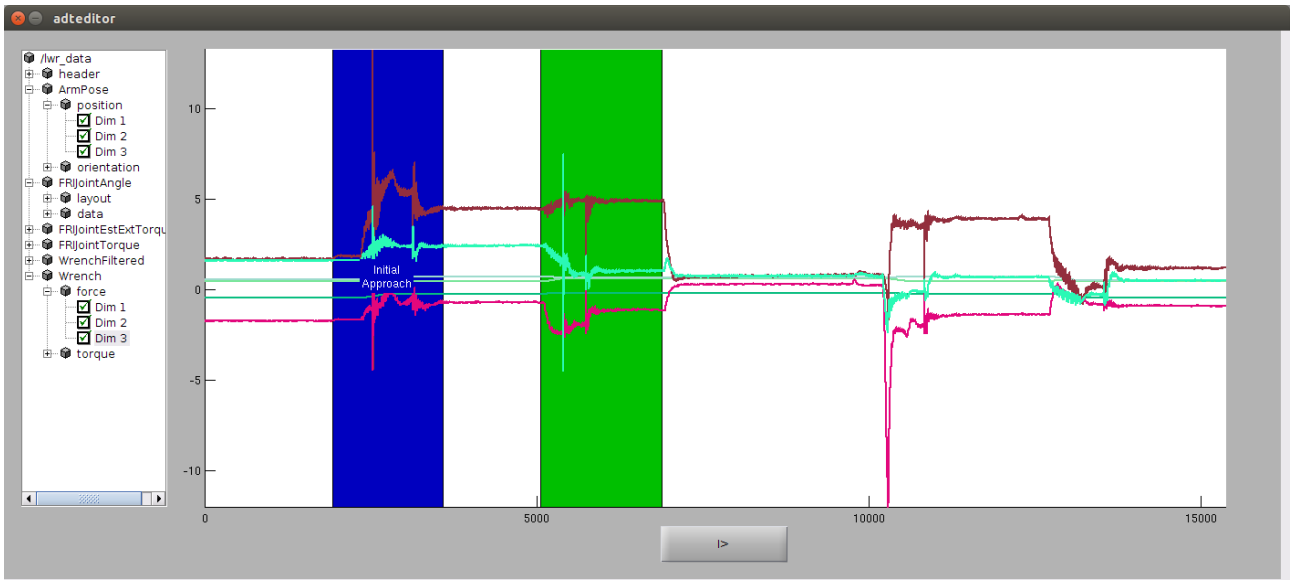

Figure 1: The ADT GUI editor with a sample rosbag being used. The blue selection indicates an action chunk, the green selection is made by the user.

A collection of tools for handling Action Data Tables (ADTs), in particular creating, filling or verifying ADTs, have been developed during ACAT. Two of these tools are organized in a suite which can be downloaded using github[1] (detailed information and examples are given in D1.2, figure 1 shows the main interface) and comprises the following:

ADT XML Tool facilitates the generation of new, or population of existing ADT XML files using ROS bag recordings. The ADT Tool creates a SEC matrix based on topics in the rosbag indicating whether objects touch or not and includes action chunk timestamps as well as information on object poses if available.

ADT GUI Editor acts as an interface to ADT Tool and can be used to both visualize the content of the rosbag that is associated with an ADT and to annotate the content.

Furthermore, the following tools are accessible separately:

[1] `https://github.com/barryridge/acat-adt-tools`

Extraction of movement primitives can be downloaded using git[2]. With this tool Movement primitives are extracted using rosbag data on trajectories and forces and are based on signal property changes inside action chunks of the ADT.

The following movement primitives are extracted:

- $arm\_move(goal\_pose)$: Move the robot arm to the pose defined by $goal\_pose$
- $arm\_move\_periodic(V, \omega)$: Move the robot arm periodically along the direction of V with frequency ω.
- $exert(f)$: Exert the force equal to f at the robot end effector.
- $hand\_move(goal\_angles)$: Move the robot hand to the configuration specified by joint angles denoted by $goal\_angles$
- $grasp()$: Grasp the object which is inside the fingers of the hand
- $ungrasp()$: Ungrasp the previously grasped object.

Script for transforming Open-EASE data into ADTs can be downloaded using git[3]. This is Python code for transforming action logs, collected at UoB, from the OWL format into ADTs. Contact events and trajectory information are automatically extracted from the logs and an ADT is generated.

[2] `https://git.physik3.gwdg.de/ACAT/movement_primitives`
[3] `https://git.physik3.gwdg.de/ACAT/adt_transforms`

3 Open-EASE

OpenEASE is a web-based knowledge service providing robot and human activity data. It contains semantically annotated data of manipulation actions, including the environment the agent is acting in, the objects it manipulates, the task it performs, and the behavior it generates. The episode representations can include images captured by the robot, other sensor datastreams as well as full-body poses. A powerful query language and inference tools, allow reasoning about the data and retrieving requested information based on semantic queries. Based on the data and using the inference tools, robots can answer queries regarding to what they did, why, how, what happened, and what they saw.

In section 3.1 we give a brief overview on the chemistry experiments (details can be found in D5.5) for which the knowledge base is accessible and section 3.2 covers the natural-language understanding which is available via a web-interface. Furthermore, an extension of the system for execution action plans is outlined in section 3.3 and a tools for learning and reasoning are covered in section 3.4.

3.1 The ACAT Chemistry Experiment

The ACAT chemistry experiment is centered around robotic assistants in chemistry labs and consists of different actions performed by Raphael, our PR2 robot:

- Fetching and placing of chemistry tools
- Pipetting of liquids
- Screwing and unscrewing of bottle and tube caps

Chemical laboratories are very structured environments which host activities that should be executed according to precise procedures. The manipulation of dangerous substances for different experiments makes chemical laboratories hazardous. These aspects make autonomous mobile robots ideal candidates for performing chemical experiments. Our robot successfully performed several steps of a DNA extraction procedure for "Ocean Sampling Day", an event organized by "Micro B3", a project aiming to stimulate the analysis and indexation of the whole planetary ocean DNA.

Typically, the instructions of a particular chemical experiment are described in natural language. Our autonomous mobile robot relies on a couple of mechanisms for performing an experiment from such a description. Initially, it infers the meaning of each natural language instruction to be performed and uses the background knowledge to chose and parametrize the most appropriate plan from the plan library for each given instruction. Then each chosen plan queries the vision system for the objects to be manipulated and runs the constraint motion controller to perform the necessary motions. Finally the logging mechanism records all details concerning how was the chemical experiment performed.

The logs can be used to answer a wide range of questions concerning the performed experiment, improving the way future experiments can be performed. The Knowledge Base for this experiment is accessible through the web platform[4].

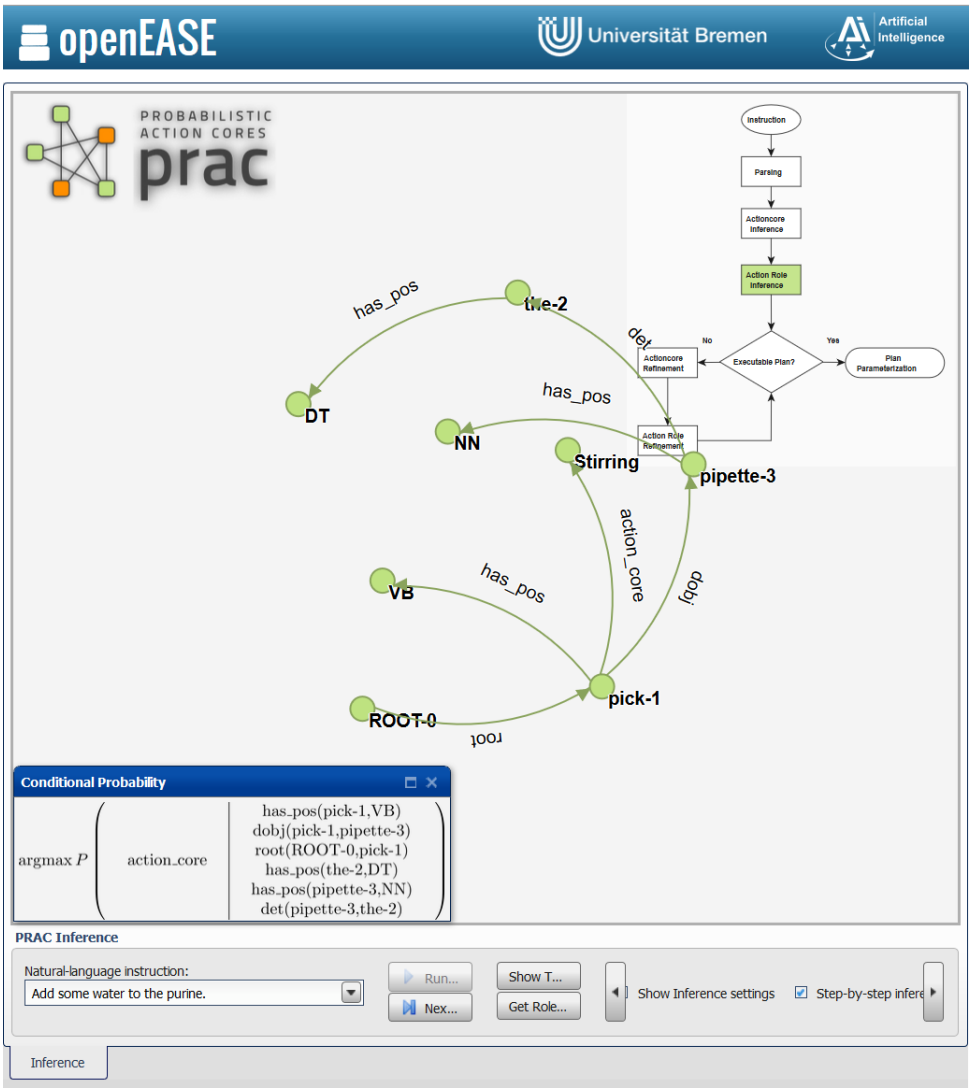

Figure 2: Screenshot of PRAC interface processing the instruction "Add some water to purine"

3.2 Natural-language understanding for intelligent robots

Knowledge about actions and objects is represented as Probabilistic Robot Action Cores (PRAC) [R3], which can be thought of as generic event patterns that enable a robot to infer important information that is missing in an original natural-language instruction. PRAC models are represented in Markov Logic Networks, a powerful knowledge representation formalism combing first-order logic and probability theory. On the openEASE web platform an implementation of PRAC can be directly accessed[5], figure 2 shows a screenshot of the web interface.

[4] `http://www.open-ease.org/robotic-agent-performing-chemical-experiments-overview/`
[5] `https://data.open-ease.org/prac/pracweb`

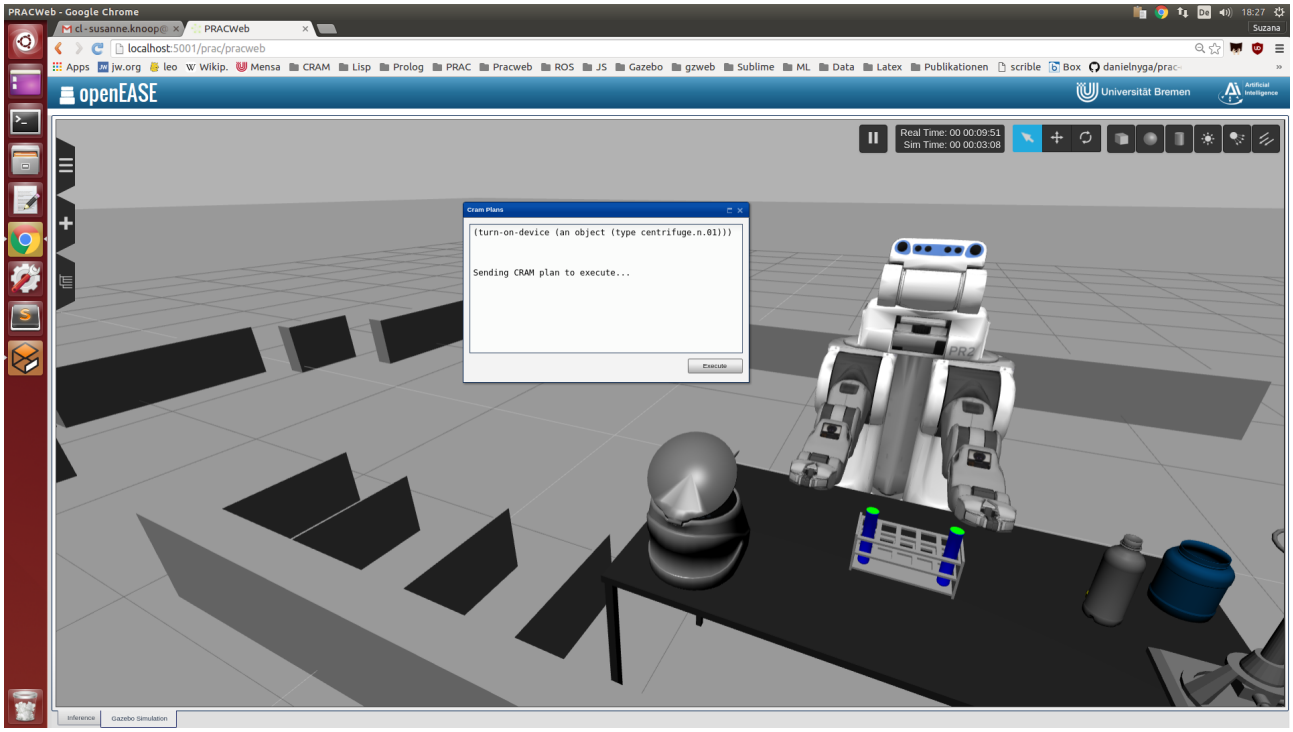

Figure 3: Screenshot of the PRAC system executing the natural-language instruction "start the centrifuge" in the browser-based Gazebo simulator.

3.3 Execution of natural-language instructions in a robot simulator

The PRAC reasoning system has been extended by a connection to the CRAM[6] plan executive, which is able to interpret the formal robot plans generated by PRAC. CRAM executes the generated robot plans either on a real robot or in the Gazebo[7] simulator. In order to make the natural-language instructions directly executable in PRAC, the browser-based web interface to PRAC was enriched with the *GzWeb* extension of Gazebo, a JavaScript interface for making Gazebo accessible from a browser. A screenshot the PRAC system executing a natural-language instruction in the web simulator is shown in figure 3.

3.4 Tools & algorithms for statistical relational learning

pracmln[8] is an open source toolbox for statistical relational learning and reasoning and as such also includes tools for standard graphical models that has been developed in context of ACAT and represents the learning and reasoning engine of PRAC. pracmln is a statistical relational learning and reasoning system that supports efficient learning and inference in relational domains.

pracmln was designed with the particular needs of technical systems in mind. Our methods are geared towards practical applicability and can easily be integrated into other applications. The tools for relational data collection and transformation facilitate data-driven knowledge engineering, and the availability of graphical tools makes both learning or inference sessions a user-friendly experience.

[6] `http://www.cram-system.org`
[7] `http://www.gazebosim.org`
[8] `http://www.pracmln.org`

Scripting support enables automation, and for easy integration into robotics applications, we provide a client-server library implemented using the widely used ROS (Robot Operating System) middleware.

4 Compiler for textual instructions

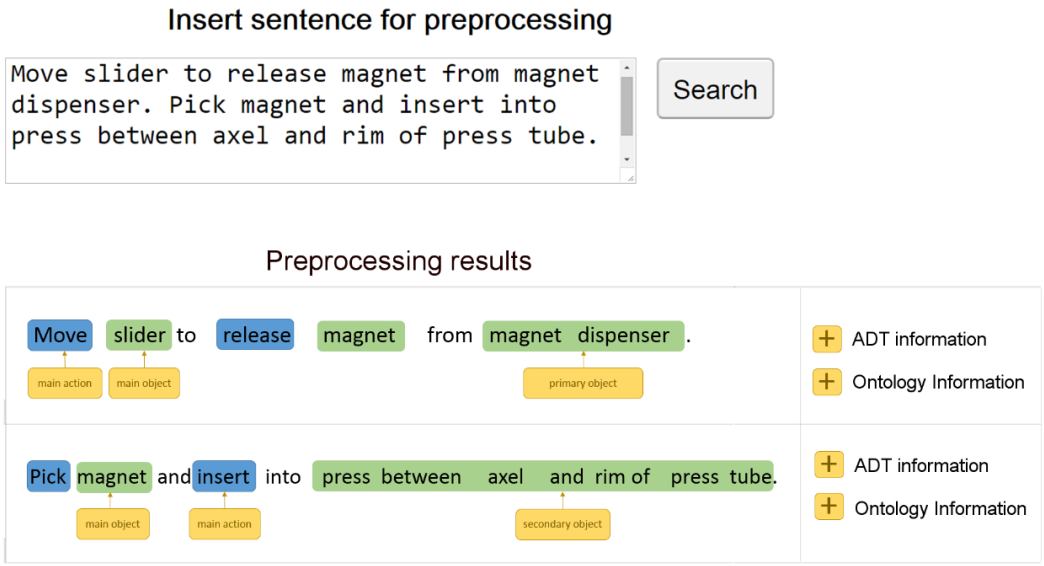

Figure 4: First step of textual instruction compiler – parsing of a textual instruction.

The textual compiler developed in the ACAT project is accessible through a web-interface[9] (illustrated in figure 4). The interface allows to enter textual instructions into a web page form, and displays the results of the textual instruction compiler:

- semantic roles, recognized in the instruction sentence - main action, main object, primary object, secondary object
- missing action related information extracted from the ontology
- instruction-specific data by means of Action Data Tables (ADTs)

The initial step for the compilation is to enter the instruction in the compiler dialog window. The following step of the compiler execution consists of the parsing of the instruction. Results for a sample instruction are shown in figure 4.

Further, in the next step, links to Action Data Tables (ADTs) that are related to semantic role keywords, identified in the instruction, are provided (figure 5). The best matching ADT is used to form an "ADT Blueprint", an ADT pre-filled with symbolic information for the corresponding instruction. All ADTs are directly downloadable via this interface.

Detailed information on the textual instruction compiler is provided in deliverable D3.1.

[9] http://strazdas.vdu.lt:8080/

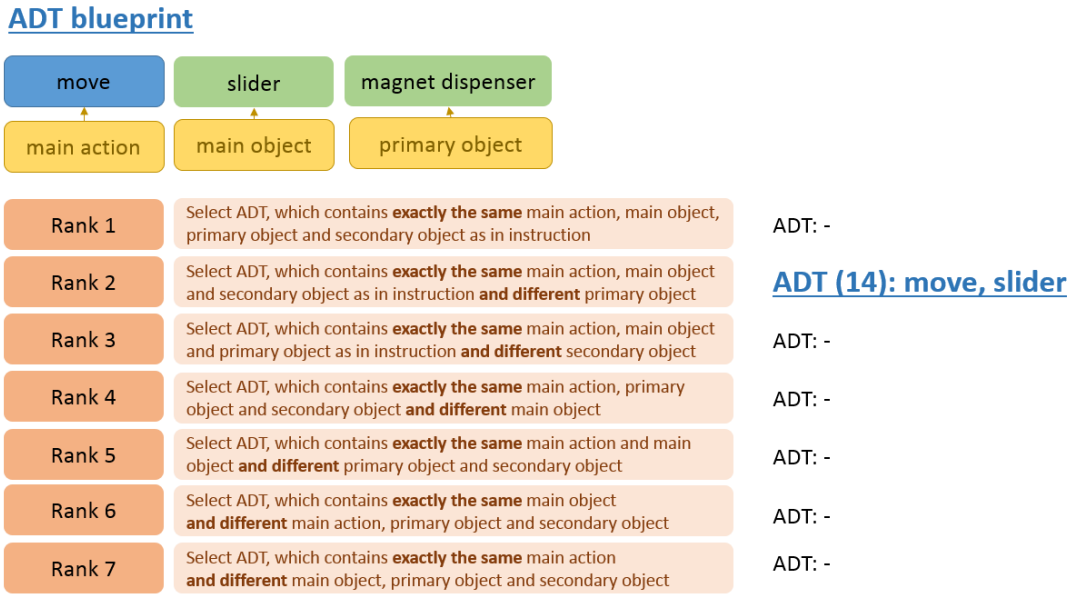

Figure 5: Links to corresponding ADTs provided by the compiler.

5 Software for computer vision – CoViS

The Cognitive Vision Software (CoViS) is the implementational basis for the Vision system which is being developed at SDU. The CoViS library has thus been utilized in various contexts ranging from drivers assistance to robotic manipulation and is also the basis for the work on pose estimation [R2] done in the ACAT project (mainly associated to WP4). CoViS can be downloaded free of charge from GitLab[10].

6 Software for robotics and simulation of robotic processes

RobWork [R1] is a collection of C++ libraries, accessible at RobWork.dk[11] for simulation and control of robot systems. It provides amongst others tools for kinematic modelling of various types of robots, path-planning and optimization, and grasp planning. In addition a graphical user interface is provided (RobWorkStudio) as well as a dynamic simulator (RobWorkSim). In ACAT in particular the simulation tools have been applied and developed which was the basis for e.g. the learning of task-specific grippers[R4]. While we in ACAT utilized several workstations running in parallel (see also D5.3), the software can also be used on a standard computer without modifications.

[10] https://gitlab.com/caro-sdu/covis
[11] http://www.robwork.dk

7 Benchmarks

In the ACAT project benchmarks, focusing on following three different aspects, have been defined (see also deliverable D5.2-update for further details on benchmarks and key performance indicators):

1. End-User oriented benchmarks
2. Language oriented Benchmarks
3. Basic Research-Oriented Benchmarks

While End-User oriented benchmarks are targeting aspects relevant to industry, the two other benchmarks are rather relevant in an academic context. In the following we provide an overview of the three benchmark groups and the associated key performance indicators (KPIs). The KPIs are provided on the ACAT web-page[12] and will be filled with concrete results achieved by the ACAT project as soon as benchmark evaluations are completed.

End-User oriented benchmarks focus on the degree of involvement of the end-users, e.g. the operator of the robot. Various key-performance indicators target different types of interaction such as the setup-time or the robustness of the system. Obviously, lower setup-times and fewer interactions for compensating failures during run-time imply that the system is more valuable for the end-user.

The key-performance indicators for this benchmark group are defined as:

KPI1.1a Setup time for execution of a known task

KPI1.1b Setup time for execution of a known task with unreliable pose inputs

KPI1.1c Setup time for execution of a semi-known task with unreliable grasp inputs

KPI1.1d Setup time for execution of a semi-known task with similar model information

KPI1.2 Robustness during setup

KPI1.3 Robustness during execution

KPI1.4 Cycle time during execution

KPI1.5 Training time required

KPI1.6 Demonstration efficiency

KPI1.7 User-Friendliness: The combination of training time, the setup time and the demonstration efficiency define the user friendliness of the system.

[12] `http://www.acat-project.eu/index.php?page=benchmarks`

Language oriented benchmarks target the domain-oriented ontologies which form a backbone in the ACAT project. The ontologies are formed by extracting verbs and associated objects from textual sources such as instruction sheets, documentation or guides. If available, the texts can be supplemented with additional information, e.g. image or shape information. Since a successful processing of new instructions requires relevant information to be available in the domain specific ontology, the amount of information reflected in the ontology indicates the applicability of it. The key-performance indicators therefore address e.g. the number of action verbs the associated objects and the number of filled ADTs stored in the process memory.

The key-performance indicators for this benchmark group are defined as:

KPI2.1 Linguistic action ontology: Number of action verbs in the ontology and number of synsets

KPI2.2 Object categories: Number of object categories saved in the process memory

KPI2.3 Number of action grounding instances

KPI2.4 Action categories: Number of ADTs saved in the process memory

Basic Research-Oriented Benchmarks target the benchmarking of research platforms, in particular cognition-enabled robots, where the measurements of e.g. execution times or robustness are not indicative for the capabilities if the system. Therefore, these benchmarks rather focus on the ability of the system to interpret potentially incompletely formulated tasks correctly or to communicate the understanding of the scene as well as the consequences of executing an action in this scene. Since such abilities are difficult to quantify, the key-performance indicators rather build on directing a set of request to the system and identify how often e.g missing information on objects or or quantities could be inferred correctly.

The key-performance indicators for this benchmark are defined as:

KPI4.1 Causal relations correctly understood

KPI4.2 Vague quantities: Inferring vaguely formulated quantities

KPI4.3 Missing objects: Inferring missing objects and roles

KPI4.4 Disambiguation: Inferring correct meanings of ambiguous words

8 Summary

With this deliverable we provide an overview of the different software tools and libraries, which have been developed or extended during the ACAT project. Table 1 provides a concise overview of the different components mentioned in this deliverable and indicates links to the relevant work packages. In total 10 software components have been made available.

Furthermore, this deliverable provides a concise overview of benchmarks defined in ACAT project and indicates the web address on the ACAT web-page where the results of benchmarking will be submitted before the end of the project.

Titel	Partner	Related WP	Source
CoViS	SDU	WP4	Git repository
RobWok	SDU	WP2, WP5	SVN repository
ADT Tool Suite	JSI	WP1	Git repository
OpenEASE	UoB	WP3, WP4	Web-interface
pracmln – Markov logic networks in Python	UoB	WP2, WP3	Git repository
PRAC – Probabilistic Action Cores	UoB	WP3	Web-interface
Textual Compiler	VDU	WP3	Web-interface
Extraction of movement primitives	UGOE	WP1	Git repository
OWL to ADT conversion	UGOE	WP1	Git repository

Table 1: Overview over publicly available components.

References

[R1] L.-P. Ellekilde and J. A. Jorgensen. "RobWork: A Flexible Toolbox for Robotics Research and Education". In: *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK)* (2010), pp. 1 –7.

[R2] L. Kiforenko, A. G. Buch, and N. Krüger. "Object Detection Using a Combination of Multiple 3D Feature Descriptors". In: *Computer Vision Systems*. Ed. by L. Nalpantidis, V. Krüger, J.-O. Eklundh, and A. Gasteratos. Lecture Notes in Computer Science. Springer International Publishing, 2015, pp. 343–353.

[R3] D. Nyga and M. Beetz. "Cloud-based Probabilistic Knowledge Services for Instruction Interpretation". In: *International Symposium of Robotics Research (ISRR)*. Sestri Levante (Genoa), Italy, 2015.

[R4] A. Wolniakowski, J. A. Jorgensen, K. Miatliuk, H. G. Petersen, and N. Kruger. "Task and context sensitive optimization of gripper design using dynamic grasp simulation". In: *Methods and Models in Automation and Robotics (MMAR), 2015 20th International Conference on*. 2015, pp. 29–34.