About us

The Institute for Artificial Intelligence (IAI) at the University of Bremen, Germany, headed by Prof. Michael Beetz, investigates methods for cognition-enabled robot control. The research is at the intersection of robotics and Artificial Intelligence (AI) and includes methods for intelligent perception, dexterous object manipulation, plan-based robot control, and knowledge representation for robots.
Robots performing complex tasks in open domains, such as assisting humans in a household or collaboratively assembling products in a factory, need to have cognitive capabilities for interpreting their sensor data, understanding scenes, selecting and parameterizing their actions, recognizing and handling failures and interacting with humans. In our research, we are developing solutions for these kinds of issues and implement and test them on the robots in our laboratory. A particular focus of the group is on the integration of individual methods into complete cognition-enabled robot control systems and the release of the developed software as open-source libraries.
More information about the institute, the team and our research projects can be found on our website: http://ai.uni-bremen.de/
Two software libraries that origin from our laboratory and that are now used and supported by a larger user community are the KnowRob system for robot knowledge processing and the CRAM framework for plan-based robot control.

KnowRob – Robot Knowledge Processing KnowRob is a knowledge processing system that combines knowledge representation and reasoning methods with techniques for acquiring knowledge from different sources and for grounding the knowledge in a physical system. It provides robots with knowledge to be used in their tasks, for example action descriptions, object models, environment maps, and models of the robot's hardware and capabilities. The knowledge base is complemented with reasoning methods and techniques for grounding abstract, high-level information about actions and objects in the perceived sensor data.

KnowRob became the main knowledge base in the ROS ecosystem and is actively being used in different academic and industrial research labs around the world. Several European research projects use the system for a wide range of applications, from understanding instructions from the Web (RoboHow), describing multi-robot search-and-rescue tasks (SHERPA), assisting elderly people in their homes (SRS) to industrial assembly tasks (SMERobotics).
KnowRob is an open-source project hosted at GitHub[1] that also provides extensive documentation on its website – from getting-started guides to tutorials for advanced topics in robot knowledge representation[2].

[1] http://github.com/knowrob
[2] http://www.knowrob.org/

CRAM – Robot Plans CRAM is a high-level system for designing and performing abstract robot plans to define intelligent robot behaviour. It consists of a library of generic, robot platform independent plans, elaborate reasoning mechanisms for detecting and repairing plan failures, as well as interface modules for executing these plans on real robot hardware. It supplies robots with concurrent, reactive task execution capabilities and makes use of knowledge processing backends, such as KnowRob, for information retrieval.

CRAM builds ontop of the ROS ecosystem and is actively developed as an open-source project on GitHub[3]. It is the basis for high-level robot control in many parts of the world, especially in several European research projects covering applications from geometrically abstract object manipulation (RoboHow), multi-robot task coordination and execution (SHERPA), experience based task parameterization retrieval (RoboEarth), and safe human robot interaction (SAPHARI).
Further information, as well as documentation and application use-cases can be found at the CRAM website[4].

In our group, we have a very strong focus on open source software and active maintenance and integration of projects. The systems we develop are available under BSD license, and partly (L)GPL.

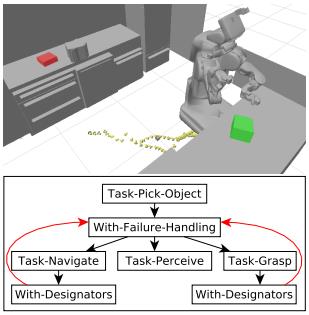
[3] http://github.com/cram-code
[4] http://www.cram-code.org/

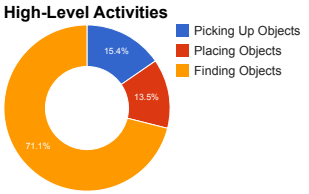
Proposed Topics for Summer of Code-Projects

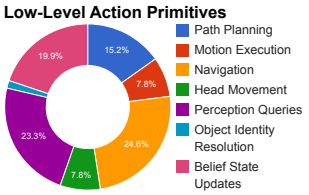
In the following, we list four proposals for Summer of Code topics that contribute to the two aforementioned open-source projects CRAM and KnowRob.

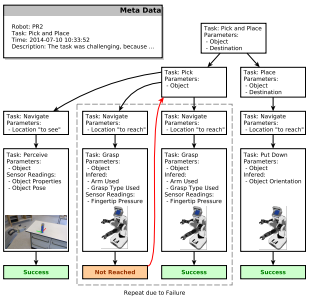
Topic 1: CRAM – Report Generation from Robot Mobile Manipulation Activities

Main Objective The development of algorithms for intelligently interpreting, comprehending, and filtering complex episodic memories of robot activities. The activities to analyze feature a real mobile robot (Willow Garage PR2) that performs mobile manipulation tasks (fetch and place). The algorithms to develop should generate a PDF report card featuring the most important tasks in the experiment (varies with activities), their outcome, experiment statistics, and the overall course of action.
The goal is to allow researchers and developers to get the most important or appealing information of complex experiments at a glance, and to identify potential problems or anomalies in robot behaviour.
This involves developing Prolog predicates for analyzing the experiment logs, and writing functions in Java for LaTeX code generation. The produced code will be part of the larger CRAM system and will have a large impact on its daily usage for many robotics researchers world-wide.







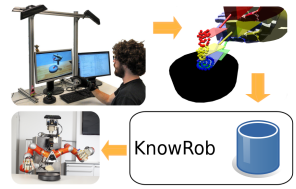


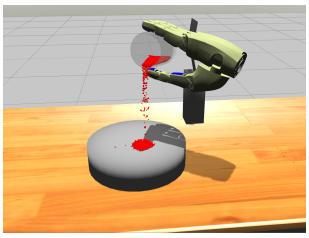
Task Difficulty The task is to be placed in the intermediate difficulty level, as it requires forming an intuition for log data of arbitrary experiments in the domain of mobile fetch and place.

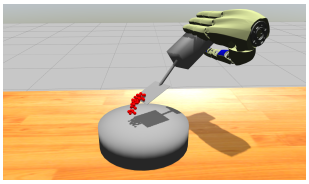
Requirements Basic programming skills in Java and Prolog. Basic knowledge in ROS helps, as the CRAM system is built upon ROS. Experience with autonomous robots helps.

Expected Results We expect operational and robust algorithms that produce experiment comprehensions, and are easily maintainable and extendable.

Topic 2: Kitchen Activity Games – Developing a GUI to interact and learn from a robotic simulator







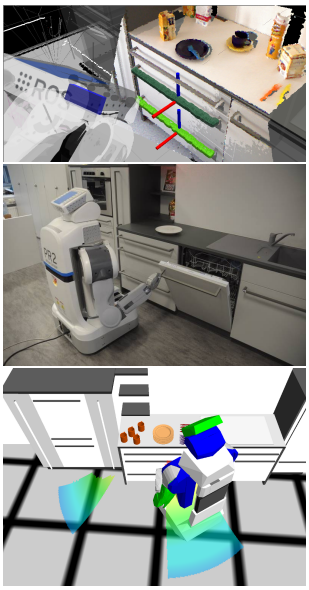
Main Objective Creating a gamelike Graphical User Interface to interact with the robotic simulator Gazebo[5]. The simulator will be used as a library, from which different scenarios (worlds) can be selected and run. Every scenario would have a description and a given task to be performed that will be shown to the user.
After finishing a scenario, the game logs should be able to be stored, deleted or post processed. When post processing is selected, a given algorithm will run on the data and it will return a score for the user, showing how well he did in the task. Further, the physics related data will be stored and used to teach robots common sense knowledge.
The goal is to have a system that easily allows non expert users to get familiar with our simulation environment and to be able to execute everyday manipulation tasks from which robots can learn.

Task Difficulty The task is to be placed in the easy difficulty level, as it requires less algorithmic knowledge and more programming skills.

Requirements Good programming skills in C++. Good knowledge of the Gazebo API. Experience with QT GUI programming.

Expected Results We expect to have a Graphical User Interface that allows users to log into our simulation based game, select from different scenarios and being able to execute the given tasks. After finishing a scenario, post processing will be run on the logged data, that will return a score for the user and physics related data for robots to learn from.

Topic 3: CRAM – Symbolic Reasoning Tools with Bullet



Main Objective Extending a cognitive robot control framework (written in modern Common Lisp (SBCL) and partly C++) by mapping the environment of the robot to its internal world representation and tracking the changes in the environment to keep the internal world state up to date while the robot performs manipulation tasks in human environments.
Possible sub-projects:

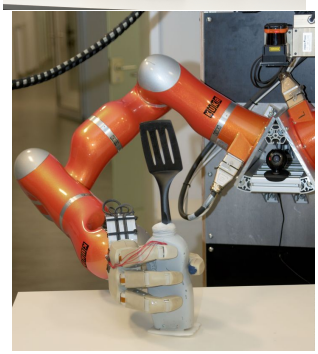
- assuring the physical consistency of the belief state by utilizing the integrated Bullet physics engine, i.e. correcting explicitly wrong data coming from sensors to closest logically sound values;
- improving the representation of previously unseen objects in the belief state by, e.g. interpolating the point cloud into a valid 3D mesh;
- improving the visualization of the belief state and the intentions of the robot in RViz (such as, e.g., highlighting the next object to be manipulated, printing textual information about the goals, etc.)
- update the internal world state to reflect the changes in the environment such as "a drawer has been opened" (up to storing the precise angle the door is at after opening), and emitting corresponding semantically meaningful events, e.g. "object-fell-down" event is to be emitted when the z coordinate of the object drastically changes;
- many other ideas that we can discuss based on applicants' individual interests and abilities.

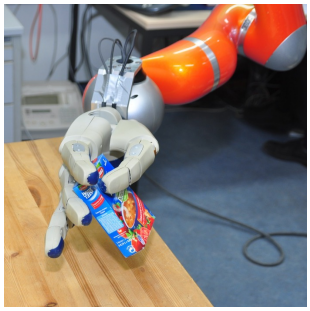
Task Difficulty Moderate, considering that most of the sub-projects deal with diving into a small chunk of an already existing system and extending it from within, as opposed to developing plug-ins and similar.

Requirements Familiarity with functional programming paradigms. Minimal functional programming experience (preferred language is Lisp but experience in Haskel, Scheme, OCaml, Clojure or any programming language in general that supports functional programming, as, e.g., Julia or Java 8, will do). Experience with ROS (Robot Operating System) is a plus. Good understanding of geometric shapes and coordinate system transformations is a plus.

Expected Results We expect operational and robust contributions to the source code of the existing robot control system including minimal documentation and test coverage.

Topic 4: Multi-modal Big Data Analysis for Robotic Everyday Manipulation Activities





Main Objective Consider an autonomous robot preparing breakfast pancakes. In order to perform such everyday manipulation activities the robot's control program needs powerful perception algorithms: It needs to see where a spatula is to grasp it, feel when its fingers made contact, detect the load change in its joints, know where its arm is in the world, and integrate all these information to verify that it successfully grasped a spatula.
We propose to guide a Google Summer of Code participant through the process of extending an action perception system which can answer question about robotic everyday manipulation episodes. We will provide the student with MongoDB logs of the sensory data produced by our high-end research robot Boxy `http://ai.uni-bremen.de/research/robots/boxy` during several manipulation episodes. The existing prototype action perception system which the student shall extend uses the Open-Source Apache UIMA Framework `https://uima.apache.org/`.
The action perception system shall answer question like this:

- Which of the cooking tools did you grasp first?
- Which of your fingers made first contact with the spatula?
- Did all of your fingers feel contact during that grasp?
- What was the heaviest tool you grasped?
- During which transport motion did you feel slippage?

The datasets contain the raw haptic, and proprioceptive sensory data produced from the robot's arms, hands, and force/torque sensors. Additionally, the recordings contain the output of the visual object perception pipelines of the robot. Finally, we provide video recordings of Boxy performing its tasks to help the student understand the content of the datasets.
The existing prototype follows the unstructured information management architecture (UIMA) paradigm for building multi-modal analysis engines. So far, the prototype only considers the proprioceptive data of the robot's arm. Following that example, we will guide the student to add routines to process the data from the robot's hands and force/torque sensors, and then leverage UIMA's capabilities for multi-modal integration.

Task Difficulty This is a challenging but exiting task. It asks the participant to learn developing within the complex UIMA software framework. Additionally, the developer has to familiarize herself with the basics of the robotics domain to implement meaningful analysis algorithms.

Requirements The ideal participant is cocomfortablee with programming in Java and has passed an introductory course on Linear Algebra. Less importantly, she would have prior experience in using github and know the basics of robotics.

Expected Results Contributions to our Java prototype in the form of documented and tested library functions for basic hand and force/torque sensory data processing and multi-modal action analysis. A nice optional result would be utilities to visualize the answers calculated by the analysis engine.