Research Methods Intelligent Robot Systems

Maj Stenmark, maj.stenmark@cs.lth.se

June 22, 2016

1 Thesis Topic: Intelligent Robot Systems

My PhD thesis work is carried out in the Robotics and Semantic Systems group at the Computer Science department at Lund University. The main objective of the thesis is to make industrial robot programming faster and simpler. At present, human workers in low-cost countries carry out manufacturing tasks because the cost of automation is too high. While the hardware of industrial robots is relatively cheap (the latest dual-arm robot costs roughly 500,000 SEK), the sensor integration and programming is time-consuming and requires skilled engineers. Hence, the RobotLab at Lund University has participated in several European research projects aimed at automating robot programming so that the shop-floor worker can deploy a new robot task within a day (I've participated in the EU projects Rosetta, PRACE, SaraFun). When approved, the projects have a *Description of Work* which specify in detail what solutions to implement and what functionality to demonstrate.

Experimental software is implemented and tested in the robotics lab using the ABB dual-arm robot YuMi, it's predecessor Frida shown in Fig. 1 and other small industrial ABB robots (IRB120 and IRB140).

The methodology used is more geared towards engineering or technoscience than the classical scientific method from natural sciences. However, it is possible to set up hypotheses and falsify them: e.g, *When programming robots, it is more efficient for humans to guide the robot physically and using speech than using only a graphical user interface.* However, it is necessary to add the specifics of the experimental implementation, for example *using a non-scary-looking robot, English with British accent and compare this to a terrible user interface.* The details of the experimental setup are so many, that general claims are very hard to make. A competing research group can create a nice-looking user interface and have a ten minute tutorial and come up with a contradicting result. An additional problem is that reusability of software solutions between research groups is extremely limited, and repeating other's research is, if not impossible, not valued from the community.

1.1 Formal Methods

The investigated research problems have been determined by the projects, the idea of the Rosetta project was to let experts create advanced robot skills that could be parameterized and reused. In order to reuse robot skills these had to be described using an *ontology* that represents *objects* and *actions*. In this case objects refer to physical work pieces that the robot has to assemble, sensors and different types of robots, and actions are motions and computations carried out by a robot (or a human). The descriptions are created from case studies of commonly used assembly scenarios. This is a *formal* or *mathematical* approach were logical symbols are created and reasoned upon automatically.

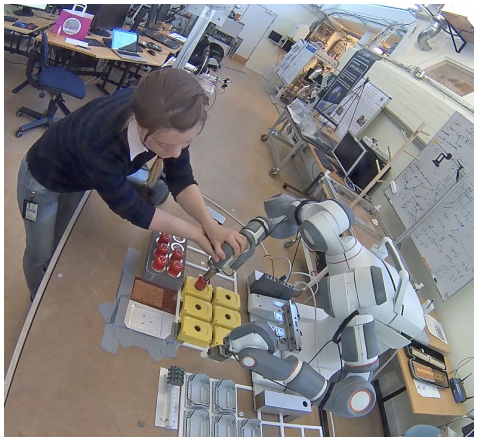

Figure 1: Recording positions for a pick and place task using an ABB dual-arm robot.

Experiments were carried out in the lab to create robot action representations and transfer tasks between robots, but these experimental implementations were intended as a *proof of concept* rather than actually evaluating the expressiveness of the ontology. The current EU project focus on extracting a task representation from human demonstrations (of the assembly) and generating code for the robot system. The usability of the representation will be evaluated in a user study.

1.2 System Design and Implementation

In order to make our robots "smarter" we developed a distributed software architecture. Components of the system were the physical robot, the real-time robot control system, a programming environment and a so-called knowledge server where data and (modular) ontologies were stored and heavy computational services were carried out. The system itself has been presented at workshops for e.g., cloud-based robotics systems, but for publications we only look at one small piece at a time.

First, we implemented a graphical user interface and generated executable (force) sensor-based programs for the robot. This reduced the amount of code the user had to input to the system significantly and all parameters had human readable descriptions. Here, the system could be evaluated using both qualitative and quantitative criteria using a few experimental tasks (running on both simulated and real robot systems).

Then I added natural (human!) language support for English and Swedish so that the user could describe the task in unstructured language and all parameter values were initialized with default values. The correctness of the language understanding was evaluated using a small set of (a few hundred) sentences with instructions of varying grammatical structure and "difficulty". In this context, when sentences describe multiple conditions and actions in parallel (*While moving in the x-direction until contact or timeout hold 4 N in the z-direction and ...*) it can be difficult to extract all conditions. When I had both English and Swedish natural language programming implemented I could compare those two, but nobody else had an application that I could compare my work to. Hence, one of the challenges in my filed is to find good metrics.

2 Current Research Methods

In September, we'll carry out a user study to evaluate the usability of the programming system and the representation of the robot skills. We'll measure the time it takes for the users to program a small assembly task and by comparing two groups, one that reprograms the same task from scratch using slightly different parts, and another that reuse and adjust the skill that they've created. We can also measure the robustness of the programs (success rate) and get feedback on our system by letting them answer a survey. Most importantly, we'll have an indication wether robots need to "understand" their programs with an abstract representation of objects and actions, or if this can be replaced by a nice user interface. The user study also has an explorative aspect, since we will observe unexperienced robot programmers interacting with the system and collect their feedback.

3 Summary Research Methods

To formulate concepts and models we start with case studies. Then we formulate a framework and conceptual models that are evaluated using laboratory software experiments. Current research methods involve laboratory experiments with humans (although not on humans!) and explorative surveys.

The field suffers from a methodological problem since software solutions are seldom shared and thus it is difficult to repeat experiments. Since the field develops artifacts and models, it's a challenge to come up with suitable metrics and evaluation criteria, especially general benchmarks.