iProgram: Intuitive Programming of an Industrial HRI Cell

Jürgen Blume,
Alexander Bannat and Gerhard Rigoll
Institute for Human-Machine Communication
Technische Universität München
80333 Munich, Germany
Email: blume@tum.de

***Abstract*—This paper introduces a concept for intuitive programming of an industrial HRI cell for non-experts. The main idea of this concept is to combine recently available technologies ranging from speech recognition, 3D visual surveillance (person tracking and object recognition) and handheld devices over to teach-in, visual programming and instruction based programming of compliant robots for heavy loads. With these components an easy method for programming or adapting workflows within an industrial packaging cell was realized and tested in a real factory environment.**

***Index Terms*—Robot programming; Human robot interaction**

I. Introduction

Programming a robot can be done in several different ways. While most of the industrial robots still are programmed by experts using a teach-panel, lots of efforts have been made to provide more intuitive and easier methods and simplify the programming procedure. An overview about some of these methods is given in [1] featuring kinesthetic teach-in [2], generalization of trajectories, programming by demonstration [3], learning tasks on a semantic level and by instructions [4]. Furthermore, visual programming tools, for example realized in the Microsoft Robotics Developer Studio or Choreographe for the NAO robot [5], allow the user to easily manipulate the behavior of the robot in a drag and drop environment.

We realized our concept for programming the robot and related hard- and software components within our industrial HRI packaging cell by combining some of the above described programming methods. In the following, the research project together with the developed components are briefly described followed by the programming concept and a short application scenario for a user study and first promising results.

II. The CustomPacker project

The main goals of the EU research project CustomPacker are the design, development and assembly of a flexible industrial HRI packaging cell, which is capable of handling different products varying in size, weight and form. Furthermore, it is desired to program the HRI workflow also by non-expert users in the field of robotics. In the following, the relevant hard- and software components developed within the project are shortly introduced.

1) Compliant heavy load robot and flexible gripper: In the HRI cell, an industrial robot is used together with a human worker performing a joint packaging process. Therefore, the robot must be able to work together with the human worker in the same work place safely, efficiently and interactively. Within the project, a safe compliant robot is developed by FerRobotics [6]. This robot is capable of handling loads of up to 50 kg.

The flexible gripper is developed and produced by Tekniker [6]. It is equipped with four fingers and a special selected rubber material which allows a careful handling of electronic consumer goods. The fingers can be exchanged and moved in two axis. The gripping fingers can be controlled either based upon a desired position or force.

2) Worker Tracking and Activity Recognition: The worker tracking is done by VTT [6]. For the worker tracking within the HRI cell, a Microsoft Kinect sensor and a capacitive sensor mat on the shop floor is used. The worker tracking is responsible for compensating false positives and tracking errors for subsequent processing components. Based upon the worker tracking, the activity of the human worker can be estimated by using the available workflow information.

3) System Interaction: The system interaction provides the interface for the worker using a GUI, text-to-speech (TTS), speech and gesture recognition. The GUI is realized in HTML and the TTS is a commercial solution. For speech recognition, Apple's Siri engine is interfaced using the Siri Proxy of [7]. A plugin for the proxy was created using keyword spotting and regular expressions to extract the information relevant for the workflow within our HRI cell. For gesture recognition, DTW from [8] was applied using motion energy thresholds on the skeleton data of the Kinect Sensor.

4) Object Recognition: The object recognition was developed by Profactor [6]. Their Software reconstructMe [9] is implemented to find the position and orientation of the object to be packed using a point cloud provided also by a Kinect Sensor. The reference model for the object recognition is provided with a CAD-Model of the product to be handled.

III. Concept

The concept for intuitive programming of the industrial HRI cell is depicted in Figure 1.

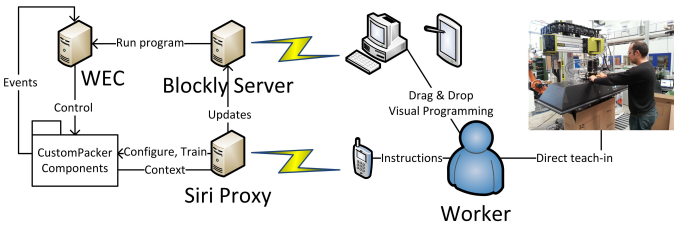

Fig. 1. Concept for intuitive programming of an the industrial HRI cell

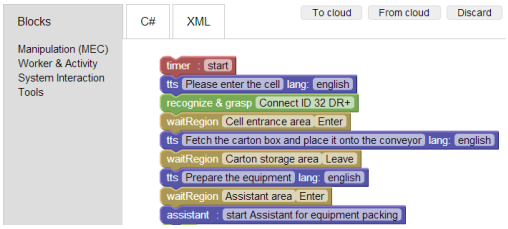

Fig. 2. Exemplary workflow representation using blockly

The CustomPacker components should be easily configurable by the user to program a new workflow. Therefore, teach-in methods with instruction based learning methods were combined into a visual programming scheme. The visual programming language blockly [10] was selected for a block based representation of the workflow. Each component of the HRI cell has one or more blocks.

With this block representation, new workflows can be easily programmed by the worker. Blockly is based on javascript and can thus be easily run on handheld devices, like tablet computers or mobile phones via drag and drop in a browser window. These devices are synchronized using websockets. Furthermore, connected clients are updated based on new instructions from the worker.

Together with the context information coming from the CustomPacker components (e.g. worker position, skeleton data and robot position), the worker has for example the following options to interact with the HRI cell: move the robot and manipulator via speech command or direct teach-in; save current positions of robot and manipulator with comments like *above tv-set*; create and learn new gestures and store them with user defined names; create and monitor new regions for the activity recognition with user defined names. These newly created regions, gestures, positions are added automatically to the blockly representation. The sequence of the blocks can be further manipulated and individually configured by the worker using drag and drop in the browser window.

After this procedure, the system has a representation of the packaging sequence (c.f. Figure 2). It is on the one hand machine readable to be executed by the entire HRI cell (including the robot), whilst on the other hand the representation is also human readable to be further adjusted by the worker if necessary. The generated machine code is transmitted to the Workflow Execution Control (WEC) and the connected clients are updated for track and trace functionality.

IV. First Results and Conclusion

In our user study, the participants had to program the last part of a packaging process. The robot had to wait until the human places a cardboard box on a conveyor belt. Then, it has to move above and into the box and back out again. After this, the robot had to wait for a gesture from the human before starting the next cycle.

The user study was done with 20 participants (aged between 22 and 40, 4 females). Seven participants had no experience in programming robots at all. However, all of them managed to complete the sample scenario successfully.

The meantime for programming (teaching positions, regions, gesture and adapting the workflow) was only 8.15 minutes. The worker was correctly detected in the trained regions in 95% of the trials. Gestures were recognized correctly in 40% of the trials. A reason for this might be that the gestures were only trained once by the users in a non-lab environment.

After the experiment, the participants had to fill out a questionnaire. This questionnaire included also a System Usability Scale, where we scored 89.4 out of 100. 90% stated that it was *easy* or *very easy* to program the workflow.

These preliminary results look very promising for the introduced concept. The participants of our study were able to intuitively program a new packaging process with the developed system interface, although they were inexperienced in terms of programming industrial robots. Therefore, we will continue working on our concept and improving the overall system.

Acknowledgment

The authors would like to thank all partners of the project for their support. This research project is supported by the European Commission under the 7th Framework Programme through the 'Factories of the Future' action under contract No: PPP-FoF-260065.

References

[1] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*. Berlin, Heidelberg: Springer, 2008. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-30301-5

[2] M. Hersch, F. Guenter, S. Calinon, and A. Billard, "Dynamical system modulation for robot learning via kinesthetic demonstrations," *Robotics, IEEE Transactions on*, vol. 24, no. 6, pp. 1463 –1467, dec. 2008.

[3] C. Eppner, J. Sturm, M. Bennewitz, C. Stachniss, and W. Burgard, "Imitation learning with generalized task descriptions," in *Proceedings of the IEEE International Conference on Robotics and Automation*, ser. Imitation Learning with Generalized Task Descriptions, 12-17 May 2009, pp. 3968–3974.

[4] C. Breazeal, A. G. Brooks, J. Gray, G. Hoffman, C. D. Kidd, H. Lee, J. Lieberman, A. Lockerd, and D. Chilongo, "Tutelage and collaboration for humanoid robots." *I. J. Humanoid Robotics*, vol. 1, no. 2, pp. 315–348, 2004.

[5] Aldebaran Robotics, "Choregraphe." [Online]. Available: http://www.aldebaran-robotics.com/en/Discover-NAO/Software/choregraphe.html

[6] CustomPacker, "The CustomPacker Project Partners." [Online]. Available: http://www.custompacker.eu/content/partners

[7] P. Plamoni, "Siri proxy for apple's siri," 2011. [Online]. Available: https://github.com/plamoni/SiriProxy

[8] Rhemyst and Rymix, "Kinect sdk dynamic time warping (dtw) gesture recognition," 2011. [Online]. Available: http://kinectdtw.codeplex.com/

[9] PROFACTOR GmbH, "Reconstructme," 2012. [Online]. Available: http://reconstructme.net/

[10] Google (maintained by Neil Fraser), "Blockly - a visual programming editor," 2012. [Online]. Available: http://code.google.com/p/blockly/