Programming Concept for an Industrial HRI Packaging Cell

J. Blume[1], A. Bannat[1], G. Rigoll[1],
M. Rooker[2], A. Angerer[2] and C. Lenz[3]

Abstract— **This paper presents an overview about a programming concept for an industrial HRI cell designed for packaging of electronic consumer goods. The focus of this work lies on the interplay of the involved software components. Furthermore, developed methods for programming the software components of the HRI cell are described within a sample use case. Finally, the usability of the programming concept and a short evaluation of relevant components are presented.**

I. MOTIVATION

The packaging and handling of heavy and highly individualizable electronic consumer goods (like subwoofers, TV sets or microwave ovens) is still very often done manually by human workers at most production sites, especially for small lot sizes where a complete automation is not affordable or economic. However, automating the packaging process will decrease the production cycle time (and thus costs) also for mixed variant production lines, thus allowing that several production lines can be merged to a reduced number of flexible packaging stations. This also allows an optimization with regard to the actual demands of the (various) goods (i.e. number of items produced per day). In order to achieve the realization of these challenging goals for a highly flexible packaging station, CustomPacker tries to combine the highly adaptable skills of a human worker together with the precision and ability of robots to carry and manipulate heavy goods.

Therefore, different components are developed to be combined depending on the actual product and use case. Ideally, these components can be programmed by non-experts to adapt the flexible cell towards new products and thereby reduce the time and costs compared to having experts do the job.

Obviously, there have been lots of efforts in research and industry to reduce the complexity of programming robots, like programming by demonstration [1], direct kinesthetic teach-in [2], instruction based learning [3] and visual programming toolkits like Microsoft Robotics Developer Studio [4] or NAO choreographe [5]. However, most of the existing tools provide or focus on one programming method only.

With our concept presented in this paper, we try to combine methods for programming the technical side of the components as well as integrating a programming scheme for the human worker related part within the workflow. Additionally, this workflow is represented visually to be more comprehensive and allow easy modification. This concept was developed within the project CustomPacker (abbreviation for Highly Customizable and Flexible Packaging Station for mid- to upper sized Electronic Consumer Goods using Industrial Robots) to allow the human worker teaching the packaging station how to handle and pack new products.

The rest of this paper is structured as follows: The developed programming concept with an overview about the involved components are explained within the next section in more detail. The results of a usability study of the programming concept and experimental results from the related components are followed by a conclusion and an outlook.

II. COMPONENTS AND PROGRAMMING CONCEPT

The selected software architecture for our programming concept follows a modular design approach as it can be seen in Figure 1. The communication between the components is realized with event based message handling using TCP. The main component on the top is orchestrating the connected components and modules and is called Workflow Execution Control (WEC). The WEC is also responsible to synchronize the workflow between the robot and the human and is supported by two further subsystems. One system is responsible for the machine related manipulation of the electronic goods and is called Manipulation Execution Control (MEC). Another subsystem is responsible for the worker surveillance and assistance and is called Worker Interaction (WI). The MEC subsystem deals with all aspects regarding recognizing the objects position and orientation and the collision free path planning from the parking position of the robot to a deposit position of an object to be packed. Furthermore, the MEC handles the generation of the robot-specific movement program and the transfer of this program to the packaging robot. The WI subsystem is responsible to localize and detect the worker within the cell, reject false person hypothesis and monitoring and providing assistance functionalities for the worker.

1) Robot Control (RC) and Gripper Control (GC): The robot control is responsible to provide the position or joint control for the robot and also deliver the status information

*This work was supported by the European Commission under the 7th Framework Programme through the 'Factories of the Future' action under contract No: PPP-FoF-260065. supported by any organization

[1]J. Blume, A. Bannat and G. Rigoll are with Faculty of Electrical Engineering, Technische Universität München, 80333 Munich, Germany {blume, bannat, rigoll}@tum.de

[2]M. Rooker and A. Angerer are with PROFACTOR GmbH, 4407 Steyr-Gleink, Austria {martijn.rooker, alfred.angerer}@profactor.at

[3]C. Lenz is with Faculty of Computer Science, Technische Universität München, 85748 Garching, Germany {lenz}@in.tum.de

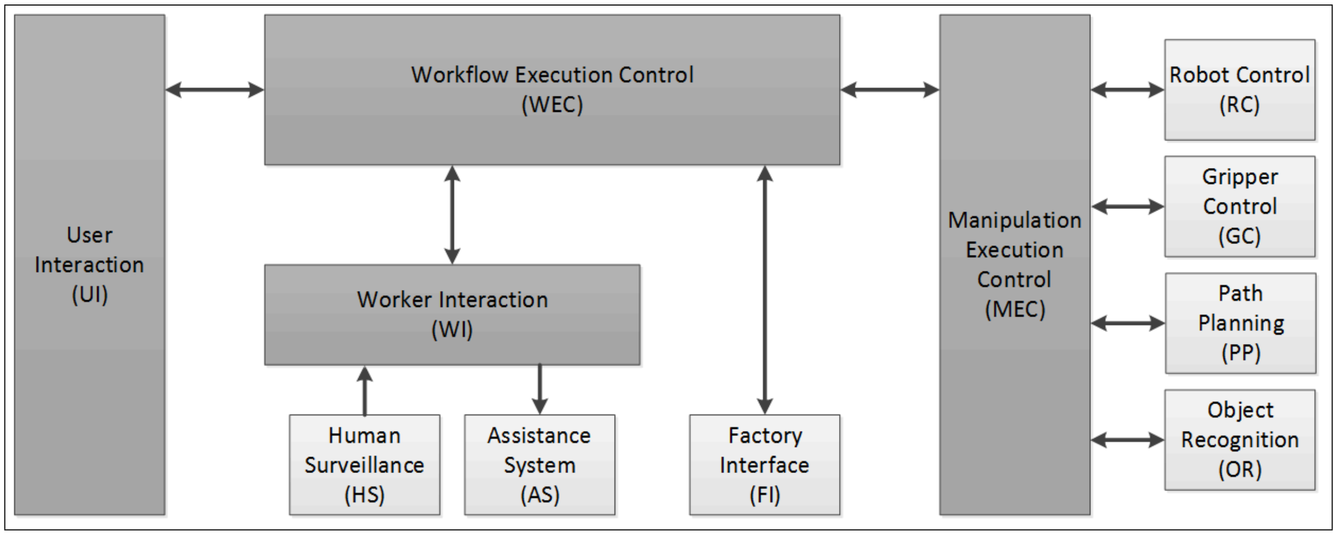

Fig. 1. This is a schematic representation of the CustomPacker software architecture. The main components are the User Interaction (UI), the Worker Interaction (WI), the Workflow Execution Control (WEC) and the Manipulation Execution Control (MEC) (dark gray blocks). The remaining involved modules are connected to those as depicted (light gray blocks).

for the WEC. In our case the industrial robot is used together with a human worker performing a joint packaging task. Therefore, the robot must be able to work together with the human worker in the same work place safely, efficiently and interactively. Although there are already existing compliant robots like the UR10 [6] from universal robots handling up to 10 kg or the KR 5 SI [7] from MRK, which is certified and can handle up to 5 kg, the electronic consumer goods to be handled can have weights of 30 kg or more. Therefore, a new compliant robot was developed within the project by FerRobotics [8]. This new robot is capable of handling loads of up to 50 kg.

The gripper control is required to operate and get feedback about the gripping device. In our case the gripper (produced by Tekniker [8]) can receive force or position commands. It has four fingers, which can be moved along two axis, and a special selected rubber material which allows a careful handling of electronic consumer goods .

2) Object Recognition (OR) and Path Planning (PP): The object recognition was developed by PROFACTOR [8]. The software for recognizing objects uses the commercial ReconstructMe [9] software (also developed by PROFACTOR). The object recognition is implemented to find the position and orientation of the object that has to be grasped, relative to the packaging robot. Therefore a point cloud is recorded by a Microsoft Kinect sensor (or similar devices like the Asus Xtion Pro or the Softkinetics device). The software compares a reference model of the object with the data of the recorded point cloud by searching for special features. This reference model is provided by using a CAD model of the product to be handled. The CAD model contains also the necessary data for grasping the object in a subsequent process of the packaging cycle, like grip points. After recognizing the object, the path planner calculates a collision free path from the parking position of the robot to the defined grasping position and afterwards back to a programmed deposit position. A simplified model of the robotic packaging cell and the robot (both generated in CAD) is needed for planning the collision free path through the workcell. It should be mentioned that the collision free path is checked only against static geometries in the packaging cell. Mobile obstacles are handled by using a compliant robot and additional pro-active safety systems, like worker tracking. In the very last step, a movement program for the packaging robot is generated by the path planning algorithm.

3) User Interaction (UI): The user interaction provides the interface for the worker to train the system during the programming mode or inform about the ongoing working cycle during the productive mode of the system. Therefore, the UI features a graphical user interface (GUI), speech recognition, text-to-speech (TTS) and a gesture recognition. The GUI is designed to be run in browsers on mobile devices using HTML 5 and websockets to communicate with the WEC. This allows the user to program the workflow directly within the cell. However, the GUI can also be run on desktop PCs, if desired. Furthermore, the GUI enables drag & drop in the programming mode. For this drag & drop functionality and the visual representation of the workflow Google's Blockly [10] was applied and adapted. This fits perfectly to map the CustomPacker components to blocks. In the productive mode the GUI shows the current step within the workflow and additional process information. For speech recognition, Apple's Siri engine is interfaced using the Siri Proxy of [11]. The speech recognition is only used in programming mode to give instructions, which trigger programming of the other components and create new blocks within the visual representation. Therefore, a plugin for the proxy was created using keyword spotting and regular expressions to extract the information relevant to configure the component related blocks. The TTS is a commercial solution, which is applied for confirmations during programming mode and instructions in productive mode. For the gesture recognition, dynamic time warping [12] was applied using motion energy thresholds on the skeleton data of the Kinect Sensor. In the programming mode the user can start programming of new gestures by using instructions like 'Learn a new gesture called cycle done'. Afterwards, the operator can demonstrate the new gesture. In the productive mode, these gestures can be recognized by the UI and reported as event to the WEC.

4) Human Surveillance (HS): For the human surveillance within the HRI cell, a Microsoft Kinect sensor is used. Furthermore, a capacitive sensor mat is installed on the shop floor of the HRI cell. With these devices the position of the human worker is identified.

The Kinect sensor produces a wireframe model of the human body. It directly calculates the position of the tracked body in world coordinates. By calibrating the cell coordinate system with the Kinect coordinate system, the position of the worker is available from the Kinect data.
The capacitive sensor mat gives sensor data when objects are moving or standing on the shop floor. Since the sensors on the mat have a specific pattern, the position of the data producing objects can be directly calculated. This sensor mat delivers robust information about objects and people placed on its surface. Together with the Kinect data, the HS is capable of delivering robust estimates about the workers position within the cell.

Based upon the worker position data, the activity of the human worker can be estimated by using the available workflow information about the packaging process. With a rule-based approached, relevant positions are combined with the workflow information. The rules are used as event-triggers for estimating the current worker action. For example, if required packaging material is stored at a specific location, the worker has to fetch this material. By entering and afterwards leaving this area, the HS-logic produces the event *fetch item* from this observation. The workflow execution control can use these events to synchronize the workflow between human and robot and trigger next steps.

5) Assistance System (AS): The assistance system is designed to help the worker fullfill his dedicated tasks for new product variants to be manipulated. The system is equipped with two modalities for supporting the worker. It features a visual and an accoustic information channel. A projection unit mounted above the HRI cell uses projected light patterns to guide the worker to important areas in the cell or highlight regions relevant to perform the next step.

In Figure 2, the projection of this unit is highlighting the region marked in green light. In this sample, the worker has to place the cardboard box on the conveyor belt at the marked spot. Once the worker reaches the region, he also receives further instructions via voice gernerated from the UI. Trained workers, who are already used to the required processes can also reduce the level of information presented by the assistant or even can choose to turn the AS off completely.

6) Factory Interface (FI): The final component is the factory interface. The FI is responsible for synchronizing the packaging HRI cell with the rest of the factory environment. This typically features additional machinery, like conveyor belts, sensors and trigger signals. The required information about the next type of the electronic consumer good to be packed is also transmitted via this component to the WEC. After the task within the packaging HRI cell is completed successfully the event *cycle completed* is also signaled via this component to the superior control of the factory environment.

Furthermore, it can be distinguished between a productive mode and a programming mode of the system. The productive mode is the operational mode in which the electronic consumer goods are to be packed by robot and the human worker, like it is depicted in Figure 2.

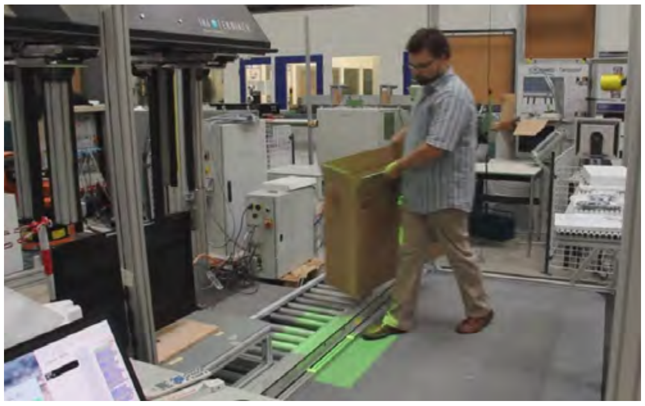

Fig. 2. An image of the CustomPacker HRI packaging cell prototype showing a worker putting the carton on the conveyor belt during the productive mode. The assistance system is providing instructions and light patterns (green light on shop floor) to guide the worker.

The programming mode is the state in which new products and sequences can be trained by the operator.

The main idea of the programming concept is to combine different programming methods to program and synchronize the workflow between the human and the robot more easily. Obviously, this requires the different components involved in the HRI cell to be programmable by the worker. Therefore, each component needs to provide the puzzle pieces as depicted in Figure 3.

The WEC handles the workflow execution during the productive mode. Therefore, this workflow has to be generated during the programming mode. The subsystems provide their functionality to the WEC in so called blocks. These blocks are an abstract representation of the skills of the components and include configurable options. As it can be seen Figure 3 these blocks contain executable code for the WEC and also a visual representation for the worker via the UI. Additionally, they have to provide a programming interface using either demonstration, instructions or teach-in to configure existing or add new options. The sequence of this blocks can furthermore be edited using drag and drop.

For a better understanding of how this programming concept works, consider the following use case: You have to teach a new good to be packed. This good has to be recognized and grasped autonomously, while this is done you have to place an empty carton onto the conveyor belt, where

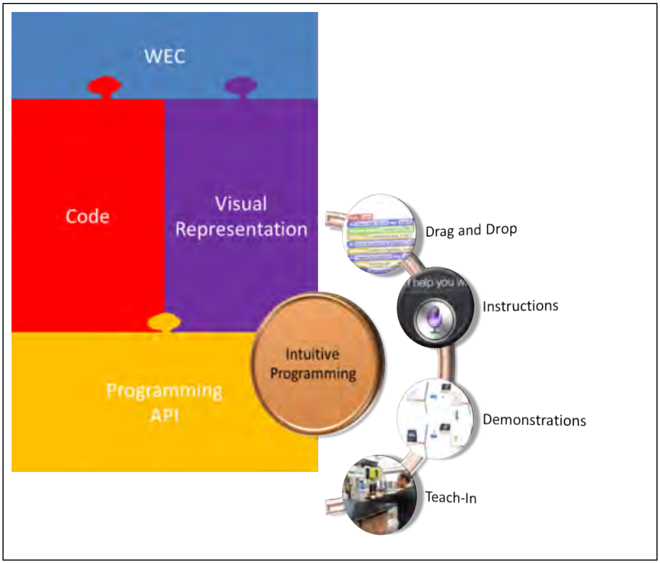

Fig. 3. Our idea of intuitive programming is to combine different programming approaches and have one visual representation of the created programm. Furthermore, not only the robot should be programmed in this way, but also the other relevant components of the cell (e.g. human surveillance and assistance system).

this good will be stored. Some additional packaging material and accesoires have to be prepared as well and stored into this box together with the good.

Thus, the first step is to prepare the MEC for the new manipulation of the good.

Therefore, an object has to be added to the system (if it hasnt been added before) by generating a reference model. This is done by scanning it with a preparing software (also based on ReconstructMe [9]). This reference model has to be aligned with the CAD model of the object which contains also the information for grasping the part with the flexible gripper. This alignment is done by using the open source software tool Meshlab [13]. The calculated transformation matrix of the alignment between CAD model and scanned reference model has to be stored, thus the path planner knows the correlation between grasping points and object position. Also the deposit position of the new object has to be programmed once for generating the proper movement. If an object has been added once to the system there is no need to teach it in again. The MEC only needs to know, which part should be grasped in the next packaging cycle(s) and thus the system can be reconfigured in a very short time.

After the automated grasping of the good has been trained, the manipulation and grasping procedure for this good can now be chosen from the list of options in the corresponding MEC block. Further positions (e.g. like the deposit position or the position over the carton) required for the robot can be added by either moving the robot directly via teach-in or instructing the robot to move in a direction with a certain distance by speech or using a move block and enter the x,y,z position directly in a visual programming editor. All of these programming methods would result in the same block representing the desired robot position.

For training the human surveillance module the following procedure is integrated into the system. The relevant workflow information from this module for the WEC is the position of the worker to monitor the completion of his task. This position information is gained by evaluating the sensordata of the installed sensitive floor where the worker stands on. If a specific region is activated for a specified amount of frames, an event is send to the WI informing the system about the probable worker position. This event is then forwarded to the WEC. Teaching new regions of interest for monitoring the worker position is also possible. Therefore, the HS-module records raw sensor data over time in the programming mode. The operator can just walk to a location, which is of interest for the workflow process. When he has arrived at a region of interest relevant for the cell workflow, he can save his current position in the HS with a meaningfull name, e.g. using instructions. The current position is estimated by building a heat-map from the sensordata over a specific time frame. The region where the operator currently is standing on has a much higher data value, than the other regions within the cell. By a thresholding operation after the data capturing, this region is then isolated and stored into the HS. With saving this location, the operator has to provide a human-readable name for the current event region. This name is then transmitted to the WEC. Via his mobile interface, the operator can also add a describing comment to this region, which is then automatically available in the UI. In the WEC, the corresponding region-block (in blockly) the human-readable name is appended to the list of available regions on the cell floor. This enables the workflow designer to choose from these events to enable a smoothly running worker integration. The system can then map the worker position and name to the visual representation. The operator can select wether entering or leaving of this area should be monitored.

The assistant system involves compontents for displaying visual and auditive instructions. Adding new assistance instructions can be done using a drag & drop design tool. For each instruction, the designer can place geometric objects on the screen varying in color, size and shape. During programming mode, the operator gets a live view of the objects, he wants to place. This enables the operator to highlight desired regions by simply moving the objects to the corresponding location. He can use a standard PC for this operation, or the already mentioned mobile device. Using mobile devices makes the programming mode more effective, since the operator can move within the cell while planning the regions for the objects to be highlighted. Thereby, the operator gets direct feedback wether the visual instructions are located on the right places. Furthermore, each visual instruction can also be accompanied by auditive instructions, which are played back to the worker using TTS in the productive mode. Thereby, the worker does not need to read the instructions on an additional display, since they are given just in time for the next step in the workflow.

III. Evaluation and Results

This sections provides evaluations and results about the manipulation execution control, the human surveillance and usability study results of the programming concept.

A. *Evaluation of the manipulation execution control*

The software components of the MEC recognize the objects with an accuracy of about +/- 2mm as it can be seen also in Figure 4. The Microsoft Kinect sensor has a frame rate of 30 FPS and so it is possible to record a point cloud in less than 100ms. The recognition of a single object takes about 1 second. Also the path planning algorithm calculates a collision free path through this packaging cell in less than 1,5 seconds. Experiments in grasping showed, that all objects could be grasped and be packed without problems with the chosen hard- and software components.

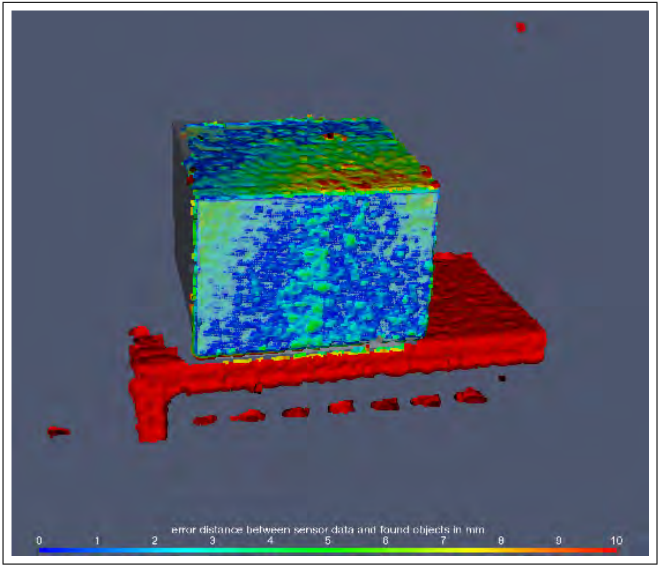

Fig. 4. A sample result of the object recognition showing the color coded error distance between sensor data and the detected object. As the subwoofer in this case is mostly colored in dark and light blue this is a good match.

B. *Evaluation of human surveillance*

During a further short trial run, the human surveillance module was in focus of evaluation. This evaluation was done while the subjects performed a complete packaging sequence. The participants were one expert user, who already had experiences with the CustomPacker system. Furthermore, two novice users were also asked to participate in the evaluation.

The following data was recorded during the evaluation: Each participant had to complete five packaging procedures. Therefore, a total of 15 packaging procedures were recorded. The relevant information gathered from these procedures is the number of events, correcty recognized by the HS-module. To complete the packaging task, 11 events had to be correctly recognized per person and cycle. This results in 165 events for the complete evaluation. Of all those events, only 8 events were missed in total. This results in an overall recognition rate of:

$$p_{recognition} = 1 - \frac{8}{165} = 95,15\% \tag{1}$$

For the cycle time, the recorded data showed a total of $2074,93s$ overall time. Thus, the average cycle time is $138,33s$.

Due to the geometric structure of the applied sensor mat (cf. SensFloor [14]), the spatial resolution of one sensor field is limited to one sensor triangle (approx. $0,25m$).

C. *Usability Study of Programming Concept*

We conducted first experiments to study the functionality of the developed programming concept and the usability of this concept. Therefore, we created a small sample application within the packaging domain. The subjects had to program the scenario on their own and afterwards got the chance to fill out a questionnaire including a system usability scale part.

The sample scenario for the subjects was to program the last steps of a tv-packaging application. The subjects had to program the rest of the sequence, after the object recognition had located and grasped the TV set already. Thus, they had to progam a new location for the system to be monitored to make sure the worker placed the cardboard box on the conveyor before the robot tries to put the TV set into the box. Additionally, the subjects had to program three robot positions, one as waiting position with the TV set, one above the box and one in the box. Finally, they had the chance to train a sequence completed gesture.

A total of 20 subjects (including participants from marketing, translators and business studies; aged between 22 and 40 with 4 female participants) first conducted the sample scenario. Afterwards, they evaluated the system with a questionnaire including the well known system usability scale (SUS) and attrakdiff [15] part. 85% had much or very much experience in handling technical systems (smartphones, PCs, etc.). However, 30% had no and 25% only very little experience related to industrial robots. 90% said it was easy or very easy for them to program the sequence for the scenario. This correlates with the question how good they managed to cope with the overall programming concept. Here, 22% said they get along well and 72% got along very well with this programming concept. Thus, every participant managed to complete the scenario successfully. The mean time for programming was 8,15 minutes with a deviation of 3,66 minutes. Considering the single programming methods, the direct teach-in could be made more sensitive with respect to the vertical movements.

The logfiles of five participants were compared with hand written spoken text annotations to calculate an average Word Error Rate (WER) according to equation 2:

$$WER = \frac{S + D + I}{N} \tag{2}$$

Within this equation S is the number of substitutions, D is the number of deletions and I the number of insertions to reproduce the originally spoken N words. Based on this calculation the WER using the siri-proxy[11] and one phone for all participants with german language resulted in 15,9%.

The SUS has a maximum score of 100 points. However, according to [16], which averaged over 234 SUS studies, the mean value is not 50 but around 70,1. In our study the mean score was $\mu = 89,375$ with a standard deviation of $\sigma = 6,38$ and thus can be considered an above average usability also taking into account the higher mean value.

In Figure 5 the graphic result of the attrakdiff study is depicted. The transparent orange rectangle shows the confidence of the 20 participants in the two corresponding dimensions hedonic and pragmatic quality of the concept and the center P shows the mean value averaged over all participants. Similar to the SUS the attrakdiff results also show a desired solution by the users in both dimensions.

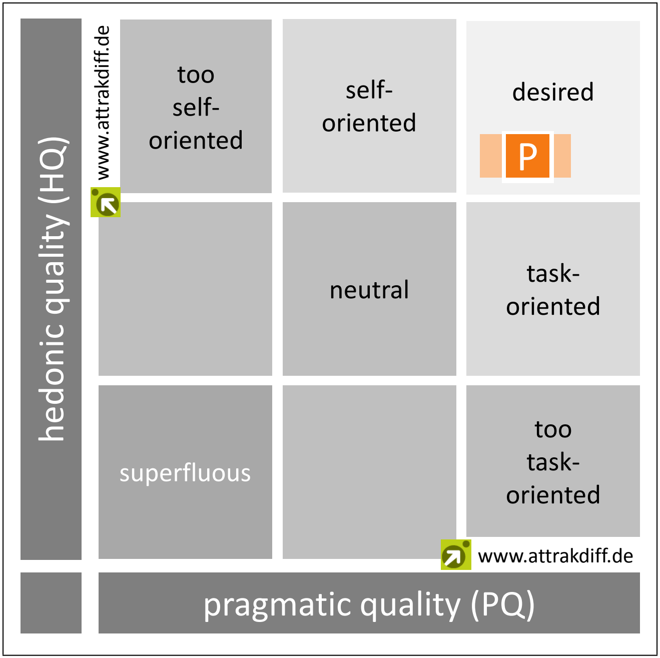

Fig. 5. Results from the attrakdiff for the prototyped programming concept based on a user study with 20 participants.

IV. CONCLUSIONS AND OUTLOOK

In this paper, we presented our approach of combining several different programming methods for the cell components with a common visual representation. The user can choose between voice instructions, direct teach-in, drag & drop and demonstration of new regions and gestures to be monitored. The communication between the WEC and the software components is designed to be flexible and modular, thus allowing easy adaption to new use cases, even without robots. Additionally, the conducted evaluation of the core components within the prototyped cell shows promising results concerning the usability of the programming concept for new workflows within the industrial HRI cell. However, the current implementation is mainly based on a single line of execution with limited threading capabilities. This could be enhanced in future work towards a graph based representation for supporting multiple lines of execution like a petrinet. Furthermore, we are looking at extracting packaging step information by demonstration of a human directly from point clouds using key frames and a mobile observing robot plattform (cf. Figure 6).

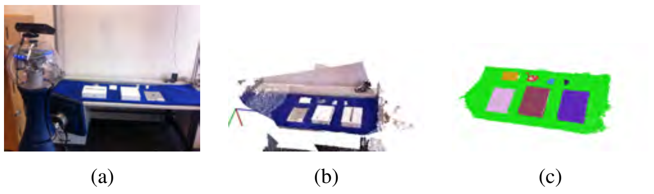

Fig. 6. **Mobile robot for extracting packaging steps from point clouds** - (a) lab setup (b) registered point clouds (c) segmented plane and objects

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] 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.

[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. 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.

[4] Microsoft, "Microsoft robotics developer studio 4," 2012. [Online]. Available: http://www.microsoft.com/robotics/

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

[6] Universal Robots, "Collaborative Robot Solutions." [Online]. Available: http://www.universal-robots.com/GB/Products.aspx

[7] MRK-Systeme GmbH, "KR 5 SI (SafeInteraction)." [Online]. Available: http://www.mrk-systeme.de/e_produkte_interaction.html

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

[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/

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

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

[13] P. Cignoni and G. Ranzuglia, "Meshlab, the mesh processing system for 3d scanning and printing." [Online]. Available: http://sourceforge.net/projects/meshlab/

[14] Future Shape, "SensFloor – large-area sensor system." [Online]. Available: http://www.future-shape.de/en/technologies/23

[15] User Interface Design GmbH, "Attrakdiff." [Online]. Available: www.attrakdiff.de

[16] A. Bangor, P. T. Kortum, and J. T. Miller, "An empirical evaluation of the system usability scale," *International Journal of Human-Computer Interaction*, vol. 24, pp. 574 – 594, 2008.