On Cognitive Robot Woodworking in SMErobotics

Mathias Haage, Lund University, Lund, Sweden, mathias.haage@cs.lth.se
Stefan Profanter, fortiss GmbH, Munich, Germany, profanter@fortiss.org
Ingmar Kessler, fortiss GmbH, Munich, Germany, ikessler@fortiss.org
Alexander Perzylo, fortiss GmbH, Munich, Germany, perzylo@fortiss.org
Nikhil Somani, fortiss GmbH, Munich, Germany, somani@fortiss.org
Olof Sörnmo, Lund University, Lund, Sweden, olof.sornmo@control.lth.se
Martin Karlsson, Lund University, Lund, Sweden, martin.karlsson@control.lth.se
Sven Gestegård Robertz, Lund University, Lund, Sweden, sven.robertz@cs.lth.se
Klas Nilsson, Lund University, Lund, Sweden, klas@cs.lth.se
Ludovic Resch, Mivelaz Techniques Bois SA, Mivelaz, Switzerland, resch@onlywood.ch
Michael Marti, Güdel AG, Switzerland, michael.marti@ch.gudel.com

Abstract

This paper details and discusses work performed at the woodworking SME Mivelaz Techniques Bois SA within the SMErobotics FP7 project. The aim is to improve non-expert handling of the cell by introduction of cognitive abilities in the robot system. Three areas are considered; intuitive programming, process adaptation and system integration. Proposed cognitive components are described together with experiments performed.

1 Introduction

Today's robot systems have limits in their ability to deal with frequent changes in the manufacturing process. While robots are able to carry out repetitive tasks to a high standard, they do not meet the demands of small- and medium-sized enterprises (SMEs); small lot sizes, high number of product variants, frequent re-configuration and re-programming, and little or no in-house robot expertise. Also, for the setup and operation of robot systems in an SME environment, which is typically less structured and involves more uncertainties than large-scale or mass-production industries, the currently available solutions result in overly complex system integration.

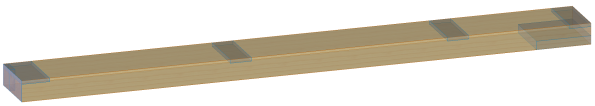
The SMErobotics[1] project vision is to significantly lower the threshold for introducing and meeting productivity demands with robot automation on the SME shop floor. Cognitive abilities need to be included per default in the robot system to aid the operator in setup and during process with little or no robot expertise needed. To this end, robot system components and techniques are defined to target several aspects of SME robot automation, such as human-robot interaction, system integration and process adaptation. Together the components present cognitive "add-ons" to an industrial robot system.

In this paper a SME robot cell set in the woodworking domain is targeted by a selection of SMErobotics techniques. The production cell is located at Mivelaz Techniques Bois SA, an SME located in Switzerland that manufactures pre-fabrication houses, and performs wooden wall panel assembly as well as free-form milling. A picture of the cell is shown in Figure 1. Several issues for the specific SME are targeted, including intuitive adaptation of machining operations for execution, automatic assistance in selection of process parameters, and robust system-level integration of equipment and software components.

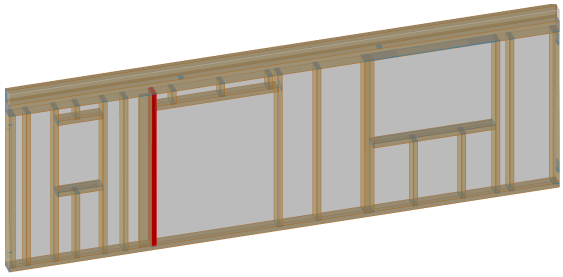
Figure 1 Woodworking gantry-robot cell at SMErobotics partner Mivelaz Techniques Bois SA, performing wall panel assembly for pre-fabrication houses and free-form milling.

The paper is outlined as follows; A description of the production cell is followed by a description of cognitive components added or suggested for the cell. This is followed by descriptions of experiments performed and finally conclusions.

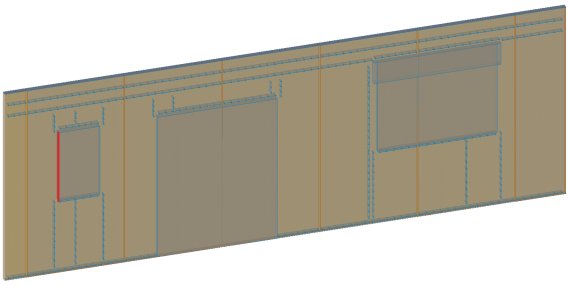
[1] http://www.smerobotics.org



(a) Wall struts are manufactured by CNC.

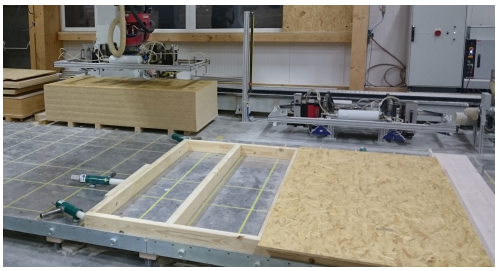


(b) Struts are manually assembled into wall structures.

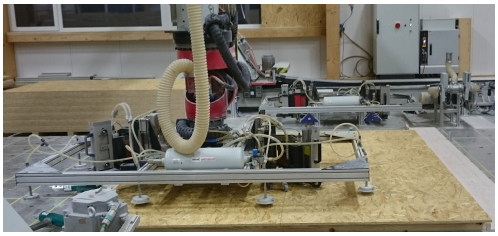


(c) Panels are assembled and fitted to wall structures using the gantry robot cell.

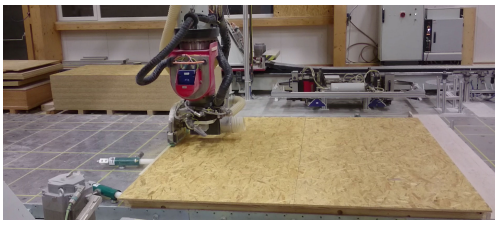
Figure 2 The basic shop floor production flow at the SME.



(a) Pick and place of panels.



(b) Nailing of panels to wall structure.



(c) Sawing of panels to fit wall structure.

Figure 3 Gantry wood panel operations utilized for the panel assembly in Figure 2(c).

2 Robot Woodworking in an SME

This section describes the woodworking robot cell located at Mivelaz Techniques Bois SA. The SME produces one-off pre-fabrication houses based on in-house CAD/CAM[2] expertise. The on-site shop floor manufactures the required house parts (walls and other details) that are then shipped for assembly at the customer site.

The gantry robot[3] cell is part of a production flow that starts with CAD/CAM resulting in machining description formats and blueprints for the shop floor, followed by machining and assembly operations on the shop floor. The major production steps are depicted in Figure 2. CNC[4] machines manufacture struts that are manually assembled into wall structures. These are then moved onto the gantry robot work table where panel assembly is taking place. Additional operations, such as quality inspection, adding insulation material, or turning the wall structure for panel assembly on the opposite side, is performed manually.

An assembly consists of picking panels and placing them a side of the wall structure. This is usually followed by nailing of panels to the structure and concluded by sawing and milling the panels to fit the structure. Since wood is a live material, fitting is usually performed by material removal according to nominal CAD dimensions (with an added fault-tolerance offset) followed by manual adjustment of the sawing and milling operation and re-execution for exact fit. The gantry robot has two work zones to allow several walls to be assembled at the same time for higher throughput.

2.1 Issues

Aiming for non-expert handling, interviews revealed dependencies on in-house robot and robot-related expertise:

- General robot expertise is needed from time to time (estimated once per two weeks) to solve issues that are beyond the knowledge of an ordinary robot operator. Typically this amounts to resolving issues with CAMed machining operations on the shop floor.
- Process adaptation due to tool wear is currently done manually, requiring operator process experience.
- Introduction of new sensors and other hardware equipment requires PLC expertise and experience in the current cell setup.

[2] Computer Aided Design/Manufacturing

[3] The gantry has 6 degrees of freedom. Panel assembly utilizes five while free-form milling uses all six DoFs.

[4] Computerized Numerical Control

3 System Architecture

As a step towards addressing the issues listed in Section 2.1, cognitive software components and techniques from three SMErobotics areas were adopted for the cell; intuitive programming, sensor adaptation and system-level integration. The added software was physically hosted on a PC during experiments, but in preparation for permanent installation of part of the software an industrial PC was also integrated in the cell setup.

Figure 4 shows an architectural overview for the introduction of cognitive components in the robot system; Intuitive programming is represented by several of the components. Process adaptation is listed as a component, but at the moment of writing actual packaging of the adaptation software as a component has not been performed. Finally, system-level integration is concerned with interoperability of equipment and software components and is represented in (some) of the arrows in the figure. Robot operations are stored in a woodworking shop floor format called BTL and are imported to a database. The operations then need to be selected, possibly changed and annotated with further process data. This is done using the interaction manager with associated GUI. The task execution engine allows visualization of operations through a generic simulation environment, as well as generation and deployment of robot programs, and monitoring of a running task. Two deployment targets are available in the system, a robot-specific simulation environment (ABB RobotStudio[5]) capable of executing the generated robot program and the physical controller. During execution sensor data is collected for process adaptation. Task execution is a matter of interacting with both the cell PLC and the robot controller. This is done by communicating with the PLC through the robot program.

The implemented architecture is more complicated than the architecture described so far. Since components were adapted from different project partners from existing solutions it resulted in incompatibilities that needed to be resolved for components to interoperate. Incompatibilities such as differing data structures and execution environments motivated an investigation into and application of (some) SMErobotics system-level integration techniques.

3.1 BTL parser

BTL[6] is an open data format mainly maintained by Lignocam[7]. It was created by two CAD companies, SEMA[8] and CADWORK[9]. It is a data transfer format for wood constructions containing both part geometries and machining operations which is understood by woodworking CNC machines. As an example, the geometry in Figure 2 is part of a house described in one BTL file. The dark areas in Figure 2(a) represent material removal operations, the red box

[5]http://www.robotstudio.com
[6]http://www.design2machine.com
[7]http://lignocam.com
[8]http://www.sema-soft.de
[9]http://www.cadwork.com

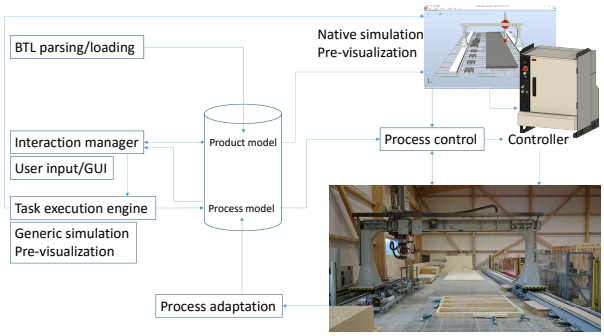


Figure 4 Proposed architecture for the introduction of cognitive components in the woodworking robot system.

in 2(b) indicate a selected strut, and the red line in 2(c) represents a sawing operation. Since solutions for BTL interpretation for industrial robots were scarce during the project, it was necessary to develop a BTL industrial robot component specifically for the SME need[10]. The component is used to make BTL data accessible to the other components in the system.

3.2 Intuitive Adaptation of BTL processes

As part of SMErobotics several components were developed to make robot automation accessible to non-expert robot operators. Operators are experts in their respective domains and should be able to communicate their intentions to the system, without being experts in robotics. In the interaction manager, this is done by explicitly modeling the required knowledge in a way, such that the robot system is able to understand the language of the domain expert. A semantic description language has been developed to specify processes, interaction objects, and the workcells in which the processes shall be executed [3, 4]. For interaction, an intuitive touch interface serves as the primary modality for inspecting and parameterizing manufacturing processes. This interface serves as a frontend to the semantic process description. For the woodworking application the interaction manager was adopted to read the BTL description through a semantic transformation to the process description language. It then manages the dialogue with the robot operator, assisting in decision making and fault diagnosis of BTL execution. It assists in task-level programming and configuration of the BTL process and works in close co-operation with the task execution engine. Together the two components aim at providing an intuitive approach to programming and execution in the SME robot system.

3.3 Task Execution and Monitoring

The output of the BTL parser (Section 3.1) and the intuitive instruction GUI (Section 3.2, [6]) is an object-centric task description which can not be executed directly by the robot. E.g., a BTL command defines a sawing line on an

[10]BTL for industrial robots will kindly be supported and available through Cognibotics, http://www.cognibotics.com/

object, but not the corresponding coordinates in the base frame of the robot. The description has to be mapped accordingly and a robot program has to be generated using the task execution engine to execute the desired steps.

The task execution engine checks all preconditions for each task and maps an abstract task description to one or more specific commands, called skills. For this mapping different sensor values and additional data is required, e.g., the relative position of the workpiece to the robot has to be measured and the task parameters adapted to match the offset pose. After all the mappings are calculated and the process is ready to be executed, the generated program can be deployed to, and simulated within the ABB Robot Studio simulation environment. After successful simulation, the program can be sent directly to the robot for execution.

The task execution engine also performs monitoring. In order to make robots more viable in SMEs, not only their instruction and normal operation need to be more intuitive but also their operation when errors occur, because it cannot be expected that there is always an expert on site to resolve them. Errors are detected by equipping the robot system with additional sensors and detecting when an execution no longer produces the expected sensor values. By performing the error detection and handling on the task level, the robot system knows the expected sensor values and how to handle potential errors. This error handling can be automatic or it can include the shop floor worker. In the latter case, a semantically meaningful and human-understandable error message is shown to the worker in the user interface. The worker is given a description of the error and the initial, automatic actions taken by the robot (such as pausing of all movements), a recommendation on how to resolve the error and several options for commanding the robot system on how to proceed afterwards. Tasks can not only include a semantic description of the normal execution of the task, but also descriptions of how to recover from certain types of errors, e.g., by simply repeating a task.

3.4 Process Adaptation

Automatic or semi-automatic process adaptation may improve quality and lower the cognitive burden on the operator by providing assistance in optimal selection of process parameters. The example found in the SME was adaption of path speed during sawing which in this case was done manually based on listening on the sound of the cuts and adjusting speed according to the experience of the operator. Tool wear such as bluntness of the sawing blade are an example of important process parameters that are usually not part of task programming unless automatic means for estimation are available. For comparison, classifiers for several sensors were developed, including audio, accelerometer and robot joint torques. From a cost perspective torque data is already available in the system. Audio provides cheap but potentially noisy sensing. Accelerometer data is accurate but is also the most expensive input modality.

3.5 Interoperability

For simple setup, a robot system must be robust and flexible in integration of new equipment and software. The SMErobotics project therefore offers a number of tools offering support for integration at the system level. At the core, the basic communication channel between devices and software components is proposed to be enhanced with in-band syntactic and semantic descriptions for the channel to become self-describing.

Self-describing data channels may give a human operator early and precise information about incompatibilities during set-up. Thus, they provide robustness against unintentional changes to interfaces and assist the human in identifying and fixing mistakes. It also allows to detect and handle mismatches in communication between equipment and software from different vendors, which is often important in an SME setting. It enables (semi-)automatic generation of communication bridges that connect incompatible devices. This is exemplified in [2], which generates bridges for the case of semantically compatible but structurally different message formats, such as between ROS and LabComm. The message translation can be either manually specified, or automatically deduced from semantic descriptions, or a combination of the two. In the SME semi-automatic bridge building is used for transforming messages between the BTL parser and the process description language used by the interaction manager. Other potential uses include assistance in PLC and sensor integration.

A prototype protocol called LabComm[11] is being developed [1]. LabComm is an example of an in-band, self-describing protocol[12] and is used to illustrate features of low-level data channels, and how they can be used to bridge and mitigate differences. LabComm ensures stable identities/signatures of different message types on communication channels. This can, for instance, be utilized for semantic grounding of messages towards external resources. In turn it allows the receiver of messages to verify that the set of messages the other party may send is precisely the ones expected. In other words, any change to a message format, or addition of a new message type, is detected before operation commences. Several tests were performed to evaluate LabComm in the SME context. Since components existed in two different execution contexts, ROS[13] and OSGi[14], LabComm was used to create communication bridges between these. Also, since the communication between the task execution engine and the robot controller was untyped, LabComm was used to create a typed and semantically grounded controller communication channel.

Self-describing data has seen much use in web-service systems, with protocols like Apache Avro[15] and Google Proto-

[11] http://www.control.lth.se/Research/tools.html

[12] Each time a communication channel is established between two devices, a *signature*, describing all messages that can be transmitted on the channel, is transmitted before the transmission of data begins.

[13] http://www.ros.org/

[14] https://www.osgi.org/

[15] http://avro.apache.org

col Buffers[16]. Systems for automation and motion control, on the other hand, tend to rely on standards (e.g., CANopen device profiles). In line with the open-world assumption, we use self-describing data to make it possible to bridge incompatibilities in order to enable agile setup and reconfiguration. An important difference between the LabComm self-describing data channels and systems like Avro is that LabComm focuses on stability, requiring exactly matching signatures, whereas Avro supports *schema evolution* with mechanisms for automatic handling of added, removed, or modified fields.In business and web-service applications, supporting such evolution is valueable, but in robot control, stability and safety is paramount.

4 Experiments

A number of experiments were performed on the industrial workcell from Figure 1 which are described below.

4.1 Assembly of wooden house walls

The main experiment consisted of evaluating the full software stack of our Cognitive System Architecture and hardware functionalities. For this task a simple wooden house wall was constructed (see Figure 5) using CADWORK and then exported as a BTL file. The operations within the BTL include placing two panels, nailing and fixating them on a woooden frame, and cutting off overlapping parts of bigger panels using a circular saw attached to the robot.

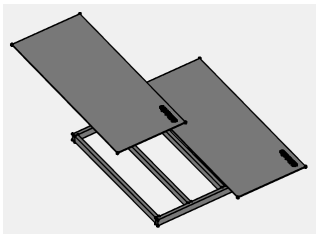


Figure 5 A simple wooden house wall with two panels which need to be placed, nailed, and sawed.

As depicted in Figure 4 different components are integrated into the system and play different roles. This experiment was carried out to prove that all the components work as expected. The BTL interpreter (Section 3.1) is used to interpret the file generated by CADWORK, and to store it within the semantic storage (Section 3.2). After the file is successfully parsed, it can be visualized in the intuitive GUI (Figure 6).
This intuitive GUI supports different input modalities for different parameter types [5]. If necessary, task parameters can be modified (e.g., place-positions of the panels or length and speed for a sawing line). When the process is ready to be executed, it is passed to the interaction manager and task execution engine (Section 3.3). After all task parameters are mapped to the corresponding robot skills, the process can be simulated in ABB Robot Studio. For communication between different subsystems we use the interoperability approach as described in Section 3.5. As soon as the simulation completed, the shopfloor-worker can decide to execute the process on the real robot (Figure 3). During execution, the interaction manager is monitoring different system parameters to detect anomalies and to notify the worker if necessary (Section 3.3).

[16]https://developers.google.com/protocol-buffers

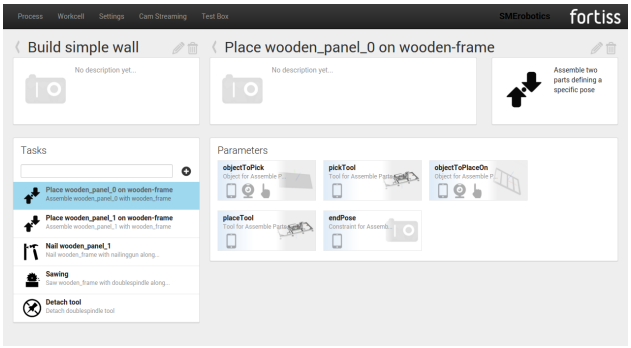


Figure 6 Woodworking process parsed from BTL and visualized in the intuitive instruction GUI. Parameters for each task can be modified before execution.

4.2 Automatic assessment of tool wear

To gather data for the audio-based classifier, cuts were made under different conditions while the sound was recorded using a microphone. In three cases, the cuts were made with a blunt blade at max feed rate. These form positive examples, and it is desired that the classifier detects such events, so that a decreased feed rate or change of blade can be recommended. Subsequently, a set of negative examples was formed, where the type of blades as well as the feed rate were varied. Blunt blades occurred in the negative set as well, but at lower feed rates than for the positive examples. The data was then divided into training data and test data. The corresponding normalized frequency spectra were estimated, Figure 7 shows one positive and one negative example. Inspection of the training data revealed that the positive samples contained more energy in the high-frequency range. Further, these samples covered a wider range of the spectrum, compared to the positive ones. For this reason, the mean μ_f and the standard deviation σ_f of the frequency were chosen as features for the machine learning algorithm using Support Vector Machines (SVM). These are given by

$$\mu_f = \int_0^\infty f R(f)\,\mathrm{d}f \tag{1}$$

$$\sigma_f^2 = \int_0^\infty (f-\mu_f)^2 R(f)\,\mathrm{d}f \tag{2}$$

where $R(f)$ is the normalized power spectral density for the frequency f. The resulting classifier is visualized in Figure 8.

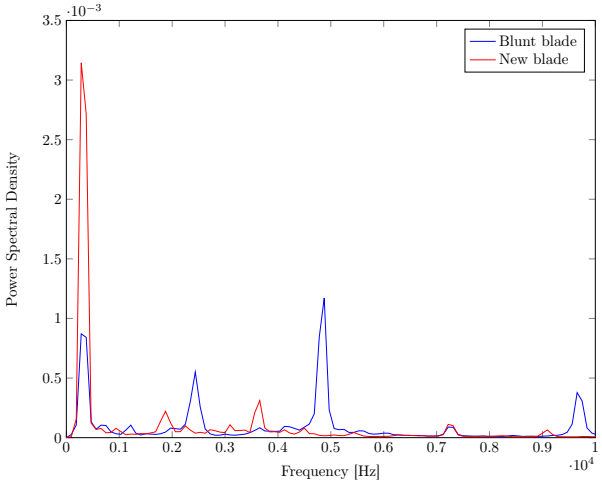


Figure 7 The normalized frequency spectra of the sound from cutting with a blunt and a new blade, respectively, at full path speed.

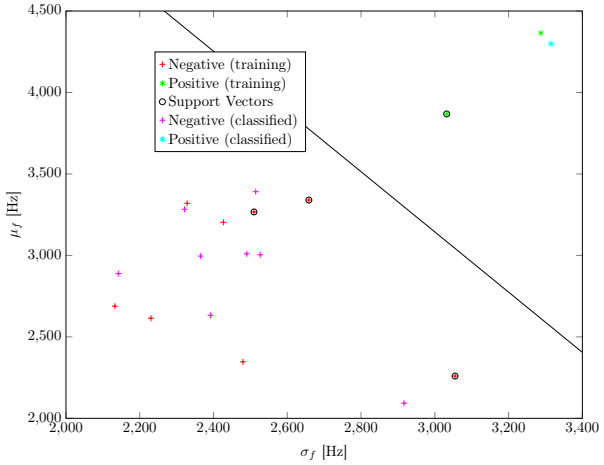


Figure 8 Training data and decision boundary (black line), as well as test data of the SVM. Half of the samples were used for training the SVM, and the other half was used as test data. After training, the SVM classified the test data correctly.

4.3 Additional sensors

In addition to audio data, measurements from an accelerometer and the robot joint torques were considered. The accelerometer was attached to the robot end-effector and was used to gather acceleration data during the experiments. The motor torque of the translational joint that was used to carry out the cut was also considered. Four different blades were used; a sharp, semi-sharp, blunt, and resharpened blade. In the top plot of Figure 9, the mean square error of the euclidean norm of the acceleration is displayed as a function of the feed rate, for each of the four blades. In the bottom plot of Figure 9, the mean of the motor torque in the sawing direction is shown. It is to be noted from both plots that, as expected, sawing with the sharp blade exhibits less vibrations and resistance for the robot. Also, a linear

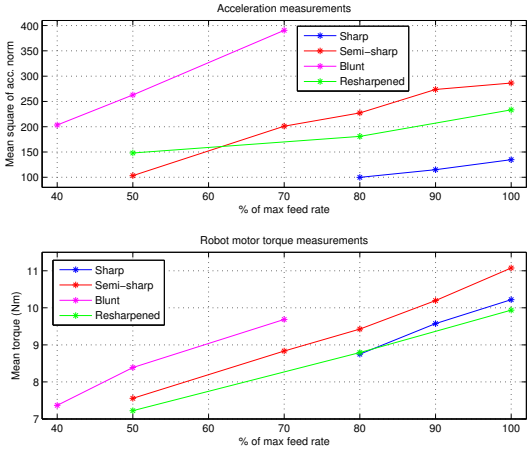


Figure 9 Acceleration and torque measurements shown as function of the feed rate, for four different sawing blades.

relationship between the torque/acceleration and the feed rate can be assumed.

From the gathered data, a model can be formed and used to identify the status of an unknown sawing blade, and in turn optimize the feed rate in order to avoid poor sawing results. However, looking at the acceleration and motor torque for the resharpened blade, the results differ a bit - the acceleration suggests that the blade is not fully sharp, while the motor torque does. This is explained by the fact that the blade has been resharpened - it is sharp, but most likely slightly warped as a result of extensive previous use. Therefore, the use of acceleration data is more advantageous than torque data, as it can detect warped blades.

4.4 Task-level error handling

Two experiments were performed with the woodworking robot in order to exemplify the task-level error handling. For both experiments a laser sensor on the panel gripper is used. This sensor is used to measure the distance from the gripper to the panels on a palette to determine how far the robot should move down. In both experiments, the process consisted of picking up two wooden-based panels from a palette and placing them on a wooden frame, nailing both panels to the frame and then sawing off the parts of the panels extending beyond the frame.

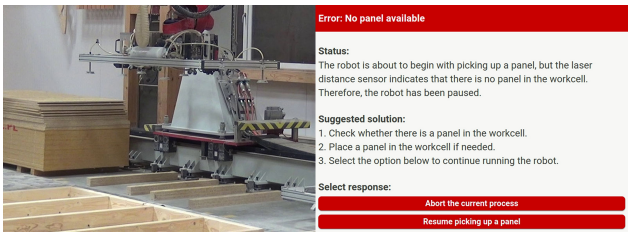


Figure 10 Missing panel experiment: *left:* missing panel, *right:* semantic error message.

In the first experiment a human-based error was introduced: The shop floor worker forgot to place new wooden-based panels on the palette until there were none left (see Figure 10 left). Without any error detection, the palette or the concrete floor could be detected as the top-most wooden-based panel resulting in robot movements that could potentially damage the robot or its tools. Using error detection the robot system would detect whether there were any panels left on the palette and alert the shop floor worker with a semantic error message in the user interface if the palette was empty (see Figure 10 right). After the suggested solution (refill panels) was performed and confirmed by the worker, the system automatically checked again if the palette was filled, and if so continued with normal execution by picking up the panel.

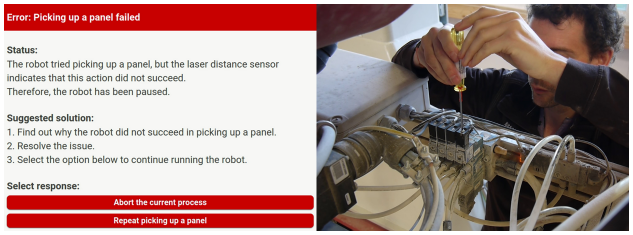


Figure 11 Gripper error experiment: *left:* semantic error message, *right:* repairing the gripper.

In the second experiment a hardware-based error was introduced: The robot detected and tried to pick up a wooden-based panel from the palette, which failed. Without any error detection, the robot system would not notice this error state and the process would ultimately fail. Fitted with error detection, the robot system detected this error by using the laser distance sensor and notified the worker (see Figure 11 left). The worker diagnosed and fixed the hardware problem on the panel gripper (see Figure 11 right) and selected the respective response option to retry and continue process execution.

5 Conclusion

We present an integrated system for robot woodworking based on cognitive technologies, that make it suitable for use in SMEs. For this purpose, we use an existing robotic hardware setup (with the addition of a few sensors for process monitoring/adaptation) in a real SME scenario. Our software supports the commonly used BTL format. This makes it compatible for use with standardized CAD/CAM tools for design and technical specification of woodworking tasks. The augmentation of this system with cognitive add-ons is a major contribution of this work. Our intuitive GUI enables untrained users to handle parsed BTL process plans. We use machine learning techniques to assess the tool wear using data from audio sensors and accelerometers. The system also detects some human errors or hardware failures at the task level, and presents the operator with semantically grounded error messages. Through several experiments involving typical woodworking tasks such as sawing, nailing and assembly, we demonstrate the feasibility of using this cognitive approach in SME scenarios, and also evaluate its specific advantages in terms of intuitiveness of use, online/offline process adaptation possibilities and error handling strategies.

6 Acknowledgement

We would like to thank Jacek Malec[17], Anders Robertsson and Rolf Johansson[18] for help and suggestions during the work. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 287787 in the project SMErobotics.

7 Literature

[1] Sven Gestegård Robertz et al. SMErobotics deliverable D4.1: First release of interoperability solutions, February 2015. Grant agreement number: 287787.

[2] Erik Jansson and Tommy Olofsson. Configuration of Software for Distributed Mobile Manipulation. Master's thesis, Lund University, Sweden, 2014.

[3] Alexander Perzylo, Nikhil Somani, Stefan Profanter, Markus Rickert, and Alois Knoll. Toward efficient robot teach-in and semantic process descriptions for small lot sizes. In *Robotics: Science and Systems (RSS), Workshop on Combining AI Reasoning and Cognitive Science with Robotics*, Rome, Italy, July 2015. `http://youtu.be/B1Qu8Mt3WtQ`.

[4] Alexander Perzylo, Nikhil Somani, Markus Rickert, and Alois Knoll. An ontology for CAD data and geometric constraints as a link between product models and semantic robot task descriptions. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, September 2015.

[5] Stefan Profanter, Alexander Perzylo, Nikhil Somani, Markus Rickert, and Alois Knoll. Analysis and semantic modeling of modality preferences in industrial human-robot interaction. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, September 2015.

[6] Nikhil Somani, Andre Gaschler, Markus Rickert, Alexander Perzylo, and Alois Knoll. Constraint-based task programming with CAD semantics: From intuitive specification to real-time control. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, September 2015. doi: 10.1109/IROS.2015.7353770. `https://youtu.be/qRJ1JmNoFEw`.

[17] Computer Science, Lund University, Lund, Sweden
[18] Both from Automatic Control, Lund University, Lund, Sweden