Robotic Technologies for Fast Deployment of Industrial Robot Systems

Emmanuel Dean-Leon, Karinne Ramirez-Amaro, Florian Bergner, Ilya Dianov, Pablo Lanillos and Gordon Cheng
Institute for Cognitive Systems, Technical University of Munich
Arcisstrasse 21, 80333 München, Germany

***Abstract*—The development of breakthrough technologies helps the deployment of robotic systems in the industry. The implementation and integration of such technologies will improve productivity, flexibility and competitiveness, in diverse industrial settings specially for small and medium enterprises. In this paper we present a framework that integrates three novel technologies, namely safe robot arms with multi-modal and auto-calibrated sensing skin, a robot control framework to generate dynamic behaviors fusing multiple sensor signals, and an intuitive and fast teaching by demonstration method that segments and recognizes the robot activities *on-line* based on re-usable semantic descriptions. In order to validate our framework, these technologies are integrated in a industrial setting to sort and pack fruits. We demonstrate that our presented framework enables a standard industrial robotic system to be flexible, modular and adaptable to different production requirements.**

I. Introduction

The demand for an increasingly high productivity level in industrial scenarios requires both shorter task execution times and faster/easy robotic systems programming methods. Automation and robotics are expected to deliver the required reduction on production costs and increase in productivity. For this purpose, an automated process using robots needs to be programmed fast and to perform as efficient as a human worker in various domains, for example packing and quality checking of products, polishing of steel molds or filling of a spray-painting machine. However, the setting up of a robotic system takes, in general, at least 3 months [1], which implies the need of robot expert programmers and higher costs. This is more prominent for Small and Medium Enterprises (SMEs), since they usually only have small production batches due to seasonal on-off production.

An interesting method to extend the flexibility and capabilities of a robot is to integrate it in close interaction with human co-workers. The fusion of the high adaptability of the human and the accuracy of a robot system can facilitate the automation of industrial processes. In this case, safety in physical human–robot interaction [2] is a fundamental aspect on developing robot technologies. Especially for the new way of teaching robots sequences using programming by demonstration methods which requires physical interactions with the robot. This method allows the operator to teach the robot tasks in an easy and natural way, hence an expert robot programmer is not required, see Fig. 1. Therefore, the development and integration of technologies such as fast configurable artificial skin, control schemes and robust teaching methods

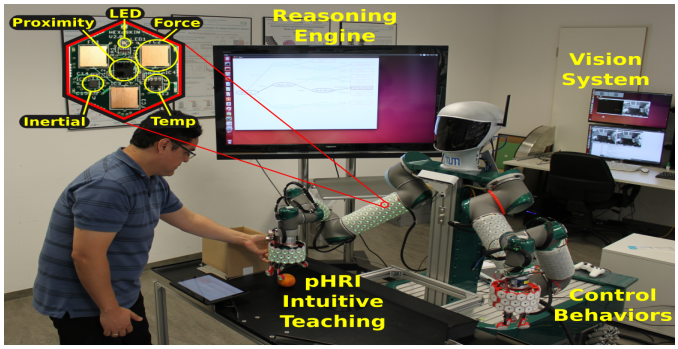

Fig. 1. Demonstration scenario: the user can intuitively teach a complete process to an industrial robot, in this case, sorting oranges. The setup is composed of an enhanced perception system (fusing artificial skin and vision), multi-modal control behaviors, and robust reasoning methods.

are needed to simplify the robot programming, to improve the safe physical interaction with robots, and to decrease the deployment time of robotic systems in the shop floors. These fast deployment technologies are needed to advance the current robot systems and it is the main focus of this paper which is being developed as part of the project Factory-in-a-Day[1].

A crucial step to properly integrate the above mentioned technologies, in a coherent framework, is the correct fusion of multiple sensors. One significant advantage of using multi-modal sensor fusion is to provide enhanced and complementary information during the parallel processing of data [3]. However, many issues arise during this fusion. This includes the adequate management, sensor synchronization and the necessity of *different levels of abstraction* to cope with signal uncertainty. Multi-modal sensor fusion requires inter-disciplinary knowledge in control theory, signal processing, artificial intelligence, probability theory and statistics. This important aspect is also addressed in this paper.

In this work, we employ a multi-modal artificial robot skin technology [4], which is fused with a vision system to extend the robot perception and interaction capabilities. In addition, we introduce a multi-modal control approach to enable different dynamic behaviors for standard industrial robots. The integration of these components is done with our novel semantic reasoning framework [5] to teach kinesthetically new activities to robots, see Fig. 1. All the previously mentioned

[1]http://www.factory-in-a-day.eu/

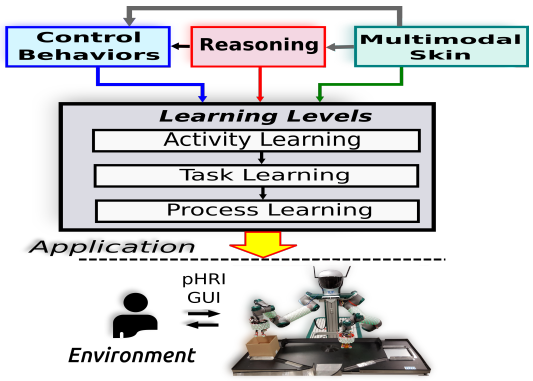

Fig. 2. Framework which integrates three main technologies for fast robot deployment: multi-modal skin, control behaviors and a reasoning engine. This framework allows an intuitive robot teaching and can re-use the acquired knowledge to different domains.

technologies will facilitate a faster installation of a robot system in industrial settings.

A. Main contributions

The key components and contributions of our framework are: a) The fusion of fast self-organizing and self-calibrating artificial skin information with visual system to enhance the robot perception, interaction and safety. b) The development of multi-modal control behaviors to allow safe physical human robot interaction beyond the standard capabilities of industrial robots. c) A multilevel approach that is capable to automatically segment and recognize robot behaviors from kinesthetic demonstrations. This approach boosts the learning phase since the demonstrated nominal trajectories are automatically segmented and recognized, therefore no off-line annotation is needed, which is typically the case in most Programming by Demonstration (PbD) methods. These three components are integrated in a framework, see Fig. 2, that allows the generation of new robot behaviors for standard industrial robots[2].

B. Related Work

One of the main problems of deploying robots in industrial processes is commonly related to the lack of knowledge in robot programming by the operator. Therefore, an expert engineer is required to manually program tasks to the robot, which is usually done using robot-specific teach-pendants [6]. This teaching task involves defining trajectories which are a set of points the robot must follow. This teaching process is time-consuming an a highly demanding task. However, in the context of industrial scenarios, programming robots in an easy and intuitive manner is an important requirement [6]. To fulfill this requirement, a new promising way for robot programming seems to be the Programming by Demonstration (PbD) [7], [8], which allows the operator to teach tasks to the robot in an easy and natural way, thus requiring no experience in robot programming. PbD methods can enable fast and flexible modifications on robot behaviors to perform a wide variety of tasks [9]. For example, [10] proposed a three-stage PbD method based on Dynamic Motion Primitives (DMPs) to Kinesthetically teach industrial robots, this method learned low-level profiles such as force and pose trajectories. Recently, [11] presented an approach to learn the task of grating vegetables based on task-space constraints (e.g. force and position of the end-effector), which are defined by a *significant* variance of the observed variables across demonstrations.

Furthermore, cognitive cyber-physical systems are used in order to develop novel robotic technologies to cope with larger variability on processes. For example, [12] presented a cognitive system applied to a concrete and well-known use case from the automotive industry, which is part of a research project called STAMINA. For fast deployment of robotic technologies the work of [13] proposes a "knowledge integration framework" which generates a generalized platform independent description of a manufacturing process. The manufacturing process is modeled by abstract tasks which contain skills. The general description is robot independent such that tasks at the higher execution levels can always be executed in the same way. The work of [14] focuses on the simplification of programming industrial robots by combining on-line and off-line programming to a more intuitive and efficient assisted on-line programming technique. Instead of teach pendants or joysticks the authors propose an intuitive input device which bases on optically tracking a special marker tool. Moving the marker tool reflects movements of the end effector. The framework uses a modular approach in which algorithms (e.g. collision avoidance) and end-effector restrictions simplify the programming process. In a different form, robot programming using augmented reality (RPAR) has been proposed by [15] where the robot programming is more intuitive and thus more flexible for SMEs.

The need of flexible and easily reconfigurable robots is discussed in [16] which addresses challenges in manufacturing process generated by the change of mindset; from mass production to mass customization. This work proposes a generic set of skills which can be combined to more complex robot tasks. This implies that the teaching process should be transformed into a high-level abstraction in order to allow generalization [17]. Furthermore, when developing new robot teaching systems for industrial applications, it is needed to fulfill the safety requirements specified by the International Organization for Standardization (ISO). The principal standards for guaranteeing safe operations of industrial robots are defined by ISO 10218-1/2 [18], [19]. In addition, since PbD methods imply physical interaction with robots, this sort of interaction is becoming a highly relevant topic inside the robotics community. Therefore, a new ISO 15066 [20] which covers safety for collaborative industrial robots has been recently published. These new standards pave the way to new robot technologies to tackle the problems in the factories of the future. In this work we proposed a new framework based on three robotic technologies to allow the fast deployment

[2]The only requirement is that the robot provides an external control interface, e.g. position, velocity or torque control.

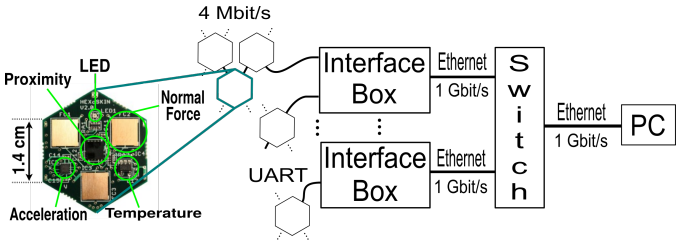

Fig. 3. Artificial skin cells and its interface to the PC. Each *skin cell* has 4 different modalities: a)Proximity, b)Normal Force, c)Acceleration and d)Temperature.

of industrial robot systems. Our framework takes in consideration, intuitive robot programming adaptable to variations on the process, knowledge-based components to enhance the robot teaching process and extended perception components to improve safety during physical robot interactions with human co-workers.

The following sections are devoted to describe the main technologies used in this work as well as experimental validation of the proposed framework.

II. Robot Multi-modal Skin

One of the keystone components of our framework is the fast configurable multi-modal artificial skin. This technology can transform a standard industrial robot into an intelligent system. It allows to easily build reactive behaviors directly in the low-level control of the robot arm to increase its safety for human robot collaboration. It also provides an extended HMI for the user, since the interface is the complete arm when the robot is covered by the artificial skin. The multi-modal sensory information generated by the artificial skin can improve the perception capabilities of a system, specially to produce local complementary information.

Our artificial skin [4] is a modularized, multi-modal robotic skin which consists of hexagonally shaped *skin cells* (see Fig. 3). Each skin cell has the same set of sensors which transduce tactile information of different modalities, such as vibrations (acceleration sensor), pressure (capacitive force sensors), pre-touch (proximity sensor) and temperature (temperature sensor). Micro-controllers located on the back of each skin sample and filter the sensor data and pack the acquired information into skin cell data packets which are then forwarded to a PC (see Fig. 3). Neighboring skin cells are connected to each other and exchange information with their neighbors. All connected skin cells realize a meshed and highly redundant skin cell network. This skin cell network is managed by the micro-controllers of the skin cells in a distributed and self-organized way. Skin cells which are directly connected to each other shape an entity called *Skin patch*. Skin Patches are connected to interface boxes (see Fig. 3) which ease the communication between skin cell network and a PC. The interface boxes allow to communicate with large skin patches via standard Gigabit Ethernet without the need for special hardware or drivers on the PC. We have developed a simple ROS driver node to integrate robotic skin within the ROS environment.

Local transformations, between *skin cells* and robot body parts, are required to map tactile information into meaningful control information, see Section III. However, to calibrate the pose of hundreds of skin cells is not a trivial task. To tackle this challenge, the artificial skin provides a 3D surface reconstruction algorithm for *skin patches* [21]. This algorithm automatically deduces the poses of every skin cell of a patch with respect to a *root cell* such that the skin cells exactly match the surface they cover. Then, the calibration task is reduced to only find the transformation of one *root cell* to a robot link. This can be done manually or using an external sensor [22].

A. Event-based signaling

Partially covering a robot arm, such as the UR-5 robot on our development platform, results in approximately 300 skin cells. Sending tactile data with a constant sampling rate of 250 Hz[3] induces a huge processing load on the PC, even when the skin is idle and no tactile interaction occurs. With 300 skin cells this processing load impacts the performance of real time control and this is more evident when the number of skin cells increases. In order to cope with this problem we use event-based signaling [23]. We take advantage of the distributed micro-controllers of the skin cells and generate events on site. In this manner, skin cells only forward tactile information whenever the local change of a sensor value exceeds a given threshold. This principle reduces the data load significantly to 20 % of the original data load and induces less computational costs which also improves the performance of the real time control.

The main goal of the artificial skin in our framework is to utilize the skin signals to teach robots in a natural way by interacting directly with the robot surface and without the need of specialized interfaces, e.g. teaching pendants, joysticks or visual markers. A virtual shield for real collision avoidance (not just collision detection) protects the robot such that the teacher must not additionally keep the robot safety in mind and can focus only on the teaching process. This will be achieved by enforcing dynamical virtual behaviors on the robot, where enforcing compliance in non-compliant robots is of particular importance. This will be explained in the following section.

III. Robot Control Behaviors

The goal of this component is two folded: a) to transform the multi-modal signals obtained from the *Robot Skin* component into meaningful information for the low-level control; b) to generate a *robot behavior* library using a composition of different low-level controls, e.g. *Kinesthetic Joint*, *Reach Joint Compliant*, see Table I. This library is the interface between the *Semantic Reasoning Learner* component, see Sec. IV, and the low-level control of the robot.

A. Transforming Multi-modal Tactile Signals into Joint Torques

In order to fuse the information from the artificial skin sensors with the different controllers available in the *Robot*

[3]250 Hz is the maximum sample rate of our artificial skin.

Behavior Library (see Fig. 6), we need to transform the sensors signals (e.g. pre-touch and pressure) to generalized force commands. In this work we use force vectors to transform tactile signals into joint torque signals. This is achieved in the following two steps.

1) Multi-modal Tactile Signals to Force Vector: Each $Cell_i$ produces a set of three *pressure signals* $f_{i_m} \in \mathbb{R}, m = 1, 2, 3$ and a single *proximity signal* $p_i \in \mathbb{R}$, see Fig. 4. The first step is to transform these signals into force vectors. By design both the pressure-signals and the proximity signal are normal to the sensor $Cell_i$, defined by its $z - axis$. Therefore the $Cell_i$ force vectors can be constructed as follows:

$$\overline{P_i} = [0, 0, w_p p_i]^T \tag{1}$$

$$\overline{F_i} = [0, 0, w_f \sum_{m=1}^{3} f_{i_m}]^T, \tag{2}$$

where $w_p, w_f \in \mathbb{R}$ are weighting gains for the proximity and pressure signals, respectively. The above equations represent the force vectors of each signal with respect to the $Cell_i$ frame, see Fig. 4. The force vector with respect to the robot base ($Link_0$) is obtained as:

$${}^{cell_i}F_{Link_0} = {}^{cell_i}R_{Link_0} \left(\overline{F_i} + \overline{P_i}\right) \tag{3}$$

where ${}^{cell_i}F_{Link_0} \in \mathbb{R}^3$ represents the total force vector produced by the tactile signals of the $Cell_i$. The rotation matrix ${}^{cell_i}R_{Link_0} \in SO(3)$ is extracted from the homogeneous transformation ${}^{cell_i}T_{Link_0}$, which is obtained as:

$${}^{cell_i}T_{Link_0} = {}^{Link_j}T_{Link_0} \left({}^{cell_i}T_{Link_j}\right), \tag{4}$$

where the set of local transformations ${}^{Cell_i}T_{Link_j}$ of each $Cell_i$ with respect to its link reference frame $Link_j$ has been obtained using the fast self-calibration feature of our artificial skin [21], [22]. In this case, $i = 1, 2, ..., s$ denotes the cell id with s as the total number of cells, and $j = 1, 2, .., n$ represent the joint id, with n as the robot's DOF.

Notice that ${}^{Link_j}T_{Link_0}$ is defined by the robot kinematic model, which can also be generated using the inertial sensors from the artificial skin [24].

2) Force Vector to Joint Torques: In the second step, the torque $\tau_i \in \mathbb{R}^n$ produced by the tactile signals of each $Cell_i$ is calculated as:

$$\tau_{Cell_i} = {}^{cell_i}J^T_{Link_0} \left({}^{cell_i}W_{Link_0}\right) \tag{5}$$

where ${}^{cell_i}W_{Link_0} = [{}^{cell_i}F^T_{Link_0}, 0^{1\times 3}]^T$ is the wrench of $Cell_i$[4]. ${}^{cell_i}J_{Link_0} \in \mathbb{R}^{6\times n}$ represents the Jacobian of the $Cell_i$ with respect to $Link_0$, which can be directly computed using the local and global calibration of the $Cell_i$. Finally, the total joint torque $\tau_{skin} \in \mathbb{R}^n$ generated by all the *skin cells* on the robot arm is computed as:

$$\tau_{skin} = \sum_{i=1}^{s} \tau_{Cell_i} \tag{6}$$

[4]We set the moment on $Cell_i = 0 \in \mathbb{R}^{3\times 1}$ since it is physically impossible to apply pure moment to an individual $Cell_i$ with respect to its own reference frame, or even measure it with the sensor.

B. Robot Control Behaviors

The mapping from tactile signals into joint torques (eq. (5)-(6)) allows the fusion of multiple controls that use the same generalized force representation, e.g. Joint Control [25], Cartesian Control [26] or Visual Servoing [27]. Thus, a simple normalized weighted-sum approach to add the contribution of each individual controller to a total joint torque output τ_Σ can be used:

$$\tau_\Sigma = w_s \tau_{skin} + \sum_{k=1}^{p} w_k \tau_k, \tag{7}$$

where $w_s, w_k \in \mathbb{R}$ are weighting values and τ_k is the control output of a controller defined by the user. We selected this fusion method to guarantee a deterministic behavior, even when local minima is present. Nevertheless, a more sophisticated approach can be used in order to select an optimal combination of controls, e.g. [28]. The weight selection for the controllers depends on the specific robot behavior that we need to generate. Some examples of the different robot behaviors are depicted on Table I. These robot behaviors are triggered by the *Skill Library*, see Sec. IV.

TABLE I
ROBOT BEHAVIORS AND ITS RELATION WITH THE CONTROLLERS.

Skill Name	Joint Ctrl	Spline Joint Ctrl	Spline Cart Ctrl	Skin Joint Ctrl	Skin Cart Ctrl	G Ctlr
Reach Joint	X			X		X
Reach Joint Goal		X		X		X
Reach Cart Goal			X	X		X
Kinesthetic Joint				X		X
Kinesthetic Cart					X	X

1) Robot Control Framework: The Robot Control Framework, depicted in Fig. 4, is designed to provide two low-level control interfaces, either *Position/Velocity* interface, available in most of the modern industrial robots, or *Torque* interface. In the case of *Torque* interface, we command directly τ_Σ to the control unit of the robot. On the other hand, for the *Position/Velocity* interface, we need to use a *Torque-to-Position Resolver*.

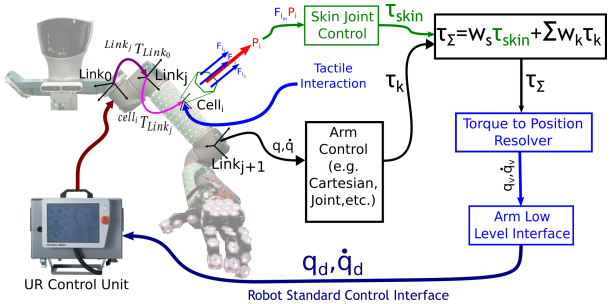

Fig. 4. Transforming skin signals into control signals.

2) Torque to Position Resolver: In order to control robots with *Position/Velocity* interface, we need to transform the total commanded joint control τ_Σ into desired joint positions/velocities. To this aim, we have implemented a torque resolver which uses the dynamic state of a nonlinear observer to generate the desired joint commands. We obtain the full dynamic model to design the observer using the kinematic models of the robot in combination with the Euler-Lagrange formulation, as explained in [29]. This observer allows to specify user-defined dynamic behaviors, e.g. it can increase the viscous friction, thus generating a slower step response to an external input (e.g. tactile interaction). The desired joint positions/velocities $(q_d, \dot{q}_d)$ generated by the torque resolver are sent to the robot using its standard control interface, see Fig. 4.

IV. Semantic Reasoning Learner

The need for an intuitive learning component is evident especially when a physical interaction with a robot is expected. An ideal solution would be a scenario where the movements from the operator are tracked, segmented and recognized by robots while kinesthetically demonstrating a new process. This is a challenging topic of research of recent years; in addition, from an industrial point of view, these methods are still not robust enough [6].

In this section, we present a novel learning by demonstration method to teach robots new processes in an *on-line* manner. This method generates compact semantic representations for automatically infer the kinesthetic demonstrations on robots. These models are robust and invariant to different demonstration styles of the same activity. Additionally, the obtained semantic representations are able to re-use the acquired knowledge to infer different types of activities. We propose a system that extracts the meaning of the demonstrated activities by means of semantic representations.

A. Workflow hierarchical structure

In this work we use the following vocabulary to recognize the robot demonstrations at different levels of abstraction. The highest level is the *Process*, which is defined as the combination of sequential *Tasks*. *Tasks* are the combination of ordered *Activities*. *Activities* are semantic descriptions of *Skills*, and finally *Skills* (lowest level) represent the primitives that robots need to execute, see Fig. 5.

Fig. 5. Hierarchical structure to define the workflow of our system. The Problem Space (marked as the blue-box) provides semantic descriptions which represents robot-agnostic knowledge, therefore it can be transfered to different domains. The Execution Space (red-box) is the specific information and routines that depends on the current robot, then only this information needs to be changed when using a different robot.

For example, the *Process* "Pack good oranges into boxes" is composed of two *Tasks* "Pick an orange" and "Place orange in box". The first task contains three activities namely "Idle", "Reach" and "Take", while the second task is defined by the activities "Put" and "Release". Each of these *Activities* is connected to a *Skill*. For example, "Reach" is linked to the "Reach Skill" primitive.

Process, *Tasks* and *Activities* are described in the *Problem Space*, and they are considered robot agnostic descriptions. They represent *what* the robot should perform, and not *how* it should be done. On the other hand, the *Skills* are defined in the *Execution Space*, and they explicitly define *how* the robot should execute an *Activity*. They represent specific routines or robot programs to execute a given *Activity*.

The main advantage of this hierarchical architecture is the re-usability and generalization of the acquired knowledge. Thus allowing the transference of knowledge generated in the Problem Space to different domains, see Fig. 5.

B. Activity inference from robot demonstrations

In order to infer the demonstrated activities, we present a two-step semantic-based approach. First, we extract the *low-level* features from the perceived environment (e.g. signals from the sensors), and as a second phase we automatically generate compact semantic rules to *deterministically* infer the robot activities from the demonstrations.

The *low-level* features are considered as the atomic representations from demonstrations, in this case the robot's End-Effector (EF) motions are segmented into one of the following categories, similar to our previous work [30]. *Move*: the EF is moving, i.e. $\dot{x} > \varepsilon$. *Not move*: the EF stops its motion, i.e. $\dot{x} \to 0$, where $\dot{x}$ is the EF velocity.

Furthermore, we also need to detect the following properties of the objects from the vision, skin, and robot sensors. *ObjectActedOn*[5] *(o_a)*: the EF is moving towards an object, i.e. getting closer to the object, $d(x_{ef}, x_{o_i}) = \sqrt{\sum_{i=1}^{n}(x_{ef} - x_{o_i})^2} \to 0$. *ObjectInHand ($o_h$)*: the object is in the EF, i.e. o_h is currently manipulated, i.e. $d(x_{ef}, x_{o_i}) \approx 0$. *GripperState ($g_s$)*: the gripper is open or closed, where $d(\cdot,\cdot)$ is the distance between the EF position (x_h) and the position of the detected object (x_{o_i}) from a common coordinate frame. The output of this module determines the current state of the system (s_s), which is defined as the quadruplet $s_s = \{m, o_a, o_h, g_s\}$. Then, we used the perceived state of the system (s_s) to obtain the semantic rules. For this, we use a similar pipeline as the one presented in [31], where the C4.5 algorithm is employed to compute a decision tree (T) which contains the semantic descriptions of the robot demonstrated activities. The obtained tree can be observed in Fig. 8.

The advantage of this abstract representation is that it allows to obtain more generic models from demonstrations, even when the information is obtained from different scenarios as well as several sources of input data. Another important aspect

[5]The information from the object can be obtained either from the vision system or the proximity sensor of the skin. The same is valid for the property of ObjectInHand.

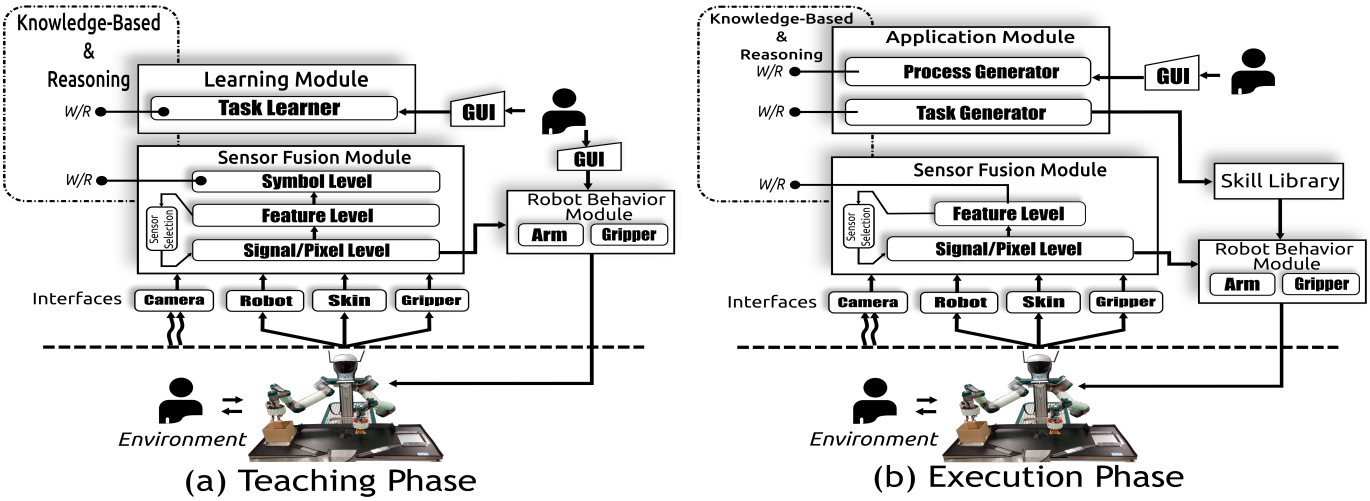

Fig. 6. General Diagrams for the Robot Teaching/Learning and Execution.

of our system is its scalability and adaptability toward different domains as presented in [5], [32]. It is important to highlight that the semantic rules are obtained off-line and can be re-used directly in multiple domains as an on-line component.

C. Task and process learner

Following the hierarchy described in subsection IV-A, after segmenting and inferring the robot activities, we can learn new tasks. In order to do this, we require the input from the user via GUI to name the learned task and to select the activities that conform this task from a list of automatically inferred activities. When the task is saved, the learned information is stored in the knowledge-base. Furthermore, per each activity a new skill is generated and stored in the knowledge-based which also contains context parameters necessary to execute that activity (see Fig. 6 (a)). For example, for the activity "Put-SomethingSomewhere", the parameters generated for the skill are *something* and *somewhere*, where *something* is instantiated when a new object is detected (e.g. orange) and *somewhere* identifies the final position of the activity (e.g. box, trash, or squeezable area).

In order to learn a new *Processes* we also need the input from the user via GUI, which during *running time* connects to the knowledge-based and retrieves the previously learned tasks and skills. Then, the available tasks are displayed and the user selects the tasks that define the new process, along with the stop criteria[6]. When the generated *Process* is executed, the system obtains the name of the object from the vision system *on-line* and this instantiates the context parameters needed to execute the inferred tasks. At this point, the system infers which tasks can be executed given the perceived data. For example, when the perceived object is an *orange* the "Squeeze" activity can be executed since *oranges* have the property of *squeezable*, however if the detected object is an *apple* this activity can not be executed due to that apples are not *squeezable*. Therefore, our system is able to adapt to these differences on the object properties without human intervention, which makes our system more general and flexible when changing objects in the production lines on SMEs without the need to re-program the robot.

[6]The stop criteria indicates when a process should stop, e.g. duration, limit weight of objects, or maximum number of objects.

V. Demonstration Scenario – Packing Fruits

Our system integrates two different phases in a single framework, these phases are *Teaching phase* and *Execution phase*, see Fig. 6. Both phases are defined by three main components a) Sensor Fusion module, b) Learning/Application module, and c) Robot Behavior module. In the Teaching phase, see Fig. 6 (a), a knowledge-based is generated and populated on-demand according to the inputs obtained from the demonstration. Afterwards this knowledge-based will be used during the Execution phase to retrieve the learned information and execute the proper robot behaviors, see Fig. 6 (b).

As a demonstration scenario, we consider the task of sorting fruits. With this scenario, we can exploit the benefits of using the tactile and proximity sensors of the artificial skin to sense the quality of the products, in this case oranges. In this demonstration scenario the human is teaching the robot the intermediate activities required to sort and pack oranges into boxes when the oranges are good (they show a rigid texture), or the oranges will be thrown to the trash container when the oranges are bad (soft texture), see Fig. 8. It is possible to observe that this complex task implies the integration of different sensors and a proper mapping method to infer the taught activities. This scenario was inspired by the standard process of orange sorting where the humans use their tactile sensation to discriminate the good oranges from the bad oranges.

Our proposed demonstration has been successfully implemented in our robotic platform Tactile Omni-directional Mobile Manipulator (TOMM), see Fig. 1. TOMM is composed of two industrial robot arms (UR-5) [33] covered with artificial skin, two Lacquey grippers also covered with our artificial

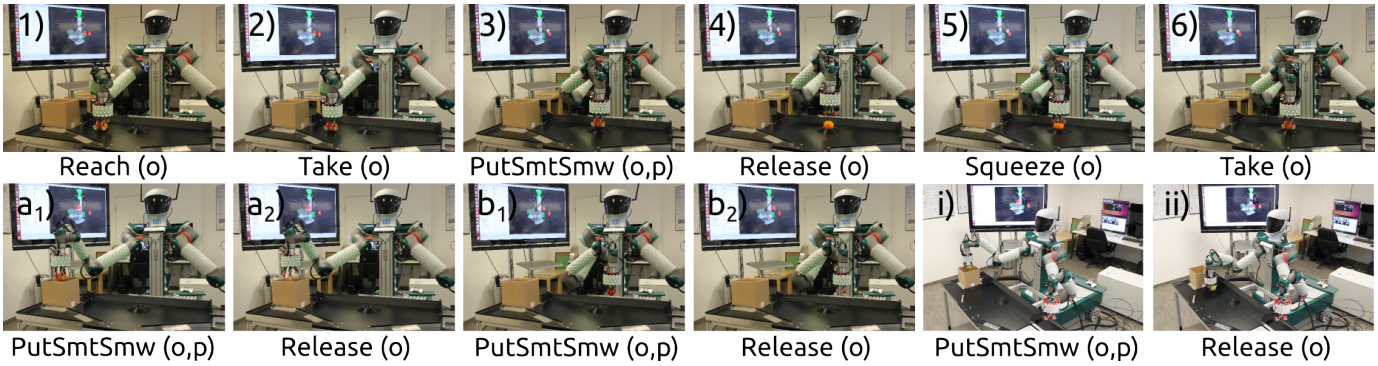

Fig. 7. Execution of the learned sequences for the scenario of packing oranges and apples. 1)–6) exemplifies the sequence of activities that are executed for sorting oranges. a1)–a2) are the activities for "good oranges". b1)–b2) are activities for "bad oranges". Finally *i*)–*ii*) are the inferred activities for apples.

skin and 2 cameras on its fixed head used to obtain the 3D position of the target objects. The artificial skin interface, control schemes and inference system are implemented in two workstations running on Ubuntu 14.04 Linux/OS. The specifications of the PCs are as follow: Intel Core i7-4702MQ CPU @ 2.20GHz. The complete system has been integrated using ROS Indigo middle-ware [34].

A. Experiment 1: Teaching Packing Oranges

In this part of the experiment, the user is guiding the robot by enabling the Kinesthetic_Cart robot behavior, see Table I, as well as controlling the robot gripper state (open/close). Then, the *Semantic Reasoning* component will automatically segment and infer the demonstrated activities. It automatically generates semantic descriptions of the tasks in each step, and stores the relevant information in the knowledge-based (e.g. skill parameters), see Fig. 8. Later, the inferred activities can be retrieved in sequential order and the user can easily create new tasks. It is important to highlight that the user will demonstrate only once the activities involved in the desired process, in this case, sorting oranges. This teaching process can be observed in the following video[7]

B. Experiment 2: Executing Packing Oranges

The tasks generated by the user are also stored in the knowledge-based and a new process can be generated. This process is generated with un-bounded variables, which will be instantiated during running time, taking in consideration the information obtained from the multi-modal perception system (tactile skin, vision, robot state, etc.). In this case, the user creates the process of sorting oranges which consists on the following sequential tasks: $T_1\{Pick\_Fruit\}$ = [1) Reach($object$), 2) Take($object$)], $T_2\{Identify\_Good\_Fruit\}$=[3) Put($object, place$), 4) Release($object$), 5) Squeeze($object$),...,6) Take($object$)], where $object = orange$ and $place = squezable\_area$. If the stiffness of the orange is high, then it is considered as "good-orange" and the following task is executed: $T_3\{Place\_Fruit\_Box\}$=[a1) Put($object, place$), a2) Release($object$)], where $object = orange$ and $place = box$. On the other hand, if the orange is soft, then it is considered as a "bad orange", then the following task is executed: $T_4\{Place\_Fruit\_Trash\}$=[b1) Put($object, place$), b2) Release($object$)] in this case $object = orange$ and $place = trash$, see Fig. 7.

[7]https://youtu.be/jdLpYUG3Td0

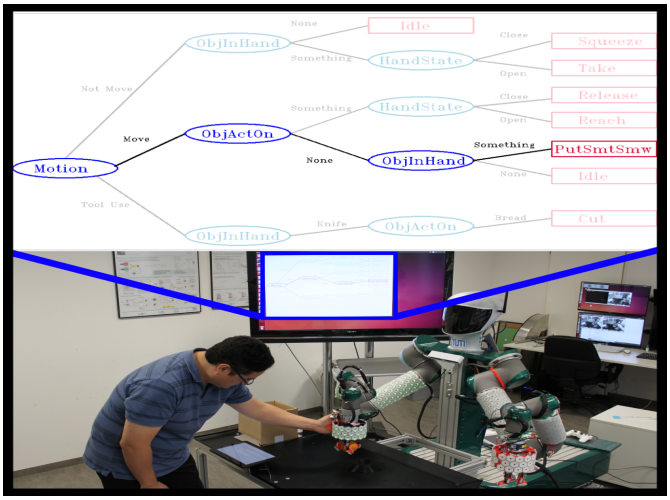

Fig. 8. Learning by demonstration the activities involved on the packing oranges scenario. The figure depicts the demonstration of "PutSomething-Somewhere" activity.

C. Experiment 2: Executing Packing Apples

In order to demonstrate the adaptability and flexibility of our framework, we place an apple (instead of an orange) in the robot work-space and execute exactly the same learned process of sorting oranges. Then, our framework automatically bounds the new perceived object (apple) and its properties to the process variable $object$. When running the process, the list of tasks is verified and only those tasks whose requirements are fulfilled will be executed. For example, the task T_2 requires that $object$ has the property $squeezable$, therefore this task can not be executed. This is automatically inferred by the *Semantic Reasoning* component and the next task will be evaluated and if it satisfied then it is executed. In this case only T_1 and T_3

will be executed, see Fig. 7 *i)* and *ii)*. Notice that for these tasks the variables are $object = apple$ and $place = box$.

VI. Conclusion

This work introduced three main robotic technologies to enable fast deployment of industrial robot systems. These technologies are integrated in a flexible framework which exploits: a) a fast self-configurable artificial skin, b) a multi-modal control framework to extend the dynamic behaviors of standard robots and c) a robust and intuitive PbD method based on semantic reasoning. The presented framework is robust, adaptable, flexible and intuitive to new situations due to the re-usability of learned rules. Our framework provides an intuitive robot teaching method since it allows a non-robotic-expert user to physically interact with the robot arm, where the acquired knowledge is defined as higher-level human-readable descriptions. Furthermore, the extended perception capabilities offered by the fusion of artificial skin and vision systems, together with multi-modal control behaviors and reasoning, creates an enhanced safety system for physical human robot interaction. This framework can be implemented in any standard industrial robot as long as it provides an external control interface, either Position, Velocity or Torque commands.

ACKNOWLEDGMENT

This work has received funding from the European Community's Seventh Framework Program (FP7/2007-2013) under grant agreement no. 609206.

References

[1] A. Jäger, C. Moll, O. Som, C. Zanker, S. Kinkel, and R. Lichtner, "Analysis of the impact of robotic systems on employment in the European Union. Final report. Luxembourg: Publications Office of the European Union, 2015," 2015.

[2] A. D. Santis, B. Siciliano, A. D. Luca, and A. Bicchi, "An atlas of physical human–robot interaction," *Mechanism and Machine Theory*, vol. 43, no. 3, pp. 253–270, 2008.

[3] R. C. Luo and C.-C. Chang, "Multisensor Fusion and Integration: A Review on Approaches and Its Applications in Mechatronics." *IEEE Trans. Industrial Informatics*, vol. 8, no. 1, pp. 49–60, 2012.

[4] P. Mittendorfer and G. Cheng, "Humanoid Multimodal Tactile-Sensing Modules." *IEEE Trans. Robotics*, vol. 27, no. 3, pp. 401–410, 2011.

[5] K. Ramirez-Amaro, M. Beetz, and G. Cheng, "Transferring skills to humanoid robots by extracting semantic representations from observations of human activities," *Artificial Intelligence*, 2015.

[6] D. Massa, M. Callegari, and C. Cristalli, "Manual guidance for industrial robot programming." *Industrial Robot*, vol. 42, no. 5, pp. 457–465, 2015.

[7] S. Calinon, F. D'halluin, E. L. Sauser, D. G. Caldwell, and A. G. Billard, "Learning and reproduction of gestures by imitation: An approach based on Hidden Markov Model and Gaussian Mixture Regression," *IEEE Robot. and Autom. Magazine*, vol. 17, no. 2, pp. 44–54, June 2010.

[8] P. Kormushev, S. Calinon, and D. G. Caldwell, "Imitation Learning of Positional and Force Skills Demonstrated via Kinesthetic Teaching and Haptic Input," *Advanced Robotics*, vol. 25, no. 5, pp. 581–603, 2011.

[9] R. Dillmann, T. Asfour, M. Do, R. Jäkel, A. Kasper, P. Azad, A. Ude, S. R. Schmidt-Rohr, and M. Lösch, "Advances in Robot Programming by Demonstration." *KI*, vol. 24, no. 4, pp. 295–303, 2010.

[10] W. K. H. Ko, Y. Wu, K. P. Tee, and J. Buchli, "Towards Industrial Robot Learning from Demonstration." in *HAI*, M. Lee, T. Omori, H. Osawa, H. Park, and J. E. Young, Eds. ACM, 2015, pp. 235–238.

[11] A. Ureche, K. Umezawa, Y. Nakamura, and A. Billard, "Task Parameterization Using Continuous Constraints Extracted From Human Demonstrations." *IEEE Trans. Robot.*, vol. 31, no. 6, pp. 1458–1471, 2015.

[12] V. Krüger, A. Chazoule, M. Crosby, A. Lasnier, M. R. Pedersen, F. Rovida, L. Nalpantidis, R. P. A. Petrick, C. Toscano, and G. Veiga, "A Vertical and Cyber-Physical Integration of Cognitive Robots in Manufacturing." *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1114–1127, 2016.

[13] A. Björkelund, H. Bruyninckx, J. Malec, K. Nilsson, and P. Nugues, "Knowledge for Intelligent Industrial Robots." in *AAAI Spring Symposium: Designing Intelligent Robots*, ser. AAAI Technical Report, vol. SS-12-02. AAAI, 2012.

[14] B. Hein, M. Hensel, and H. Wörn, "Intuitive and model-based on-line programming of industrial robots: A modular on-line programming environment." in *ICRA*. IEEE, 2008, pp. 3952–3957.

[15] Z. Pan, J. Polden, N. Larkin, S. van Duin, and J. Norrish, "Recent Progress on Programming Methods for Industrial Robots." in *ISR/ROBOTIK*. VDE Verlag, 2010, pp. 1–8.

[16] M. R. Pedersen, L. Nalpantidis, R. S. Andersen, C. Schou, S. Bøgh, V. Krüger, and O. Madsen, "Robot skills for manufacturing: From concept to industrial deployment," *Robotics and Computer-Integrated Manufacturing*, vol. 37, pp. 282–291, 2016.

[17] M. N. Nicolescu and M. J. Mataric, "Natural methods for robot task learning: Instructive demonstrations, generalization and practice," in *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*. ACM, 2003, pp. 241–248.

[18] ISO-10218-1, "ISO 10218-1, "Robots and robotic devices – safety requirements for industrial robots – Part 1: robots", Norm ISO 10218-1, Int. Organization for Standardization." 2010.

[19] ISO-10218-2, "ISO 10218-2, "Robots and robotic devices – safety requirements for industrial robots – Part 2: robot systems and integration", Norm ISO 10218-2, Int. Organization for Standardization." 2011.

[20] ISO/TS-15066, "ISO/TS15066, Safety for collaborative industrial robots. Technical Standard ISO 15066." 2014.

[21] P. Mittendorfer and G. Cheng, "3D surface reconstruction for robotic body parts with artificial skins." in *IROS*. IEEE, 2012, pp. 4505–4510.

[22] P. Mittendorfer, E. Dean, and G. Cheng, "3D spatial self-organization of a modular artificial skin," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2014, pp. 3969–3974.

[23] F. Bergner, P. Mittendorfer, E. C. Dean-Leon, and G. Cheng, "Event-based signaling for reducing required data rates and processing power in a large-scale artificial robotic skin." in *IROS*. IEEE, 2015, pp. 2124–2129.

[24] P. Mittendorfer, E. Dean, and G. Cheng, "Automatic robot kinematic modeling with a modular artificial skin," in *2014 IEEE-RAS International Conference on Humanoid Robots*, Nov 2014, pp. 749–754.

[25] V. Parra-Vega, S. Arimoto, Y.-H. Liu, G. Hirzinger, and P. Akella, "Dynamic sliding PID control for tracking of robot manipulators: theory and experiments," *IEEE Transactions on Robotics and Automation*, vol. 19, no. 6, pp. 967–976, Dec 2003.

[26] L. G. Garcia-Valdovinos, V. Parra-Vega, J. A. Mendez-Iglesias, and M. A. Arteaga, "Cartesian sliding PID force/position control for transparent bilateral teleoperation," in *31st Annual Conference of IEEE Industrial Electronics Society, 2005. IECON 2005.*, Nov 2005.

[27] E. Dean-León and G. Cheng, "A new method for solving 6D Image-Based Visual Servoing with virtual composite camera model," in *2014 IEEE-RAS Int. Conf. on Humanoid Robots*, Nov 2014, pp. 519–525.

[28] E. Aertbeliën and J. D. Schutter, "eTaSL/eTC: A constraint-based task specification language and robot controller using expression graphs," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2014, pp. 1540–1546.

[29] E. Dean-Leon, S. Nair, and A. Knoll, "User friendly matlab-toolbox for symbolic robot dynamic modeling used for control design," in *Robotics and Biomimetics (ROBIO), 2012 IEEE International Conference on*, Dec 2012, pp. 2181–2188.

[30] K. Ramirez-Amaro, M. Beetz, and G. Cheng, "Automatic Segmentation and Recognition of Human Activities from Observation based on Semantic Reasoning ," in *IEEE/RSJ IROS 2014*. IEEE, Sept 2014.

[31] ——, "Understanding the intention of human activities through semantic perception: observation, understanding and execution on a humanoid robot." *Advanced Robotics*, vol. 29, no. 5, pp. 345–362, 2015.

[32] K. Ramirez-Amaro, E. C. Dean-Leon, and G. Cheng, "Robust semantic representations for inferring human co-manipulation activities even with different demonstration styles." in *Humanoids*. IEEE, 2015, pp. 1141–1146.

[33] "UR5: Universal Robots," http://www.universal-robots.com/.

[34] "ROS: Robot Operating System," http://www.ros.org/.