An intelligent robotics modular architecture for easy adaptation to novel tasks and applications

F. Gosselin, G. Acher, B. Gradoussoff, S. Kchir, F. Keith, O. Lebec, C. Louison, B. Luvison, F. Mayran de Chamisso, B. Meden, et al.

An intelligent robotics modular architecture for easy adaptation to novel tasks and applications*

F. Gosselin, G. Acher, B. Gradoussoff, S. Kchir, F. Keith, O. Lebec, C. Louison, B. Luvison, F. Mayran de Chamisso, B. Meden, V. Molina, M. Morelli, J. Rabarisoa, C. Vienne, G. Ameyugo

***Abstract*— Industrial robots significantly contributed to the increase of quality and productivity in the industry. Still, their deployment and use remain complex and expensive, limiting their main market to mass production in large factories. This article introduces an intelligent robotics framework intended to solve this issue. It relies on a four-layer modular architecture associating a components-agnostic orchestrator coordinating software modules accessed through a standard middleware, and different hardware running the required functions. This architecture is implemented for performing various tasks in autonomy or in collaboration with a human operator, the different components being turned on and adapted on-demand according to the use-case requirements. We illustrate the proposed concept on four robotic sequences: the assembly of a representative gear unit with one arm, the same application with two robots, the Robothon® Grand Challenge and the insertion of deformable objects in a rail.**

I. Introduction

Industrial robots, which significantly contributed to an increase in productivity and quality in various industries, remain mostly limited to mass production in large factories, and a novel generation of more intelligent and flexible robots is required to address the challenge of agile production. The ability to autonomously adapt to variations of the environment and the capability to be easily reconfigured by non-expert users are now generally accepted as pre-requisites to address smart manufacturing and make robots accepted by SMEs, which have up until now rarely adopted them due to their low volume and customized productions [1]. In this context, cobots, or collaborative robots, are more and more seen as a first step towards intelligent robotics [2], especially as their safe design allows them to be easily manipulated by humans, who can guide the robot manually along the desired trajectories. This so-called programming by demonstration approach [3] proves efficient in practice for relatively simple tasks, such as pick and place, but the efficient programming of more complex tasks asks for higher-level functionalities. To this end, skills, that encapsulate complex robots behaviors in high-level functions addressed with a minimal set of parameters [4], are a powerful tool for defining elementary subtasks, with complementary approaches based on Deep Learning (DL) [5] for complex processes. However, DL approaches require a large amount of data and/or demonstrations and long training procedures to adjust the numerous models' parameters, with a lack of robustness and generalization difficulties which still prevent easy adaptation to novel tasks, novel set-ups and changing conditions. Also, even if an operator can be part of the learning process (e.g. to set-up the experiment, to provide human demonstrations …), Reinforcement Learning (RL) most-often targets full autonomy at the end of the process. This is still an issue as success rates as high as 90-95% are way lower than the reliability required for real operations in factory environments, *de facto* limiting these end-to-end approaches to laboratory and experimental settings, notwithstanding the limited ability to handle unanticipated situations. Regarding these limitations, rather than an end-to-end learning methodology, we explore in this article a more conservative multi-layer approach (here four), and natively integrate in our solution both autonomous and human-assisted modes. This approach is "Industry 5.0-oriented", in the sense that it relies on the principle of human-machine cooperation, allowing to integrate human intelligence in the loop and combine human reactivity with the precision, speed and reliability of robots

The structure of our paper is as follows: after a review of relevant related work in section II, our architecture is introduced in section III, with a first example of implementation for industrial assembly presented in section IV. In section V, we show the flexibility of the concept, which is adapted to three other configurations and / or use-cases. Finally, section VI discusses the results and section VII concludes the paper.

II. Related work

A key element for easy adaptation to different hardware is the availability of vendor-independent standards allowing to seamlessly connect and use various components and devices. In [6] for example, the authors explore the use of OPC UA, instead of proprietary protocols, to link and control robots and machines that expose different skills in a simplified and standardized way. OPC UA is also used in [7] to develop plug and produce solutions, with additional discovery mechanisms allowing the automatic detection of novel components and skills. It is evaluated in representative environments figuring workstations with different robots and gripper that are quickly discovered and made available. However, to allow for a standardized communication between various machines, industrial standards (such as OPC UA) introduce communication and synchronization overheads which require a complex software infrastructure that negatively impacts performances [8]. Also, the reported experiments are limited to workshop-level proof-of-concept setups. Industrial solutions exist that allow addressing the problem at the scale of a whole factory, but they remain proprietary, complex to implement and expensive.

*Part of this research was supported by the "Agence Nationale de la Recherche" (Carnot Institute financial support N°19 CARN 0006 01 and N°20 CARN 0006 01). Some of these developments also received funding from the European Union under the Horizon 2020 research and innovation programme (projects Merging, grant agreement No. 869963, and RobMoSys, grant agreement No. 732410).

All authors are with the Université Paris-Saclay, CEA, LIST, F-91120 Palaiseau, France (e-mail : florian.gosselin@cea.fr).

Beside these attempts of the industrial automation's community to obtain agile production systems through the application-dependent adaptation of the components used, the robotics community tried to address the problem with self-contained intelligent robots able to adapt to various situations thanks to the integration of all necessary functions in the robot itself. The humanoid robot ARMAR-6 presented in [9] includes for example grasping, mobile manipulation, integrated perception, compliant-motion execution, and natural language understanding to allow for helping a maintenance technician in his or her work on a conveyor. The designers of ARMAR-6 made the choice of a vertical integration during the system development, relying on a proprietary software architecture. While not preventing the exchange of key components thanks to a modular design and standard interfaces, its complexity, with a high number of interconnected modules that create a complicated network of dependencies, renders such evolutions complex. A similar approach prevails for the HRP-4 and TORO robots presented in [10]. Both are also highly integrated humanoid robots featuring multi-contact planning and control functionalities, walking abilities, embedded localization and mapping, visual and force control, contact detection and safety functions. They are used to perform complex assembly tasks in full autonomy. The robots achieve remarkable results regarding the complexity of the use-case but there are still limitations both in terms of robustness and performances, and these robots are not industry-ready to date. Additionally, despite their modular architecture, their integrated nature renders any evolution complex and expensive.

In addition to these humanoid designs, a diversity of "autonomous industrial mobile manipulators" (AIMM) have been proposed this last decade for logistics, machine tending or manipulation tasks in industrial environment [11]. The Little Helper system [12] consists in a robot manipulator mounted on a non-holonomic mobile platform dedicated to a large variety of tasks in semi-structured industrial environments, including navigation, pick-and-place or quality control. It relies on a software architecture consisting of distributed servers providing services for each hardware subsystem and it is used for machine tending and interactions with visitors during a trade fair. In [13], an ulterior version of Little Helper is presented. It integrates an intuitive GUI and a screwing skill used to assist workers in the maintenance of injection molds. However, the capabilities of the system are limited and have to be enhanced to address a larger scope of industrial tasks. More recently, the mobile manipulator OMNIVIL [11] was deployed in a representative factory mock-up, with three workstations between which it transports goods and preassembled parts. It integrates autonomous navigation, workspace monitoring and adaptive manipulation based on visual servoing through a software architecture divided into lower-level and higher-level tasks, with an integrated state machine that plans the global task execution. While proving able to realize tasks involving collaboration between robots and machines, it remains limited to simple manipulation tasks performed on parts equipped with augmented reality markers. Another example is the collaborative robot for the factory of the future introduced in [14]. Designed to be modular by nature, this dual-arm mobile manipulator combined the latest sensing and actuation technology at the time it was constructed in order to allow efficient navigation, dual arm manipulation and interactions with humans. These functions are necessary to address the needs of automotive or aeronautics industry (e.g. kitting). To this end, a large number of components is used (mobile base, arms and hands, force sensors and cameras …), resulting in an heterogeneous architecture which required the robot's designers to develop their own standardization framework, making evolutions here also complex.

The recent developments on flexible robotics have also led to several robotic frameworks mainly dedicated to facilitate the definition of the manipulation scenario by non-expert users. The skill-based platform SkiROS [15] is proposed for organizing the ontology of the scene and using it to divide complex procedures into skills, that are automatically combined to execute the task. The architecture is evaluated on a Fanuc robot mounted on a mobile platform to perform kitting operations in a real industrial site. MaestROB [16] introduces additional functionalities like for example exchanges with a human using natural language. It can be used to plan complex tasks after human demonstration then execute them with the same or a different robot, with the assistance of an operator if needed. The modular system CoSTAR [17] is based on Behavior Trees and relies on a relatively small set of geometric predicates that have to be combined by the operator to specify its task.

In this paper, we introduce an intelligent robotics framework that combines the best practices and advantages of automation (use of standards, modularity, proven ability to exchange major components of the system) and robotics (ability to collaborate with humans and to handle various tasks efficiently). What we claim is:

- A four-layer modular architecture allowing adaptation of the robotic set-up according to the targeted application.
- Its proven ability to adapt to four different applications with only limited efforts.
- Its capability to reach state-of-the art performances in an open competition context.

III. INTELLIGENT ROBOTICS MODULAR ARCHITECTURE

Building on the best practices learnt from past projects like HORSE (http://www.horse-project.eu/) and RobMoSys (https://robmosys.eu/), we adopted a model-based software engineering approach that relies on the principles of separation of concerns and separation of roles. The principle, which consists in addressing more abstract and general concepts at the higher levels of the architecture and specializing individual and concrete concerns in the lower layers, reduces the integration efforts required when composing modules from multiple stakeholders and enables the independence from specific execution frameworks. The definition of appropriate abstractions and interfaces between the different layers and components allows developers to focus only on their own work and responsibilities, thus drastically reducing the development and integration efforts. This approach is part of an ecosystem of initiatives for manufacturing systems that establishes a clear separation between communication protocols and data requirements (https://sparc-robotics-portal.eu/web/software-engineering/stewardship-software-engineering-systems-integration-and-systems-engineering) and enables independence from specific execution frameworks (vendor lock-in).

As shown in Fig. 1, the intelligent robotics architecture presented in this article distinguishes four layers. The system orchestrator on top of it is in charge of representing and formalizing the tasks to be executed in the form of successive elementary functions whose logic of execution can be simulated and tested before its implementation on the real robot(s). The orchestrator can be formalized in different forms such as Behaviours Trees, Finite State Machines, or even simple lists of actions. A standard middleware is used to connect it to the main software components which are implemented in the form of high-level functionalities (e.g. robotics and vision skills, HMI, Digital Twin). When such a functionality is called, its components are activated and it begins to perform the task it is in charge of. The details of the software implementation are not of interest at this level. The orchestrator only needs to know that the functionalities and skills are available and that they propose the necessary strategies to achieve the task. Only the basic information required to assess the status of the different steps (e.g. launch, success or failure of a component) are exchanged between the orchestrator and the software components. As some components can also require specific information computed by other components (e.g. an object's representation in the form of a point cloud computed by a vision module can be useful to update the digital twin used to compute a robot's trajectory), direct communication (i.e. without passing through the orchestrator) also occurs between the interested components. Here again, the most efficient standards (e.g. WAMP) are used. The hardware is found at the bottom layer; with components implemented in terms of configurable parameters, making this architecture hardware-agnostic, i.e. if we decide for example to change a robot or camera, the robotic skills and vision algorithms are reusable.

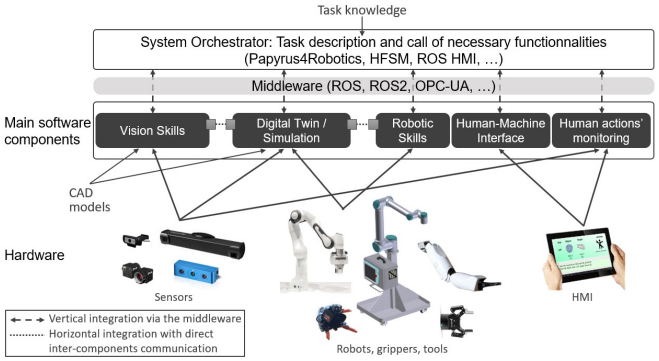

Fig. 1. Intelligent robotics modular software architecture.

IV. The Robot Companion Setup

The modular architecture presented above was first used for the design of Robot Companion, a demonstrator of intelligent interactive robotics intended to perform industrial assembly tasks autonomously or with the help of a human operator. As shown in Fig. 2, this set-up integrates various components. The robot is a Franka Emika collaborative robot equipped with a bi-digital gripper. It was chosen as it gives full access to the low level controller, allowing the fine tuning and optimization of the robot skills. It is associated with two vision systems. The first one is composed of a Logitech C930e 2D webcam and a Photoneo Phoxi 3D Scanner M 3D active camera. It is in charge of the recognition and localization of the to-be-assembled objects, while the second one, based on a Logitech C930e webcam, is dedicated to the monitoring and recognition of the human operator's activity. A digital twin of the whole platform completes the set-up, and a HMI is used to display information and interact with the operator.

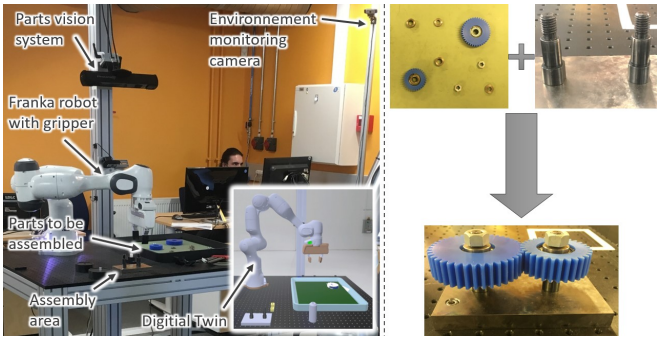

Fig. 2. The Robot Companion first demonstrator (left) and modified gear assembly use-case (right).

The target task, inspired by state-of-the art assembly challenges [18], consists in assembling a gear system composed of a fixed base plate with 2 threaded shafts, and 8 mobile parts (2 spacers, 2 gears, 2 washers and 2 nuts). It was slightly modified compared to the reference challenge: as shown in Fig. 2, the parts are initially placed randomly in a tray fixed on the table, and, as we have only one robot in our first demonstrator, the base plate is fixed on the table, and the axes are pre-assembled. This task is challenging for both vision (with small, shiny and similar parts for which we make no assumption on their localization inside the tray) and robotics (with assembly tolerances below the precision and repeatability of the robot).

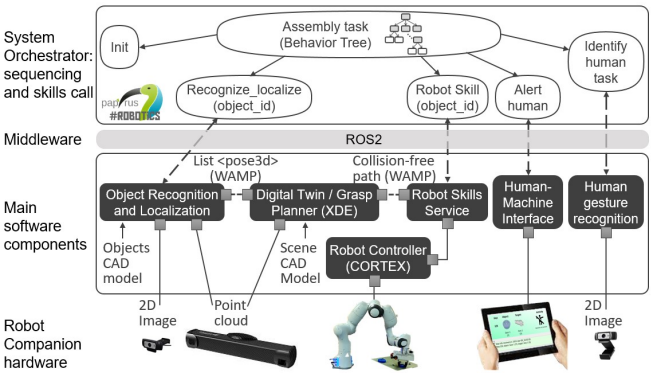

Fig. 3. Robot Companion first demonstrator software architecture.

The instantiation of the general software architecture presented in section III is illustrated in Fig. 3. ROS 2, the *de-facto* standard open-source software platform used in industry and research for the development of advanced robotics applications, is used as middleware. Conversely, the Model-Based Software Engineering platform *Papyrus for Robotics* (https://www.eclipse.org/papyrus/components/robotics/), an open-source, Eclipse-based, low-code environment that supports code generation for ROS 2 and reverse engineering of ROS 2 based software systems is used to design the orchestrator. In this framework, the task is represented as a Behaviour Tree (BT) which reflects the main phases of the assembly that are repeated until they all complete with success: selection of the next object in a

database, recognition and localization of this object, grasping and insertion or screwing. To this end, the following software components are used (see [19] for more details):

Object Recognition and Localization: a state of the art EfficientDet D2 object detector [20] is used to recognize the objects in the webcam's 2D images and identify their bounding boxes. The mapping of the latter on the 3D images acquired by the Photoneo sensor allows extracting local points clouds (LPCs) which are used to localize the objects in 6 degrees of freedom [21]. This process is GPU-accelerated and requires less than 0.5s to localize all objects with a submillimetric precision. A last step consists in using a local model of the gripper to automatically generate candidate grasping configurations which are ranked according the absence of collisions and ease of access.

Digital Twin: a Digital Twin exploiting the XDE physics engine [22] is used to run a Rapidly-Exploring Random Tree path planning algorithm and find a collision-free path between the current configuration and one of the previous grasping configurations, beginning with the best candidate and continuing until a suitable path is found. The same algorithm is subsequently used to find a collision-free path between the grasping and assembly configurations. Thanks to the ability of XDE to work with both CAD data and point clouds, the world model is updated at each step with the data provided by the 3D camera. A link with the robot controller is also provided, allowing the model to run concurrently with the real robot for monitoring purposes.

Robot Skills: the robot skills are high-level functions (eg. grasp, insert, screw) implemented in a dedicated Python framework. Internally, the skills are translated into joint position and/or torque commands transmitted to the robot and gripper controller. The latter is accessed using CORTEX, our in-house Component-Oriented Real-Time EXecution engine. CORTEX encapsulates the controller provided by the Franka, and provides a standard API for the skills, with new features and modes of operation useful for advanced skills implementation. Implemented control modes include low-level motion planners for trajectories, and hybrid force-position control for demanding assembly steps (e.g. assembly of the gears with less than 10 µm insertion tolerance).

Human-Machine Interface: the whole process is monitored by the orchestrator and a tablet is used to displays (visual or audio) warning messages in case of failure (e.g. object not found). This tablet is also used to relaunch the system once the operator's intervention is finished and the error mitigated.

Human gesture recognition: when an alert is displayed, the operator is asked to replace the missing part in the deposit tray or to perform the assembly step instead of the robot. A Panda-Net neural network [23] is used to detect the presence of the operator in the environment monitoring webcam's images and estimate his or her 3D skeleton in world coordinates from the video signal. This information is further analyzed to determine the action carried out (e.g. picking up or adjusting a part) and inform the orchestrator accordingly.

As illustrated in Fig. 4, this set-up proved able to perform the Gear Assembly in full autonomy, the parts being assembled one after another, or with the help of an operator.

Fig. 4. Insertion of the first gear in autonomous mode (left) and recovery of a failure during the assembly of the second spacer (right).

V. Adaptation to Novel Applications

A. *Two arms assembly*

While the Robot Companion can successfully perform the Gear Assembly, it requires several minutes, which is very long compared to a human being. To accelerate the process, we integrated a second cobot and updated the orchestrator so that both robots can work in parallel. Therefore, the orchestrator makes use of a second skills parameter, along with the identifier of the part of interest, allowing to assign a specific robot to each step of the assembly. In practice, we chose to use a Universal Robot UR10e as second robot. Even if its controller is less open than the Franka's one (thus limiting the possibilities to implement and optimize the robot's assembly skills based on hybrid force position control schemes), its performances are higher and it is more representative of the devices used in real factories (Universal Robot is the main provider of collaborative robots in factories to date). Also, it allows challenging the ability of our architecture to deal with different robots (since then, we also worked with Stäubli robots). This set-up is completed with an electric screwdriver (a dismantled handheld device integrated in a custom machined housing) mounted at the end-tip of the UR10e along with a RobotiQ 2F85 gripper (see Fig. 5, left).

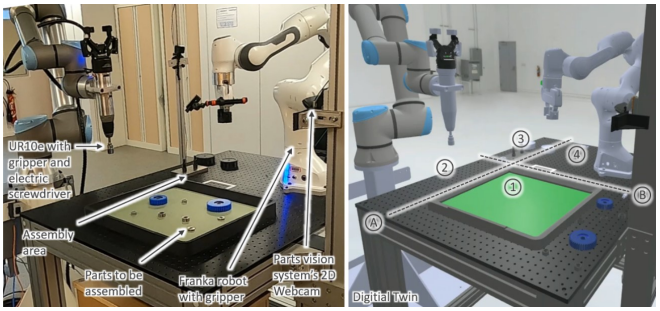

Fig. 5. The Robot Companion two-arms set-up and its Digital Twin.

The digital twin was updated accordingly, with a very low effort. The second robot's model was simply integrated in Solidworks and the whole scene was converted in Unity. It is worth noting that, each of the performed operations being feasible by a single robot in this case, the basic principle used to parallelize the process consists in having one robot performing an assembly while the other one grasps the next part. This way, we can ensure that the robots always work in different areas, drastically facilitating the management of potential collisions between them. In practice, we define four quadrants in the space above the table (see Fig. 5, right) and crossing poses (taught by demonstration) guarantying that robots always remain in different quadrants. During the

whole assembly process, anti-collision vertical virtual walls passing through the lines A and B that separate the different quadrants are activated in turn to avoid collisions when the Franka picks a part while the UR10e assembles another part in parallel and vice-versa. This simple solution allows the separate computation of collision-free paths for each robot.

Even if a full parallelization is not possible as some tasks remain currently feasible by only one robot (e.g. final screwing can only be made by the UR10e which is equipped with a screwdriver), the different steps are distributed as evenly as possible between the two robots. The principle of operation is the following: at start-up, the Franka and UR10e reach their respective crossing poses in quadrants 4 and 2. Then the UR10e begins to work by picking the first spacer in quadrant 1 and comes back to its crossing pose in quadrant 2. The next step concerns the assembly of the spacer, performed by the UR10e in quadrant 3 before it comes back to quadrant 2, while at the same time the Franka begins to work in parallel by picking the first gear in quadrant 1 before coming back to quadrant 4. Subsequently, the Franka assembles the first gear while the UR10e picks the first washer. Then the UR10e assembles the first washer while the Franka picks the first nut which is first put in place, waiting for the UR10e to perform the final screwing with the screwdriver, after a 180° rotation of its end-effector, performed in parallel with the nut pre-insertion. The second column (i.e. second spacer, gear, washer and nut) is assembled the same way.

It is worth noting that this parallelization strategy is only possible because the tasks considered in this use-case are simple enough to be performed by a single robot. To deal with assembly processes that require several arms to work concurrently (in the same space and at the same time), a more detailed approach would be needed, and a dynamic collision-free path planning would be required, as proposed for example by Realtime Robotics (https://rtr.ai/).

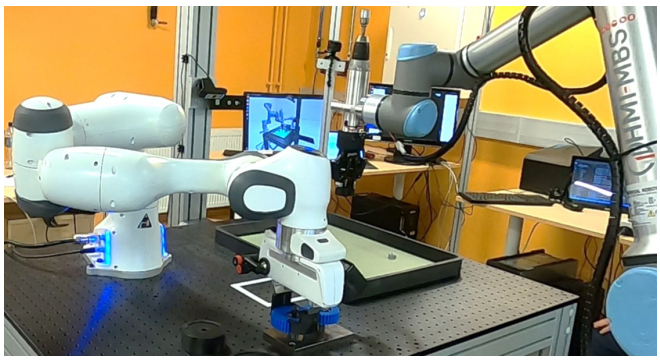

Fig. 6. The insertion of the second gear performed in parallel with the picking of the second washer (Digital Twin is visible in the background).

To accelerate the assembly, the vision system is triggered only once at the beginning of the assembly. Indeed, as the parts are, except the gears, small, shiny, and similar in size and shape, the 3D laser scanner is pushed to its limits in order to get as precise as possible points clouds. A negative consequence is that with our Class 2 laser scanner, which was selected to favor the operator's eye safety, each active scan can take up to five seconds (it would be much less with a Class 3R laser, at the price of safety concerns). In return, the information is of sufficient quality for the vision skills to recognize and localize all objects from the first scan most of the time, and it is not necessary to rescan the scene at each step. Still, in order to follow the task logic, the presence of the objects is not checked all at once at the beginning. Instead, the objects are managed according to the assembly progress, only one of them being considered at each step. A new scan is performed if and only if the current object is not detected in the original image, with the hypothesis that the recognition algorithm has less chance to fail once previous objects have been assembled as there are fewer objects left on the table. If the object is still not detected, then the system calls for human help, just as previously explained.

Regarding the robots, the Franka did not require any adaptation and works with the previously presented skills. The limited access to the low-level controller of the UR10e did however not allow to implement the same force and hybrid force-position control strategies, thus requiring slight modifications and tuning of the grasping and assembly skills. The main consequence is that the UR10e was not able to handle all parts, especially gears which are the most demanding ones.

As a whole, the vision optimization and partial parallelization allow a substantially faster work. The total assembly time is reduced from about 11 minutes for the one arm configuration (the nuts being not fully screwed) to only a little bit less than 6 minutes when the two arms are used (the nuts being fully screwed).

B. Robothon® Grand Challenge

Another use-case that was performed with the Robot Companion platform is the Robothon® Grand Challenge, an international competition and benchmarking event for measuring state-of-the-art performance in robot manipulation on tasks representative of electronics waste management (https://robothon-grand-challenge.com/).

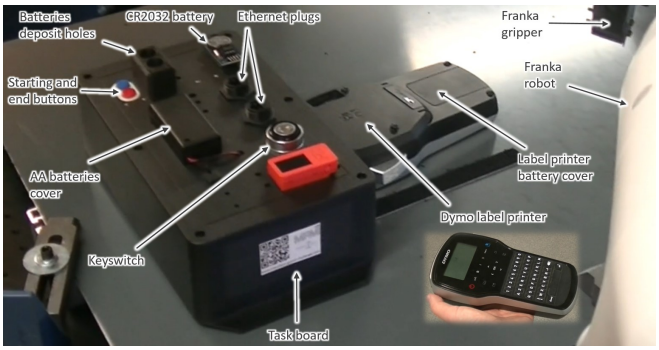

Fig. 7. The Robothon® setup (with Franka robot visible on the right).

This challenge consists in performing a given sequence of manipulation tasks 5 times in 10 minutes with one intelligent autonomous robot. This sequence is as follows (see Fig. 7):

- localize a "task board" (a plastic box integrating all other tasks) randomly placed on a table (the accomplishment of this task is validated by the ability to press a small button on the board),
- pick up a key on the task board and activate a keyswitch,
- move an Ethernet plug from one port to another,
- open a battery box, remove 2 AA batteries and place them in dedicated holes,
- and finally remove a CR2032 battery from its receptacle.

In addition, the participants had to show that their picking solution can be transferred to another similar task by proving that they are also able to extract the batteries of another electric device of their choice (our team chose a Dymo label printer). Finally, they had to demonstrate the robustness of their solution by proving that they can perform the sequence in a different order known at the last moment.

The functionalities required to perform these tasks being similar to the ones previously presented (i.e. recognition, localization, grasping and manipulation of objects), we largely reused the skills available on the Robot Companion. However, contrary to the Gear Assembly, where each object comes in a random position on the table, the key elements' positions relative to the task board are perfectly defined here. The only unknown is the configuration of the task board itself, making it the only object that has to be localized. Hence no recognition is required, and the localization module is called only once at the beginning of each repetition, the configuration of the different elements of interest being deduced from that of the task board using pre-identified transformation matrices. A similar simplification occurs for the orchestrator. The different tasks being defined in advance for all competitors and not subject to change, a dynamic tool like Behaviour Trees is not required. A simple ordered list of objects is used instead, with the possibility to change their order to adapt to the jury's demands. On the contrary, the movements required to extract, grasp, move and insert both thin elements (key, CR2032 battery) and longer ones (AA and Dymo batteries), some even with flexible parts (disconnection-preventing part of the Ethernet plugs), are more varied than for the Gear Assembly. Small metallic "nails" (that can be seen in Fig. 8 below) are integrated in the gripper's jaws to facilitate the removal of the batteries (these nails are also used to push on the buttons), and novel insertion skills (key, RJ45 plug, batteries) are tuned by trial and error.

These developments allowed us to perform almost all tasks 4 times in 10 minutes (see Fig. 8 for examples of successful steps), to easily adapt the order of the different steps, and to remove the Dymo batteries. Our team (OSCAR) ranked 6th for its first participation to this competition (the submitted video is accessible online at https://www.youtube.com/watch?v=sna7vPwEOZk and further details on the robot and results can be found at https://automatica-munich.com/en/munich-i/robothon/teams/, see in particular 'Team submissions 2022, Documentation' and 'Competition results 2022').

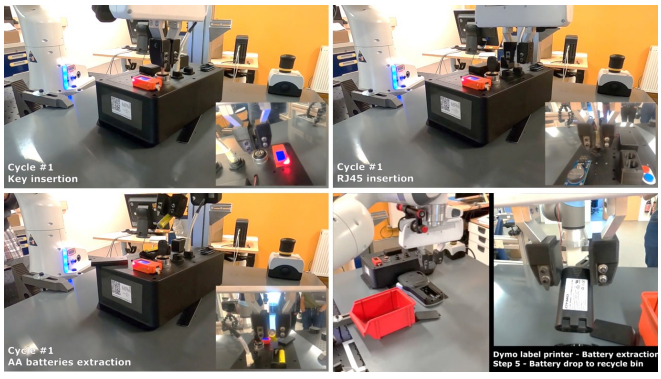

Fig. 8. Example successful steps of the Robothon Grand Challenge.

C. *Manipulation of plastic pouches*

The last application considered here is related to the European project Merging. This project aims to provide manufacturers with a versatile, easy-to-use and low-cost solution to automate the handling of flexible and fragile objects (https://www.merging-project.eu/). One of the study-cases, described below, consists in grasping plastic pouches, initially thrown on a table, and inserting them in a plastic rail, which will subsequently be used to feed an industrial pouches filling machine. This task is usually performed manually, and its realization by robots requires dedicated tools.

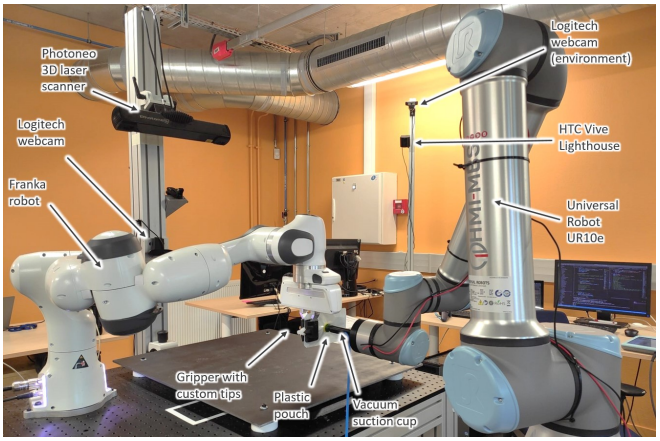

Fig. 9. The Merging plastic pouches setup (plastic rails are not visible).

As shown in Fig. 9, we largely reused the Robot Companion set-up, with important changes however. First, the UR10e is equipped with a pneumatic vacuum gripper and the Franka's gripper with custom tips allowing a stable grasping of the pouches (their design was optimized by trial and error). A 3D printed rail mock-up (not visible in Fig. 9) is also used. Besides, the 2D webcams are not used here. The plastic pouches are localized using only the Photoneo laser scanner and 3D registration algorithms. After they have been grasped by the UR10e's vacuum gripper, they are transferred to the Franka which is in charge of their insertion in the rail. The transfer configuration and entry point in the rail are taught by demonstration using an HTC Vive Lighthouse camera used to capture the position of a hand-held HTC Vive Remote Control equipped with a pointed tip used to designate points of interest in space. As the environment is free of obstacles, the Digital Twin is not used. Few passing points are defined instead for both robots. For the insertion skill itself, we implemented 6D Lissajous oscillators, whose hyperparameters were optimized prior to the realization of the task using fast converging (less than 10 minutes) CMA-ES and TPE algorithms [24] [25]. Once trained, the system proves able to realize the task (see Fig. 10).

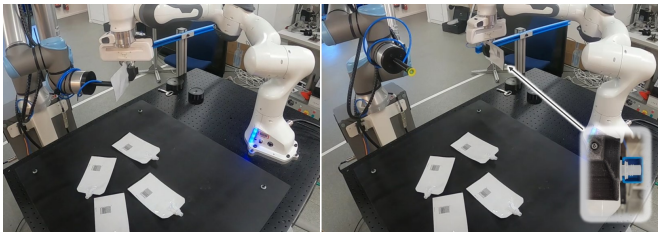

Fig. 10. Transfer of a pouch between the UR10e and the Franka (left) and sucessful insertion into the rail (right, with detail on the pouch nozzle).

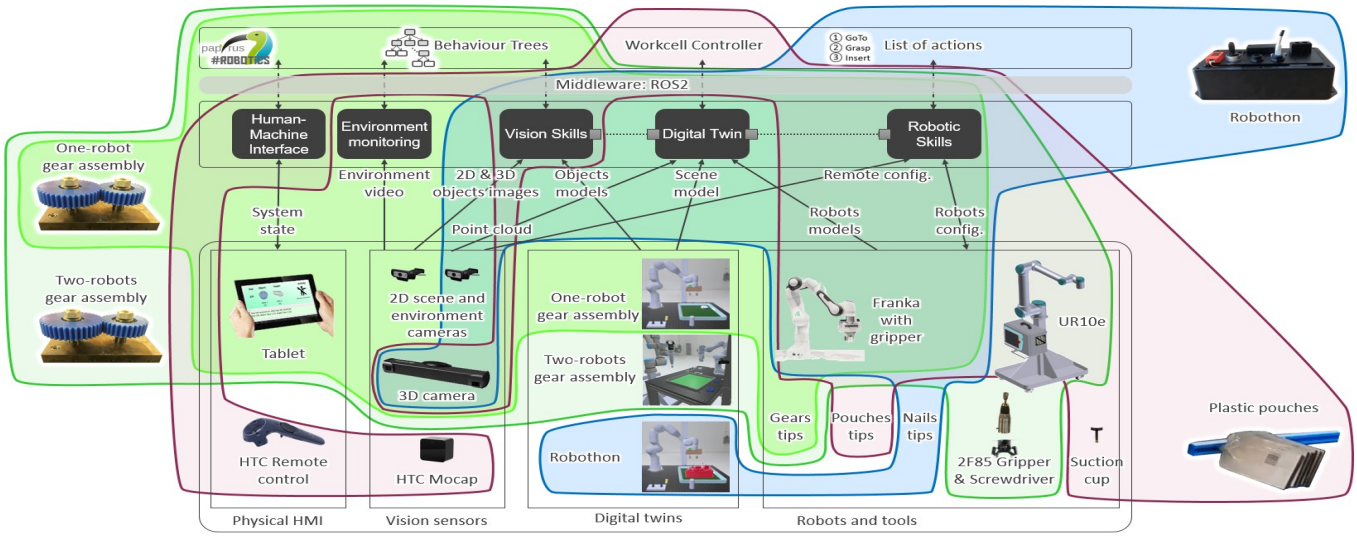

Fig. 11. Illustration of the different set-ups.

VI. Discussion

In this article, we introduce a four-layer modular architecture that associates a components-agnostic orchestrator coordinating software modules accessed through a common middleware, and different hardware running the required functions. Thanks to the respect of the principles of separation of concerns and separation of roles, a standard in software engineering, the hardware and software components can be easily adapted to the use-case requirements. The use of ROS 2, the *de facto* standard in advanced robotics, allows easily addressing the most-common robots and vision systems, especially as our orchestrator is designed using *Papyrus for Robotics*, a low-code environment that supports code generation for ROS 2.

As shown in previous sections and as illustrated in Fig. 11, the application-dependent adaptation occurs at almost all levels. Regarding the tasks orchestration, Behaviour Trees are used to address complex assembly tasks with our Robot Companion platform while simple lists of actions are sufficient to cope with the Robothon® disassembly logic, and the Workcell Controller developed by one of the project's partners is used to address the manipulation of plastic pouches within the context of the Merging Project [26]. The software components and skills also can be adapted. As an example, the list of skills used for the gear assembly with two arms is {**Locate**, GoTo_crossingPose, Goto_safePose, **Grasp**, **Insert**, PrepareScrew and Screw}, while for the pouches insertion the list of skills is {**Locate**, **Grasp**, Transport, Transfer (from UR10e to Franka), **Insert**, Detect (a failure) and Retry (in case of failure)}. And of course, the hardware can also be adapted, with different configurations of the vision system and robots depending on the type of task (autonomous or collaborative) and use-case requirements.

The ability to control and synchronize several robots and use task-dependent tools is particularly interesting in assembly tasks. As mentioned in section V-A, the assembly time was reduced from about 11 minutes to less than 6 minutes when switching from a one-arm configuration to two-arms. It is worth mentioning that, while much better, it is still much longer than a human operator. To further accelerate the process, a more detailed decomposition of the assembly process would be necessary, allowing a finer parallelization of the different steps and substeps, ideally allowing to perform vision skills and path planning in fully hidden time (at the price of a more complex orchestration). The speed of the robots, currently limited as the system is still in development phase, could also be improved, up to 0.25m/s in a cobotic mode compatible with the presence of a human being in the surroundings, and even up to about 2m/s when no one is present, provided the use of security sensors and situation awareness functions.

The latter remark takes all its sense when considering (dis)assembly challenges. As mentioned in section V-B, the architecture presented in this article was used to address the Robothon® Grand Challenge. While initially developed in another context, our solution proved able to efficiently address the associated disassembly tasks. Thanks to the ability to easily adapt the components used (this challenge did not require the full functionality of the system, and the orchestrator, digital twin and advanced visual perception were not used) and to integrate new skills in the framework, the global architecture remained unchanged and the adaptation effort was very reasonable. The performance of the 1 kHz hybrid force-position control loop implemented on the Franka robot was particularly useful for performing precise insertion tasks and allowed our team to favorably compare with other competitors. For its first participation, our team ranked 6th among the 20 candidates selected to participate, with a score of 181 on a maximum of 200. Only six teams, from which ours, were able to successfully address the six tasks of the challenge and to realize similar tasks on their own device. Our architecture was particularly efficient to perform the different tasks in a last-minute defined order. The main limitation came from the robots'

speed, which allowed us to perform the complete sequence only four times in ten minutes during the evaluations, with a fastest time of 118s. The final results showed that comparable or superior performances can be obtained with collaborative or industrial robots using classical position-based control, with fastest times between 107 and as low as 31s in the five first teams who were able to perform the whole disassembly five times in ten minutes.

VII. Conclusion

This article introduces a four-layer modular architecture composed of a components-agnostic orchestrator in charge of representing and formalizing the tasks to be executed in the form of elementary functions which are called through a standard middleware and executed on ad-hoc hardware. It was first developed for the assembly of a representative gear unit with one arm. It was then adapted with a limited effort to assemble it with two robots, to address the Robothon® Grand Challenge, and to insert plastic pouches in rails, proving its ability to easily adapt to different applications. Moreover, performances that favorably compare with the state-of-the-art were obtained in the open-competition context of the Robothon®, proving that our solution is both agile and performant, making it a valuable solution for adaptive intelligent robotics.

Acknowledgment

The authors would like to thank the following colleagues at CEA LIST who contributed to the projects presented in this article: M. Anastassova, N. Bruckmann, C. Andriot, R. Besançon, M. Boc, M. Boukallel, T. Boulmier, M. Darouich, G. De Chalendar, M. Grossard, H. Le Borgne, B. Perochon, A. Radermacher.

References

[1] SWMAS, 'Manufacturing barometer - Robot adoption: The SME challenge', *National report autumn 2019–20*, 2019, 24 pages.

[2] M. Javaid, A. Haleem, R. P. Singh, S. Rab, R. Suman, 'Significant applications of Cobots in the field of manufacturing', *Cognitive Robotics*, 2, 2022, pp. 222-233.

[3] A. Billard, S. Calinon, R. Dillmann, S. Schaal, 'Robot Programming by Demonstration', *Handbook of robotics*, 2008.

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

[5] H. Nguyen, H. M. La, 'Review of Deep Reinforcement Learning for Robot Manipulation', *Proc. IEEE Int. Conf. on Robotic Computing*, 2019, pp. 590-595.

[6] P. Zimmermann, E. Axmann, B. Brandenbourger, K. Dorofeev, A. Mankowski, P. Zanini, 'Skill-based Engineering and Control on Field-Device-Level with OPC UA', *Proc. IEEE Int. Conf. on Emerging Technologies and Factory Automation*, 2019, pp. 1101-1108.

[7] S. Profanter, A. Perzylo, M. Rickert, A. Knoll, 'A Generic Plug & Produce System Composed of Semantic OPC UA Skills', *IEEE Open Journal of the Industrial Electronics Society*, 2, 2021, pp. 128-141.

[8] S. Cavalieri, G. Cutuli, 'Performance Evaluation of OPC UA', *Proc. IEEE Conf. on Emerging Technologies and Factory Automation*, 2010, 8p.

[9] T. Asfour, M. Wächter, L. Kaul, S. Rader, P. Weiner, S. Ottenhaus, R. Grimm, Y. Zhou, M. Grotz, F. Paus, 'ARMAR-6 - A High-Performance Humanoid for Human–Robot Collaboration in Real-World Scenarios', *IEEE Robotics and Automation Mag.*, 26(4), 2019, pp. 108-121.

[10] A. Kheddar, S. Caron, P. Gergondet, A. Comport, A. Tanguy, C. Ott, B. Henze, G. Mesesan, J. Englsberger, M.A. Roa, P.B. Wieber, F. Chaumette, F. Spindler, G. Oriolo, L. Lanari, A. Escande, K. Chappellet, F. Kanehiro, P. Rabaté, 'Robots in Arcraft Manufacturing, The Airbus Use Cases', *IEEE Robotics and Automation Mag.*, 26(4), 2019, pp. 30-45.

[11] H. Engemann, S. Du, S. Kallweit, P. Cönen, H. Dawar. 'OMNIVIL—An Autonomous Mobile Manipulator for Flexible Production', *Sensors*, 20(24), 2020.

[12] M. Hvilshøj, S. Bøgh, 'Little Helper - An Autonomous Industrial Mobile Manipulator Concept', *Int. Journal of Advanced Robotic Systems*, 8, 2011, pp. 80–90.

[13] P.J. Koch, M.K. van Amstel, P. Dębska, M. A. Thormann, A. J. Tetzlaff, S. Bøgh, D. Chrysostomou, 'A Skill-based Robot Co-worker for Industrial Maintenance Tasks', *Procedia Manufacturing*, 11, 2017, pp. 83-90.

[14] A. Cherubini, R. Passama, B. Navarro, M. Sorour, A. Khelloufi, O. Mazhar, S. Tarbouriech, J. Zhu, O. Tempier, A. Crosnier, P. Fraisse, S. Ramdani, 'A collaborative robot for the factory of the future: BAZAR', *The International Journal of Advanced Manufacturing Technology*, 105(9), 2019, pp. 3643-3659.

[15] F. Rovida, M. Crosby, D. Holz, A.S. Polydoros, B. Großmann, R.P.A. Petrick, V. Krüger. 'SkiROS—A Skill-Based Robot Control Platform on Top of ROS', in *Robot Operating System (ROS): The Complete Reference*, Springer, 2017, Vol. 2, pp. 121–160.

[16] A. Munawar, G. De Magistris, T.H. Pham, D. Kimura, M. Tatsubori, T. Moriyama, R. Tachibana, G. Booch, 'MaestROB: A Robotics Framework for Integrated Orchestration of Low-Level Control and High-Level Reasoning', *Proc. IEEE Int. Conf. on Robotics and Automation*, 2018, pp. 527-534.

[17] C. Paxton, A. Hundt, F. Jonathan, K. Guerin, G.D. Hager, 'CoSTAR: Instructing collaborative robots with behavior trees and vision', *Proc. IEEE Int. Conf. on Robotics and Automation,* 2017, pp. 564–571.

[18] Y. Yokokohji, Y. Kawai, M. Shibata, Y. Aiyama, S. Kotosaka, W. Uemura, A. Noda, H. Dobashi, T. Sakaguchi, K. Yokoi, 'Assembly Challenge: a robot competition of the Industrial Robotics Category, World Robot Summit – summary of the pre-competition in 2018', *Advanced Robotics*, 33(17), 2019, pp. 876-899.

[19] F. Gosselin, S. Kchir, G. Acher, F. Keith, O. Lebec, C. Louison, B. Luvison, F. Mayran de Chamisso, B. Meden, M. Morelli, B. Perochon, J. Rabarisoa, C. Vienne, G. Ameyugo, 'Robot Companion, an intelligent interactive robot coworker for the Industry 5.0', *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2022, pp. 8918-8925.

[20] M. Tan, R. Pang, Q.V. Le, 'EfficientDet: Scalable and Efficient Object Detection', *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2020, pp. 10778-10787.

[21] Y. Guo, M. Bennamoun, F. Sohel, J. Wan, M. Lu, '3D free form object recognition using rotational projection statistics', *Proc. IEEE Workshop on Applications of Computer Vision*, 2013, pp. 1-8.

[22] X. Merlhiot, J. Le Garrec, G. Saupin, C. Andriot, 'The XDE mechanical kernel: Efficient and robust simulation of multibody dynamics with intermittent nonsmooth contacts', *Proc. Joint Int. Conf. on Multibody System Dynamics*, 2012.

[23] A. Benzine, F. Chabot, B. Luvison, Q.C. Pham, C. Achard; 'PandaNet: Anchor-Based Single-Shot Multi-Person 3D Pose Estimation', *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2020, pp. 6856-6865.

[24] N. Hansen, 'The CMA Evolution Strategy: A Tutorial', *arXiv*, https://arxiv.org/abs/1604.00772, 2016.

[25] Y. Ozaki, Y. Tanigaki, S. Watanabe, M. Onishi, 'Multiobjective Tree-structured Parzen Estimator for Computationally Expensive Optimization Problems', *Proc. ACM Genetic and Evolutionary Computation Conf.,* 2020, pp. 533-241.

[26] D. Andronas, G. Kokotinis, S. Makris, 'On modelling and handling of flexible materials: A review on Digital Twins and planning systems', *Proc. CIRP Conf. of Assembly Technology and Systems*, 2021, pp. 447-452.