Chapter 2.1
PIROS: Cooperative, safe and reconfigurable robotic companion for CNC pallets load/unload stations

Federico Vicentini, Nicola Pedrocchi, Manuel Beschi, Matteo Giussani, Niccol Iannacci, Paolo Magnoni, Stefania Pellegrinelli, Loris Roveda, Enrico Villagrossi, Mehrnoosh Askarpour, Inaki Maurtua, Alberto Tellaeche, Francesco Becchi, Giovanni Stellin, Giuseppe Fogliazza

1 Introduction

Handling and assembling applications with small batch size and high production mix requires requires high adaptability, reconfigurability and flexibility. Thus, human-robot collaboration could be an effective solution to ensure production performance and operator satisfaction. This scenario requires human-awareness in different level of the software framework, from the robot control to the task planning. The goal is to assign high added value activities to the human as much as possible, while the robot has to be able to substitute the human when needed. Team piROS faces this goal by designing a IEC 61499/ROS-based architecture which integrate safety assessment, advanced force control, human-aware motion planning, gesture recognition, and task scheduling.

2 Use case description and motivation

Team piROS activities deal with the use of collaborative robotics in handling and assembly applications, in the domain of advanced manufacturing and specifically in the use of Flexible Manufacturing Systems (FMS) for machining of metal parts. FMS is a highly-automated machining systems, in which production is dynamically optimized and balanced depending on changing conditions (production mix, batch sizes, availability of machines, etc.). Nonetheless, manual labour is still a relevant component in the preparation/verification steps of the machining processes. FMS is a relevant scenarios for hybrid human-robot production systems and collaborative

Fig. 30 Production scenario with heavy-duty manual tasks at LUSs (Cortesy of: Cembre SpA, Brescia, Italy).

robotics in manufacturing, because of the need of both collaborative assembly tasks and workspace sharing conditions along variable tasks.

2.1 Industrial Practice

In conventional FMS, operators are mostly dedicated to interfacing with CNC consoles and to completely handling all material flows. Raw materials are assembled into CNC pallets with fixtures, then machined/finished parts are removed, checked and re-assembled for further machining or prepared for shipping. All these operations happen in dedicated parts of the FMS layout, each one called Load/Unload

Fig. 31 A LUS upgraded with a robot system for a collaborative version of pallet preparation tasks: the robot system provides services from lateral approach (from left side), sharing the workspace with a single LUS operator (Cortesy of: Cembre SpA, Brescia, Italy).

Station (LUS, see Figure 30). In highly automated FMS, pallets are stored inside a (remarkable large) buffer system, and shuttled to eventual machining destinations according to optimized schedules.

A typical situation is shown in Figure 30, while Figure 31 shows the adoption of the piROS collaborative robotics solution in support of an existing, fully manual LUS.

Examples of heavy workload tasks that benefit from the use of collaborative robots in a LUS include: setting fixtures, mounting/dismounting workpieces into fixtures before/after machining, inspecting the premachining setups, inspecting the quality of machined parts. In these cases, robots can provide a number of assistive tasks such as kitting parts and tools, handling parts, supporting manual assembly, moving sensors for inspections, etc., according to *nominal* task plans or inline *alternatives* requested by operators.

While collaborative applications and shared spaces are popular production approaches [61], their efficacy and efficiency are critically dependent on the performance of individual sub-task and on the design of such sub-task allocation between humans and robots. Specifically, in a scenario of remarkable variability of parts and processes, the design and allocation of roles and actions between humans and robots may be the key for determining the target efficiency when unfeasible/inefficient tasks are identified and allocated in an optimal way.

In the proposed scenario, collaborative assembly is investigated and evaluated in terms of two principal factors: (i) the ability in performing the sub-tasks, (ii) the efficient sub-tasks allocation to the human and the robots.

2.2 Objectives

For sake of clarity, preliminary definitions are reported below:

Fig. 32 Example of FMS pallet, with bolts/nuts jigs, inside a LUS during an intermediate machining step: array of parts on left side need to be disassembled and flipped to the right side array jigs for second machining step. Additional different jigs are present on lateral mounting surfaces of the pallet (Courtesy of: Cembre SpA, Brescia, Italy).

- **variability** in manufacturing is intended here as requested changes in mixed parts, process/assembly sequences, tools used, target timing, changes in design features or materials, during a production plan;
- **flexibility** in manufacturing ([30, 105]) is the ability to maintain an efficient production goal, or at least similar operational capabilities, in presence of variability without significant modifications of setup or programs (see a review of the evolution of literature about flexibility in [22] and citations therein);
- **reconfigurability** in manufacturing is the ability to attain the same previous objectives but, unlike flexibility [39], undergoing some system changes when production/process exhibits some variability. Attitude for changes is not built-in, needs down-time to reset but can be more cost-effective depending on the size and quality of variations;
- **adaptability** in robotics is intended here as the possibility for a robot system (hardware and software) to be changed by users. As such, the robot system has to provide some built-in flexibility in terms of task allocation, pattern selection, tooling, etc. **Adaptivity**, instead, indicates the ability to automatically react to changing conditions and fit to new behaviors/targets.

In piROS use case experience, the most important property for the proposed collaborative robotics solution is *adaptability* of the platform to production and usage needs. In particular, in FMS manual tasks the following conditions hold.

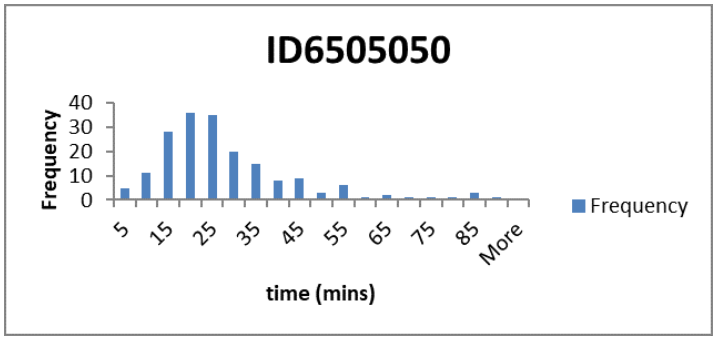


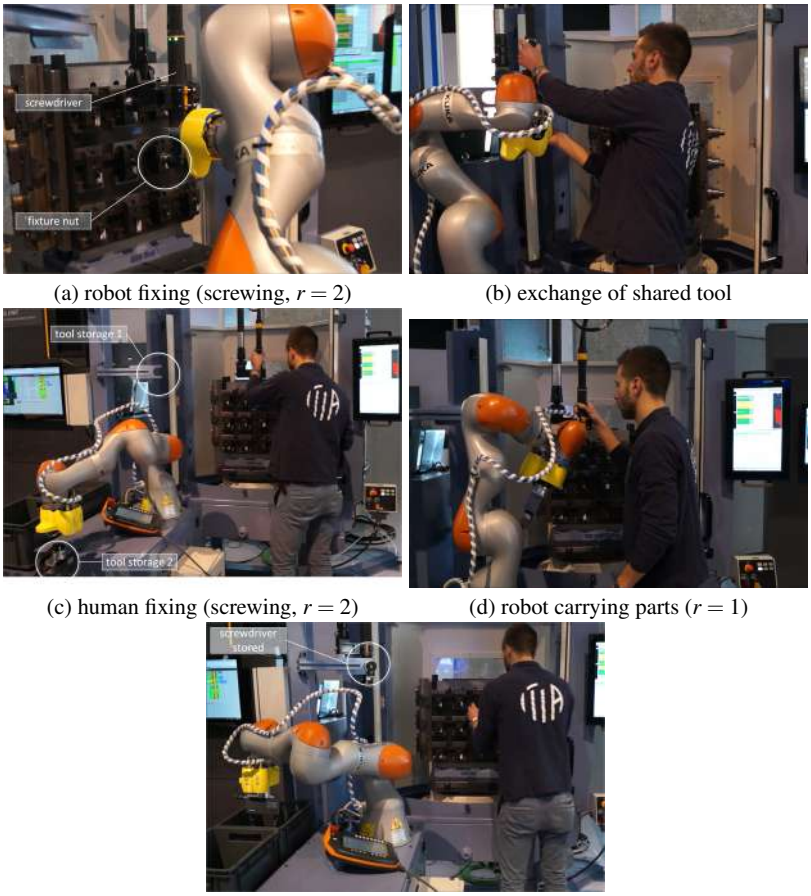
Fig. 33 Example of usage profile of LUS usage profile.

- Process flexibility: the FMS enables its hosting shopfloor to afford a variable production mix in the magnitude of 10^3 parts/year, with batch sizes ranging in $10^2 - 10^5$ parts. Machining pallets have unique fixtures to be prepared, manipulated, checked, cleaned, etc. approximately every 20 minutes (see an average LUS usage profile in Figure 33). Due to such production mix profile, the portion of plant integrating the piROS demonstrator is suitable for reconfigurable solutions. Therefore, additional modules need to cope with the different expected tasks to be performed.
- Process variability determined by human operators: their actions are little repeatable in terms of timing and pace. While assembly/disassembly tasks for CNC pallets at LUSs feature standardized operations, we observed operators doing many intermediate variants or alternate usage of time slots, frequently moving in and out of the workstation (*e.g.*go for materials, go for tools, consult machining CADs, measure, make quality control before resuming, etc). Due to such usage profile, a collaborative robotic solution needs to a remarkably large state space of possible task execution modes/sequences, in terms of both subprograms scheduling and safety conditions.

2.3 Proposed architecture

The proposed approach to reach the required value of adaptability is to design modular solutions able to perform the sub-tasks. The task planner deals with the suitable scheduling of the sub-tasks. Operators can plan and/or compose these actions at runtime, considering user preferences and execution time in case of allocation of actions to robots. The major enablers of the proposed approach is highlighted in Figure 35.

The actions have to be realized in a modular fashion, allowing easy assignment swaps between human and robot. Moreover, the programming time should be compressed as much as possible, indeed the modules could be rescheduled on-the-fly



(a) robot fixing (screwing, $r = 2$)

(b) exchange of shared tool

(c) human fixing (screwing, $r = 2$)

(d) robot carrying parts ($r = 1$)

(e) robot bin picking ($r = 1$), human inspecting ($r = 3$)

Fig. 34 Example of mixed workflow during an assembly/fixing sub-task (EuRoC field test stage).

depending on the current layout of the cell. The task planning description is in Section 3.3.

Many actions can be executed by the robot or the operator, therefore the robot manipulation abilities must ensure performance at least of the order of magnitude of the operator in terms of force overshooting and execution time. The proposed approach is a impedance control strategy with reference shaping coupled with environment stiffness estimation. The knowledge of the environment allows the shaping algorithm to avoid force overshoot and to limit the needed execution time. Manipulation is describer in Section 3.1.

In human-robot collaboration scenarios, the usability and the operator satisfaction strongly depend on the capability of the robotic system of reading and reacting the human actions. In the proposed scenario, the perception is focused on the gesture recognition in a true industrial scenario, characterized by occlusions and noise.

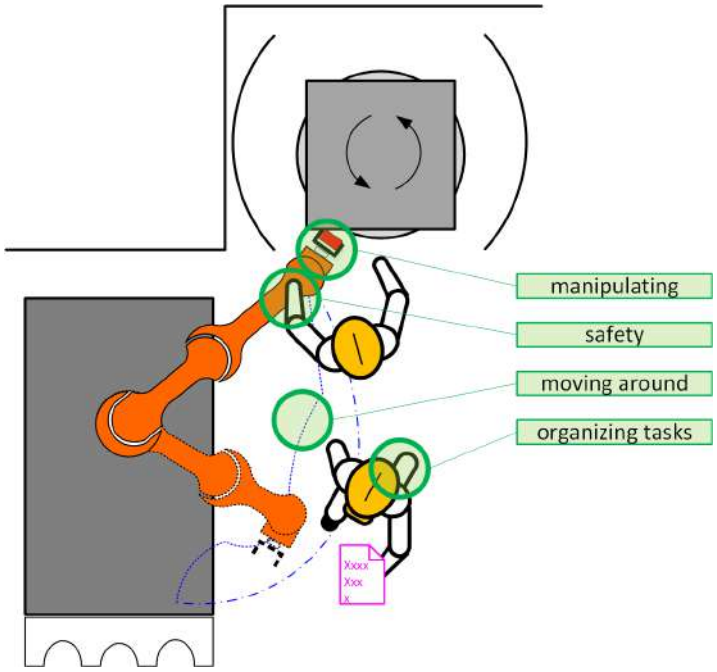


Fig. 35 Major enablers in piROS experience: *manipulating* includes all interaction control functions for assembly tasks and robot actions on materials; *safety* includes risk assessment and reduction for hazards typically featured in human-collaboration (*e.g.*accidental contacts); *moving around* includes all motion planning functions for optimizing the movements of the robot system inside the shared workspace, in terms of both safety and productivity; *task planning* includes the software for assigning roles and activities to both human and robot agents.

The goal is to understand what piece and the related activities the human wants to delegate to the robot. Section 3.4 deals with the gesture recognition activities.

The perception of the human position allows the robot motion planning to be aware of the human activities. Indeed, the robot trajectories influence and are influenced the human activities. In the proposed approach, the human position during a task is modelled as a probabilistic 3d scalar field. The field associates a occupancy probability to every point in the workspace. The motion planner considers as obstacles the volumes where the probability is higher than a threshold. In the other regions, the algorithm computed the expected execution time and the variability due to the human presence. This allows the scheduler to have a better knowledge of needed timing and to act accordingly. Section 3.2 deals with the human-aware motion planning.

The continuous reconfiguration of the cell increases substantially the efforts to realized a risk assessment. In fact, the associated risk assessment need then to consider a much higher number of assumptions at design time. Planning and scheduling generate task allocation hard to list entirely at design time. With the layout and hardware changeability, the number of risk variables is increased by partially-known objects

inside the shared workspace. Operators might undergo to errors or out-of-training behaviors, and may entail different mistaken workflows, such as circumventing instructions or attempting hazardous troubleshooting.

The proposed solution to overcome this challenge is a model-based Computer-Aided Risk Assessment. It supports the designer of safety solutions (together with the robot system manufacturer and other stakeholders) during the phases of hazard identification, risk estimation, and risk evaluation. The approach is dedicated to modeling the collaborative application with a formal model that can be checked and verified for a given objective property, which in case of HRC situations corresponds to the condition of negligibility of all risks associated with hazards potentially occurring during the task execution. Computer-Aided Risk Assessment is presented in Section 3.5.

2.4 Setup description

The robotic cell is composed by a LUS, the robot equipped with the required tools and sensors and the human-robot shared workspace. Robot, tools and sensors communicate with a Ubuntu 16.04 laptop with ROS-Kinetic framework, this computer is integrated with the FMS by means of IEC61499 standard as described in Section 3.3.

The robot system is assembled into a load/unload station located in Cembre SpA. The layout of a LUS features a frontal access zone to the processing area, which is about 900 mm wide and is enclosed within sliding doors that are opened only when a pallet is ready for manual operations at the station.

The robot system (see Figures 37 and 38) is composed of

- a lightweight robot KUKA iiwa R820, with 14 kg of payload;
- a Schunk WSG50 Co-Act gripper, custom-design pneumatic gripper for heavy load (based on Schunk PGN+ 160-2);
- a custom vacuum gripper;
- an electronic-controlled screwdriver. It enables the shared use of the tool by the robot and a human operator. The high required torque (up to 90 Nm) is balanced by a reaction carbon-fiber telescopic arm, directly attached to the tool. The tool is gravity-compensated and attached by the robot with a standard tool changer;
- Ensenso depth sensor deals with human tracking perception;
- A stereo-camera on-board system for object recognition in bin picking activities;
- Protective shells have been added around on-board cameras and the tool changer for collaborative usage purpose.

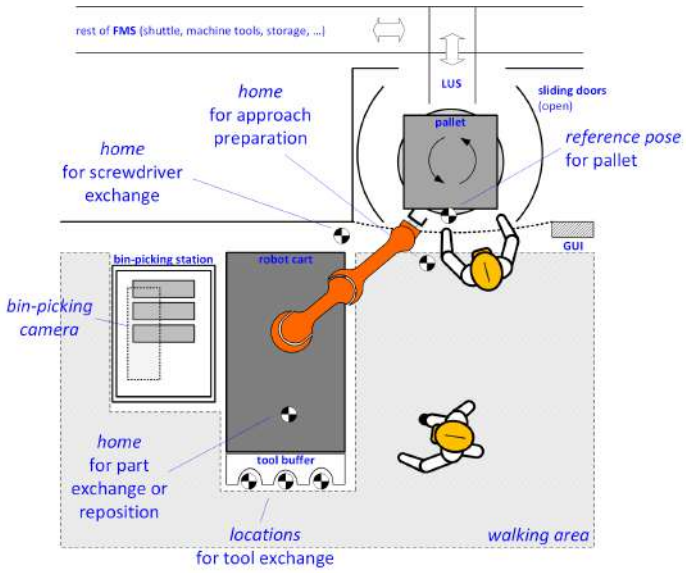


Fig. 36 Schematic top view of LUS layout, as depicted also in Figure 37. *walking area* is the zone where operators are expected to move as part of their intended use of the system. The workspace of human operators (*i.e.*including the outreach of arms and hands) extends beyond the walking area, *e.g.*to the pallet *reference pose* and fixtures, to the *home for screwdriver*, etc).

3 Activities and results

In this section the activities described in Section 2.3 will be described in details.

3.1 Manipulation

Within the application scenario, the requirements are mainly two:

- in terms of processing performances, the main quality metrics is associated with the limitation of force overshoots.
- Fast execution cannot be longer than manual insertion time (2-3 seconds) in order to maintain the scalability of the robotic solution alongside human tasks (see Figure 39)[1].

and their achievement has been addressed through the design of a proper adaptive impedance-based controller, that has been optimized in term of minimization of the force tracking error.

[1] As an example, some fixtures experienced during tests and prototypes, featured 24 slots for processed materials, half of which requiring full flipping of metal parts before insertion.

Fig. 37 Frontal view of a LUS including the piROS setup. The LUS is composed of two pallet units, of which only the left side one has been equipped with the robot system. Right hand side unit continues to operate manually (pictured with its sliding doors closed). The robot system is relocated on the left-end side of the left pallet unit (pictured with open doors and a pallet available in the processing area). The LUS, approx. 2300x3100 mm wide and high, is surrounded by the storage unit, where pallet are shelved and shuttled within the FMS. The portion of the FMS not visible in the picture include the full-length storage (approx. 30 m), another manual LUS, the shuttle delivery system travelling behind the LUSs, and 3 CNC machine tools (partially visible in the background, on the opposite side with respect the shuttle track). Notable elements of the system are labelled. (Courtesy of Cembre SpA, Brescia, Italy).

3.1.1 Methodology

The controller design is based on a two-layer architecture, with a first layer wrapping a standard compliant robot control. Second, a layer for converting a target force into an impedance set-point is designed for estimating and adapting the overall robot-environment impedance. Specifically, the main goal of a stable force tracking control problem is related to limiting or eliminating any force overshoot. The problem is therefore formulated taking into account the coupled dynamics (controlled robot - interacting environment) at the established contact conditions with the pallet fixtures.

A key point in the controller design has been the online estimation of the environment stiffness, needed in order to proper calculate the control gains to achieve the target interaction dynamics. With reference to the general control scheme in Figure 40, an Extended Kalman Filter (EKF) has been implemented in order to estimate

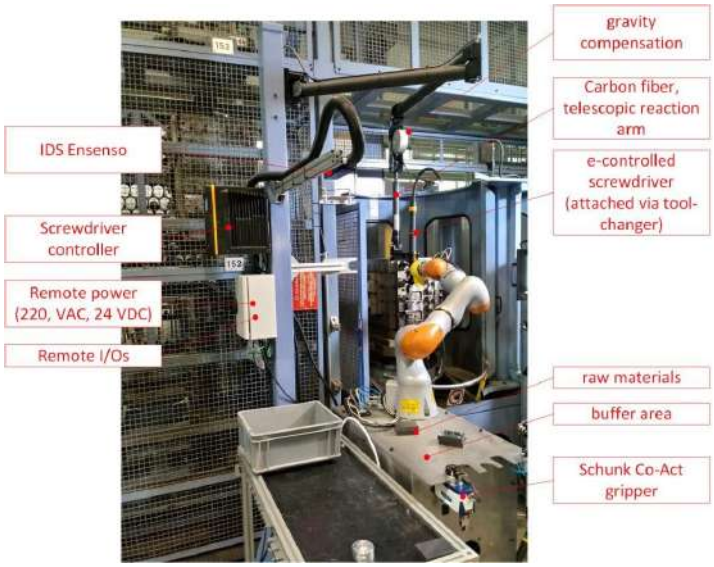


Fig. 38 Lateral view of the LUS depicted in Figure 37, showing and listing several tools included into the robot system. The screwdriver storage point (see also Figure 36) is visible at the center of the image, while the screwdriver is attached to the robot as the end effector. The buffer area on top of the manipulator cart is used for occasional re-positioning of parts or raw materials. (Courtesy of Cembre SpA, Brescia, Italy).

Fig. 39 Example of CNC pallet featuring twelve 2-slot jigs: each part is flipped between individual slots and inserted in pegs (pictured during development stages).

properly environment stiffness to the reference shaping algorithm (more details and simulation/experimental validation are in [91]).

The impedance control loop design has been therefore designed in order to guarantee the achieving of a pure decoupled second-order impedance behavior for the

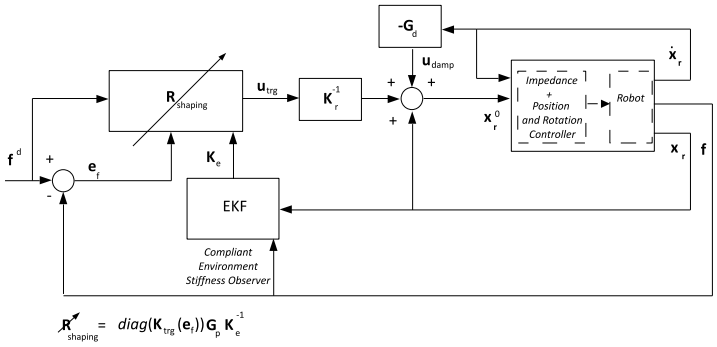


Fig. 40 Set-point deformation, including the environment observer (EKF).

Fig. 41 Example configuration of a fixture assembly task (H7/h6 tolerance).

controlled robot up to a reasonable frequency, of around $5\,Hz$[2]. Therefore, the target dynamics for the controlled robot should result in:

$$\mathbf{M}_r\ddot{\mathbf{x}}_r + \mathbf{D}_r\dot{\mathbf{x}}_r + \mathbf{K}_r\Delta\mathbf{x}_r = \mathbf{f}_r, \quad \text{with} \quad \Delta\mathbf{x}_r := \mathbf{x}_r - \mathbf{x}_r^0 \tag{1}$$

where $\mathbf{x}_r^0$ and $\mathbf{x}_r$ are the desired and actual robot positions respectively, and $\mathbf{f}_r$ is the external interacting force/torque (Figure 40). In addition, the Cartesian stiffness $\mathbf{K}_r$, damping $\mathbf{D}_r$ and mass $\mathbf{M}_r$ have to present negligible extra-diagonal coupling terms with a good approximation up to few Hz. In particular, $\mathbf{D}_r = 2\,\mathbf{h}\,\mathbf{M}_r\,\omega_0$, where $\mathbf{h}$ is diagonal matrix of the imposed damping ratio and $\omega_0 = \sqrt{\mathbf{M}_r^{-1}\mathbf{K}_r}$ is the system pulsation.

[2] The robot used, KUKA iiwa already displayed a decoupled behavior in the Cartesian space. However the approach can be easily extended also to other robots, by properly designing a control loop around the standard position controller for many lightweight industrial robots based on [97].

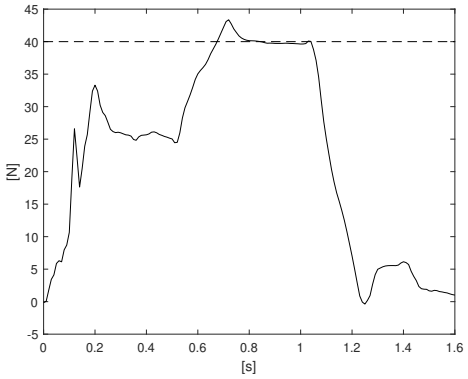


Fig. 42 Measured force vs. target force during the experiment with KUKA iiwa.

3.1.2 Results

Developed controllers were able to optimize a hard-contact assembly process with tight mounting tolerances (*e.g.*H7/h6, as in Figure 41) both in terms of force-tracking and execution time. The force overshoot f^{max} is computed at the highest peak magnitude w.r.t the target force f^d, the steady state force f^{ss} is averaged over 100 samples from the force overshoot f^{max}, the execution time t^{exe} is computed from the first contact until the opening of the gripper (end of the task) and compared with the expected target execution time t^d.

Table 3 shows obtained results for f^{max}, f^d and t^{exe} for $i = 1,\ldots,5$ iterations using the setup in Figure 41, while Figure 42 shows the measured force during the assembly task during one iteration.

Assembly Results Using the KUKA iiwa					
assembly	1	2	3	4	5
$f_i^d\,[N]$	40	40	40	40	40
$f_i^{max}\,[N]$	43.3632	43.4049	43.5111	43.6693	43.6454
$f_i^{ss}\,[N]$	39.7509	39.8195	39.7514	39.6848	39.7678
$t_i^d\,[s]$	15	15	15	15	15
$t_i^{exe}\,[s]$	1.13	1.13	1.12	1.13	1.12

Table 3 Target force f^d, force overshoot f^{max}, steady state force f^{ss}, execution time t^{exe} are shown for each i^{th} assembly. Experiments executed with the KUKA iiwa robot.

The experiment reports an average 9% overshoot and 1% error force and an average execution time of 1.13 s, corresponding to an improvement of 89% w.r.t. the target t^d. The control variable with KUKA iiwa is the position reference, making

the force tracking in impedance mode being very accurate. The first peak and settle phase in Figure 42 corresponds to the preengagement of the pegs, with the highest peak on reference being the actual forced insertion.

As an overall result, the avoidance of the force overshoots and a drastic reduction of the execution time of the assembly process proved the control strategy as an effective option for the collaborative pallet assembly scenario.

3.2 Motion planning

In collaborative applications, humans and robot behaviours are coupled, and robot trajectory influences ergonomics and the way in which the worker moves affects the robot trajectory [44, 53]. The primary reason for altering a robot trajectory is safety, *i.e.*it may be necessary to stop or to slow down in order to avoid collisions. However motion re-planning may also be involved in re-tasking robot actions on-the-fly. This coupling has a relevant impact on task planning and scheduling activities. in the state-of-the-art, however, the task planner/scheduler is not able to *reason* on the basis of uncertain robot execution times and changing human behaviours [109]. In fact, currently available approaches for task planning are limited in reasoning about robot trajectories at spatial level [98, 34].

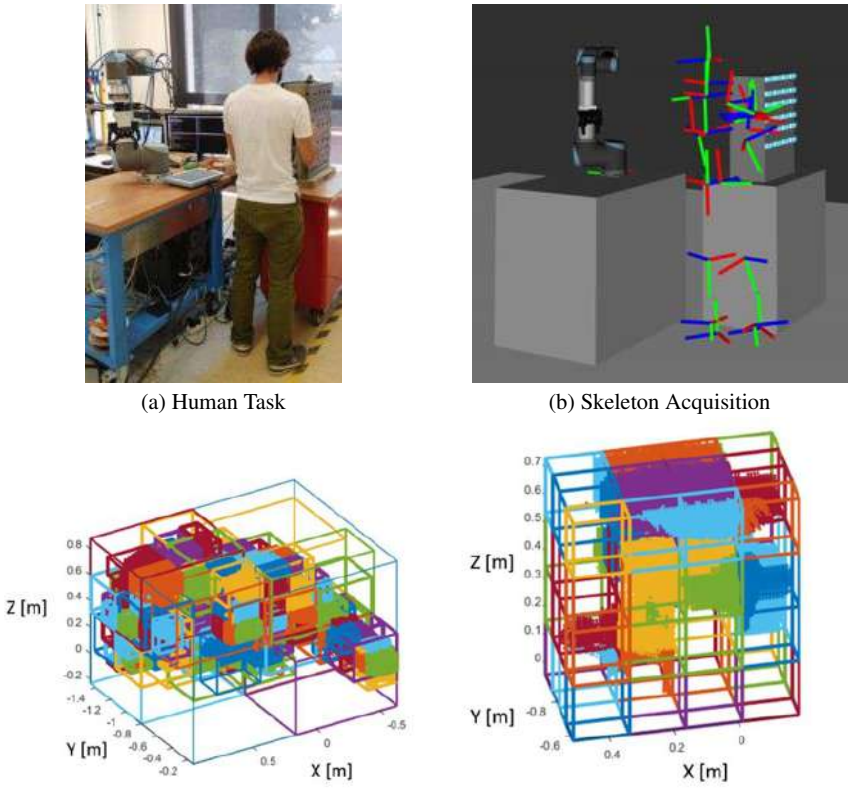
To overcome such limitation, within EuRoC, Team piROS investigated motion planning where the timing of HRC tasks has been previously estimated for enabling the optimal planning and scheduling of *e.g.*assembly procedures [82].

3.2.1 Methodology

Within piROS, the methodology consisted of a probabilistic time-description of the human occupancy close to the robot, and the estimation of how the overlapping of the robot trajectories with the probabilistic occupancy volume corresponding to the operator influences the trajectory time.

Specifically, a workspace segmentation has been performed considering the volume occupied by the human and the robot during their motion. Then, this segmentation is exploited for the definition of a set of Markov chains modelling human-robot interaction and allowing the estimation of the robot execution time. A complete overview of the methodology are in [83].

As show in Figure 43, first, the Occupancy Grid representing the collaborative workspace is defined with a step of few millimeters, typically 5mm. This resolution is sufficient to track properly the human movements as needed by the method, and it is coherent with the resolution of the Kinect One sensor. The two Kinect One sensors are placed so that occlusions are avoided, i.e., at least one Kinect One is always able to track human movements. The occupancy volumes are defined of the integral time when at least a segment of the human is in the grid cell divide by the acquisition time. The computational complexity for the problem is reduced by means of a



(a) Human Task

(b) Skeleton Acquisition

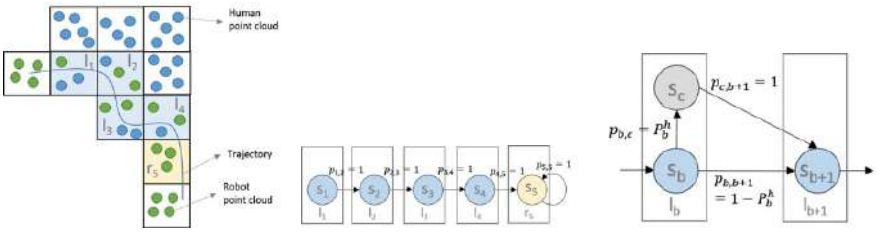
(c) Ful octree of the occupancy volumes given by tracking the human skleton

(d) Octree leaf bins

Fig. 43 The volume occupied by the human and the robot during the execution of their tasks is represented by two cloud points, than merged and organized into an octree. The leaf bins of the octree occupied by both the human and the robot will be used as nodes in a set of Markov chains.

Octree algorithm. Given the task represented as an occupancy volume, it is possible to model a multi-order Markov Chain that can be used to model the interaction of the human with the robot, while the robot intersects the human workspace. Finally, through such multi-order Markov Chain is therefore possible to estimate the probability of collision, and the best estimation for the mean time of the robot trajectory when it is repeated many times close to the human.

Within EuRoC, the approach has been tested under multiple conditions. First, the approach was tested using a high number of simulated experiments. Specifically, a simulator of the real behavior of the robot was created and used to obtain an estimation of the robot expected time for randomly generated trajectories. The results coming from the simulation and from the proposed approach were compared in terms of mean execution time and standard deviation. Second, a set of experiments



(a) Leaf bins of the octree and a trajectory that crosses the human workspace

(b) Octree with only leaf bins and a zero-order Markov chain built on it.

(c) Changes to be applied to the states and the transition probabilities of a Markov chain with order > 1 that models the stop of the robot.

Fig. 44 The volume occupied by the human and the robot during the execution of their tasks is represented by two cloud points, than merged and organized into an octree. The leaf bins of the octree occupied by both the human and the robot will be used as nodes in a set of Markov chains.

was conducted on a real setup and compared in terms of obtained mean execution time with the results coming from the approach.

3.2.2 Results

The results show to have relevant repercussions on the company profitability and productivity in terms of improved efficiency and reduction of idle time. Roughly, the method is able to estimate the mean time for a cooperative task with an error less than 5%. Given such accuracy in the mean time estimation, the software module developed in piROS is able to select the best and mean-fastest trajectory among a wide set of many different paths.

3.3 Task Flow Management

Pallet assembly/disassembly displays some common features of highly dependable human-robot interaction tasks. The modeling and programming background is framed in the modular component-based software engineering. From a workflow standpoint, state-based execution is more effective than time execution: collaborative applications are loosely based on time sequences, although robot tasks in production are almost always programmed with deterministic timing because of simplicity and design attitude. In interactive tasks, however, it is recommended that the timing is not rigidly imposed by the robot for usability and comfort purposes [36, 35]. Instead, execution states may be triggered by events made by either robots or operators (see [113, 75]).

3.3.1 Methodology

Robot task programming has proved to be a major enabler for adaptability, because robotic applications analysis can be done on the basis of elementary units of logical functions (*i.e.actions*) [102],[99]. From a software engineering stand point, component-based engineering is the key approach [23, 24] for ensuring modularity, composability and reusability of such actions and tasks.

In piROS approach such modular approach has been first pursued for a task programming standpoint, making use of the IEC 61499 standard [65, 111] for developing the application-level control structure, *i.e.*the top hierarchical layer for individual machines (*e.g.*robots). Composing tasks is a matter of assembling functional units (see Section 3.3.1), connected to the physical layer (*i.e.*device controllers). See an overview of the programming-deployment architecture in Figure 45.

While this approach has been suitable in the initial stages of EuRoC challenge for local task programming by application developers or by end users through simplified graphical interfaces, for details see Section 3.3.1. The piROS architecture later extended to include scheduling/planning problems. The allocation of human and robot tasks has been coordinated with the FMS shopfloor-level controller, extending the approach to a multiscale role and action allocation based on Flexible Timelines. An overview of an expanded programming-planning architecture are in Figure 50 while details are in Section 3.3.1.

Task programming using IEC61499 Function Blocks

IEC 61499 defines architectures and standards for logical Function Blocks (FB) which are the natural modular and reusable entities constituting one of best know methodology in the spirit of component based engineering (see examples in [100]).

Robot control through IEC 61499 dates back from [101] to recent results [114, 115]. While exploiting the similar wrapping approach for encapsulating low-level features, our development is comparatively less centralized and provided a good degree of composability. All function blocks dedicated to operators inputs and behaviour are more flexible with the possibility of being interrupted, with the possibility of introducing a broader spectrum of Temporal Allens Relations [11].

The proposed approach consists of a layered (hierarchical) organization of FBs, associated to progressively low-level functionality:

- **scheduling level** that manages on-line the allocation and execution of applications based on internal/external requests. Requests can originate from operator decisions, external plant inputs, events in other applications which are running in parallel. Queuing requests and sorting the allocation of applications is one of the simplest and most effective ways to coordinate multi-applications workcells;
- **application level** that includes all the specific application instances, namely all the branches of interconnected tasks that reflect a desired work-flow or elementary sequences of tasks. The definition of applications as interconnected, orderly triggered sets of actions (encapsulated in FBs) is fundamental for a lean structure of this layer;

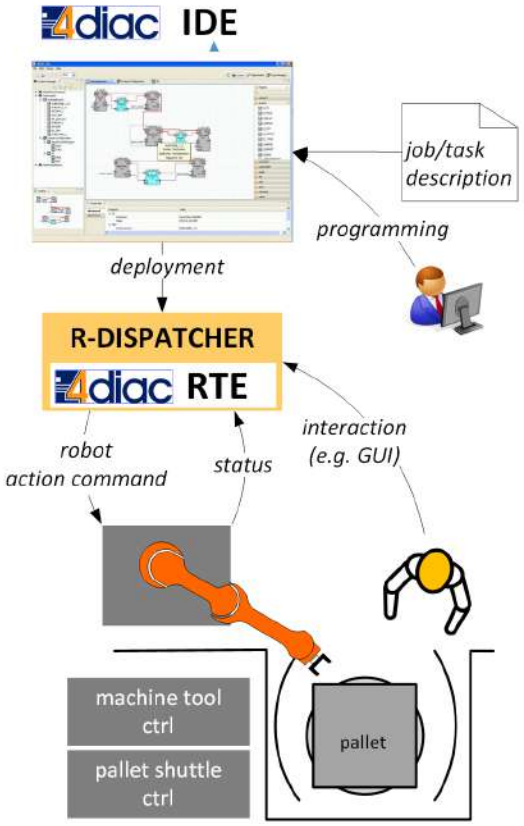


Fig. 45 Overview of the programming-deployment architecture.

- **low level** (closed-loop control) which is dedicated to machine-level behavior. As far as closed-loop control and robot trajectory generation are concerned, the Robot Operating System framework (ROS) is one of the best candidates for being wrapped inside FBs.

More details are provided in [54]. The deployment of IEC 61499 solutions according to the approach in Figure 46 is based on the composition of logic in the dedicated IDE (4DIAC).

Despite the spaghetti chart appearance, tasks composition in a IEC 61499 IDE is straightforward for *e.g.*PLC programmers. All relevant blocks for conditional execution and user-interaction are included (*i.e.*decisions, queries for options, wait for commands, alerts), although the user interface usability is limited.

Task planning with Flexible Timelines

Component-based engineering supported by IEC 61499 FBs is a powerful method for integrating local robot system scheduling and control, with factory-level supervi-

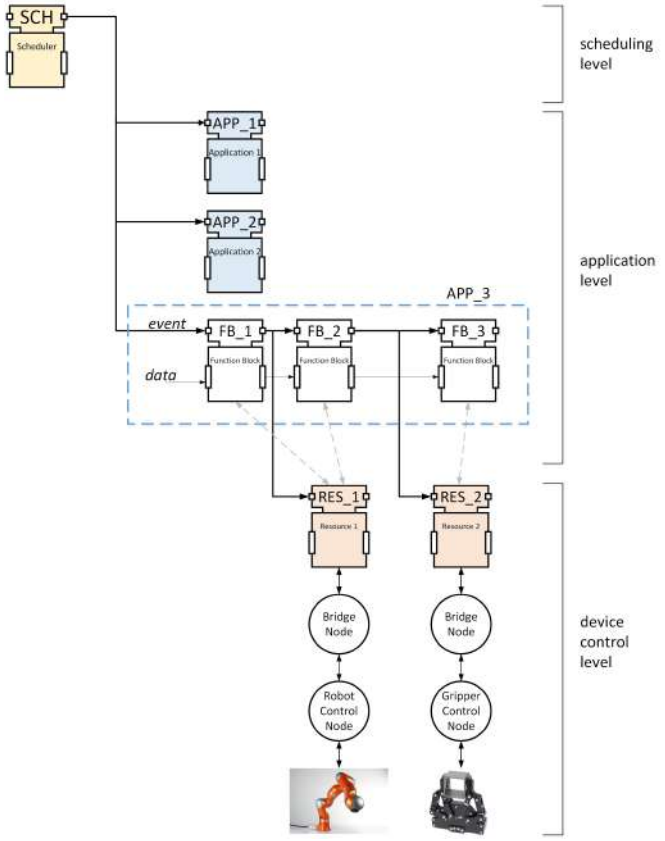


Fig. 46 Actions and function blocks.

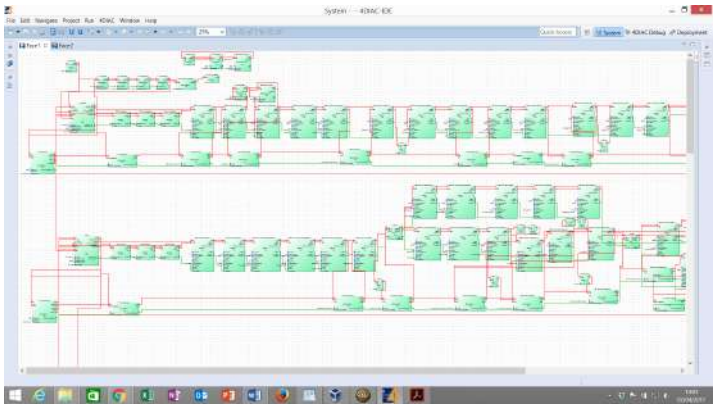


Fig. 47 Example of function blocks application in 4DIAC.

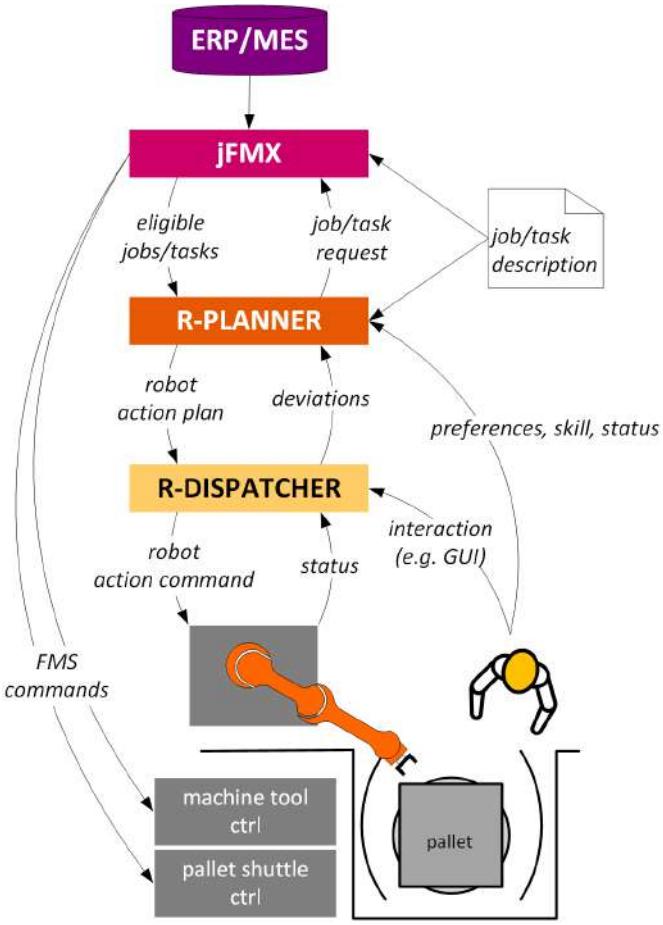


Fig. 48 Concept of vertical integration between MCMs factory-level control and robot control. (ERP = enterprise resource planning; MES = manufacturing execution system; jFMX = MCM machining supervisor; R-PLANNER = planner module of robot system; R-DISPATCHER = dispatcher module of robot system).

sory systems, like the FMS supervision module developed by piROS end user MCM S.P.A and called java Flexible Manufacturing eXecution (jFMX). The jFMX module is delivered in two versions: ”level 1” for FMS control, ”level 2” is interfaced with the factory ERP, increasing the scope and scheduling capabilities with respect to production objectives. The jFMX module is natively an even-driven architecture, whose functional logical units are CNC machine tool actions. While wrapping IEC 61499 FB to jFMX is straightforward, we found that the resulting hierarchy (from jFMX task scheduling to execution commands for robots) would have limited the flexibility of human-robot local role allocation.

While at the macro scale the jFMX module elaborates a list/queue of eligible tasks to be assigned to a LUS, the micro-scale R-PLANNER selects which single task to execute (*i.e.*calling for a specific element of the FMS queue), and how to assign the task actions between the robot system and the human operator (see Figure 48)

The possibility of variously combining synchronous, simultaneous or asynchronously supportive types of interaction between humans and robots, allows the R-PLANNER modules a higher flexibility in task assignment phase but requires a detailed knowledge of the tasks at hand, and a full understanding of the risks involved with task-centric collaborations [72], *e.g.* drifts from coordination and deployed plans. The piROS framework for robot planning is based on flexible timelines[73]. Inspirations and design motivations are derived from :

- long-standing approaches [10] that allow the robot to select and perform its tasks while taking into account explicitly the constraints imposed by the presence of humans, their needs and preferences;
- and recent works[27] about flexible-timeline control architectures based on hierarchical task decomposition in which the hierarchy of the timelines reflects the aggregation of actions performed by the system resources (operator and robot).

Flexibility in task allocation or recombination between humans and robots (under the hypothesis that same tasks can be allocated to both agents), can be conveniently attained taking into consideration important aspect of human-robot collaboration, like operators preferences (usability factors) and degree of strain (physical ergonomics factors). A generated plan (see R-PLANNER in Figure 48 and Figure 50) for assignment of specific actions may select different intensity and/or repetition patterns in pallet assembly, depending on the type of fixtures to handle, occurrences of different fixtures in the pallet queue, preference of operators for anticipating some specific tasks, etc.

3.3.2 Results

The vertical integration model in Figure 48 is designed and developed making extensive use of ROS components and packages, in order to take advantage from asynchronous communication (publishing/subscribing) and sub-task execution. The basic logical unit remain the *action* (as in Section 3.3.1). The task and action model are encoded in form of NDDL files. The actual planning capability is provided by the planner component EUROPA, which encapsulate all the resources of the robot system in form of *reactors* (see Figure 49). Reactors can be designed in functional hierarchies with senior components setting goals for junior elements, while receiving observations (state of part of the system in form of data recorded at runtime). Reactors have the capability of interfacing with executable (*e.g.*control or IO) nodes through the publishing/subscribing protocol.

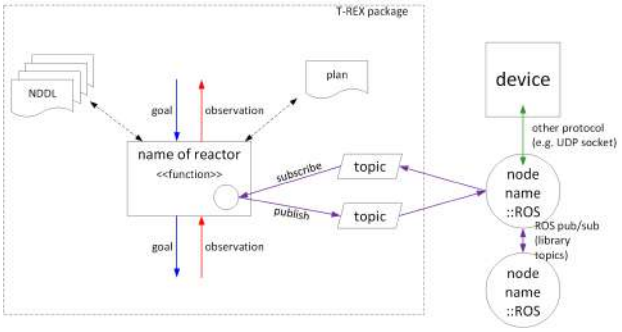


Fig. 49 Integration between T-REX package and ROS framework.

On the basis of the EUROPA foundational technology for planning algorithms on timelines, the framework of task description, planner and reactors is collectively implemented by the T-REX ROS package (see Figure 50).

Interfacing is necessary between T-REX task description and overall goals (*e.g.*pallet queue to be necessarily worked) and the resources hardware layer (*e.g.*robot controllers).

3.4 Gesture recognition

Gestures are meaningful body motions involving physical movements of the fingers, hands, arms, head, face, or body with the intent of: 1) conveying meaningful information or 2) interacting with the environment. They constitute one interesting small subspace of possible human motion. A gesture may also be perceived by the environment as a compression technique for the information to be transmitted elsewhere and subsequently reconstructed by the receiver.

3.4.1 Methodology

Gesture recognition is commonly supported by 3D cameras (stereo vision), sensors for Point Cloud (e.g., LEAP motion), 2D color cameras . Other wearable approaches, based on e.g. accelerometers and gyroscopes are also used for very specific applications questionably applicable in many manipulation-intensive tasks. Gesture information is derived by e.g. depth segmentation and skeleton tracking with 3D sensing, or by color segmentation (detection of hands, head or tags) with 2D.

The industrial challenge investigated within EuRoC have been to catch rapid human detection in cluttered environments, since such task has high requirements in terms of gesture recognition in variable conditions, with efficient processing of

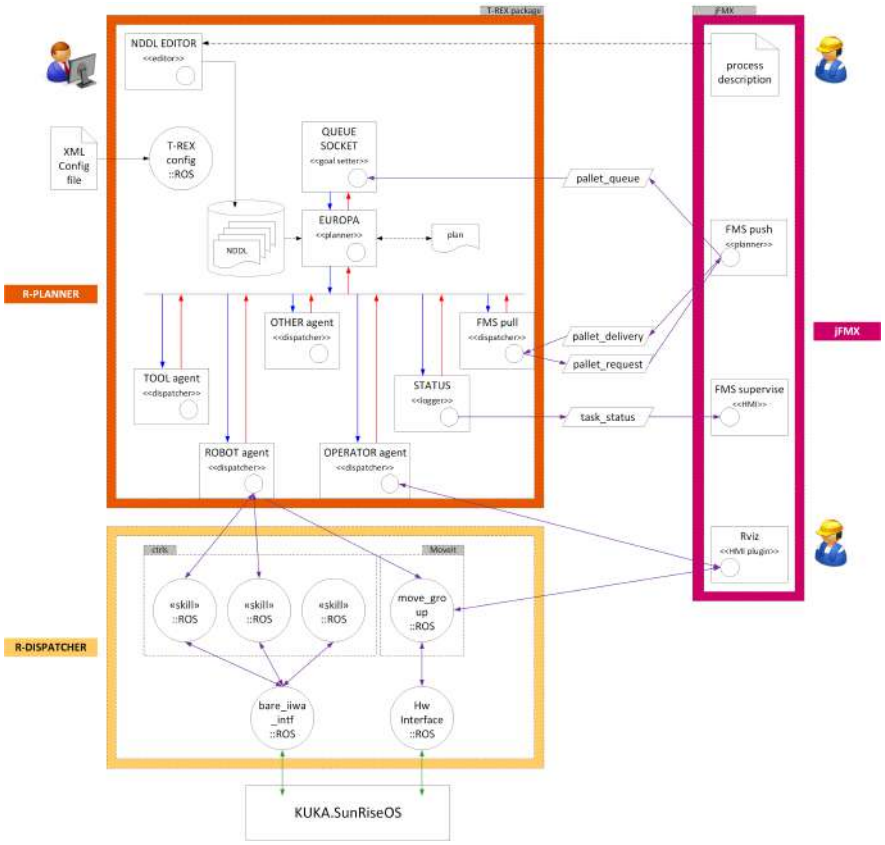


Fig. 50 T-REX ROS implementation of the framework.

point clouds. Furthermore, the reference setup is very prone to occlusions and environmental noise. Specifically the effort have been dedicated to perform a a filtering operation to detect stable gestures, i.e., the target location on tombstone is provided by the operator with gestures (e.g. finger-pointing an empty slot onto the tombstone). Specifically, a gesture is considered stable when a number N of consecutive detected points lie within a neighborhood area of radius R. N and R are configurable parameters in the pointing gesture algorithm (see Figure 52).

The pallet is defined as a grid structure. By analyzing the coordinates of the intersection point obtained in the pallet surface frame, it is possible to return the identifier of the cell the worker is pointing at. The design is intended to be modular w.r.t. the various appearances of fixtures that are possible attached to a pallet.

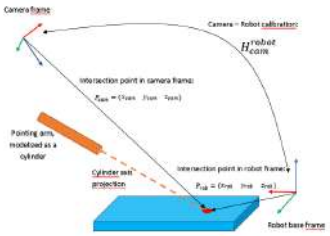


Fig. 51 User pointing to an insertion cell on face 2.

Fig. 52 Basic schema of the pointing gesture algorithm.

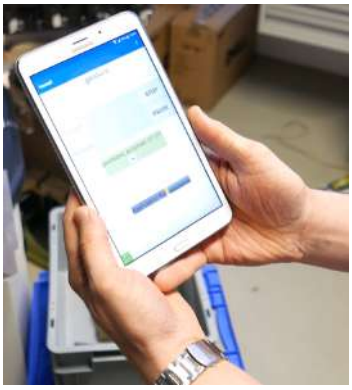


Fig. 53 Recorded cells inside the app.

3.4.2 Results

The experiments in the freestyle demonstrated the effectiveness of the methodology, with a mean time needed for the detection of the gesture of less than 0.5s, and with a percentage of success of greater than 90%.

Furthermore, during the showcase session, the industrial user considered the methodology robust and comfortable, and the usability was ranked as high.

3.5 Safety

Variability in task assignment between humans and robots, in addition to the high rate of collaboration by humans, increases the range of interaction situations possibly involving hazards. In fact, the close proximity and frequent interactions between robots and humans expand the *intended uses* (*i.e.*behaviors and tasks that do not deviate from the designed/programmed plan) of the collaborative robotics system. The associated risk assessment need then to consider a much higher number of assumptions at design time. Additionally, planning modules (see Section 3.3) generate task allocation which might be hard to list and evaluate entirely at design time. Considering, finally, the layout and hardware changeability (to adapt to different workflows or materials), the number of risk variables is further increased by partially-known objects inside a shared workspace. On top of the intended use *variability* (see Figure 54), a standard, *i.e.*manual, risk assessment is challenged also by the *complexity* of situations introduced in a collaborative application, including, but not limited to, open ended environment whenever mobile solutions are involved and trajectories are not known in advance, and variable agents assigned to the application. This latter situation is typical of dynamic assignment of personnel or different hardware resources to a robotic station, as it might be the case of a collaborative assembly task in a LUS.

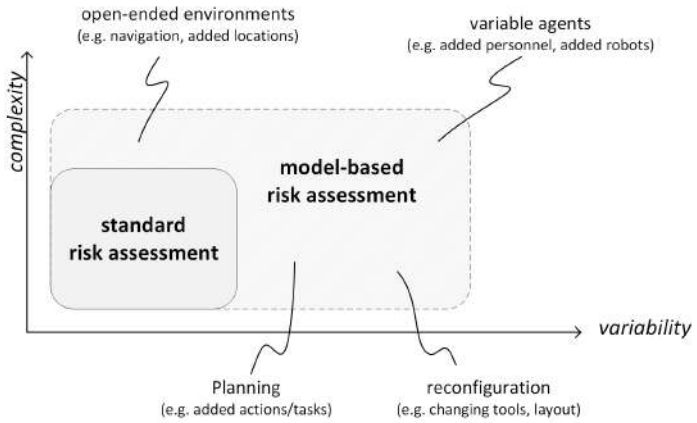


Fig. 54 Risk assessment, variability vs complexity.

Considering human aspects, operators might make errors or out-of-training behaviors, and may entail different *mistaken* workflows (*e.g.*circumventing instructions or attempting hazardous troubleshooting).The customization of applications, *e.g.*setting a preferred comfortable robot trajectory or altering the sub/task assignment during run-time execution (see Figure 34), usually improves the usability of a system, but the hazard/risk distribution is more dependent on synchronization capability of operators, their overall situation awareness when supervising multi-

ple tasks and ability to react/avoid hazardous situations during transient conditions (*e.g.*actions for catching up with a robot operation).

Hence, it is crucial that the design of the task and workspace provides suitable protection for operators in changing conditions, and that the actual deployment of applications conforms to such a safe design. piROS experience provided an outstanding use case for introducing model-based, Computer-Aided Risk Assessment (CA-RA) with the purpose of supporting the designer of safety solutions, together with the robot system manufacturer and other stakeholders, during the phases of hazard identification, risk estimation, and risk evaluation. The approach is dedicated to modeling the collaborative application with a formal model that can be checked and verified for a given objective property, which in case of HRC situations corresponds to the condition of negligibility of all risks associated with hazards potentially occurring during the task execution.

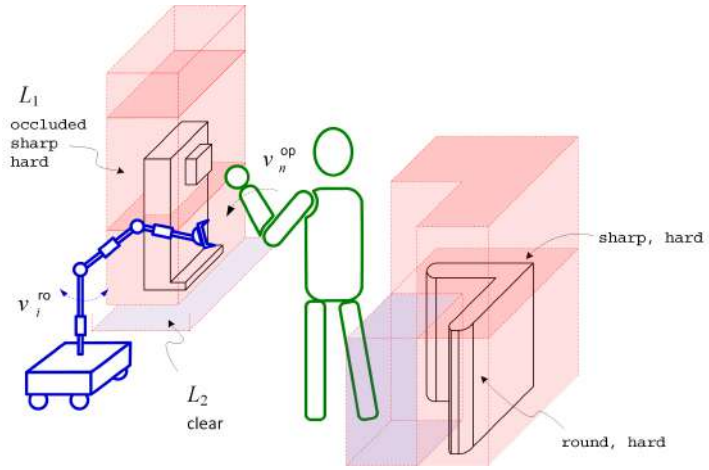


Fig. 55 Conceptual scheme of the robotic cell.

3.5.1 Formal Modeling and Verification of Safety conditions

The safety verification of target tasks is based on a formal model representing the elements of a collaborative application (operators, robots and environment), together with models of hazards (*e.g.*contacts), risks and risk reduction measures (*e.g.*limit energy). A formal verification tool is used to comprehensively explore and check all possible interactions between robots and human operators, with respect to *risks*, along the designed task. Remarkably, the verification procedure is in charge of exploring also the *unintended* interactions, where some potential hazards may occur.

The model is formulated using a temporal logic language (TRIO [43]) featuring a quantitative notion of time, *e.g.*the formal representation of typical human-robot

Table 4 List of derived TRIO operators. In particular, given a time-dependent formula ϕ (i.e., a term representing a mapping from the time domain to truth values) and a (arithmetic) term t indicating a time distance (either positive or negative), formula $\mathsf{Dist}(\phi, t)$ specifies that ϕ holds at a time instant at a distance of exactly t time units from the current one; ϕ, ψ denote propositions, and v is a variable and d is a constant value.

TRIO Operator	Definition	Meaning
$\mathsf{Past}(\phi, d)$	$d > 0 \wedge \mathsf{Dist}(\phi, -d)$	ϕ occurred d time units in the past
$\mathsf{Futr}(\phi, d)$	$d > 0 \wedge \mathsf{Dist}(\phi, d)$	ϕ occurs d time units in the future
$\mathsf{Alw}(\phi)$	$\forall t(\mathsf{Dist}(\phi, t))$	ϕ always holds
$\mathsf{Som}(\phi)$	$\exists t(\mathsf{Dist}(\phi, t))$	ϕ occurs sometimes
$\mathsf{Until}(\phi, \psi)$	$\exists t(\mathsf{Futr}(\psi, t) \wedge (\forall t'(0 < t' < t) \Rightarrow \mathsf{Dist}(\phi, t')))$	ψ will eventually occur and ϕ will hold till then
$\mathsf{Until}_{\mathsf{w}}(\phi, \psi)$	$\mathsf{Until}(\phi, \psi) \vee \mathsf{Alw}(\phi)$	weak until: ψ may never occur in the future
$\mathsf{WithinF}(\phi, d)$	$\exists t(0 < t < d \wedge \mathsf{Dist}(\phi, t))$	ϕ will occur within d time units

interaction situations like "reaching an object *before* the robot", "the robot *always* stays in a given region", etc.

Zot [86] is a bounded satisfiability model checker for TRIO language [88], where the property to satisfy is a low value of the modeled risk, for each hazard identified during the automatic model checking phase. In case the property (*i.e.*low risk) is not satisfied, Zot provides a counterexample witnessing a system execution that violates the property (*i.e.*a residual hazardous situation).

TRIO formulae are designed and suited[3] for expressing the temporal relationships of human-robot collaboration situations, including the interactions with the environment (*e.g.*machinery, fixtures, etc), that could affect safety. The formal model, in fact, includes both agents (*e.g.*humans, robots) and those interactions which can be classified as hazards (*e.g.*collisions, entanglements, shears, etc). For example, an accidental contact between an operator and the robot system may occur with different hazard profiles depending on the timing of arrival in the same location, *e.g.*both human and robot converge to a single point are likely entangling in a mutual crush; a robot arm moving away from a pallet fixture while the operator is accessing is more likely to determine a dynamic impact).

With reference to [15, 16] for complete details, the model is formulated in terms of TRIO predicates. First, three main sets of elements are defined, one for each component of a collaborative application: the human body $\mathscr{O}$, the robot system $\mathscr{R}$, and the environment $\mathscr{L}$ with a given layout (see Figure 55):

- the set $\mathscr{O}$ is a collection of eleven discrete body regions $\{O_{head}, \ldots, O_{leg}\}$ [57], where each region $O_i = \langle p_i, v_i, m_i, k_i \rangle$ is characterized by a set of predicates,

[3] We adopted TRIO formal language and its supporting tool Zot because of the generality of the language, which can be applied in natural way to different types of applications (together with MTL, TRIO is one the first temporal logic languages dealing with time in a metric way) and for the efficiency of the verification in comparison with the state of the art [18]. For a comprehensive survey and comparison among various logic languages tailored towards modeling time dependent phenomena see [42].

such as for example p_i, representing the location of body region i, v_i, representing its velocity, etc.
- the set $\mathscr{R}$ includes elements of a robot, with predicates $R_j = \langle p_j, v_j, f_j, m_j, shape_j \rangle$ for each of its n links.
- the model $\mathscr{L}$ of the layout includes a tuple of predicates $L_k = \langle shape_k, material_k, obst_k \rangle$ for each of the sections in which the layout is divided. Some predicates, like *shape*, *material* or clearance/occupancy $obst_k$ carry safety-related information.

The model evolves through states, assigned according to predicates, *e.g.* $p_i^{\mathtt{ro}} = L_k$ means the presence of the i-th link of the robot inside the k-th region of the layout. The temporal logic is formulated in terms of first-order connectives, operators, and quantifiers, as well as some composed predicates (*e.g.* $\mathsf{InSameL}_{ijk}$, etc) based on a modal operator, called Dist, that relates the *current time* to another time instant (see list of operators in Table 4).

In this way, the formal model defines formulae that capture various safety-related aspects of HRC applications, notably including hazards. For instance, a hazardous *quasi-static (Qs)* [57] contact $\mathsf{hzd}_{ijk}^{\mathtt{Qs}}$ of constrained type (for example, crushing) that occurs in section L_k of the layout between the i-th robot link (for example, the end-effector) and the j-th body part (for example, the human hand) is formulated as:

$$\begin{aligned} \mathsf{hzd}_{ijk}^{\mathtt{Qs}} \Leftrightarrow\ & \mathsf{InSameL}_{ijk} \wedge \mathsf{moving}_{R_i} \wedge \mathsf{Past}\left(\mathsf{Sep}_{ij} > \mathtt{close}, 1\right) \\ & \wedge \mathsf{Past}\left(p_i^{\mathtt{ro}} = L_k \wedge p_j^{\mathtt{op}} \neq L_k, 1\right) \\ & \wedge \left(obst_k = \mathtt{occluded} \vee \exists R_m \in \mathscr{R}\left(\mathsf{InSameL}_{imk} \wedge i \neq m\right)\right) \end{aligned} \tag{2}$$

where $\mathsf{Past}(F,1)$ means that F held in the previous instant from the current one, $\mathsf{InSameL}_{ijk}$ and moving_{R_i} are abbreviations for formulae describing, respectively, that R_i and O_j are in the same layout section, and that R_i is moving, and Sep_{ij} is a predicate capturing the current separation between R_i and O_j. The last two lines of the formula formalize, respectively, the situation of R_i and O_j getting closer to one another (with the robot arriving in L_k *before* the operator), and the presence of obstacles (*i.e.* the occlusion of section L_k) in that condition, or the internal crunching of arms inside a manipulator.

A hazard is identified as soon as the model state matches a condition defined as in 2 (recall that the actual state is generated at verification time, and not assigned *a priori*). Identified hazards are then estimated in terms of risk, whose score is assigned using TRIO predicates that capture the severity and occurrence of hazards according to normative safety methodologies [55]. In particular, the hybrid method for risk estimation in [56] is used for setting severity with discrete scores in $\{1,2,3,4\}$ and occurrence factors (frequency of situations, probability of errors and failures, probability of the avoidability of the situation) with discrete values cumulated in an aggregate score in $\{1,\ldots,15\}$. A discussion on the determinism of score assignment and update of values is provided in [14].

Upon estimation of unacceptable risk level, verification of the formal model highlights a condition of unsatisfiability of the target safety property. The model needs to be updated including Risk Reduction Measures (RRM), which are the models of safety strategies to be implemented. For instance, a collision avoidance method for preventing the contact hazard hzd_{ijk} compliant with the safety mode Speed and Separation Monitoring (SSM) [57] is forumlated as:

$$\begin{aligned}\mathsf{RRM}_{ijk}^{SSM} \Rightarrow &\left(\mathsf{Sep}_{ij} < \mathsf{Sep}_{min} \Rightarrow \mathsf{Futr}\left(v_{ij} = \texttt{none}, 1\right)\right) \wedge \\ &\left(\mathsf{Sep}_{min} \leq \mathsf{Sep}_{ij} \leq \mathsf{Sep}_{mid} \Rightarrow \mathsf{Futr}\left(v_{ij} \leq \texttt{mid}, 1\right)\right)\end{aligned} \tag{3}$$

where $\mathsf{Futr}\left(F, 1\right)$ means that F holds in the next instant from the current one, and Sep_{min} and Sep_{mid} are two thresholds on the distance between O_j and R_i, moving at relative speed v_{ij}. Sep_{mid} is used as a threshold for slowing down R_i, so to maintain a Sep_{min} threshold.

Recall that after the update of the model with RRMs, the hazard identification and risk estimation steps described above are repeated along the *new* evolution of states of the model, for checking the satisfiability of the safety property, which needs to be verified for each combination ijk identified as potentially critical:

$$\mathsf{Alw}\left(\begin{array}{l}\mathsf{risk}_{ijk} \leq \tau \vee \\ \\ \mathsf{risk}_{ijk} > \tau \wedge \exists y \left(\begin{array}{l}\mathsf{RRM}_{ijk}^{y} \wedge \\ \\ \mathsf{Futr}\left(\mathsf{risk}_{ijk} \leq \tau, 1\right)\end{array}\right)\end{array}\right) \tag{4}$$

where τ is the risk threshold value decided at design time, usually set to $\tau = 1$ (negligible/low risk) for scores $\mathsf{risk}_{ijk} \in \{0, 1, 2\}$ assigned using the ISO/TR 14121-2 hybrid method.

Such mechanisms of generation and evaluation of states of the model in every instant of its evolution are enabled by the Zot tool,[4] which implements several techniques—employing a so-called bounded satisfiability checking approach based on off-the-shelf Satisfiability Modulo Theories (SMT) solvers—for checking the satisfiability of TRIO formulae in an automated manner [18].

3.5.2 Experimental results for piROS setup

The operational scenario and layout (see Figure 31) are reproduced in the formal model (see Figure 56). The robot is supposed to move between L_8 and the pallet repetitively, to bring and load pieces on the pallet. Each piece first be placed in L_4 and kept held until human operator terminates the part manipulation task.

Here we briefly report a couple of outputs and refer you to find more details on customizing the formal model in [14] and formalizing the executing task (i.e., UML profiles to logic formulae) [64]. In the experimental results reported in the rest of this section, only one potential feasible trace has been analyzed which about 410

[4] Available from github.com/fm-polimi/zot.

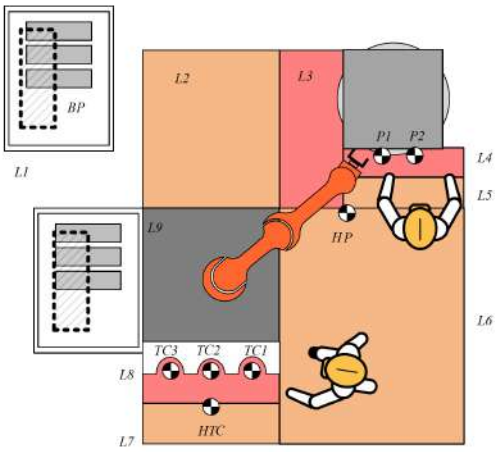


Fig. 56 Layout of the use case with its critical zones identifies with colors.

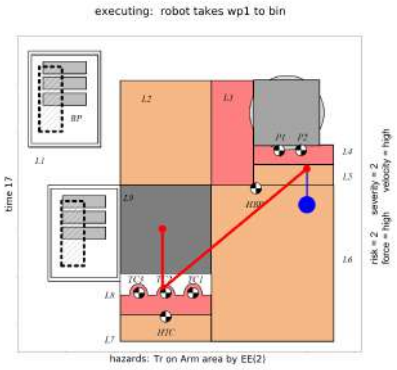


Fig. 57 This is an example of a situation in which RRMs are excluded from the model. operator head and arm are depicted in blue, while robot links and endeffector are in red. The operator arm and endeffector are collided at $t = 17$ and the risk value exceeds the tolerable risk threshold.

sec long, verification bound is set equal to 40 time units, and each action takes at least three time units to complete.

First, the model is checked against the safety property without including RRM module to show that there are risky situations generated by the model that requires mitigation. Figure 57 pictures an example of occurrence of a hazard with a risk higher than two. There is no RRM to monitor and constraint the values of force and velocity. Hence, the severity of potential collisions could be quite high.

Second, we include RRMs in the model to show that same situations detected above are not critical as before because of RRMs. In this case the negation of safety

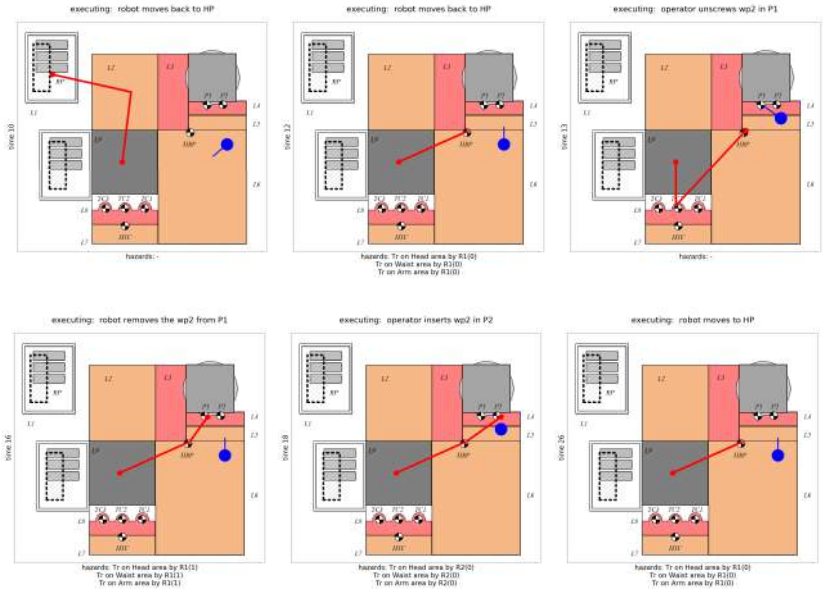


Fig. 58 An example of partial task execution in presence of RRMs which shows whenever hazards are detected, their risk is tolerable (less than two).

property is never true and thus the tool reports **UNSAT**. However, in order to show how the system works, we removed the safety property and got a possible execution trace which is shown in Figure 58. The safety of execution does not mean that hazards do not happen at all. The operator and robot have to be present in same locations to execute several actions and hazards are inevitable. However, RRMs eliminate the associated risk value to each hazard. For example, $t = 12$ in Figure 58 is an instance of hazardous system state in-which the caused severity of the hazards is constrained by reducing the velocity of the moving robot parts.

The formal model is also able to capture hazardous situations that are caused due to human erroneous behavior. For example, Figure 59 shows the same time unit in two different verification runs: one containing RRMs and the other one without RRMs. Both snapshots picture a time unit when the operator tends to execute an inspection at the wrong time. The operator should not execute an inspection while the robot is moving and getting close to the pallet, because it could cause a `Tr` hazards, as detected by the tool as well.

The critical difference between the time unit depicted in 59(a) and the one in 59(b) is the risk value. When RRMs are considered in the model, the risk value will not grow up to two since RRMs are invoked as risk reaches value one and try to maintain or reduce the risk by reducing severity via constraining velocity and force values. The picture on left has two ongoing RRMs that keep the relative force less or equal than `low` (meaning: `mid` or `low`) and the relative velocity less or equal than `mid` (mean-

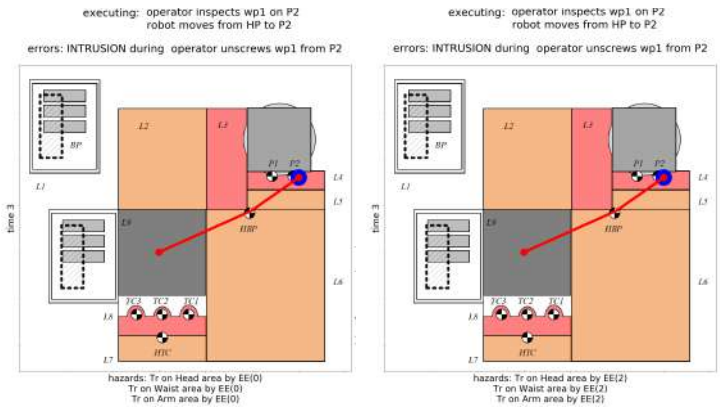


Fig. 59 An instance of an intrusion error.

ing: `mid`, `low` or `none`). The relative parameters here are pertaining to those parts of human body and robot parts which are involved in the definition of a detected hazard.

The model update is an explicit design action, which is ultimately under the responsibility of an overall safety supervisor. Different types of measures y (*i.e.*different RRM^{y}_{ijk}) can be chosen for mitigating risk_{ijk}, depending on the capabilities of the robot system (*i.e.*controls, sensors, etc.), and also on the possibility to modify the task itself, if necessary. As a consequence of the compliance with safety requirements (*i.e.*RRMs are necessary), production routines may be variously affected by the RRM strategy. For instance, slower safe modes in robot controls, re-arranged layouts, or forcing constrained action sequences by controlling access to locations, all have a substantial impact on the organization and/or efficiency of tasks.

3.6 Global Key performance index

Beyond the challenges related to each piROS module, the overall approach has been investigated. Indeed, the piROS challenge has been to deploy an ecosystem of modules that may improve dramatically the factory productivity and quality, not only tailored on the FMS use case, but with the ambition to be generally applicable to a wide field of applications.

The global Key Performance Index (KPI) for this objective is defined as the Added Value Index (AVI), which is the normalized ratio of high-ranking sub-tasks out of all sub-tasks:

$$AVI = \frac{\sum_i st_{i|r=3}}{\sum_i st_i} \tag{5}$$

where the rank r is a value-added score defined by the user, and st_i is a generic sub-task. Specifically,

- $st_{i|r=3}$ are sub-tasks of rank 3 like visual inspection, measurement on fixtures, resolution of faulty conditions on jigs or parts, etc.;
- lower value tasks—or sub-tasks that are labour-intensive—, *e.g.*pick and place, preparing insertion of parts in jigs, screwing/unscrewing, etc are ranked 1 or 2 by design since they introduce less value added.

When low-rank tasks are removed form the denominator in (5), the index increases, reflecting the fact that "better" sub-tasks are denser in operators' schedule (see an example in Table 5).

description	rank	default assignment	optimal plan
sub-task A	1	H	R
sub-task B	3	H	H
sub-task C	2	H	R
sub-task D	3	H	H
sub-task E	2	H	R
sub-task F	2	H	H
$\sum_i st_{i\|r=3}$		6	6
$\sum_i st_i$		13	8
AVI		46%	75%

Table 5 Added Value Index (AVI) index computation for a real case.

Simple, intuitive procedures for reviewing task/sub-task assignment before launching the task are fundamental for increasing AVI figures. The main indicator of performance is the seamless vertical integration of modular components able to share software interfaces, protocols and programs. The tools developed for this purpose are described in in Section 3.3.

The general KPI of the system to provide sustainable, effective support for human operators is to display an AVI $> 75\%$ for most of the production tasks (*e.g.*every time a new pallet is presented to the operator). The soundness of the index relies on three main points:

- the swap of the task between human and robot is possible, and the sub-task performance is preserved, as stated in the above technical sessions.
- the planing allows in real-time the best allocation to reduce the index according to the real conditions (*i.e.*able to react to changing situations)
- the safety, the easiness, and the usability are always granted (as shown in the Section above)

4 Conclusion

The human-robot collaboration in Load/Unload stations has been considered. Team piROS approached this challenge by adopting a modular solution. Actions have been modelled as function blocks. A scheduler assigns these actions to the human or the robot in order to optimize the Added Value Index (AVI), that is to assign sub-tasks with high added value to the human as much as possible.

The optimization of the AVI required high adaptability of the cell. This level of adaptability has been obtained by: *i)* improving the manipulation abilities of robot, allowing the swapping of low added values activities between human and robot; *ii)* realizing a human-aware motion planner which takes into account the human presence to plan the trajectory and to estimate the possible time exection variability; *iii)* interacting with the operator by means of a gesture recognition; *iv)* adopting a function block modular architecture to replan the task which requires a minimum effort of the operator; v) developing a model-based Computer-Aided Risk Assessment to support the designer of safety solutions during the phases of hazard identification, risk estimation, and risk evaluation.

Experimental results show the effectiveness of the proposed approach in a true industrial scenario.

4.1 Future developments

The piROS end-user is very satisfied of the human-robot collaboration framework designed during the project. In fact, he is interested to extend this paradigm to other cells of the plant, as well as to redesign the shopfloor flows to adapt them to reorganized layout as in Section 4.2.

For this reason, some members of Team piROS are involved in the European Project Sharework [38] and in the *Assuring Autonomy* Project Recoll [17]. Within these projects, the considered LUS cell will reorganized by dividing the assembly and inspection of the tombstone from the mounting of the pallet of the FMS. This will improve ergonomics and the shared workspace between the robot and the human.

4.2 Lessons learned

The human-robot collaboration in LUS handling and assembly is a challenging scenario, which implies the integration of many software and hardware components as well as designers skills. As Team piROS, the main lessons we learnt is the required paradigm shift to convert a LUS cell to a human-robot collaborative cell. In fact, the collaborative robotic framework is not an add-on component which can be mounted on a non-automated cell without requiring changes. In fact, the same task

imposes different timing and spatial constraints when performed by a human or a robot. Moreover, in many cells there could be a workspace limitation that does not allow the robot and the human to perform parallel tasks in proximity.

The implementation of a human-robot collaboration should always be used to rethink the way a cell works. In fact, a layout reorganization (w.r.t a simple addition of the robot in the preexisted scenario) improves ergonomics, the sub-task performance, and the number of tasks that the robot and human can make in parallel. Moreover, it can allow the designer to improve the material workflow.