Open-Digital-Industrial and Networking pilot lines using modular components for scalable production

Grant Agreement No	:	**101017141**
Project Acronym	:	**ODIN**
Project Start Date	:	**1st January, 2021**
Consortium	:	**UNIVERSITY OF PATRAS – LABORATORY FOR MANUFACTURING SYSTEMS AND AUTOMATION** **FUNDACION TECNALIA RESEARCH & INNOVATION** **KUNGSLIGA TEKNISKA HOEGSKOLAN** **TAMPEREEN KORKEAKOULUSAATIO SR** **COMAU SPA** **PILZ INDUSTRIEELEKTRONIK S. L.** **ROBOCEPTION GMBH** **VISUAL COMPONENTS OY** **INTRASOFT INTERNATIONAL SA** **GRUPO S21SEC GESTIÓN, S.A.** **FUNDACION AIC AUTOMOTIVE INTELLIGENCE CENTER FUNDAZIOA** **DGH ROBOTICA, AUTOMATIZACION Y MANTENIMIENTO INDUSTRIAL SA** **PSA AUTOMOBILES S.A.** **AEROTECNIC COMPOSITES SL. U.** **WHIRLPOOL EMEA SPA** **WHIRLPOOL MANAGEMENT EMEA SRL**

Title	:	ODIN Core Enabling technologies for smart programming and HR interaction interfaces - Final version
Reference	:	D2.5
Availability	:	Public
Date	:	28/12/2023
Author/s	:	TECNALIA, TAU, LMS
Circulation	:	EU, consortium

Summary:

This document focuses on the presentation of the final version of a) easy programming interfaces and b) human side interfaces

ODIN project has received funding from the European Union's Horizon Europe research and innovation programme under Grant Agreement No 101017141.

Co-funded by the European Union

Table of Contents

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

This document provides a detailed description of the final prototypes of core enabling technologies for easy programming and human robot interaction interfaces.

- Easy programming interfaces – Task 2.4. The following key point technologies will be presented in this document:
    - ODIN use cases specific skill development,
    - Advanced behaviour management,
    - On-site interactive skill programming system.
- Human robot interaction interfaces – Task 2.5. The initial prototype of the following key technologies will be presented in this deliverable:
    - ODIN Augmented Reality Application,
    - ODIN Smart User interfaces, and
    - Virtual Reality safety training.

Each technology will be integrated inside ODIN pilot lines based on end users' requirements as documented in D1.1.

Running the M36 of ODIN, the final prototypes of the previously documented technologies have been prepared by the partners and will be presented in this deliverable alongside with their final implementation tests.

LEGAL DISCLAIMER

The ODIN project is co-funded by the European Union's Horizon 2020 research and innovation programme under the Grant Agreement No 101017141. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the granting authority can be held responsible for them.

1. INTRODUCTION

The use of Skill Based Programming (SBP) frameworks allows implementing robotic applications sequencing configurable pre-programmed blocks (skills). It has been demonstrated being very useful for easy and fast deployment of relatively simple operations such as pick and place, inspection, handling a device for performing a process (e.g. drilling, deburring or screwing), navigation, etc. The operations that, can be generalized, may benefit of SBP approaches allowing a high degree of re-usability. The SBP techniques, combined with CATIA software-based CAD Programming concept for skill configuration, allow the plant operators to take advantage of the intrinsic information that CAD models contains in order to configure new processes using pre-programmed skills. The use of an extended and well-known software that the operators are habituated to use, empowers them for easy and quickly adaptation of existing robot programs. In this way, the flexibility of the line is increased. As mentioned above, SBP is ideal for tasks that can be generalized for a future re-use. However, there are operations that could require dexterous movements, or with a limited accessibility which makes difficult the collision free trajectory planning and execution.

Smart User Interfaces (UI) create new interaction modalities that enable informative and Real-Time communication in industrial human-robot assembly scenarios. The integration of digital light processing (DLP) projector and depth sensors provides operator's position detection and by utilizing marker-based approaches, a UI can be projected on a surface – even a mobile one. This application is utilized in pick and place tasks where virtual buttons and textual descriptions allow intuitive communication between human and manufacturing resources. The setup can also be used to provide help annotations for the operator, for example highlighting part position in the assembly operation. Visualizing work area of the robot with the projectors can increase operator's awareness of the robot movement and therefore decrease the number of violation events.

A virtual safety training system provides a safe training environment for industrial robotic applications, where the human and robot are working in close proximity. Lack of communication between robots and humans creates challenges for human perceiving and decision making during hazardous events. The virtual training application represents a replica of a robotic work cell and familiarizes the operator with safety measures. Feel of presence in virtual reality applications has been studied and proved to have a positive effect on human perception. In this set-up, the Unity [1] game engine features are utilized for creating 3D visualization content of the pilot line. It is worth mentioning that all visualizations could be exported as a template for further use in other applications. These templates improve the speed of application development at next use cases. The assembly training phase provides assembly tasks while the operator can see effect of violation mistakes with industrial indicators in the manufacturing sector. This two-phase training provides an intuitive immersive environment for acknowledging the operators regarding safety measures before stepping into the real robotics environment.

Easy programming interfaces combined with human side interfaces provides a powerful ecosystem for empowering the operators in different industrial settings belonging to disparate sectors.

The final prototype of easy programming interfaces will be presented in Section 3, while final prototypes of human side interaction interfaces will be documented in Section 4.

2. FINAL PROTOTYPE OF EASY PROGRAMMING INTERFACES

The main objective of the development of the easy programming interfaces is to generate more accessible tools to create, modify and manage robotic applications in base of templates named skills. To create these powerful skills, but maintaining their user-friendliness, special efforts have been made on the following sub-tasks:

- The process specific robot skill can be generated reusing previously developed robotic skills, but depending on the use case peculiarities new skills must be implemented. To create these new skills, an analysis, task sub-division and generalization iterations are being performed.
- Advanced behaviour management: The Skill Based Programming (SBP) could provide with an infinite set of skill combination, but the possible links between blocks determines overall system usefulness. Taking into the account the logic complexity, the usage of the state machines and behavioural trees will be necessary for handling the interaction graphically.
- On-site Interactive Skill Programming system (OISP). In some cases, various tasks require a large sequence of movements (final assembly steps, sealant application, complex trajectory execution, etc.). The combination of the SBP with OISP enables the SBP to integrate the operator approach into the robot environment simplifying the difficult associated to the robot programming.
- CAD Programming interface: An easy of use approach for visual task programming, Due to the new flexibility improvements in the CAD Programming interface, is possible the integration and creation of the new skills dynamically in execution. Taking into the account the CAD Programming interface was finished for the month 18, all the related information is compiled in the deliverable D2.2.

2.1 ODIN use cases specific skill developments

Throughout the ODIN project, a diverse set of skills have been developed. These skills will envelop the required sequence of robot movements, sensor captures, IO signals, end effector operations, mobility, etc. These skills are presented in three categories: in the aeronautics pilot, automotive pilot and auxiliary skills. In the following sections the developed skills details are presented (Table 1-Table 15).

2.1.1 Aeronautic Pilot

For the aeronautic pilot three use cases where selected, these can be defined in the next main scenarios:

- **Fan cowl transportation**:

In this case the robot transports the fan cowl from one station to another where some tasks are being performed. To complete this scenario the following skills are available:

  - **Navigate:**

The robotic platform is able to navigate into an appropriate location. This navigation parametrization can be done in three different skills.

Table 1: Navigation skill

Parameter	Description
Goal_pose	Target goal for the mobile platform defined in Pose2D (X, Y, Yaw).
Frame_id	Parent frame name.

Table 2: Navigation to station skill

Parameter	Description
Goal_pose	Target goal for the mobile platform defined by station list.
Frame_id	Parent frame name.

Table 3: Navigation to position skill

Parameter	Description
Goal_pose	Target goal for the mobile platform defined by current position of the robot.
Frame_id	Parent frame name.

- **Fan cowl grab/release:**

The robotic platform can be able to attach and detach the fan cowl transportation trolley.

Table 4: FanCowl grab/release skill

Parameter	Description
Grab	True for trolley attach and False for detach.
Camera	Requested camera.
Marker_1_id	Requested marker_1.
Reference_1	Distance to the marker 1 reference defined in Pose2D (X, Y, Yaw).
Tolerance_1	Requested maximum tolerance error for marker 1.
Marker_2_id	Requested marker_2.
Reference_2	Distance to the marker 2 reference defined in Pose2D (X, Y, Yaw).
Tolerance_2	Requested maximum tolerance error for marker 2.

- **Transport fan cowl:**

The robotic platform requires being able to transport the fan cowl trolley in the factory safely. For this skill, three different skills are available to select:

Table 5: Transport FanCowl skill

Parameter	Description
Goal	Target goal for the mobile platform defined in Pose2D (X, Y, Yaw).
Frame	Parent frame name.

Table 6: Transport FanCowl to station skill

Parameter	Description
Goal	Target goal for the mobile platform defined by station list.
Frame	Parent frame name.

Table 7: Transport FanCowl to position skill

Parameter	Description
Goal	Target goal for the mobile platform defined by current position of the robot.
Frame	Parent frame name.

- **Part inspection:**

In this case, the robot uses a sensor with high resolution and precision to scan and analyse the part in order to check the correctness of it. To complete this scenario the following skills have been developed:

- **Execute trajectory:**

The robotic platform requires being able to execute trajectories previously created with the teaching skill.

Table 8: Execute trajectory skill

Parameter	Description
Traj_id	Desired trajectory id for executing. This must exist in the database.
Tag	Refer trajectory by name. This could exist in the database.
Arm	Desired robot group to use.
Tool_id	Desired tool to use.
Vel	Desired velocity ratio of the trajectory execution.
Sim_io	True if simulated IO are used.

- **Inspect part:**

The inspection use case requires a skill which allows performing an inspection of installed devices in a FC. Inspect skill uses the data generated by the part learn skill.

Table 9: Inspect part skill

Parameter	Description
Fc_id	Fan Cowl id.
Arm	Desired robotic arm.
Trajectory_ids	Trajectories that the robotic arm will follow.
Camera_id	Camera to be used.
Topic	Topic with image source.
Reference_model	Reference model to be compared against.

- **Part drilling:**

In this case, the robot identifies the current template and with it can detect where the drillings are. To complete this scenario the following skills have been developed:

- **Docking:**

The robotic platform can dock in an appropriate location.

Table 10: Docking skill

Parameter	Description
Dock	True for docking and False for undocking.
Camera	Desired camera.
Marker	Requested marker.
Reference	Distance to the marker reference. In pose2D (X, Y, Yaw).
Tolerance	Requested maximum tolerance error.

- **Execute trajectory with vision:**

The robotic platform requires being able to execute trajectories previously created with the teaching skill. The main difference between this skill and previously descripted is the capacity to be more flexible in some learned trajectories due the usage of the cameras installed in the robot.

Table 11: Execute trajectory with vision skill

Parameter	Description
Traj_id	Desired trajectory id for executing. This must exist in the database.
Tag	Refer trajectory by name. This could exist in the database.
Arm	Desired robot group to use.
Tool_id	Desired tool to use.

Parameter	Description
Cam_id	Desired camera to use.
Detection_service_name	Service used to detect the marker during the teaching.
Vel	Desired velocity ratio of the trajectory execution.
Sim_io	True if simulated IO are used.

- **Drill:**

The drilling skill starts detection the desired template and after the trajectories to perform the drilling is calculated to continue with the drilling process. This skill which performs collision free trajectories and drilling machine management is required.

Table 12: Drill skill

Parameter	Description
Id	Identifier for traceability.
Robot _name	Desired robot manipulator.
Gripper_name	Desired tool name.
Camera_name	Desired camera name.
Pose_estimation_service _ns	Service for the pose estimation.
Template_id	Drilling template to be used.
Timeout	Max allowed time to complete approximation (sec).

- **Grab/release tool:**

The robotic platform requires being able to equip drilling machines at drilling stations for starting operation.

Table 13: Grab/release tool skill

Parameter	Description
Grab	True for tool grab and False for release.
Arm	Desired robotic arm to use.

2.1.2 Automotive Pilot

For the automotive pilot only one case is defined, screwing while moving:

- **Screwing while moving:**

In this case, the robot ahead to the screwing station, there docks with the conveyor where the engine is and finally screw the screws on it. To complete this scenario the following skills have been developed:

- **Navigate:**

The robotic platform requires being able to navigate into an appropriate location. The explanation of this skill is previously explained in fan cowl transportation use case.

- **Mobile visual servoing:**

The screwing visual servoing skill allows screwing the provided screws on the motor for required parts installation on it. This skill uses the visual servoing techniques developed on T2.3.

Table 14: Mobile visual servoing skill

Parameter	Description
Id	Identifier for traceability.
Arm	Desired robot group to use.
Tool	Desired tool to be used.
Flange_H_tcp	Pose of the tool in the robot flange.
Mass	Mass of the tool.
Cog	Center of gravity of the tool.
Use_stored	Use the previously stored information for the camera.
Camera_H_marker	Desired marker pose for the camera (optional).
Marker_H_manipulation_dest	Desired position of the arm to perform screwing (optional).
Z_manipulation	Approach distance of the tool.
Manipulation_time	Desired timespan to complete approximation (sec).
Timeout	Max allowed time to complete approximation (sec).

2.1.3 Auxiliary skills

These skills are created in order to supply an easy way to make the tasks more accessible. They are used offline and cannot add in the main process but can be used during the process creation:

- **Teaching skill:**

This skill provides an easy usable way to help in the creation of new trajectories of the arms groups and the usage of the tools provided by the robot.

Table 15: Teaching skill

Parameter	Description
Arm	Desired robotic arm.
Tool_id	Desired tool to be used.
Cam_id	Camera to be used.
Estimate_pose	Current estimate pose.
Estimate_pose_service	Name of the service to get estimate pose.
Joystick	Usage of the joystick.

2.2 Advanced behaviour management

One of the most relevant achievements is the generation of a simplified and functional system that allows the creation of robotic applications in an easy way. For this, an intuitive and accessible GUI and advanced backend that feeds this GUI with the available required information have been created. The skills are presented to the users through this human-machine interface to be used.

In Figure 1, is presented the sequence for easy robot programming process in ODIN. That follows the next steps:

1. Drag and drop skills for sequencing the desired operations.
2. Parametrize skills through the Skill Parametrization Wizard.
    a. The parametrization can be done inserting raw values directly.
    b. Some skills can be parametrized through Skill Teaching Assistant the skills that require an onsite interactive teaching process (described in Section 3.3).
    c. Also, taking the current status of the robot.

3. Generate executable files with the configured process.

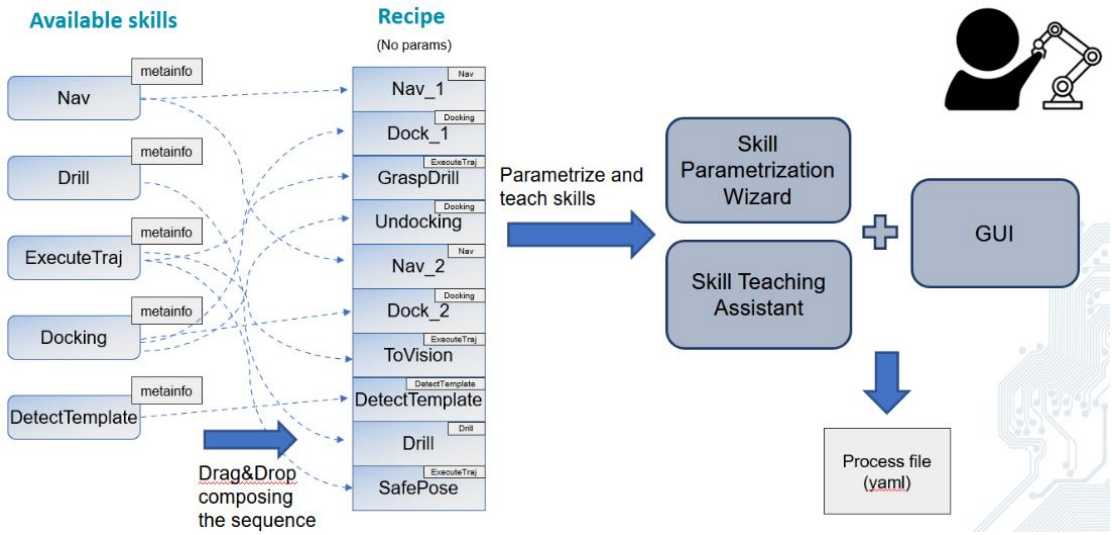

Figure 1: Conceptual design for easy programming process

Based on the Blockly concept, it has been implemented an easy and usable interface to generate a Drag and Drop styled menu. The available skills are automatically recovered from the data stored in the metainformation files where the robot capabilities are defined. In the following Figure 2 can be seen an example of the representation of this menu in the GUI to the user.

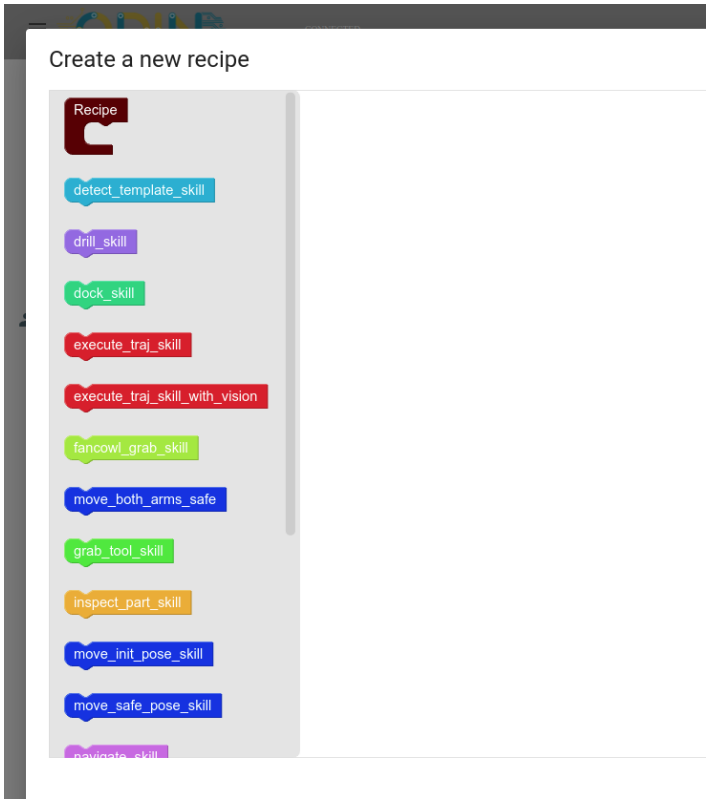

Figure 2: Blockly based GUI

The metainformation developed in the ODIN project contain the following fields:

- Spec: The skill specification, i.e., the type or the location of the skill executable code.
- Params: The parameters of the skills. Contains the type, source, default values, and information.

- Result: The result of the skill.
- Pre-conditions: Conditions that must be met before starting the execution.
- Hold-conditions: Conditions that must be met during the execution.
- Post-conditions: Effects that the skills could produce.
- Teaching assistant: The information related to skills that can use Skill Teaching Assistant.

The metainformation is stored in a YAML file and each skill needs to have a corresponding metainfo file to be used by the system. Depends on the skill type the metainfo will have different specs. The types are skill, action and service. For the params definition, the same structure is used in all the cases. At Figure 3, Figure 4 and Figure 5 an example of each type can be seen.

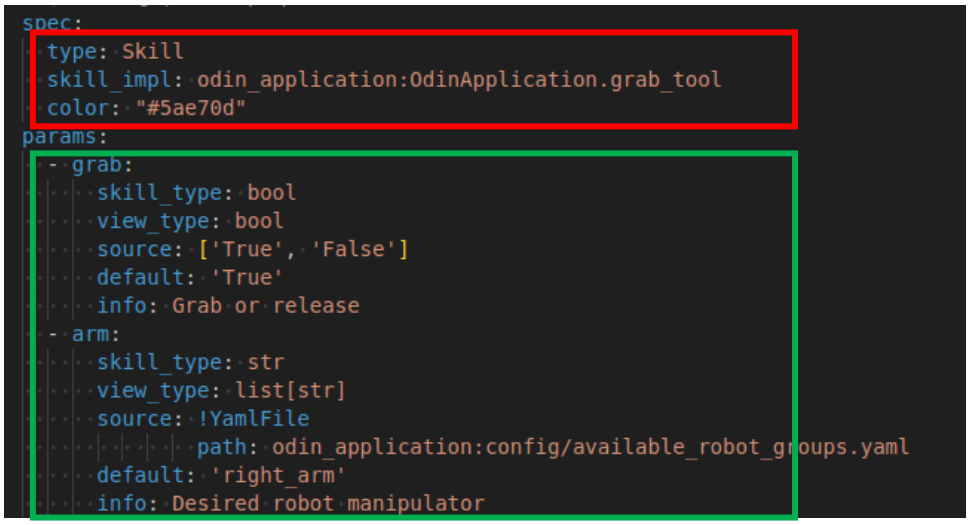

Figure 3: Metainformation file for the skill case

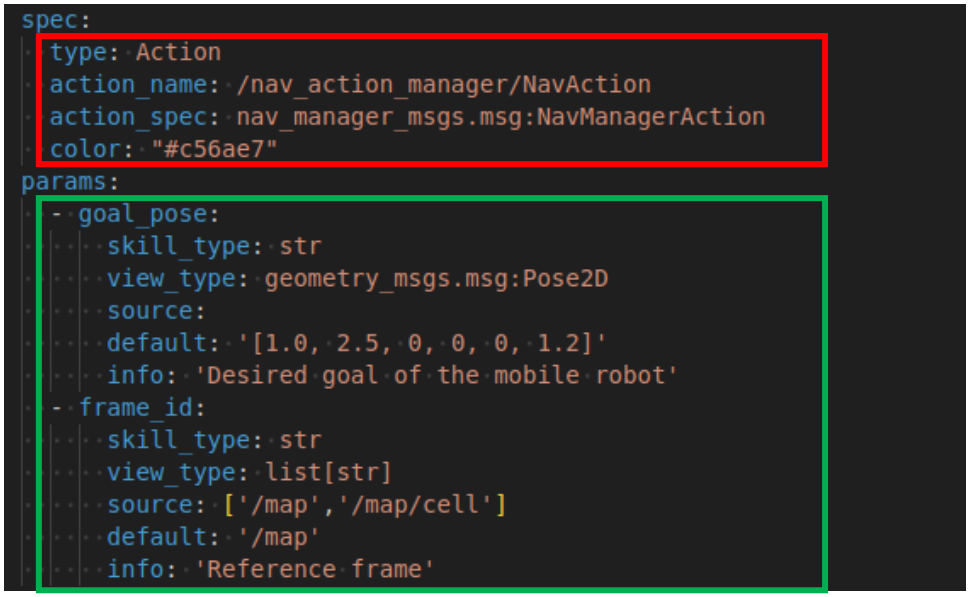

Figure 4: Metainformation file for the action case

```yaml
spec:
  type: Service
  service_name: /iiwa_state_recorder/execute_trajectory_with_vision
  service_spec: iiwa_state_recorder_msgs.srv:ExecuteTrajectoryWithVision
  color: "#d32222"
params:
  - traj_id:
      skill_type: str
      view_type: str
      source:
      default: '1'
      info: 'Trajectory id. This trajectory must exist in the database'
  - tag:
      skill_type: str
      view_type: list[str]
      source: ['move_to_vision_pose_1','pick_part_traj_1','pick_part_traj_2','place_part_traj_1',
      default:
      info: 'Trajectory tag. This tag could exist in the database'
  - arm:
      skill_type: str
      view_type: list[str]
      source: !YamlFile
          path: odin_application:config/available_robot_groups.yaml
      default: 'right_arm'
      info: Desired robot group to use
```

Figure 5: Metainformation file for the service case

After the application template has been generated, the user can open it in order to create a new process. In this step, the Blockly menu shows the full sequence at a glance, and there the user selects the different skills to parametrize. Only when the full parametrization is completed, the application can be saved to be used later. Every time where a skill is selected, one popup window shows the required parametrization data provided with the information contained in the metainfo YAML files. The next Figure 6 shows a representation of this popup.

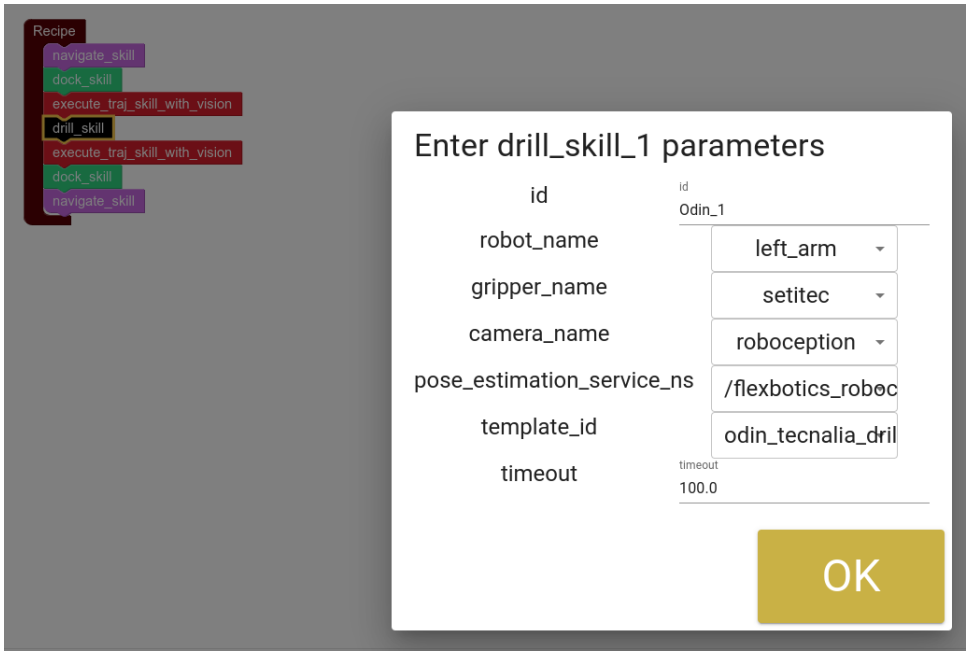

Figure 6: Skill parametrization based on metainformation

One of the claimed innovations is the concept of "source" of the parameters. This block allows feeding the interface from different sources of information with a simple syntax. The system allows the following sources of information that area available:

- Fixed lists of elements (implemented on M24)
- ROS [2] TF queries + filters (implemented on M24)
- ROS Topic queries + filters (implemented on M24)

- ROS Service queries. (implemented on M24)
- YAML files. (implemented on M24)
- Result values of previously executed skills (implemented in the last release)
- Real time usage of the Skill Teaching Assistant (implemented in the last release)
- Current status of the robot (implemented in the last release)

2.3 On-site Interactive Skill Programming system

Some complex operations cannot be easily implemented using SBP approaches. For filling this gap an On-site Interactive Skill Programming system has been developed in ODIN project. Some examples of these operations are the precise movements in constrained spaces, an intuitive pose recording, large part manipulation, etc. The OISP allows taking advantage of human dexterity, or by using a 6D joystick in case of dangerous environment, to perform the best guidance of the robot (Figure 7).

Figure 7: Dexterous trajectory teaching through manual guidance or 6D joystick

Through the GUI, the operator is able to select the arm, the tool, a camera and if the joystick is used in order to start the teaching operation using a simple and easy to use interface (Figure 8).

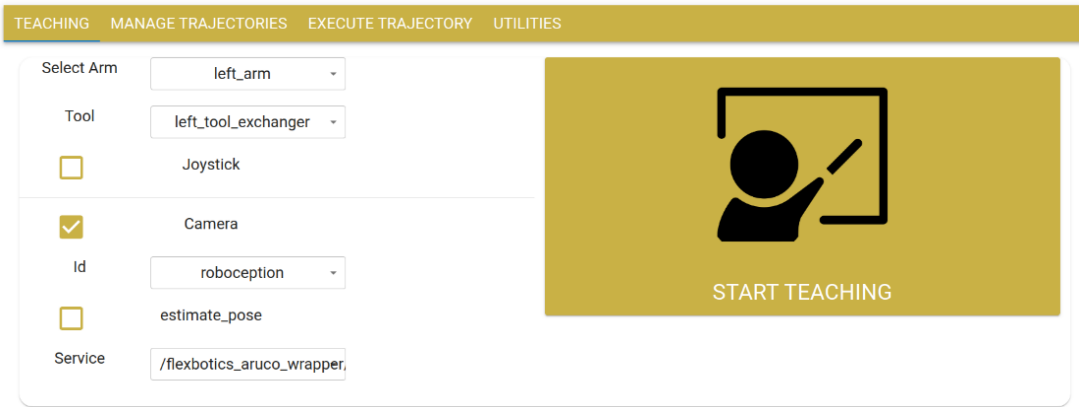

Figure 8: Start Teaching Assistant window

Relevant poses may be added during the teaching of new robot trajectories. These relevant poses add functionalities to the trajectory. Through the "Add relevant pose" button, the system will ask the typology of the relevant pose and it will be stored with the trajectory in a database (Figure 9).

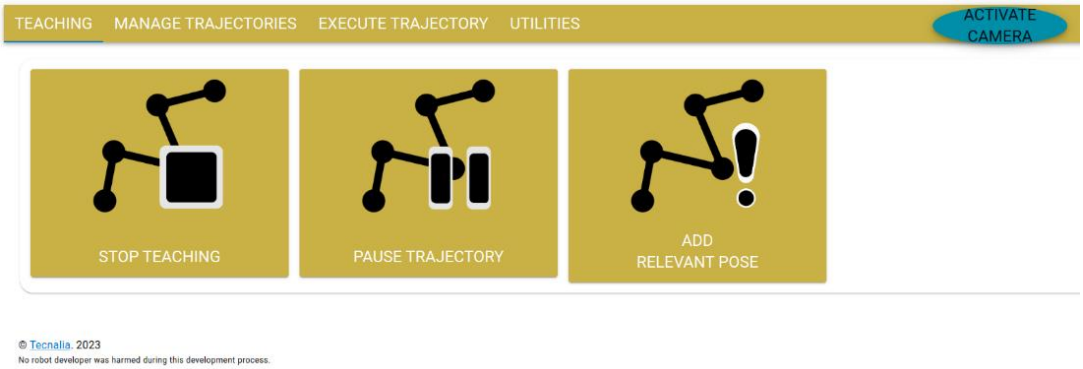

Figure 9: GUI for adding relevant poses in the taught trajectory

To create a non-positional constrained trajectory, is possible add an external marker to it. With this, the trajectory is stored as relative to the marker, avoiding problems related with the correct positioning of the robot in the workspace, allowing the appropriate reproduction of the trajectory in all the cases. Additionally, Figure 10 visualizes an example set of relevant poses that can be added for other scenarios. These relevant poses are directly related with the features of the robot.

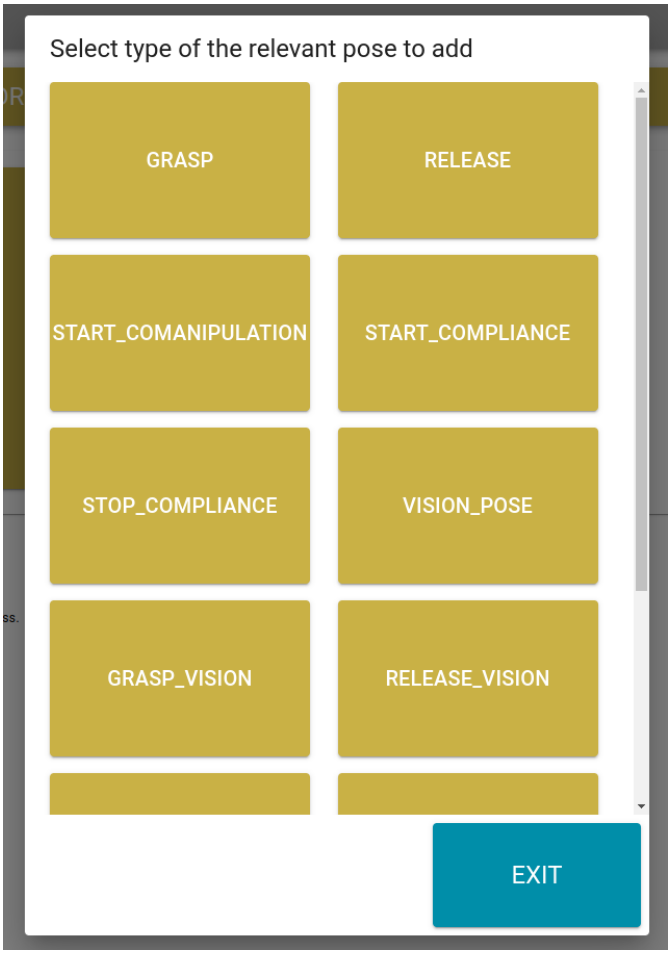

Figure 10: Available relevant poses

After the desired trajectory has been taught, the operator can visualize it in the Manage Trajectories tab (Figure 11). Here the operator it's able to select which information wants to be shown for the selected trajectory, that allows to view all the information part by part and considering the different possibilities available during the trajectory creation.

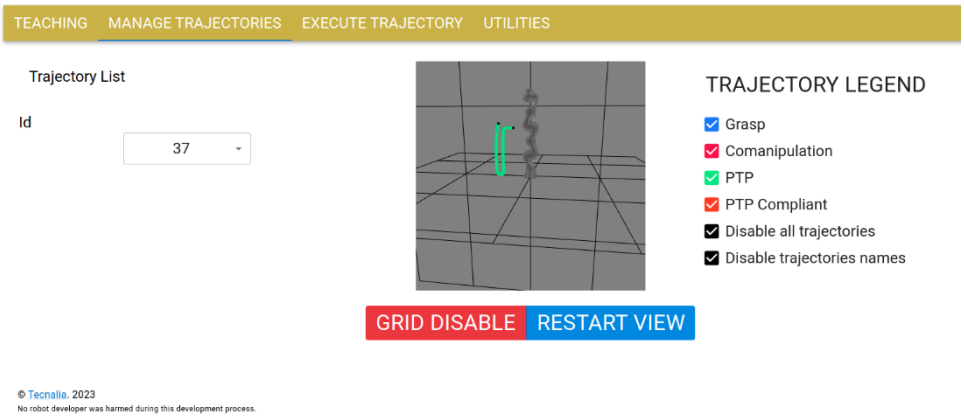

Figure 11: Taught trajectory visualization.

Also, another tab called Execute Trajectory exists, used to test the taught trajectory with the real robot in the conditions that operator can define, generating a very flexible environment for testing and development. For example, is possible to change the group and the tool, so even if the taught is realized with the left arm the execution can be performed with the right arm or the two at the same time. In the same way, in case of a camera is required, this is not limited to original hardware and the associated service to it. Figure 12 presents the aspect of this tab.

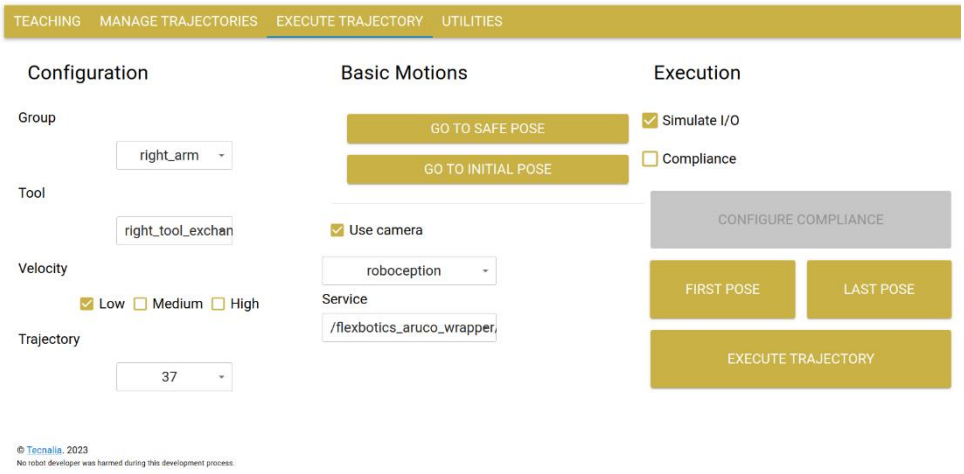

Figure 12: Taught trajectory execution.

2.4 CAD Programming

As mentioned in the introduction of the current section, more detailed information can be found in previous deliverables. However, as this is the last deliverable of WP2, a brief summary of the features of the CAD programming module developed in the ODIN project is presented.

CAD Programming module is a software in continuous development, principally throughout various European projects. Its main differential value is the concept of taking advantage of CAD information of a 3D scene for integrating with robotic skills. Thanks to a simple GUI, which provide a set of pre-programmed skills, robot programming sequences can be easily parametrized and sent to the robot, ready to execute (Figure 13).

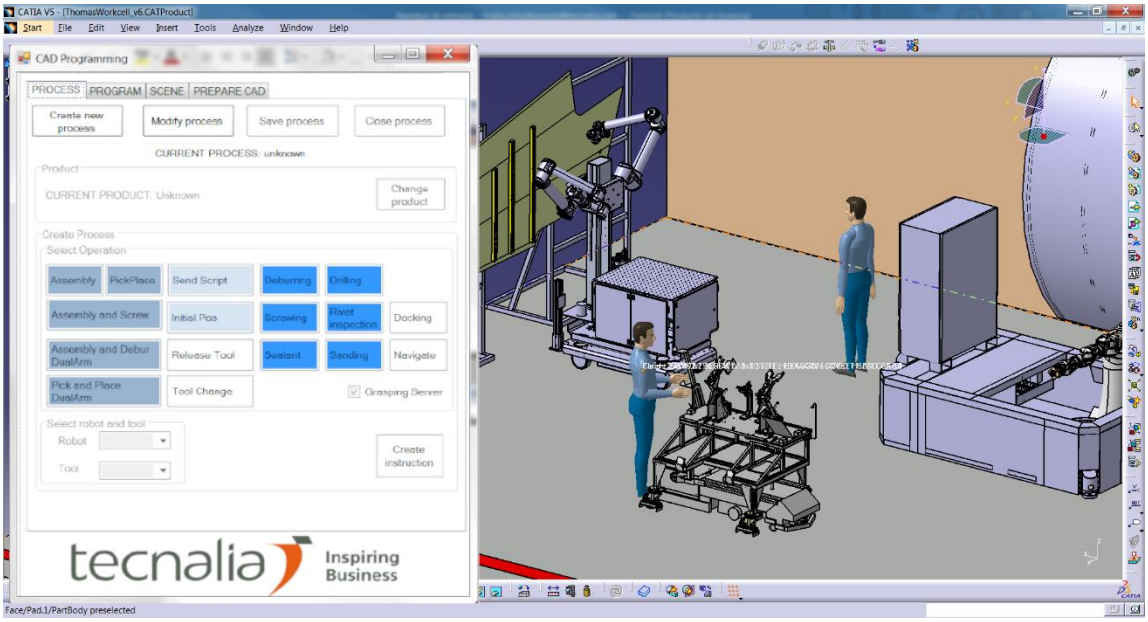

Figure 13: CAD Programming module

Along ODIN project the following features has been implemented:

- Skill categorization depending on the parameter type. Different types of skills (no parameters, Catia frames, Catia object, string parameters, etc.) have been defined.
- Adding dynamically new skill buttons. Thanks to skill categorization, new skills can be added to the GUI without further development. The new skills are defined at XML files, which contain the skill description (parameters, target skill specification, etc.) and allow mapping the skills presented in the GUI and the skill library.
- Automated scene export to Gazebo compatible format (.world).

3. FINAL PROTOTYPE OF HUMAN SIDE INTERFACES

3.1 ODIN Augmented Reality (AR) Application

3.1.1 AR application features

The Augmented Reality application of ODIN has been designed and developed in order to assist human operators in different industrial applications in a multi-pillar way [3]. For this reason, the specific application consists of different interfaces enabling the intuitive interaction of human operators with the production line through the main orchestrator framework (OpenFlow) [4].
Based on the already submitted deliverable D2.3, ODIN AR application consists of several features in terms of:

- Assembly guidance provision.
- Controlling of mobile and static robots.
- Interaction with robot gripping tools.
- Production system security breach notifications provision.
- Assembled products quality inspection results provision.
- Resilience of the production system in case of unexpected events.
- Safety awareness for the operators.
- Interaction with mechatronic devices.

In addition to these features, a connection between the AI-based decision making module and the AR application of ODIN has been accomplished. The operators equipped with the AR headset are able to check the production schedule generated by the AI Task Planner inside the virtual world of the AR application.

Table 16: ODIN AR features and implemented pilots

Feature	Title	Automotive Pilot	White Goods Pilot
AR_F1	Visualization of Human assigned tasks through AR	X	X
AR_F2	Task Completed virtual button	X	X
AR_F3	Robot trajectory visualization inside the virtual world	X	X
AR_F4	Easy robot programming using interactive virtual tool	X	X
AR_F5	Security alarm visualization	X	X
AR_F6	Error handling in case of detection process failure	X	X
AR_F7	Resilience – execution error recovery	X	X
AR_F8	Robot's safety zones visualization through AR Glasses	X	
AR_F9	Quality inspection results visualization, digitally mark for validation and instructions provision for corrections	X	
AR_F10	Alarm indicating that the robot is working in collaborative area		X
AR_F11	Interaction with mechatronic devices		X
AR_F12	Production schedule visualization	X	X

- AR_F1: Visualization of Human assigned tasks through AR and AR_F2: Task Completed virtual button

This functionality of these two features is focused on the interaction of human operators with OpenFlow for human-assigned tasks execution. The initial version of this feature implementation has already been presented in deliverables D2.2 and D2.3. In its final version, the ODIN AR application consists of the required interfaces for its connection with OpenFlow module towards the visualization of all human-assigned tasks in terms of the Automotive use case.

Depending on the assembly tasks assigned to the human operators, different holograms like arrows and screws are visualized inside the virtual world of the AR application guiding the operators towards the correct and fast execution of the assembly steps.

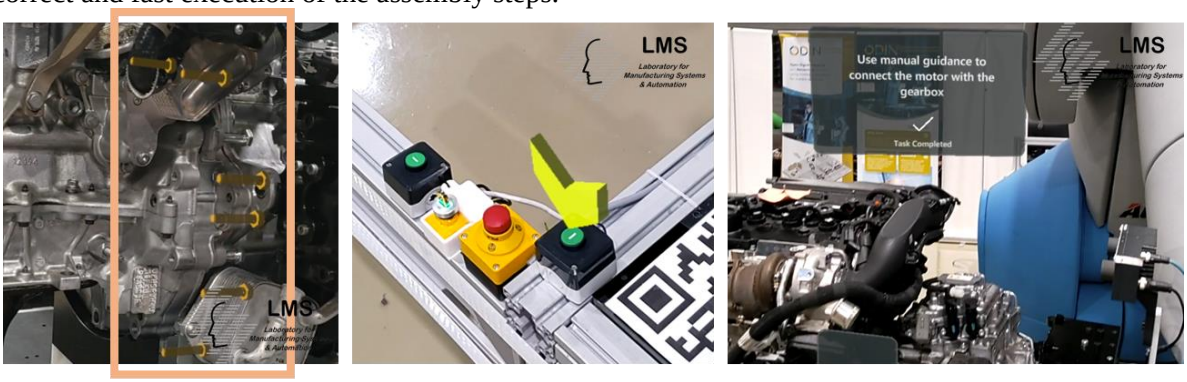

Figure 14: AR hologram-based instructions for assembly tasks' execution

- AR_F3: Robot trajectory visualization inside the virtual world

As already mentioned in deliverable D2.2, this feature increases human awareness about robots' upcoming trajectories through the visualization of robots' holograms. In the final version of the AR application, the library of the supported robots has been updated to include also a) the AURA collaborative robot used under Automotive pilot Operation 1 [5] but also b) the COMAU mobile robot [6,7] utilized in the Automotive pilot Operation 3.

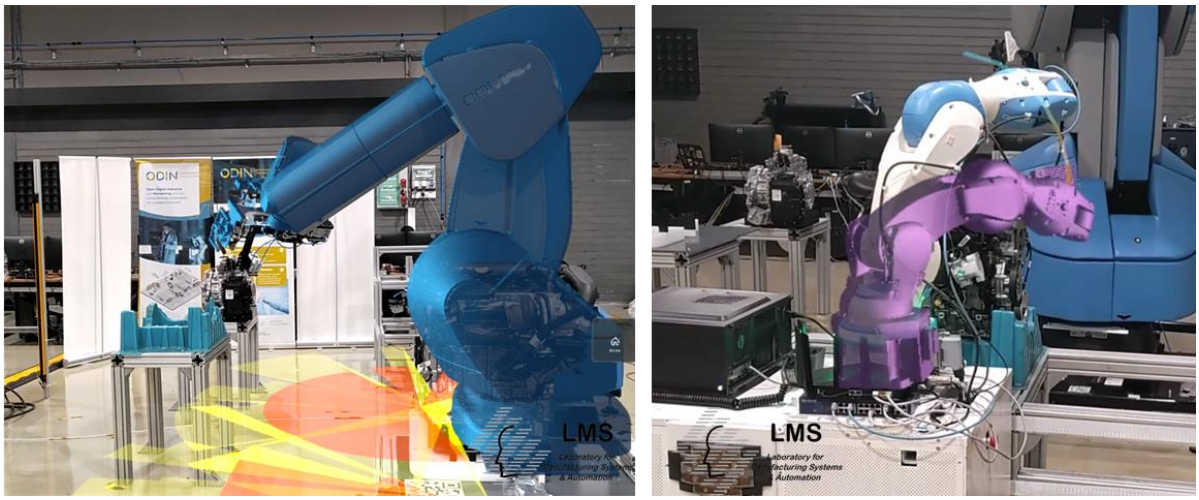

Figure 15: Robot trajectory visualization inside the virtual world

- AR_F4: Easy robot programming using interactive virtual tool

Based on the description provided in deliverable D2.2, through the easy robot programming interface of ODIN AR application, the operators are able to interact with the robots of the White Goods and Automotive pilots and define new poses for them. In the final version of the AR application, this feature has been upgraded to support different robot end effectors connected also with the COMAU controllers for integration with COMAU robots.

Figure 16: Interaction with COMAU robots using the developed AR application

- AR_F5: Security alarm visualization

Thanks to developed AR application's interfaces for its connection with the OpenFlow and the cyber security module, the human operators of the ODIN production lines are aware of cyber security threats detected inside the network of the investigated lines. These threats are handled internally by the cyber security module and the OpenFlow while the human operators are aware of the production line status.

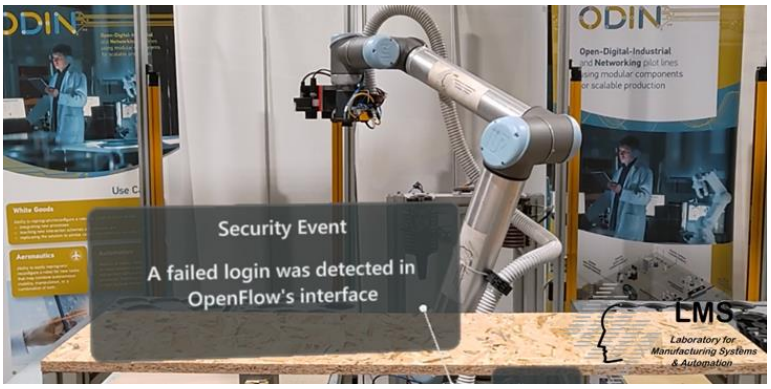

Figure 17: Security alarm visualization inside the virtual world

- AR_F6: Error handling in case of detection process failure

In case of unexpected failure of the process detection algorithm, operators are informed about the status of the production line through textual information visualized in the virtual world of the AR headset [8]. The operator is able to change the pose of the robot using the AR_F4 of the application and request the re-execution of the previously failed detection action from the OpenFlow using a virtual interactive button.

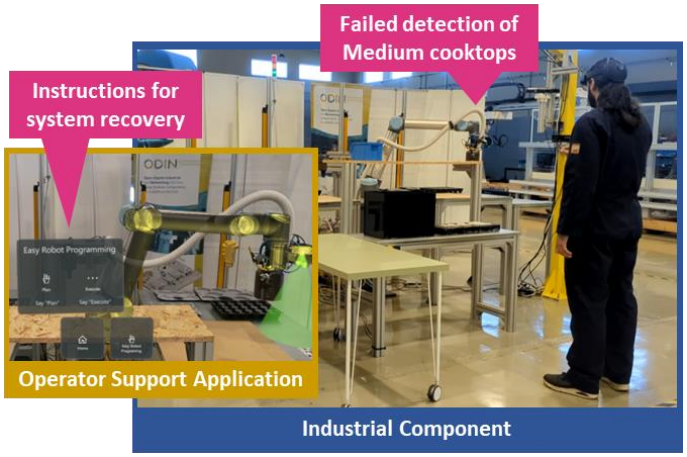

Figure 18: Error handling in case of detection process failure procedure

- AR_F7: Resilience – execution error recovery

The AR application enables the recovery of the system in case of different unexpected failures, safety zones violation, robot's failure to execute a specific motion or failed execution of a human task. The OpenFlow is monitoring the execution of the different tasks and through a set of recovery functions developed inside the AR application, the users of the application are able to interact with OpenFlow and request the recovery of the system on-the-fly through the usage of interactive virtual buttons.

Figure 19: Interaction with virtual objects for system's recovery

- AR_F8: Robot's safety zones visualization through AR Glasses

In order to increase human safety in the pilot lines of ODIN, the installation of several safety devices has been investigated. One of the most important integrated safety components are the laser scanner sensors able to adapt the detected safety areas on-the-fly based on the executed task but also on the speed of the robot according to related safety regulations. When the operator infringes the protective zone of a laser scanner, the OpenFlow state changes to "Emergency mode" and it waits for operator's distancing from the detected sensor's volume. In this case, it is very important for the operators' to be aware of the safety zones' shape and position to avoid robots' stop but also system's recovery in case of infringement. Using the real-time sensor data from the scanners, operators equipped with an AR headset are able to check all the safety zones inside virtual world of the AR application as presented in the following figure.

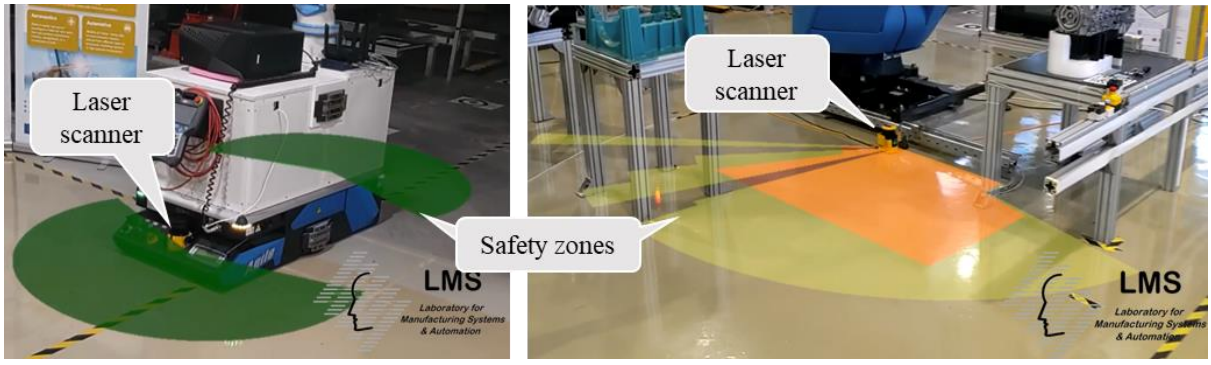

Figure 20: Safety zones' visualization inside the virtual world of ODIN AR application

- AR_F9: Quality inspection results visualization, digitally mark for validation and instructions provision for corrections

A human operator is involved in Automotive pilot operation 3 for the execution of corrective actions based on the output of the quality inspection task of the COMAU mobile robot. The operator is informed

about the fault inspection results through the AR application in the form of a list (Figure 21). Each element of the list is matched with one inspection issue detected on the motor assembly and instructions for corrective actions are provided to human operators using virtual holograms of bolts, connectors and valves of the motor. After each corrective action's execution, the operator is able to inform the OpenFlow regarding his/her action using virtual buttons of the corrective actions' interactive list.

a) b)

Figure 21: a) List of quality inspection faults and b) hologram-based instructions for inspection corrections

- AR_F10: Alarm indicating that the robot is working in collaborative area

A common characteristic of Human-Robot Collaborative workstations is the shared working areas for human operators and robotic manipulators. Based on the safety concept of ODIN pilots and the risk assessment for the White Goods pilot case, a shared working area has been defined in the layout of this pilot. Inside this area, human operators and robots are working together for the execution of specific assembly tasks. When the UR10 robot of this pilot enters these areas, corresponding alarms are sent to the operators through the AR application to inform the operator about robot's presence inside the shared areas.

Figure 22: Alarm indicating that the robot is working in collaborative area

- AR_F11: Interaction with mechatronic devices

Another critical feature of the AR application is the interaction between the operator and the robot's gripping tools. This feature is highly beneficial during assembly operations when robots need to pick up various assembly parts and provide them directly to the hand of the operator. When the operator places one of his/her hand under the gripping tools, a stopwatch of specific seconds appears in the virtual world of the AR framework (Figure 23). At the end of this duration, the gripping tool is disabled,

and the manipulated object is released into the hand of the operator. If the operator removes his/her hand from the position under the gripping tool before the end of the stopwatch, the disabling of the tool is canceled, and the stopwatch goes to zero value. This feature enables the operator to easily take parts from the robot and continue the assembly process.

Figure 23: Interaction with robot's gripping tools through the AR framework

- AR_F12: Production schedule visualization

Through the AR application of ODIN, human operators are aware of the production schedule plan. This task plan consists of the assembly tasks to be performed but also resources' (robots or human operators) responsibilities per task. The list of the assembly task plan is available inside the AR application thanks to application's connection with OpenFlow and ODIN AI Task Planner.

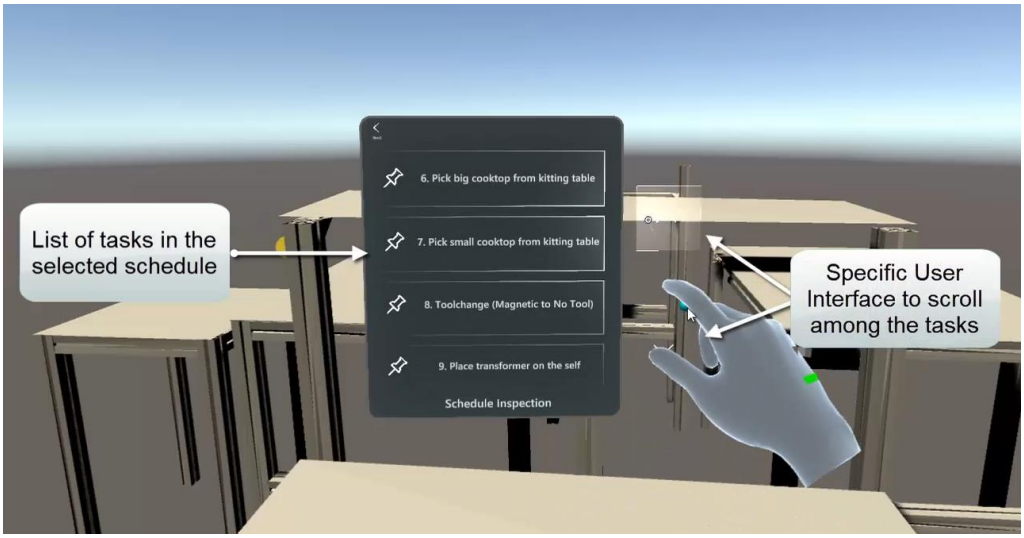

Figure 24: List of assembly tasks

3.1.2 AR application integration with OpenFlow

For the realization of previously mentioned functionalities of the AR application, a direct connection of the application with the main orchestrator of the production line (OpenFlow) is required. As presented in deliverable D4.2, OpenFlow is directly connected with all the software modules of ODIN. AR application use various type of data derived from the OpenFlow and ODIN modules to achieve the required Human-Robot interaction in the investigated pilots. The AR application is indirectly connected with the following modules of ODIN through the OpenFlow:

- Reconfigurable robot tooling – Interfaces for connection with the robot grippers is required in terms of AR_F11.
- Quality inspection module - Interfaces for connection with the quality inspection module are required in terms of AR_F9.
- Safety module - Interfaces for connection with the safety PLC and the safety module of ODIN are required in terms of AR_F10.

- Cybersecurity module - Interfaces for connection with the cyber security module are required in terms of AR_F5.
- AI Task Planner - Interfaces for connection with the AI Task Planner of ODIN are required in terms of AR_F12.

3.2 ODIN Smart User Interfaces

3.2.1 Projector HMI

The goal of the smart user interface module is to provide a visual interaction between the operator and the robot by using projector-based solution. An interface is displayed on a surface (table, floor, on product) with elements that can be clicked to trigger the robot or display instructions for the operator. Or there can be various annotations e.g. pick this object or place it into this position.

In addition, there is a possibility for the operator to trigger the drawing of a safety border around the robot to ensure safety. In case of border violation, the robot can be stopped. Safety zones from safety devices, such as laser scanners or safety light curtains, can be projected and made visible for the operator.

A functionality is implemented for static borders for the robot. These are used for monitoring certain smaller areas on the cooperative workspace shared by the robot and the operator. These smaller areas are used for placing objects within them by the robot/operator and the operator/robot can pick up them later from these slots. The static borders are also monitored from operator's violation in temporal fashion, in case should this happen at prohibited moment, the robot is stopped. A larger SW module manages the interactions between the static borders, the robot, and the operator via a booking system. This is demonstrated in the white goods use case.

3.2.1.1 Calibrations

In order to display the user interface on a surface, a calibration has to be performed to generate homographs that will avoid images distortion. Several different calibrations are needed:

1. Calibration of projector(s) and camera(s)
2. Calibration of positions and homographs between the robot and the camera(s) frames.
3. Calibration of camera with robot for global depth map generation.
4. Calibration of camera and projectors to display smart interface and static borders.

Here, ArUco library is being used in order to calibrate the surface of user interface projection and the cameras. A sample of this can be seen under Figure 25.

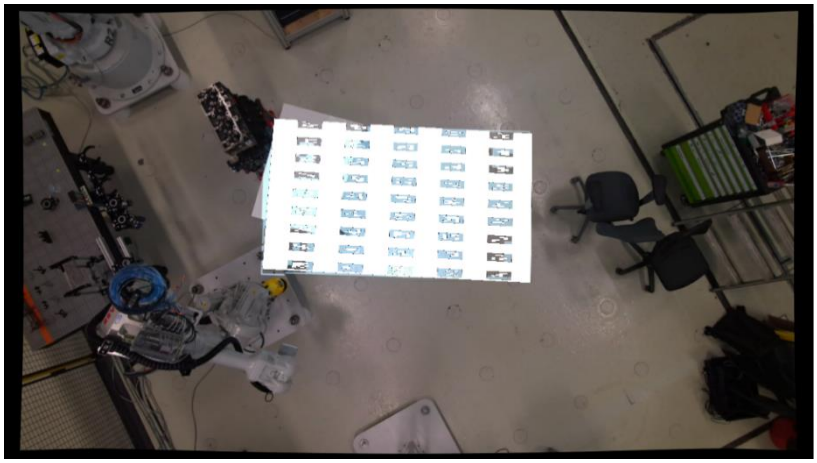

Figure 25: Calibration of the user interface projection with projected ArUco markers

Then, the robot alignment with the camera is being performed for the detection of interactions. Two alternative methods can be used: a) a green light mounted on robot's end effector and that is filtered through computer vision (illustrated in Figure 27). b) ArUco marker [9] mounted on robot's end effector (illustrated in Figure 28: Calibration of the robot and camera coordinate systems with ArUco marker at TAU Robolab.. In both alternatives, several positions for the light/marker are necessary to capture in order to perform an accurate calibration.

The calibration between the camera and the robot is done through a wizard to generate a global scene (illustrated in Figure 26).

Figure 26: Calibration of the robot and camera coordinate systems.

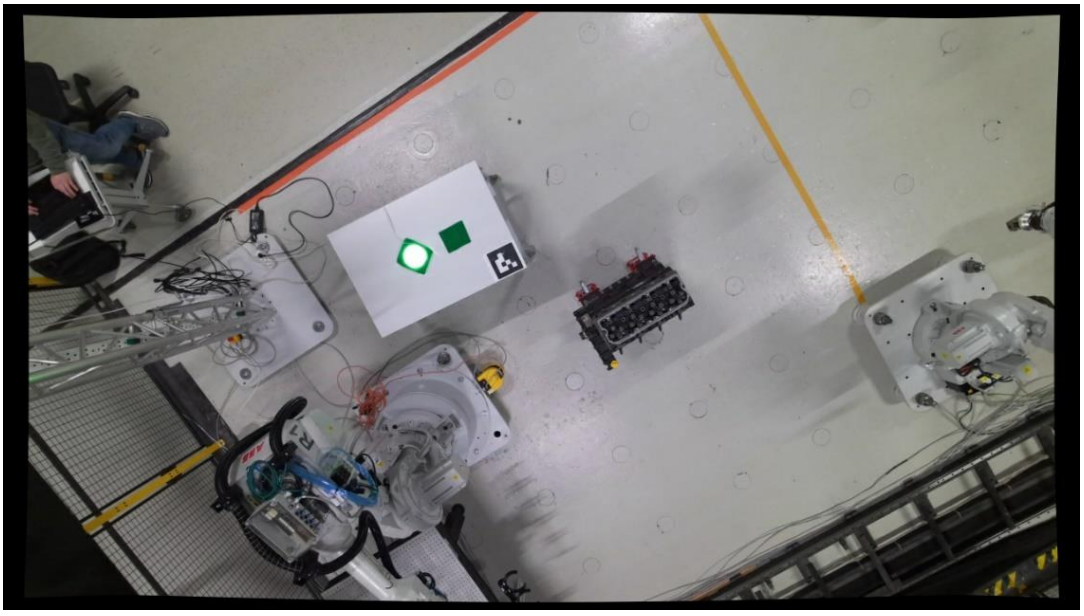

Figure 27: Calibration of the robot and camera coordinate systems with green illuminator or target at TAU's HRC small-scale pilot.

Figure 28: Calibration of the robot and camera coordinate systems with ArUco marker at TAU Robolab.

The final step is to calibrate the display zone where it is possible to display the smart user interface. The application can have multiple of such display zones. The calibration needs the four (4) corners of the display zone to be marked with specific ArUco markers. Physical ArUco markers are placed on the surface determine the display zone corner locations (illustrated in Figure 29).

Figure 29: Marking the four corners of the display zone.

Then, the same method is utilized to determine where the mobile smart user-interface and the static borders will be displayed, still with the help of ArUco markers (Figure 30).

Figure 30: Left – Marking the corners for display zone on a mobile table. Right – Defining and calibrating the table for static borders.

3.2.1.2 Mobile Table Interface

Once the calibrations are complete, the user interface can be projected on the selected movable surface with the help of an ArUco marker. The marker supports a dynamic projection on the table, meaning that the projection location is updated if the operator moves the table in the industrial environment. Naturally also static interfaces are available to be projected, not only the dynamic ones.

Figure 31: Mobile user interface following the operator's table

3.2.1.3 Dynamic Safety Border

Regarding the operator's safety, the projection of the dynamic safety border around the robot is computed according to various joint angles from the robot (tool, flange, base, and certain links). With these values, a hull is generated that will be projected around the robot on the floor in real time (Figure 32). When the robot is in motion, the dynamic border is updated so the operator knows when not to cross the robot's border. Under Deliverable document D2.4, it is also represented a detection of border violation events, in case user or an unknown object violates the drawn dynamic safety border. This will trigger the stop for the robot.

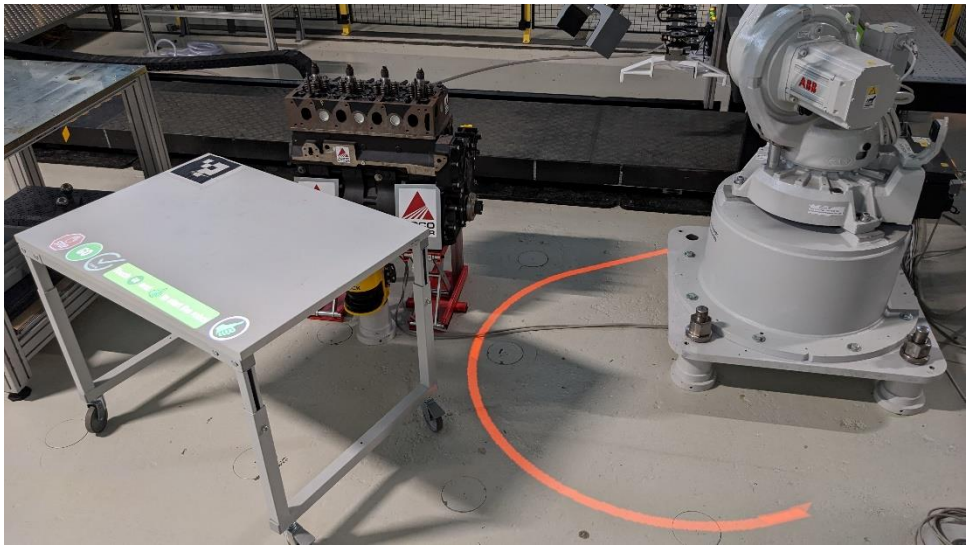

Figure 32: Projection of the safety border around the robot

3.2.1.4 Static Borders and Slots

Besides the display of a dynamic border around the robot, it is also possible to display static borders to help the robot and the operator to collaborate. The static borders are a rectangle drawn on a surface where the robot will place object and the operator is in charge to pick them up, or wise versa. A booking system is implemented to control and monitor the state of multiple static borders i.e. slots.

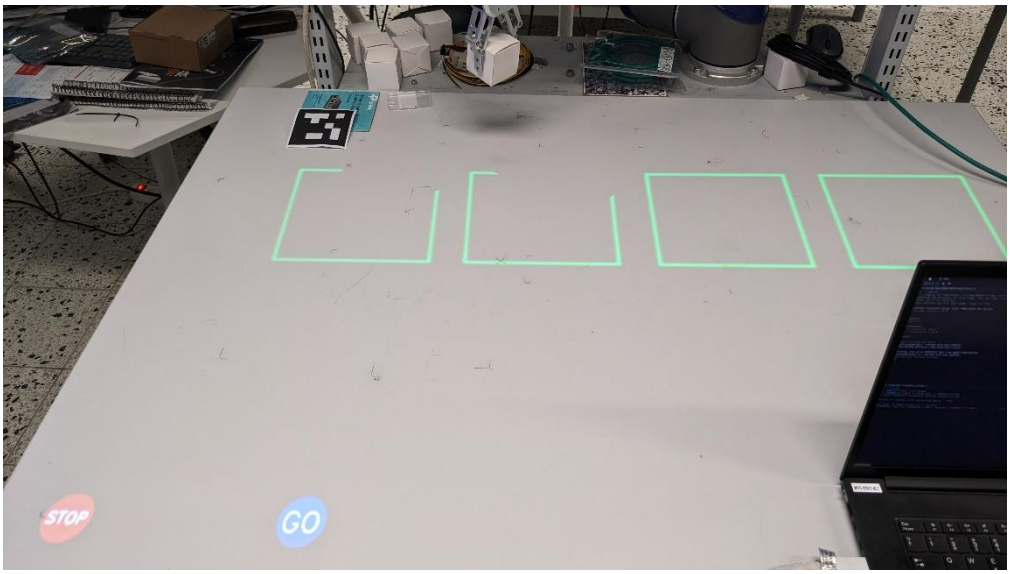

Figure 33: Display of borders before any booking. The buttons Stop and Go can be hit by the operator to start or stop the robot.

To ease the interactions, a booking system was developed where the robot books a border, signifying the operator that the robot is going to place an object within this one and thus no crossing should be allowed. During the booking, it is also possible to book adjacent borders if the robot is large enough to cross them. It is advisable to book the slots next to the target slot for operator's safety reasons (Figure 34). Once the robot placed the object to a slot, the system releases the booking of this slot.

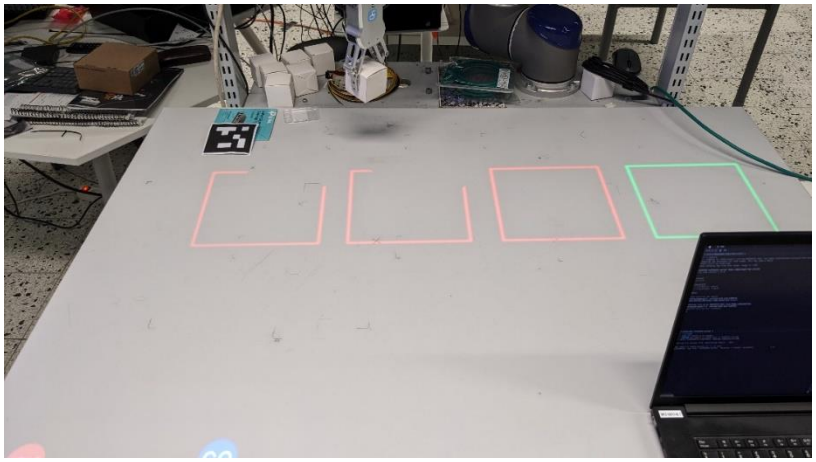

Figure 34: The second slot from the left is booked along with the adjacent borders

Figure 34 illustrates the situation, where the robot is going to place an object on the second slot from left. Thus, it is booked along with the adjacent slots by the robot, and this is marked with red coloured borders. The green border signifies the availability of this one for the operator.
Once the robot has placed the object, the system releases the booking of this slot and the adjacent ones. Action is followed by highlighting it with a different colour so the operator knows the object is ready for pickup, or even more strongly – operator should pick next the object inside this border. These two phases are illustrated in Figure 35.

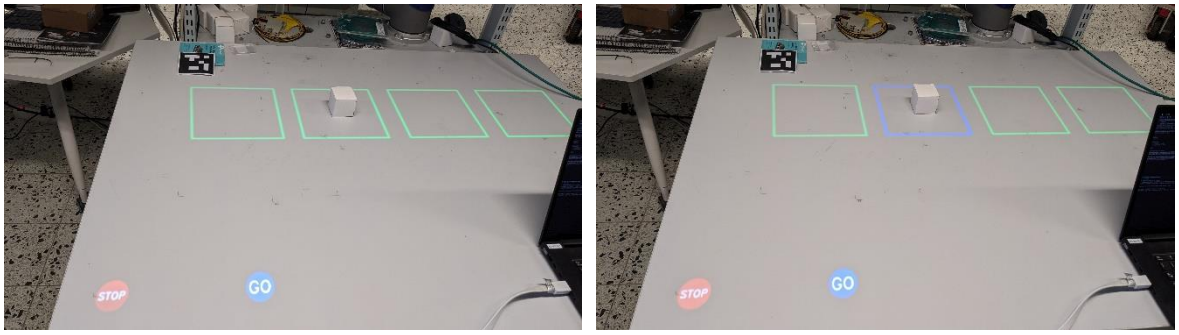

Figure 35: Left - The second border from the left is released along with the other ones. Right – the system highlights the border occupied by an object so the operator can pick up the object.

3.2.1.5 Projecting Safety Zone Violations

Apart from projecting borders, the system is also able to display safety zone on the floor associated with safety system e.g. a laser scanner installed next to the robot. In this case, the safety system is controlling the environment in real time, and according to the safety standards and specifications. In a large industrial environment where the operator can move freely around a robot, it is necessary to indicate safety zones to the operator in complement to the dynamic border drawn around the robot, because the monitored safety zones are invisible for the human.

To enable this feature, the laser scanner was calibrated to recognize obstacles and classify them according to their distance. Then, if the operator is far enough from the robot, a safe zone or its boundary will be drawn on the floor. This zone indicates where the operator can move freely. A second zone closer to the laser sensor will indicate a warning area, advising the operator to go back from the safety zone. Finally, the closest area to the sensor displays a danger zone, clearly indicating there is a risk for the operator. If the operator step in this area, the system will throw a violation event and stop the robot. This feature is solely for operator information and enabling smooth operation of the production environment. All safety related controls including sensors and actuations are made with safety rated devices, in which the projector interface does not have any effect.

3.3 Virtual Reality safety training

The virtual reality safety training, integrated into the ODIN project, is a carefully crafted tool intended to support human operators in diverse industrial environments, including those in the White Goods use case. This training system, while innovative, is an evolving solution, reflecting a moderate approach in its application and effectiveness. The VR-based safety training is structured into two distinct phases. In the first phase, the user will be introduced to all the safety measures, gaining a comprehensive understanding of the communication protocols between the user and the robot, as well as the consequences of violating any of these safety measures. This phase is crucial for establishing a foundational knowledge of safe operations. Subsequently, in the second phase, the user is required to undergo training focused on assembly procedures. This phase builds upon the knowledge acquired in the first phase, using it as a reference to prevent disruptions during the assembly process. The training in this phase is more hands-on, emphasizing the practical application of safety measures in real-world scenarios. By the end of the training, users will not only be well-versed in theoretical safety knowledge but also adept at applying these principles in practical settings, significantly reducing the risk of accidents, and enhancing overall efficiency in the workplace.

An investigation into the TAU HRC small-scale pilot revealed that the development process for such a VR application is time-consuming, which could undermine the importance and benefits of VR safety training for future use cases. Consequently, the development of the safety training components, as mentioned below, has been oriented towards modularity and reusability in subsequent applications. To this end, a template-based component has been created, which can be utilized in this project to reduce development time. These templates can be easily recreated for any project and integrated into a Unity project using a drag-and-drop method, requiring minimal modifications.

Core Components of the VR Safety Training System consist of:

1. **3D Visualization for Workspace:** The system utilizes dynamic 3D representation of robot workspace to replicate reachability of robot in assembly station. Hence, operators could be aware of regions that are strictly forbidden to enter (Figure 36).

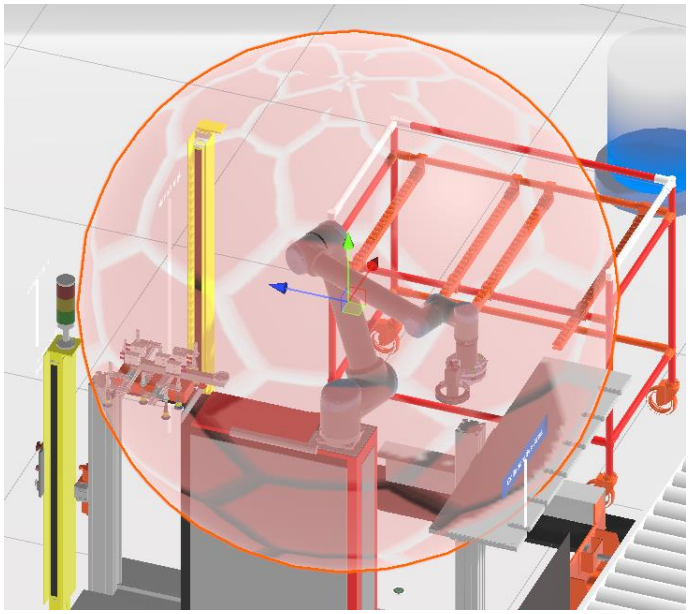

Figure 36: Robot working space

2. **Hazardous area (**Figure 37**):** Equipped with VR headsets, operators are introduced to a virtual assembly station. Here, they can learn about safety measures, and identify potential hazards and required action when violating hazardous areas. With this, operator should learn causes and effects in the system such as impact on robot state after violating the area, and what will happen then and how to avoid triggering such violations. The interactive behaviour of this area is complemented

with audio warnings. These are intended to instruct operator on current system status and to provide steps to avoid these situations. The hazardous areas are based on risk assessment and calculation of safety distances, which are basing on the rules and equations origin from the safety standards. The visual representation is aligned with the selected safety device, such as arc, cube, or volume.

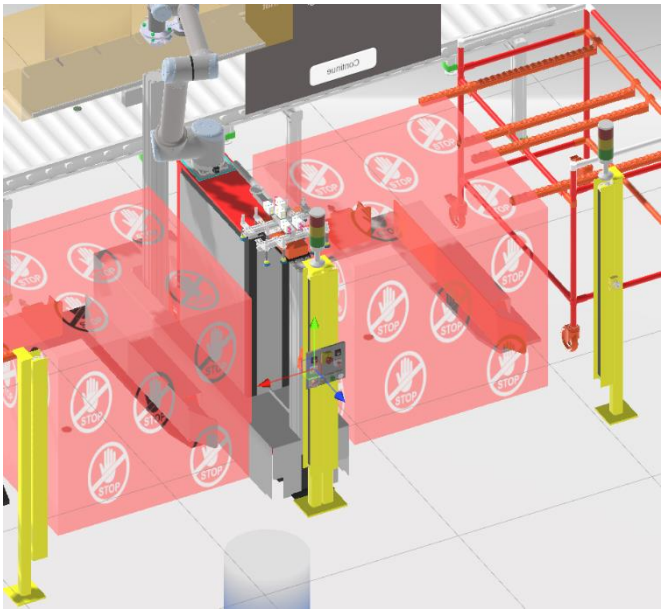

Figure 37: Hazardous area

3. **Assembly Sequence:** The training focuses on educating operators about the correct sequence of assembly tasks and about the standard work. Operators use 3D UI in VR to accomplish assembly tasks while monitoring the robot movements during each stage of assembly. Colour-coded place holders assist operator to understand exact position and orientation of assembled components. The red colour represents that component doesn't yet fit on its place and highlight what is the correct position of component (Figure 38). When component reach to correct position, placeholder colour will change to Green to inform user that it is correct placement.

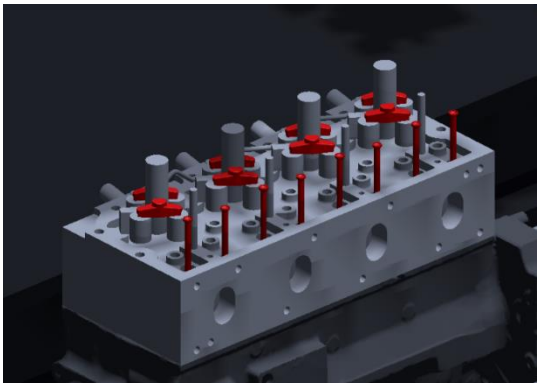

Figure 38: Place holders for assembled components in VR

4. **Interactive Safety Features:** Interactive elements such as simulated light curtains and its control panels are included and programmed. These are designed to mimic real-world safety mechanisms due to triggered safety events. While the operator is performing assembly, safety events might be triggered by the action of the operator such as violating the light curtains or accessing the robot workspace while the robot is operating. This interactive safety feature tends to teach for the operator the strict safety procedures and how to recover from these incidents to bring the system back into operational state.

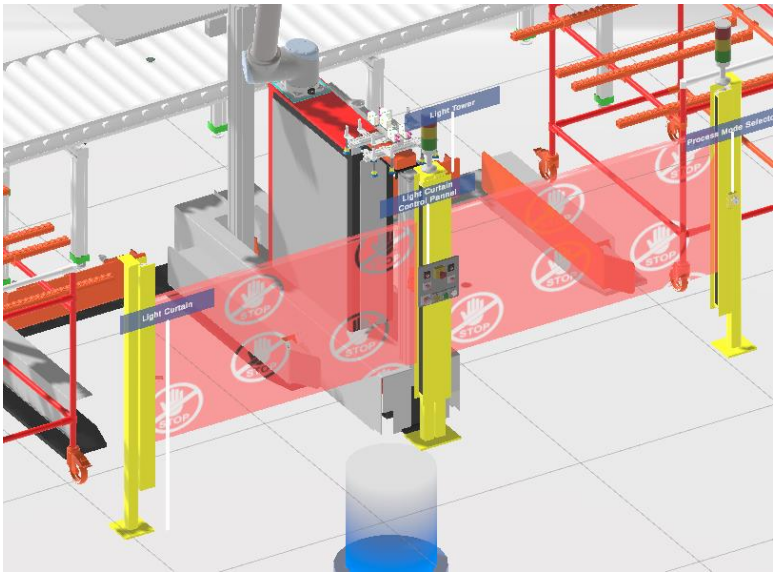

Figure 39: Light curtain interactive functionality based on distance of user to detection area

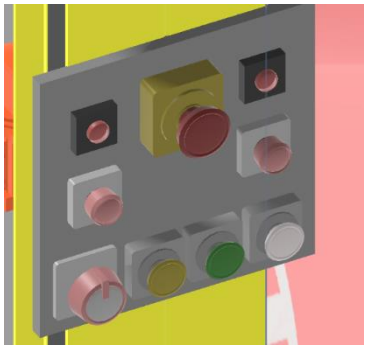

Figure 40: Interactive control panel for activate/deactivate or reset of each light curtain

5. **User Interface (UI) (**Figure 41**):** The system employs a user interface to guide operators through tasks and safety protocols. They are informative panels, containing text and images. The sample UI is tailored to the specifics of White Goods manufacturing.

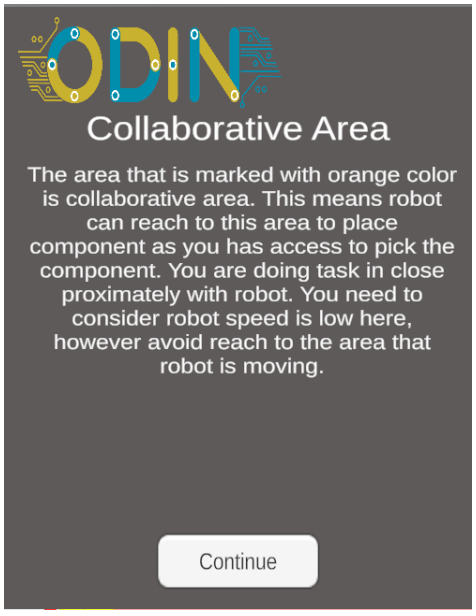

Figure 41: Example of basic UI

6. **Product Assembly Simulation:** A key part of the training involves simulating the product's assembly process. This part consists of whole assembly sequence with addition of visualization of safety measure such as violation of hazardous area. While this simulation aims to mirror actual assembly processes, it serves as an introductory tool rather than a complete substitute for on-the-job training.

Table 17 provides a structured overview of how the virtual reality safety training features are adapted and applied in the context of the ODIN project. They are mainly applied in the White Goods pilot, even though exactly same methods could be applied e.g. in automotive pilot.

Table 17: ODIN VR features and implemented pilots

Feature	Title	White Goods Pilot	Automotive Pilot
3D Workspace Representation	Detailed 3D model of the manufacturing environment, showcasing collaborative robot working space	X	P
Hazardous Area	Marked areas indicating potential hazards specific to production, based on feeding system and robot placement	X	P
Interactive Safety Features	Simulation of a light curtain as a safety mechanism	X	P
Control Panel for Light Curtain	A virtual control panel allowing users to interact with and understand the light curtain's functionality.	X	P
UI Instruction	Customized UI prompts guiding operators through specific tasks and safety protocols	X	P
Assembly Demo	A VR-based demonstration of the assembly process, focusing on assembly of microwave oven and cooktop components	X	
Assembly training	Interactive environment for assembly of two products (transformers, and cook tops) in assembly station	X	

X = implemented, P = Possible to implement

4. CONCLUSIONS

The final prototypes of easy robot programming and human robot interaction modules have been presented in this document. At M36, the presented core enabling technologies are completely developed and tested under the Small-Scale Pilots (D2.6) in order to be ready for being integrated in the Large-Scale pilots. This document together with D2.4 provides the end-users with a range of possibilities to integrate state-of-the-art technologies for mobility, perception, manipulation, easy programming interfaces and human-robot interaction.

The technology development ends here, but throughout WP5 these technologies will be integrated into the Large-Scale pilots of the end users. The final peculiarities and configurations of the modules under the pilot lines, will be documented on D5.5 of M48.

5. GLOSSARY

CAD	Computer Aided Desing
DLP	Digital Light Processing
FC	Fan Cowl
GUI	Graphical User Interface
IO	Input/Output
OISP	Onsite Interactive Skill Programming
ROS	Robot Operating System
SW	Software
UI	User Interface
VR	Virtual Reality

6. REFERENCES

1. Unity Real-Time development platform, https://unity.com/
2. ROS, “ROS - Robot Operating System”, available at www.ros.org
3. S. Aivaliotis, A. Papavasileiou, C. Konstantinou, T. Anastasiou, C. Gkournelos, S. Koukas, Sotiris Makris, “An interactive Augmented Reality based framework assisting operators in human-robot collaborative assembly operations”, 17th CIRP Conference on Intelligent Computation in Manufacturing Engineering (ICME), 12-14 July 2023, Gulf of Naples, Italy
4. S. Koukas, N. Kousi, S. Aivaliotis, G. Michalos, R. Bröchler, S. Makris, "ODIN architecture enabling reconfigurable human – robot based production lines", Procedia CIRP, Volume 107, pg 1403-1408, (2022)
5. COMAU Aura Collaborative Robot, accessed online https://www.comau.com/en/competencies/robotics-automation/collaborative-robotics/aura-collaborative-robot/
6. COMAU Agile 1500, accessed online https://www.comau.com/en/competencies/robotics-automation/collaborative-robotics/automatic-guided-vehicles-agv/
7. COMAU Racer Cobot, accessed online https://www.comau.com/en/competencies/robotics-automation/collaborative-robotics/racer-5-0-80-cobot/
8. Microsoft HoloLens 2 AR glasses, https://www.microsoft.com/en-us/hololens/
9. Aruco Markers, https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html