Mobile dual arm robotic workers with embedded cognition for hybrid and dynamically reconfigurable manufacturing systems

Grant Agreement No	**:**	**723616**
Project Acronym	**:**	**THOMAS**
Project Start Date	**:**	**1st October, 2016**
Consortium	**:**	**UNIVERSITY OF PATRAS (LMS)** **PEUGEOT CITROEN AUTOMOBILES S.A. (PSA)** **SICK AG (SICK)** **FUNDATION TECHNALIA RESEARCH & INNOVATION (TECNALIA)** **ROBOCEPTION GMBH (ROBOCEPTION)** **DGH ROBOTICA, AUTOMATIZACION Y MANTENIMIENTO INDUSTRIAL SA (DGH)** **AERNNOVA AEROSPACE S.A.U. (AERNNOVA)** **INTRASOFT INTERNATIONAL SA (INTRASOFT)**

Title	**:**	Perception skills for process execution – Final Version
Reference	**:**	D3.5
Availability	**:**	Public
Date	**:**	30/09/2019
Author/s	**:**	ROBOCEPTION, TECNALIA, LMS, INTRASOFT, PSA, AERNNOVA
Circulation	**:**	EU, consortium

Summary:

This document provides an overview of the final version of perception skills developed for process execution, including object and tag pose estimation, pose refinement, and integration with the skill engine and overall demonstrators. The document contains descriptions of the software pipelines and interfaces that connect the developed modules to the overall system architecture of the Mobile Robot Platform (MRP) developed within THOMAS. Current implementations of the perception solutions on the use cases of THOMAS in AERNNOVA and PSA are also described.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

1. EXECUTIVE SUMMARY

The main purpose of this document is to provide a description of the final implementation of perception skills for process execution, including pose estimation for objects and tags. In this sense, the document is an evolution of the deliverable D3.3, "Environment and process reasoning modules – initial prototype", finalized in M18. The deliverable includes the description of:

- Tag detection
- Detection of objects
- Pose refinement
- Integration with the skill engine
- Integration with the overall robot control

The final demonstration of the perception skills implemented for solving the two use cases within Thomas are also summarized here. Perception skills can still evolve during the last year of integration (WP7), but the working principles for the solutions are described in this deliverable.

2. INTRODUCTION

This document presents the final implementation of perception skills for process execution, including pose estimation for objects and tags. Object's detection process in both use cases based on two different rc_visard sensors placed on THOMAS MRPs as has already documented in deliverable D3.3.

In the third section of this deliverable, the perception pipeline including the detection algorithm and its integration inside THOMAS is being described. This detection process's integration inside THOMAS Aeronautics and Automotive use case is briefly presented in the following two section of this deliverable accordingly.

3. PERCEPTION PIPELINE

This chapter provides an overview of the modules developed for environment and process perception and reasoning. An initial implementation of these modules was sketched in D3.3, "Environment and process reasoning modules – Initial prototype."

3.1. Perception modules

3.1.1. Regions of interest – April tag detection

The first module developed by Roboception is used for detection of Apriltags and QR codes using the *rc_visard*. The intention of this module is to provide an initial guess for the expected region for localizing a given object. Usually, this process is followed by a CAD-based localization to detect the full 6D pose of the object of interest. Figure 1 illustrates this process.

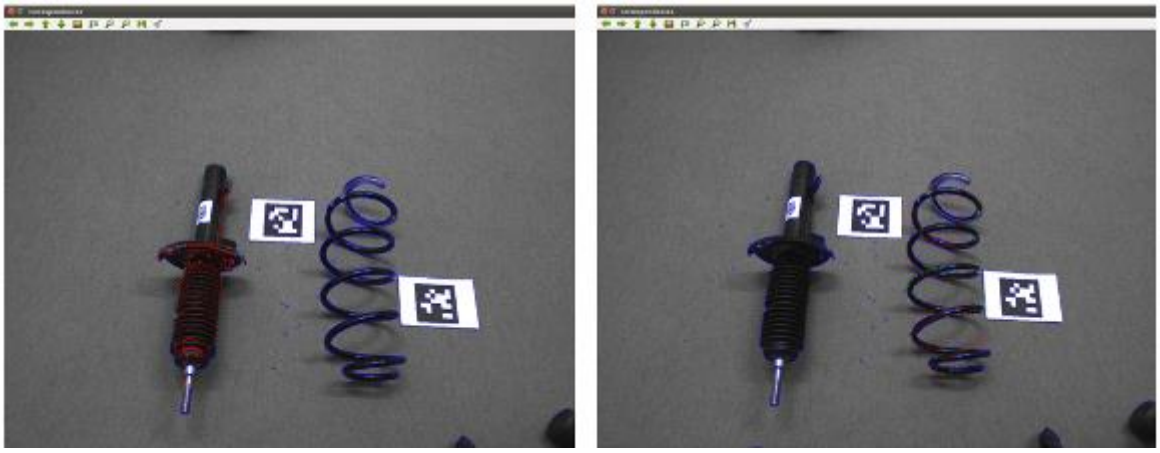

Figure 1: Region of interest indicated by an Apriltag (PSA use case). Left: initial guess. Right: refined pose estimation for the objects of interest

The component computes the position and orientation of each tag in the 3D camera coordinate system. Multiple tags can be located at the same time on the image. The tag detection module follows these general steps:

- Tag reading in the stereo image pair. The quality of this process can be controlled through a quality parameter that controls the downscaling of the images, which increases the speed but reduces the accuracy of the detection process. This parameter influences also the maximum distance of the tag with respect to the camera. As a general indication, for the Apriltag family 36h11 (recommended), with 4 cm width and composed by 8 modules, the maximum distance to the camera is 1.1 m. For a QR code with a width of 8 cm and using 29 modules, the maximum distance is 0.49 m.
- Estimation of the pose for each tag. For each tag detected on the image, the pose of the tag in the camera coordinate frame is computed, assuming that the tag is visible in both left and right images. The coordinate frame for the tag is aligned as shown in Figure 2, with the Z axis pointing into the tag. The size of the tag is also estimated at runtime, assuming that the tag is a square. However, if the size of the tag is known beforehand, it can be provided, which improves the results of the detection.
- Re-identification of previously seen tags. Each tag has an ID; for AprilTags it is the family plus tag ID, for QR codes it is the contained data. However, these IDs are not unique, since the same tag may appear multiple times in a scene. For differentiating these tags, the TagDetect components also assigns a unique identifier to each detected tag. To help the user identifying an identical tag over multiple detections, tag detection tries to re-identify tags; if a tag was previously seen, it receives the same unique identifier again.

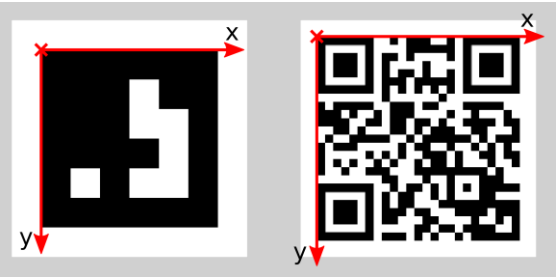

Figure 2: Apriltag and QR code reference frame

Both QR codes and AprilTags have their pros and cons. While QR codes allow arbitrary user-defined data to be stored, AprilTags have a pre-defined and limited set of tags. On the other hand, AprilTags have a lower resolution and can therefore be detected at larger distances. Moreover, the continuous white to black edge around AprilTags allows more precise pose estimation. If user-defined data is not required, AprilTags should be preferred over QR codes.

Systematic tests were carried out to determine the expected precision for the Apriltag and QR code detection, which mainly depends on the distance to the tag (and the camera resolution). Table 1 and Table 2 summarize the results, assuming that the *rc_visard*'s viewing direction is roughly parallel to the tag's normal direction, and tests were performed with the highest quality given by the module. The tables contain results for lateral precision (i.e., in x and y direction) and precision in z direction.

Table 1: Results of precision tests for apriltag detection

Distance	*rc_visard* 65 - Lateral	*rc_visard* 65 - z	*rc_visard* 160 - Lateral	*rc_visard* 160 - z
0.3 m	0.4 mm	0.9 mm	0.4 mm	0.8 mm
1.0 m	0.7 mm	3.3 mm	0.7 mm	3.3 mm

Table 2: Results of precision tests for QR code detection

Distance	*rc_visard* 65 - Lateral	*rc_visard* 65 - z	*rc_visard* 160 - Lateral	*rc_visard* 160 - z
0.3 m	0.6 mm	2.0 mm	0.6 mm	1.3 mm
1.0 m	2.6 mm	15 mm	2.6 mm	7.9 mm

For interfacing to these services, suitable REST-API calls were created. In addition, a webGUI was created to interact with the module using a web browser (Figure 3). ROS interfaces were also created and are publicly available in [1] . A tutorial for usage of this module is also available in [2] .

This module is part of the Exploitable Result ER3, "Generic Perception Modules for Robot Guidance." The module has received the commercial name of "*TagDetect*" and is commercially available since April 2018 (It was officially launched during Hannover Messe 2018).

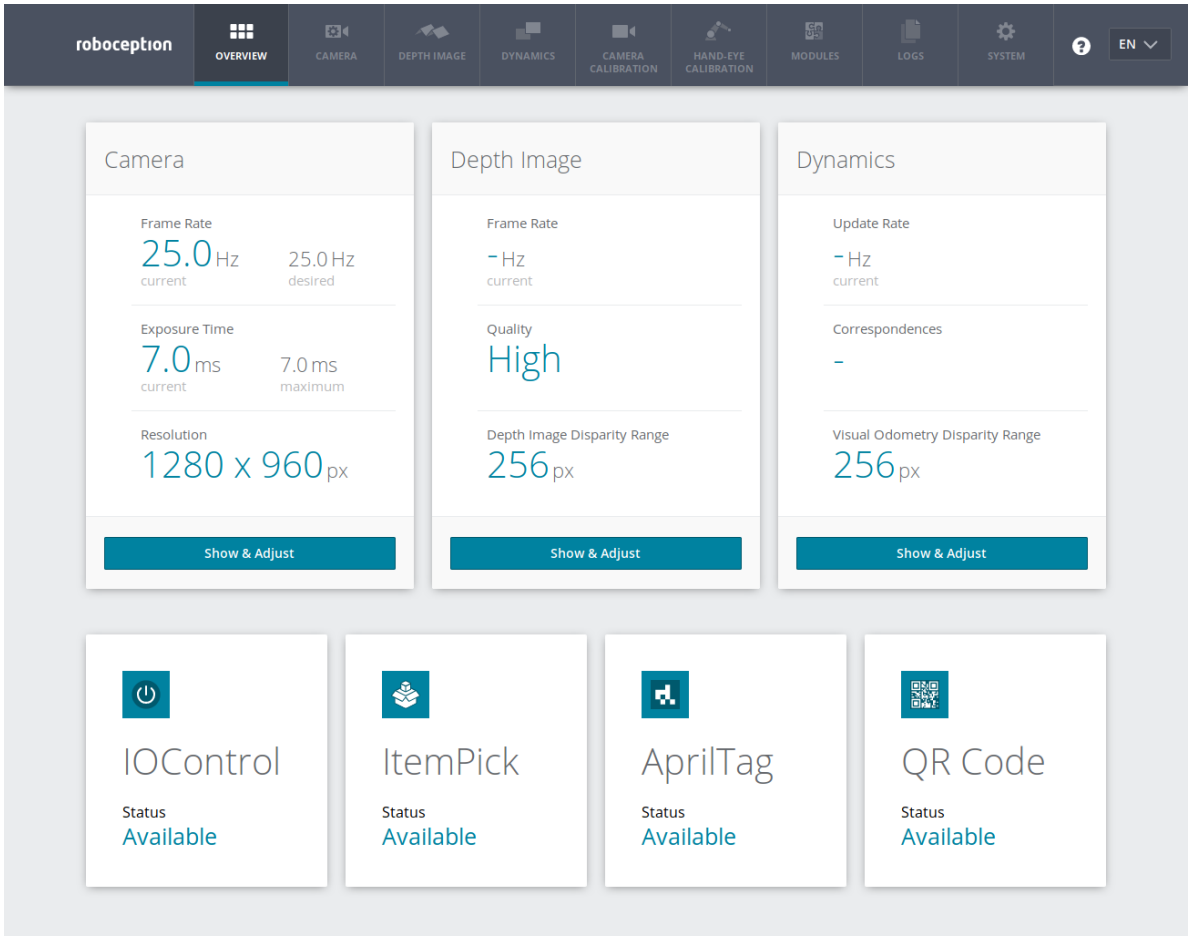

Figure 3: WebGUI for using the services of AprilTag or QR code detection

3.1.2. 3D Object detection

The final pipeline for object detection based on CAD models of the object is illustrated in Figure 4, using the detection of templates for the drilling operation in the AERNNOVA use case as an example. The CAD information of the object should be provided as an .OBJ format. The detection pipeline contains the following steps:

- Acquisition of images and pointcloud data with the *rc_visard* sensor.
- Initial detection. Using information of the process, e.g. if the part is located on a tabletop scenario or if there is a tag signalling a region of interest, an initial pose of the object is provided. The ROS service for this process is *detect.*
- Final refinement. For this step, the edges on the image are extracted, the edges from the CAD model are projected to the scene using the initial pose, and the distance between the edges of interest is minimized for obtaining a final pose estimation. The ROS service for this process is *refine.*

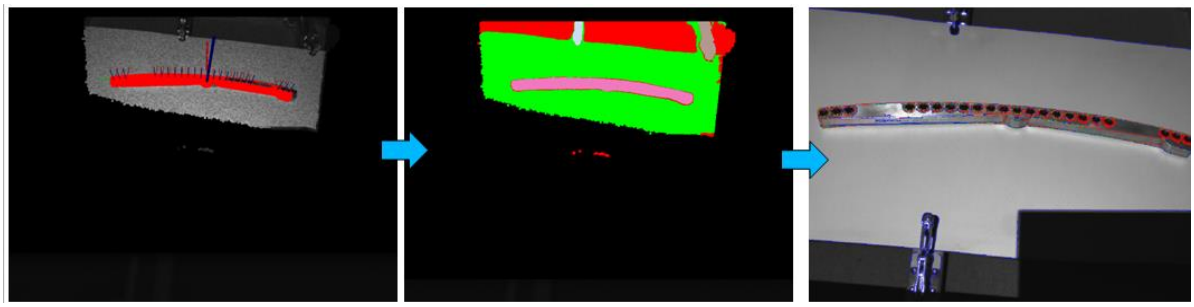

Figure 4. Pose refinement via edge alignment. From left to right: 3D data, initial detection and final refinement

The behaviour of the calls is different, depending on two possible scenarios used for providing the initial pose of the object:

- Tabletop scenario (Figure 5): this type of scenarios restricts the object pose to be in a plane, therefore only 3 parameters are required to specify the pose of the object. This is the case that suits the AERNNOVA scenario described in Section 3.

  For this scenario, there is a dominant plane on the detection; this plane is a large flat surface, which can be easily segmented from the pointcloud. The points corresponding to the dominant plane are removed from further processing. The cluster of points representing the object is then selected, its inertia tensor is computed, and the axis of inertia provides a pose prior. The center of mass of the object is assumed to be in the center of the bounding box.

  For the refinement, the normal direction of the dominant plane, which corresponds to the Z axis, is retrieved from the *detect* call, which also provides the distance of the plane to the camera. It is assumed that these parameters do not change since the *detect* call. The refinement of the edges for obtaining the final pose is only performed using the left camera image, and the pose is obtained by solving an optimization problem that provides the 3 parameters describing the object pose on the plane.

- General scenario (Figure 6): in this type of scenarios the object can be located anywhere in space. In total, 6 parameters are required to specify the pose of the object. This is the case that suits the PSA scenario described in Section 4.

  For this scenario, an Apriltag located in the working environment signals the region of interest which provides an initial guess of the object pose.

  For the refinement, the edges from the CAD model located at the initial pose are rendered on the image and compared with the edges obtained from the visual data in both left and right images. The pose of the object is optimized such that the deviation in pixels between the model and the visual edges is minimized simultaneously in the two frames. The refinement is not stable enough from a single camera image, as the required pose of the object is in 6D. Using simultaneously the two images provides robustness to the pose estimation process.

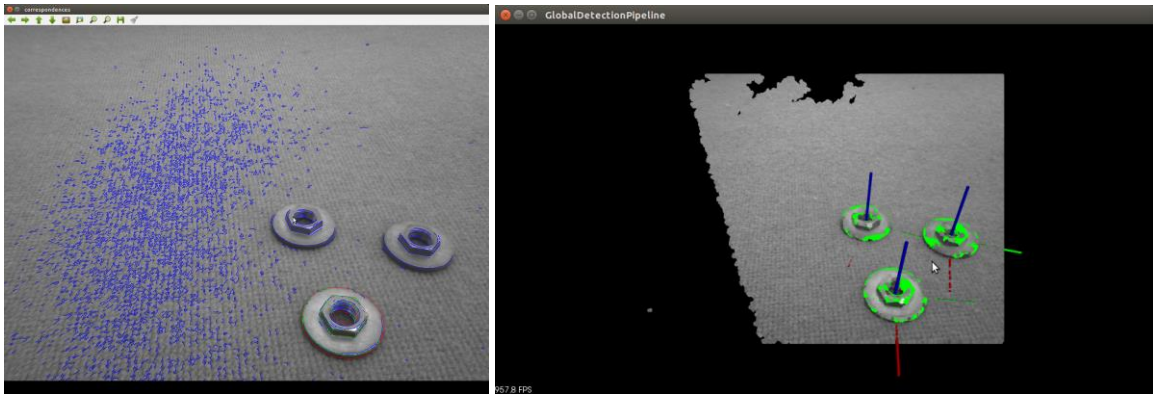

Figure 5. Detection of nuts on a tabletop scenario. Left: correspondence between edges in the model and the image. Right: reference frames for the three nuts detected in the scene

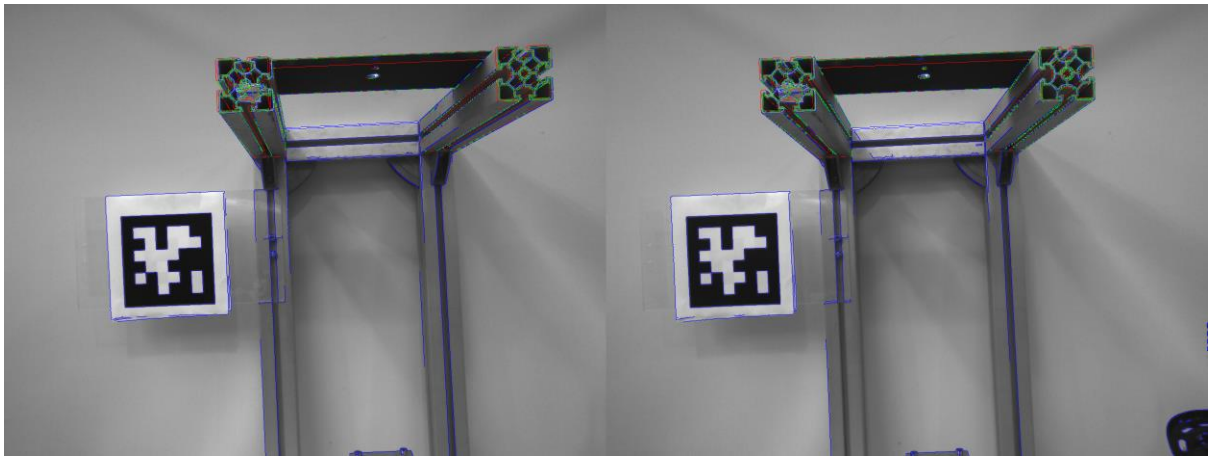

Figure 6. Detection of the upper structure of the mockup machine (PSA use case) using Apriltags to provide an initial pose estimation. Both left and right images are used for the pose estimation process

The general limitations detected so far on the pose estimation process provide mainly from the quality of the CAD model available for the part. The detection of the edges on the CAD model is very sensible to the way that the CAD model is constructed, e.g. whether there are coincident points at the same location but they are not merged, or if there is excessive detail that provides a very large amount of edges that are not detectable on the image (e.g. internal holes or small details in the part). From the perception process, sometimes edges are not visible enough, or the colour of the part (especially for black or shiny parts) makes it difficult to get enough points to define the edges.

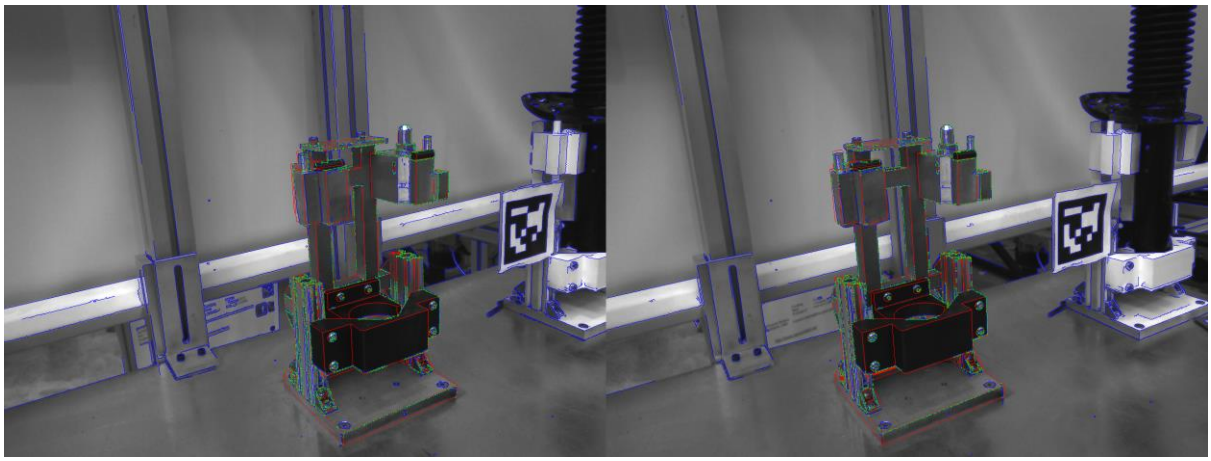

Figure 7. Low accuracy on the part detection due to lack of visual information (in this case, due to the black part, which occupies a significant part in the front of the object to be detected).

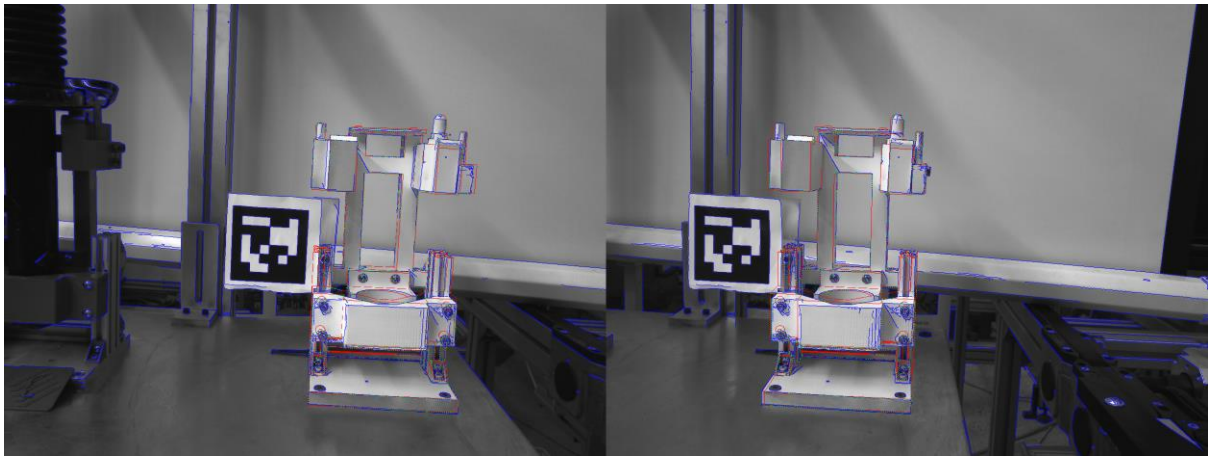

Figure 8. Low accuracy on the part detection due to mismatching CAD model and real part

An evaluation of accuracy was performed for the whole template detection in the AERNNOVA use case, to provide an indication of the expectable accuracy for the real robotic application. Three different templates were placed at different distances from the sensor, as shown in Figure 9, and the average pixel deviation and accuracy were computed for the templates. Table 3 summarizes the results. The accuracy is expected to be close to 1 mm in the standard registration process for the overall template. To improve the accuracy and achieve 0.2 mm, as required for the real implementation, a second refinement step using an additional *rc_visard* sensor was included, which takes a close image of a region of the template, thus allowing higher precision in the location of the drilling holes. The expected error on the implementation with the real robot is expected to be higher, since it would include the effects of robot accuracy and the hand-eye calibration process, mainly.

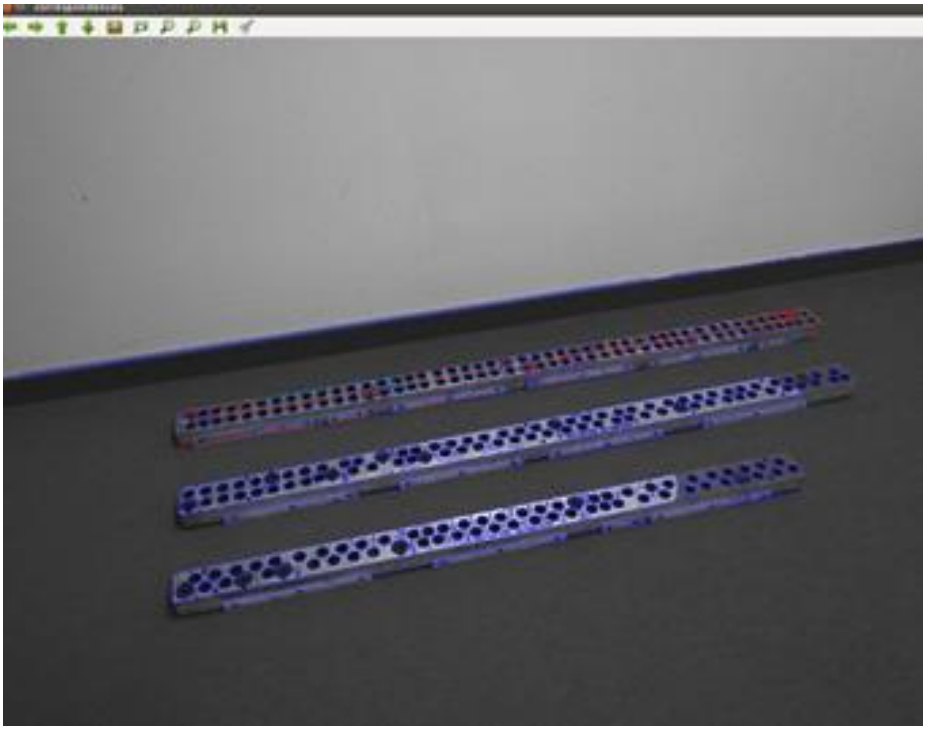

Figure 9. Accuracy evaluation for template detection

Table 3: Results of accuracy tests for initial template detection

Distance	Average pixel deviation	Average accuracy
1.40394m	1.063 px	0.00138835m
1.28625m	0.807753px	0.000966534m
1.16165m	0.80547px	0.000870442m

Note that the detection of parts when there is not enough natural texture requires the projection of an additional pattern on the part, to be able to acquire enough information for the stereo matching process. A *RandomDot projector* is a specifically tailored projector that can be used as an enhancement to the *rc_visard* when the perception of particularly difficult scenes with little or no natural texture is required. It is controlled directly from the *rc_visard*, using the *IO Control* module. This module uses also RestAPI interfaces to control the type of exposure and synchronization of the projector with the image acquisition of the sensor.

The CAD-based detection module is also part of the Exploitable Result ER3, "Generic Perception Modules for Robot Guidance." The module has received the provisional internal name of "*CADMatch*" and is currently under testing in Thomas. With the feedback received, we will move to Beta testing with other potential customers. The planned release of the module is Q3 2020.

3.2. Interfaces and integration with other modules

This section provides and overview of the interfaces used with the perception modules for connection with the world model. The concept for integration of the perception modules with the skill engine was presented in D3.3, "Environment and process reasoning modules – Initial prototype." and has not changed since the deliverable was presented.

3.2.1. ROS connectivity of *rc_visard* and additional software modules

The connection of components is based on ROS services. Roboception has published all the interfaces to the *rc_visard* as open source. They are available in the GIT repo in [3] .

The information related to the sensor drivers for the basic connectivity and for the optional software modules running on the *rc_visard*, as well as accompanying tutorials, is available in [4] .

For instance, the ROS driver for the TagDetect module is described in [5] and the source code is available in the GIT repository above mentioned.

3.2.2. Overall integration architecture

The integration of the perception process consists of the integration of the *rc_visard* sensors with the THOMAS world model. The communication system that makes the sensor data available to the THOMAS system as well as the Task execution sequence that use this data are documented in Deliverable D3.3. The diagram for the interaction of the process perception modules with THOMAS system, for which INTRASOFT (supported by LMS) is responsible, is shown in Figure 10.

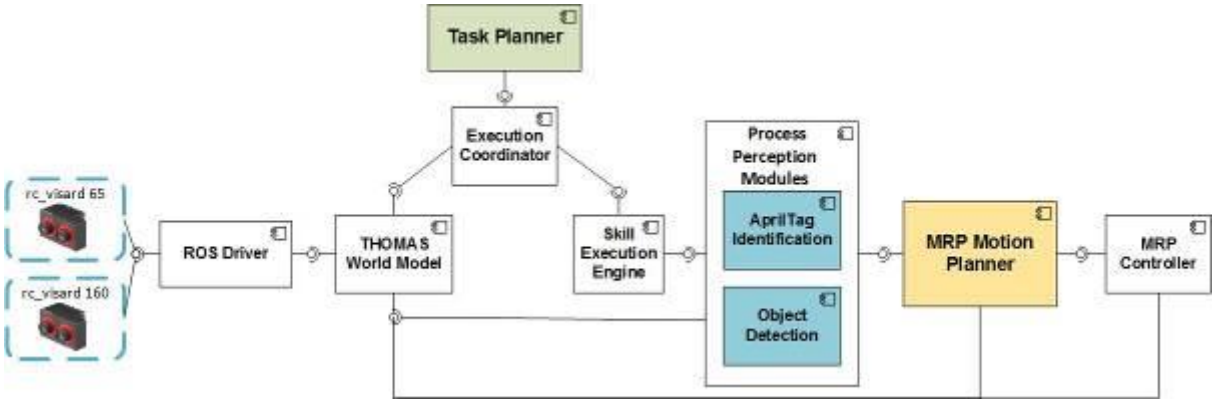

Figure 10: THOMAS Process Perception modules interaction with THOMAS system

As shown in the diagram, the *rc_visard* sensors are integrated in the system through the THOMAS world model, which connects to the ROS Driver of the *rc_visard* sensors.

The *rc_visard* sensors are accessed directly from the THOMAS World model interface by the following modules:

- The April Tag Identification and Object Detection Process Perception modules.
- The MRP Controller Module
- The Execution Coordinator Module.

Other modules can indirectly utilize the functionalities of *rc_visard* sensors by utilizing functionalities of the aforementioned software modules. The integration at this software level is more goal-specific and utilizes specific API that facilitates the needs of the required functionality. For instance, the Skill Execution engine can use the Process Perception Modules in order to execute the object/apriltag detection skills. The Apriltag Identification and Object Detection Process Perception modules utilize the THOMAS World Model, which is integrated with the *rc_visard* sensors through the ROS Driver.

The MRP has two *rc_visard* sensors attached, the *rc_visard* 160 on the right arm and the *rc_visard* 65 on the left arm.

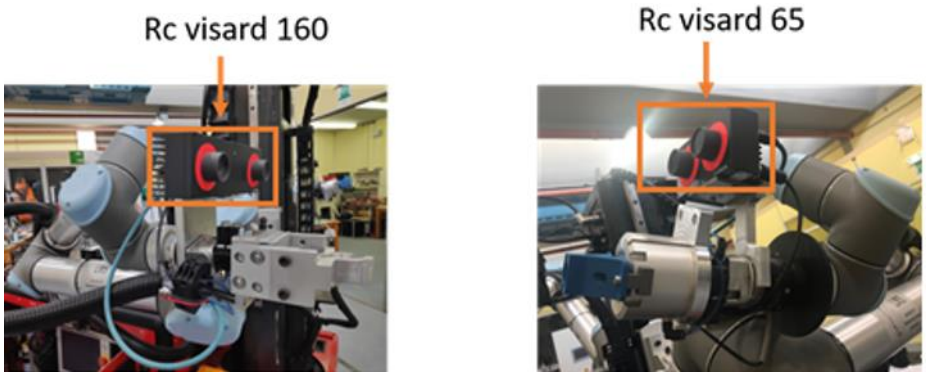

Figure 11. *rc_visard* sensors on the MRP

Table 4. Sensor topics

Sesnor ID	Camera(left/right)	Topic name	Server topic
rc_visard160	left	/rc_160/stereo/left/	/rc_160/detection_handler_server
	right	/rc_160/stereo/right/	
rc_visard65	left	/rc_65/stereo/left/	/rc_65/detection_handler_server
	right	/rc_65/stereo/right/	

The topics of the two cameras mounted on the MRP are shown above (Figure 11), as well as the topics (Table 4) of the Server of the Skill Execution Engine with which the detection services of the cameras are integrated. The function of the Skill agent and the communication of Clients and Servers using the aforementioned topics are described in Deliverable D5.4.

The position of each camera in reference to the MRP is generated through a hand-eye calibration provided by ROBOCEPTION. The URDF is then updated by adding an extra link for each camera.

The Perception process is responsible for the detection of the parts. It is structured in the form of an actionlib server–client interaction. The server and the client communicate via a "ROS Action Protocol" built on top of ROS messages. The client sends its message to the detection server's topic which is shown in Table 4.

The detection process creates a frame with the position of the detected object in reference to the position of the camera and therefore with respect to the robot. The link between the camera and detected object frame is shown in Figure 12.

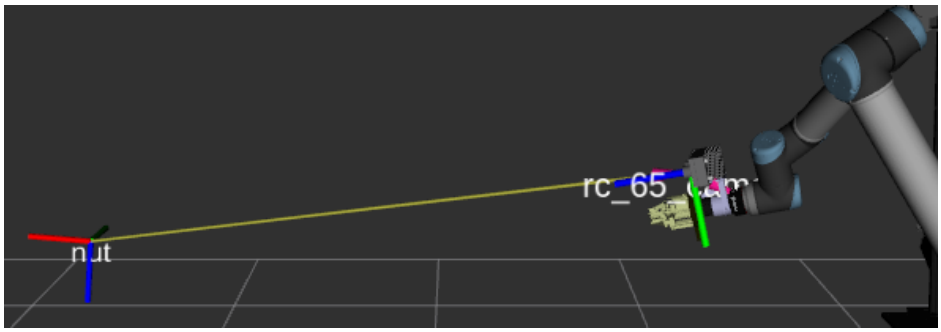

Figure 12. Sensor (*rc_visard* 65) and frame for the detected part (nut)

Through the integration of the detection process with the overall architecture, the motion planner uses the frame of the detected object to proceed with the grasping or placing operations.

4. AERNNOVA USE CASE

The use cases at the AERNNOVA plant were described and analyzed in D1.1, "Use-case definition and evaluation metrics", and D1.2, "End users requirements analysis". Three operations were analyzed: drilling, rivet inspection and paint sanding (Figure 13). The MRP platform has undergone several improvements to fit all the necessary devices, including the stereo cameras required for perception processes, to perform the drilling task in the AERNNOVA plant. This task was selected as the key use case for development of the perception components, as explained in D1.2. A physical set up has been deployed at TECNALIA following the requirements of the AERNNOVA use case.

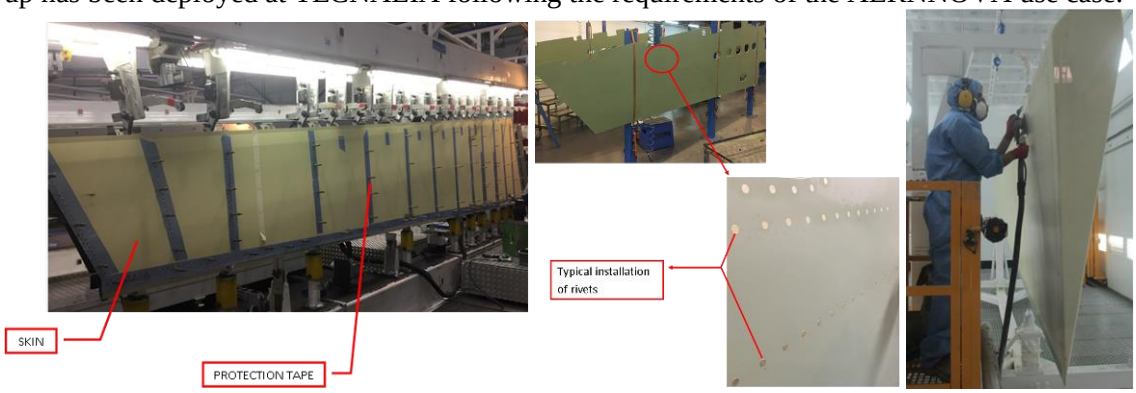

Figure 13. Scenarios considered in the AERNNOVA use case. From left to right: drilling, rivet inspection, and sanding

A prototypical implementation of the drilling use case has been implemented at TECNALIA, where the robot is able to navigate autonomously, perform a precise positioning in front of the structure to be drilled, and after that, drill the holes. The robot finds the area of interest in front of the docking station. The positioning algorithm is based on the detection and correction of the position of the MRP by a proportional control based on a given reference. For that purpose, an AprilTag located in the docking station is tracked using the IDS camera mounted in the front of the robot. Figure 14 shows the AprilTag located in the docking stations. Moreover, a drilling template can be seen in the structure placed vertically, thus recreating the real use case where templates are placed above the actual surface of the aircraft wing.

Two *rc_visard* sensors placed in the robot are used to reference the templates. As explained in section 2.2.1, the perception system has a sensor that references the template (template detection sensor, *rc_visard* 160), which needs more than one meter of distance between the robot and the template due to the large size of the template. The other arm has mounted the precision sensor (hole detection sensor, *rc_visard* 65) that works optimally in the vicinity of the template, in a workspace easily reachable by the arm.

The detection process uses two images, one with a projected pattern (to improve registration of the point cloud), and one without projected pattern (to acquire the raw image for the refinement step), as illustrated in Figure 15. The pipeline described in Section 2.1.2 is then executed to locate the drilling templates. Then, the location of each hole (coordinate system of the hole, extracted from the CAD system) is provided as output of the detection module after using the hole detection sensor to refine their location. This information is required to perform the drilling process afterwards.

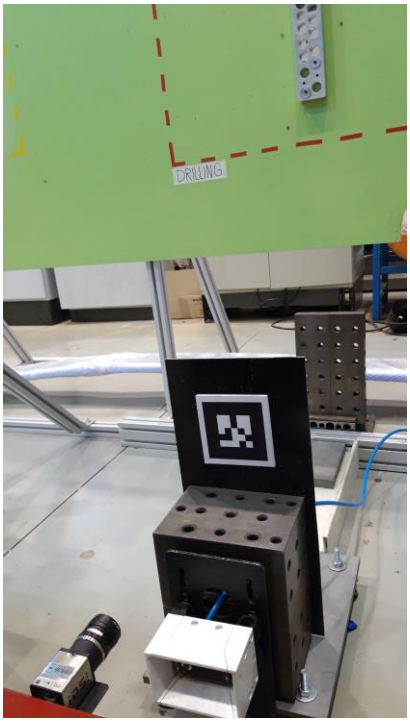

Figure 14: The AprilTag used for precise positioning allows detection and usage of the drilling templates located above it

Figure 15. Detection of template. From left to right: acquisition of images with and without projected pattern, and detection of the holes

The pilot station developed at TECNALIA currently contains only a unique drilling template. Even though multiple instances of drilling templates can be detected simultaneously (as can be seen at Figure 15), the efforts are focused on assuring the robustness of the template and hole detection first for an individual template. On the one hand, Figure 16 shows how the *rc_visard* 160 sensor locates the template placed on a real aircraft wing provided by AERNNOVA. On the other hand, Figure 17 illustrates how the *rc_visard* 65 sensor is able to detect precisely the holes of a section of the template, allowing the insertion of the drilling machine and the consequent drilling process.

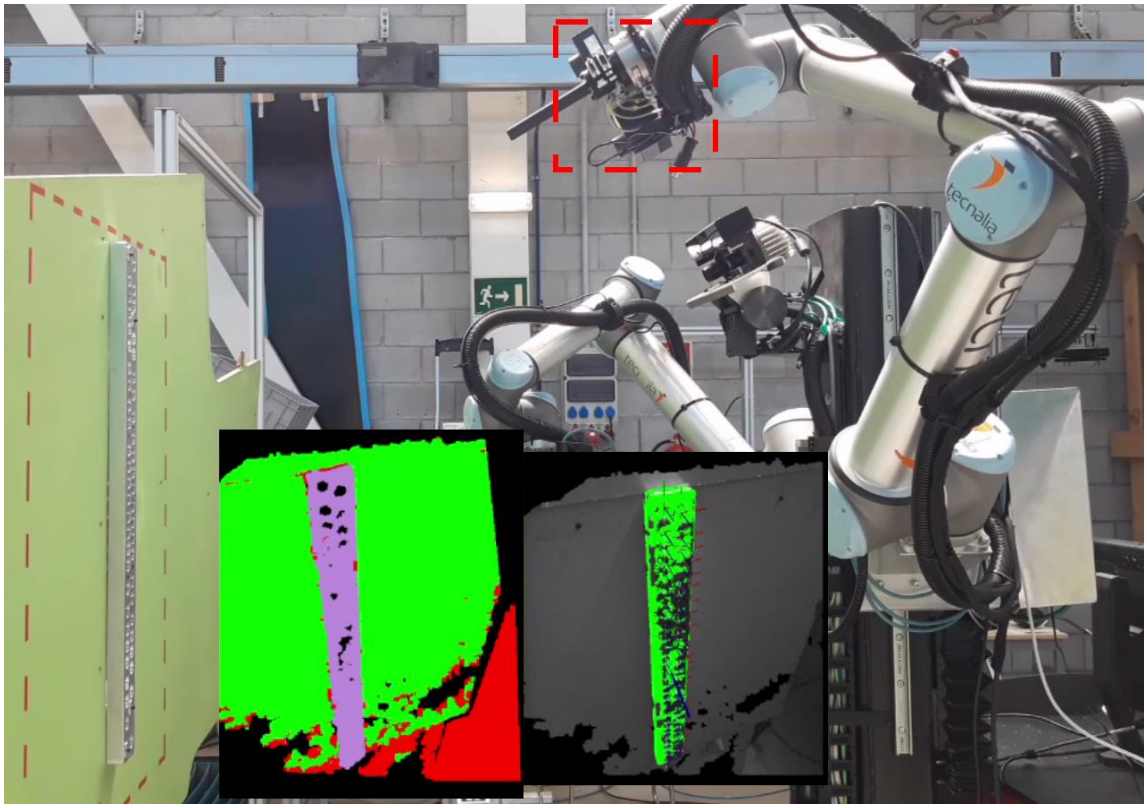

Figure 16: Template detection using *rc_visard* 160 sensor in the pilot station

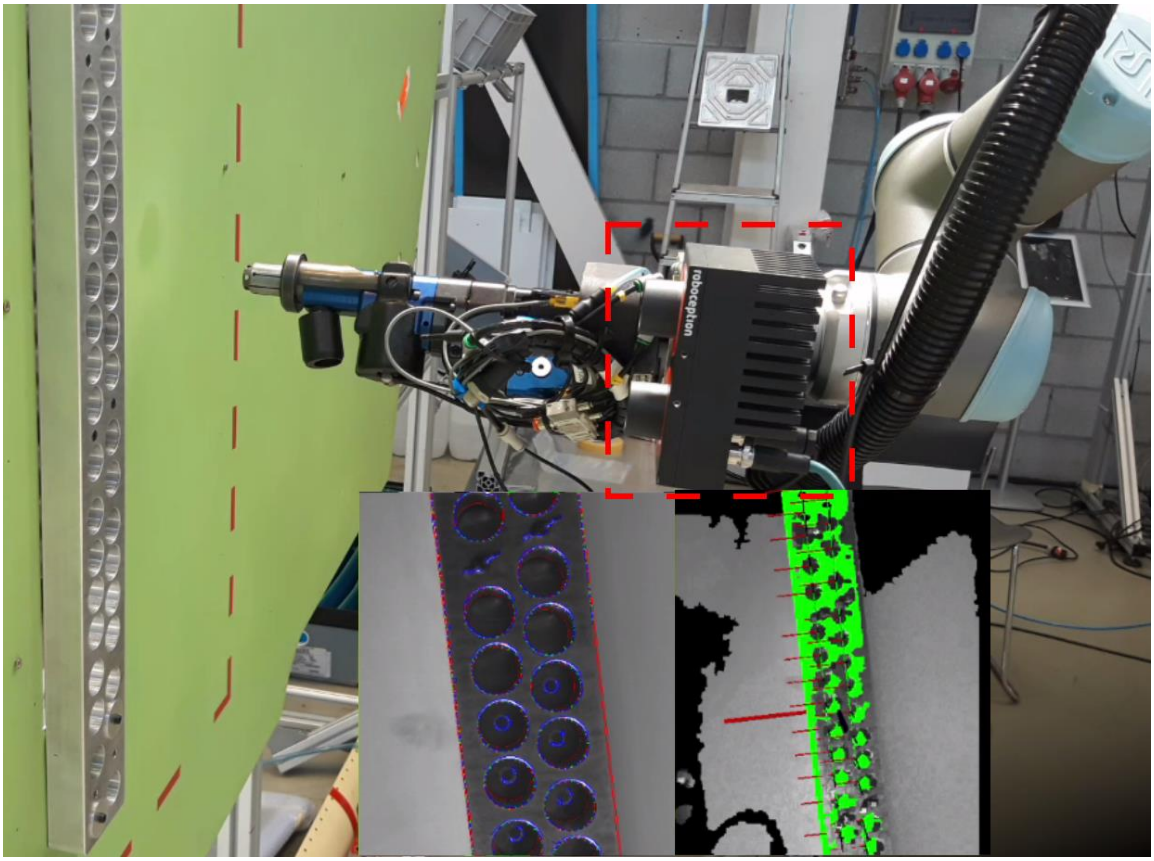

Figure 17: Hole detection using *rc_visard* 65 sensor in the pilot station

5. PSA USE CASE

THOMAS vision for the automotive scenario has been described in detail in D1.1, "Use-case definition and evaluation metrics" and updated in D2.1, "Perception for HR interaction - Design". The perception components are required to detect the region of operation for loading / unloading the parts and detect the parts that need to be manipulated in the scenario. A physical set up has been deployed at LMS following the requirements of the PSA use case (Figure 18).

Figure 18. Scenario considered in the PSA use case. From left to right: parts on the Dolly, compression machine, and assembly and screwing on the AGV

The current scenario consists of the MRP (Mobile Robot Platform) moving to the working areas of the cell shown in Figure 19 performing picking, placing as well as screwing operations through the implementation of object detection using *rc_visard* sensors. There are three working areas, the **pre-assembly**, the **compression machine** and the **MPP (Mobile Product Platform)**. In particular, the MRP will be responsible for picking the pre-assembled damper from the pre-assembly table, placing it on the compression machine, inserting the nut to the damper's upper part and removing the alignment rod, picking the compressed damper from the compression machine, transporting and placing it on the MPP and finally, for the mobile screwing operations. The design of the required modules to perform damper's detection as well as the software interface of camera have already been introduced in deliverable D3.1 "Environment and Process Perception module - Design", while the functionality of the software has been introduced in deliverable D3.3. The current implementation of the detection cases is described in this section.

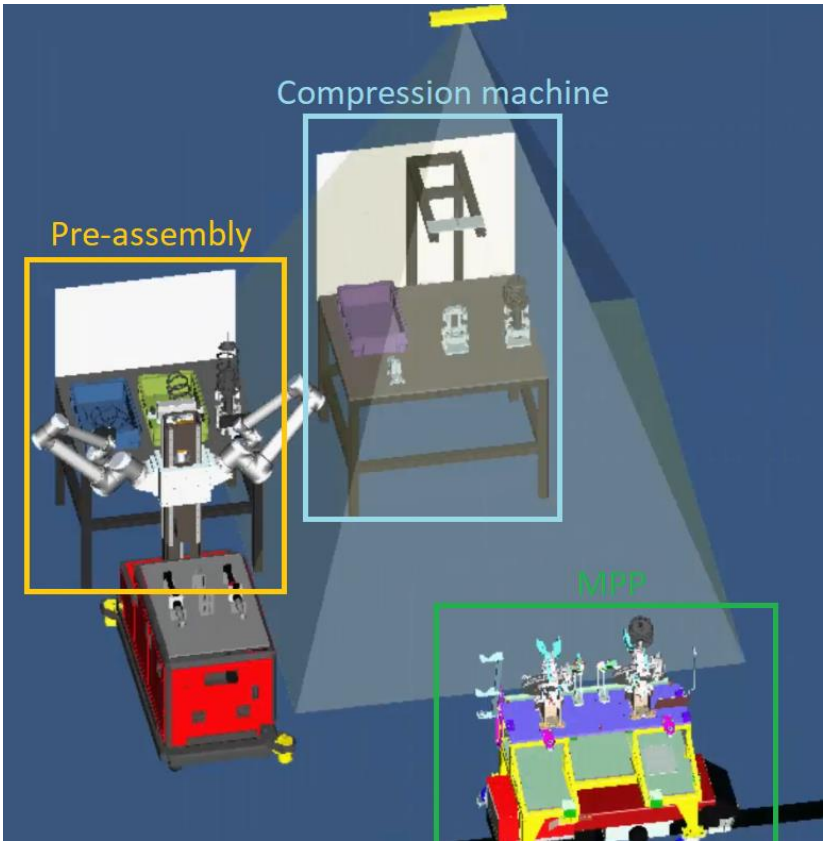

Figure 19. Current designed cell consisting of pre-assembly table, compression machine and MPP

Sensors: For the detection cases two *rc_visard* sensors are used (Figure 20). Mounted on the right arm is the *rc_visard 160* camera, used primarily for the detection of the damper. Mounted on the left arm is the *rc_visard 65* camera, used primarily for the nut, fixture and upper structure detections.

Grippers: Additionally, for the grasping procedures, two pneumatic parallel gripers are installed on the UR10's end effector. The weights of the objects that need to be grasped were taken into account for their selection. The shape of their fingers was dependent on the shape of the objects as shown in Figure 20, where the right arm gripper must grasp dampers and the left arm gripper must grasp the nut and the alignment rod.

Figure 20. Rc_visard sensors on MRP

Physical set up: Based on the current designed scenario, the physical set up has been deployed at LMS. The setup of the cell is presented in Figure 21, and each working area is further analyzed with respect to the detection cases.

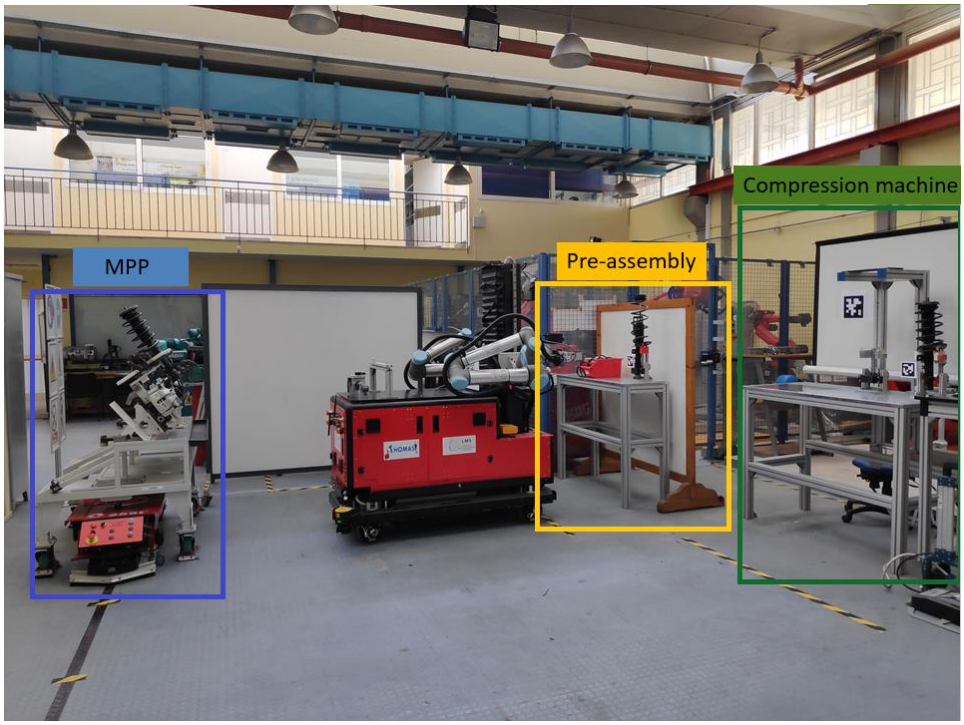

Figure 21. Current physical cell consisting of pre-assembly table, compression machine and MPP

Pre-assembly table: The pre-assembly area (Figure 22) contains the pre-assembled damper that needs to be detected and picked. For this detection the *rc_visard 160* sensor is used.

Figure 22. Pre-assembly table

- **Apriltag detection:** At the beginning of the process, the camera detects the Apriltag located at the left side of the damper and on a fixed position compared to the origin point of the damper (Figure 23). In this way, using one transformation matrix between the damper's origin

frame and one April tag's corner, the position of damper relative to the Apriltag is defined. Note that the Apriltag only provides a rough estimation of the pose of the damper, which is refined in the next step.

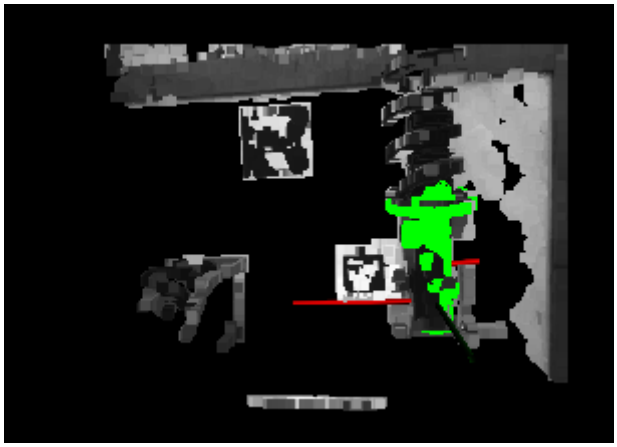

Figure 23. Apriltag detection

- **Refinement:** To refine damper's position, the software compares the position of the physical damper and the position coming from the detection of the Apriltag (Figure 24). The process implements a CAD matching algorithm. After this comparison, the final position of damper is refined, and the position of the detected grasping point relative to the camera frame is returned as output from the object detection algorithm.

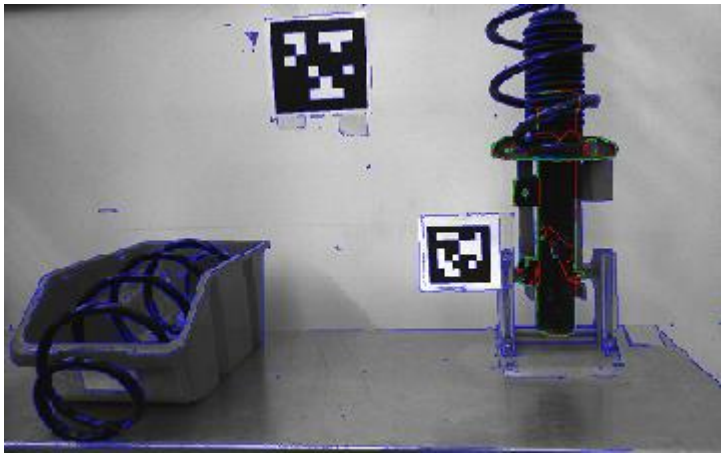

Figure 24. Refinement of detection

Compression machine: At the Compression machine, the MRP's responsibilities are the placing of the pre-assembled damper, the insertion of the nut to the damper's upper part, the removal of the alignment rod, and the picking of the compressed damper. For the placement of the damper, the detection of the fixture is necessary; for the alignment rod and the nut, the detection of the upper structure is necessary; and for the picking of the compressed damper, the detection of the damper is required. The detection of the damper is performed by the *rc_visard 160* sensor, while for the other cases the detection is performed by the *rc_visard 65*. The location of the Apriltags used on the compression machine is shown in Figure 25.

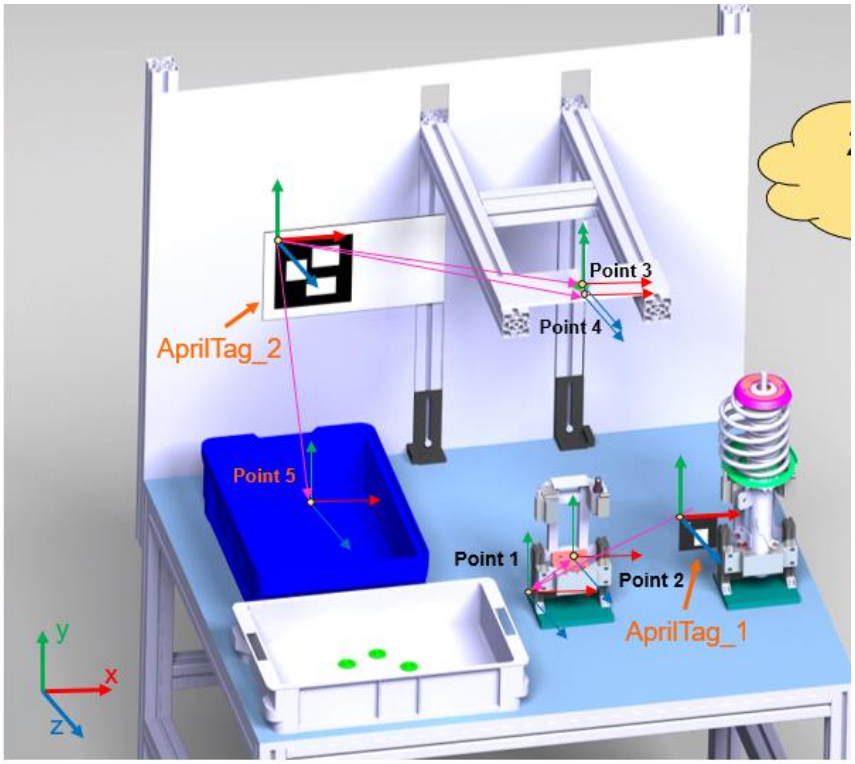

Figure 25. Apriltags used on compression machine

According to this design, the Apriltags were arranged on the physical compression machine as shown in Figure 26.

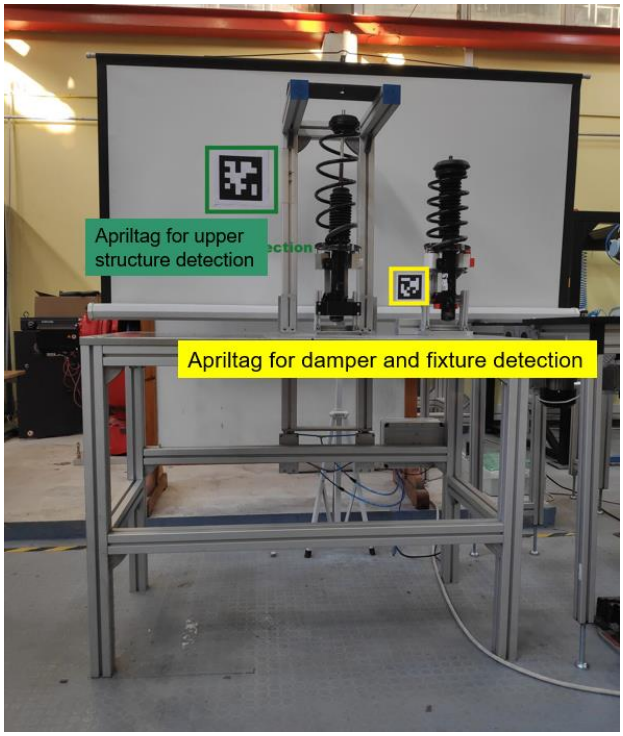

Figure 26. Compression machine.

- **Fixture detection:** The fixture detection and refinement process (Figure 27), which is necessary for the placement of the pre-assembled damper, is similar to the damper's process, but now using the CAD model of the fixture as prior information.

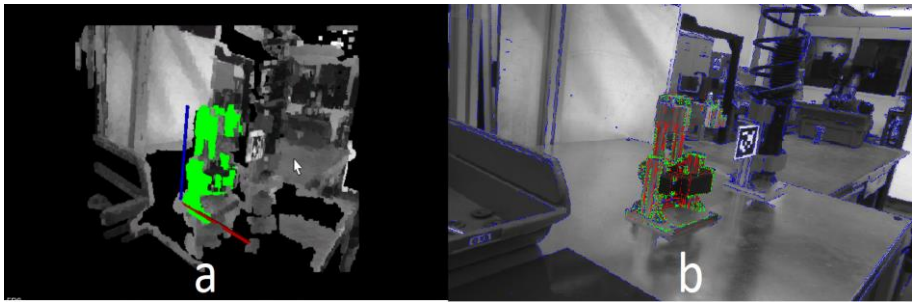

Figure 27. Fixture Apriltag detection (a) and refinement (b).

- **Nut detection:** For the detection of the nut (Figure 28) no Apriltag detection is used. The process finds the nut's physical position applying CAD matching (using the tabletop scenario), and proceeds with the refinement.

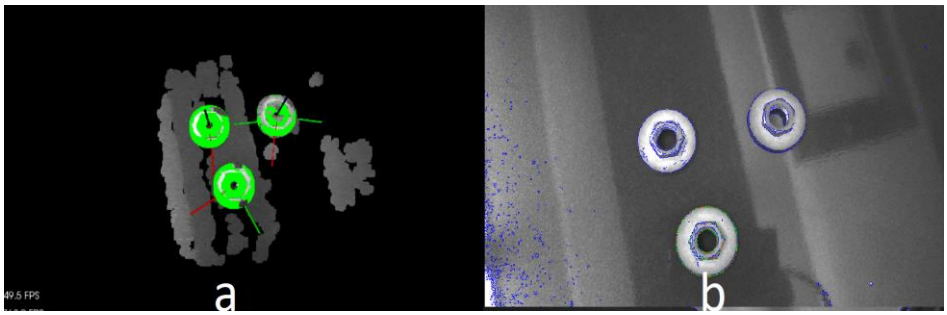

Figure 28. Nut detection (a) and refinement (b).

- **Upper structure detection:** The upper structure detection and refinement process (Figure 27), is also similar to the damper's process, but now using the CAD model of the upper structure.

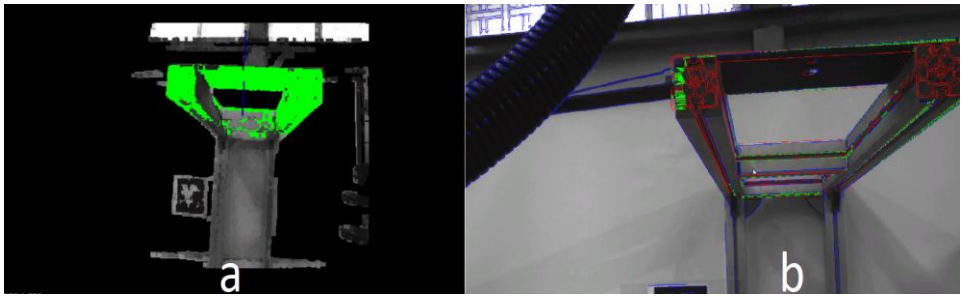

Figure 29. Upper structure Apriltag detection (a) and refinement (b).

- **Compressed damper detection:** The compressed damper detection process (Figure 30) is the same as the pre-assembled damper's detection. Note that the CAD model used for both the pre-assembled and compressed damper is the lower part of the damper, making it suitable for both detections regardless of their upper parts being different.

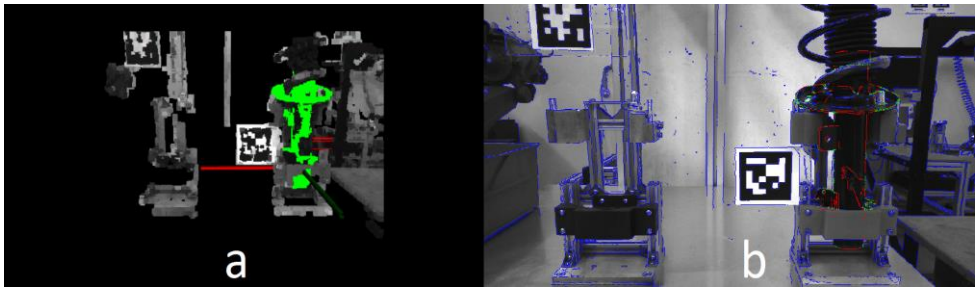

Figure 30. Compressed damper Apriltag detection (a) and refinement (b).

MPP: At the MPP working area the MRP's responsibilities are to place the compressed damper and to perform the screwing tasks. For these tasks, the left and right disk on the MPP must be detected. Both the *rc_visard 65* and the *rc_visard 160* are used. The location of the Apriltag and the origin points are shown in Figure 31.

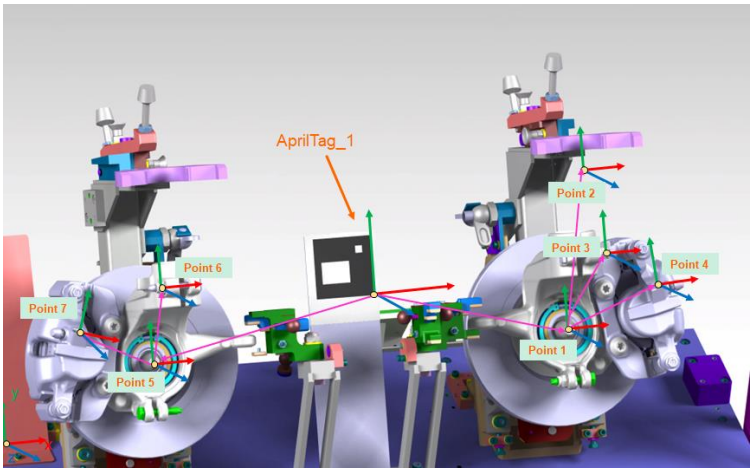

Figure 31. Apriltags use for MPP.

According to these designs, the physical Apriltag formation was created (Figure 32).

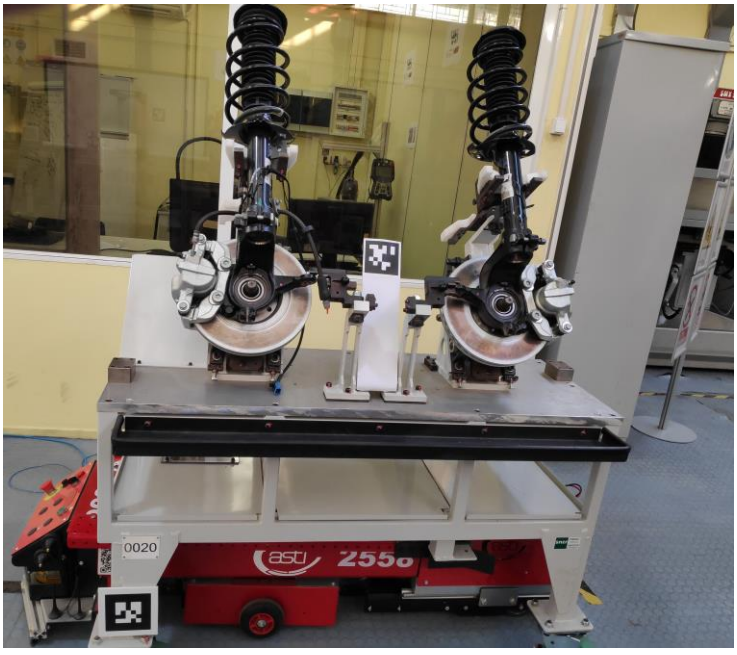

Figure 32. Physical MPP

- **Left disk detection:** The left disk detection and refinement process (Figure 33), is also similar to the damper's process, using now the CAD model of the left disk. The detection of the left disk is necessary for the placement of the compressed damper and the screwing operations of the right arm.

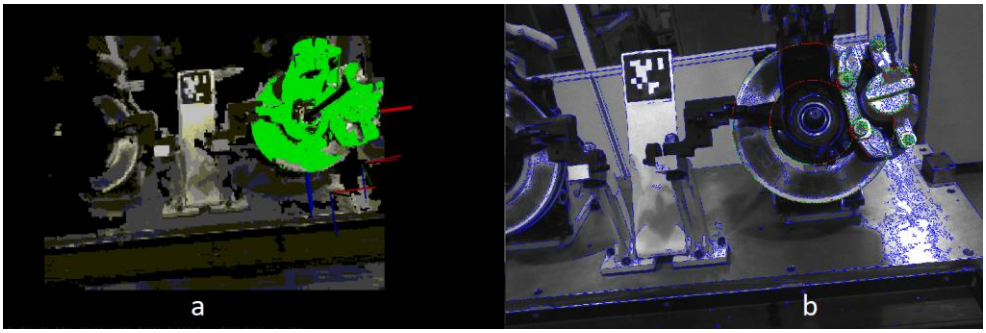

Figure 33. Left disk Apriltag detection (a) and refinement (b).

- **Right disk detection:** The right disk assembly detection and refinement process (Figure 34), is also similar to the damper's process, using now the CAD model of the right disk. This detection is necessary for the screwing operations of the left arm.

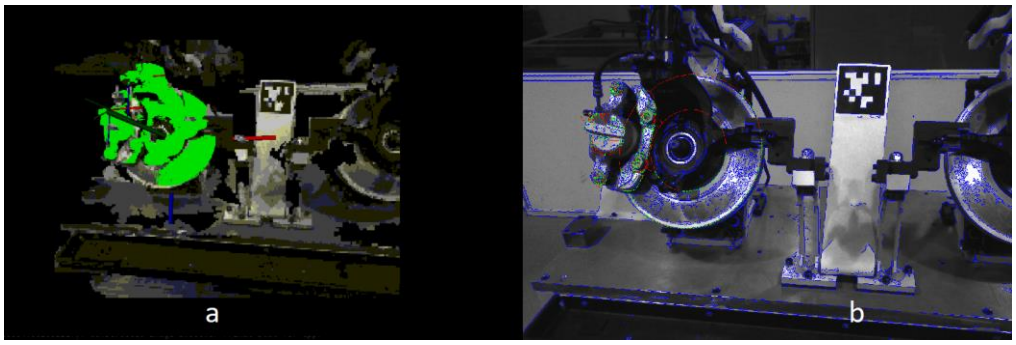

Figure 34. Right disk assembly Apriltag detection (a) and refinement (b).

6. CONCLUSIONS

This document has presented the final result of the perception skills for process execution, developed within WP3, which includes the perception pipelines for object detection and process awareness within the THOMAS project. The description includes two modules, CAD based object detection and Apriltag detection. Both modules are exploitable results of the project.

The software pipelines and integration with the skill engine and MRP control were also presented. The document also includes the description of the final implementations with the MRP for the use cases of AERNNOVA (implemented at TECNALIA) and PSA (implemented at LMS). These implementations served to evaluate the functionality of the components in relevant scenarios and will be further refined within WP7 – "THOMAS demonstrators and assessment" to obtain the final implementation of the perception skills on the MRP.

7. GLOSSARY

MRP	Mobile Robot Platform
MPP	Mobile Product Platform

8. REFERENCES

[1] URL RC_VISARD ROS package in github https://github.com/roboception/rc_visard_ros

[2] URL Tutorial for RC_VISARD ROS package use http://wiki.ros.org/rc_visard/Tutorials/TagDetect%20Module

[3] URL rc_visard interfaces in GIT https://github.com/roboception/rc_visard_ros

[4] URL Tutorial of RC_VISARD drivers http://wiki.ros.org/rc_visard

[5] URL ROS driver for the TagDetect module http://wiki.ros.org/rc_tagdetect_client?distro=melodic