Human-robot co-working system for warehouse automation

Rafael Rey, Marco Corzetto, Jose Antonio Cobano, Luis Merino and Fernando Caballero
reyarcenegui@gmail.com, m.conflan@gmail.com, jacobsua@upo.es, lmercab@upo.es, fcaballero@upo.es
Service Robotics Laboratory
Universidad Pablo de Olavide
Seville, Spain

***Abstract*—This paper addresses the material handling problem (MHS) in warehouse automation by proposing a system that uses an automated guided vehicle (AGV) in industrial environments. The aim is to optimize the picking task with respect to manual operation in a paint factory. The work describes the whole system to perform all the automatic tasks. The process is controlled by the Manufacturing Process Management System (MPMS) and an autonomous co-worker robot execute the mission in partially known environments. The navigation system implemented is safe and robust. It considers the people detection and unknown static obstacles. Also, an ultra-wide-band localization system is implemented by offering new capabilities for situation awareness in factories. Experiments with a real holonomic platform called ARCO are performed to validate the approach.**

***Index Terms*—AGV, warehouse automation**

I. INTRODUCTION

In the last years, the industrial robotics in the smart factories of the future is being fostered. Advances of smart sensors, electric and electronic technologies, and manufacturing technology are conducting to the intelligent manufacturing technology. Currently, Europe, United States and China are paying more attention to this area [1] [2] [3]. The International Federation of Robotics (IFR) estimates that by 2019 more than 1.4 million new industrial robots are being installed in factories around the world [4].

The smart factory, also known as Industry 4.0 [5], represents a leap forward from more traditional automation to a fully connected and flexible system. Concretely, Industry 4.0 refers to a phase in the industrial production evolution that focuses heavily on inter-connectivity, automation, machine learning, and real-time data processing.

Therefore, the current context requires more and more intelligent robotics solutions for the collaboration between humans and robots. Particularly, mobile robots play an important role to successfully enable key improvements in the manufacturing technology.

Automated guided vehicles (AGVs) are an example of efficient warehouse systems by replacing human with mobile robots or collaborating with them. The popular Kiva robot [6] from Amazon is a good example, although other system can be also found in the market. These systems are used in large-sized factories.

Different approaches have been presented in the literature for AGV on intelligent warehouses, many of them based on forklifts. Simulated forklift AGV systems with focus on the path planning problem is presented in [7]. It allowed that robotic forklifts executed transportation tasks in an intelligent warehouse-like environment. All was done in simulation and authors did not consider unknown obstacles like people walking or other vehicles moving in the warehouse. A large number of studies has been done in [8] including AGV design, flowpath layout design, collision and deadlock avoidance, route selection and scheduling problem. The work presented in [9] considered several robots and compare two collection strategies in simulation. In [10], a navigation system of a flexible AGV intended for operation in partially structured warehouses is presented. However, person detection and tracking is not considered by the approach.

Generally, the factory space is split between human and robot areas. This is counterproductive to perform coordinated joint human-robot tasks. Moreover, the autonomous navigation areas or paths of the AGVs are constrained to spaces with reduced presence of people, reducing the efficient of the factory.

This paper addresses the material handling problem (MHS). AGVs are widely chosen by companies to implement robust, safe and truly MHS. The work presents a human-robot co-working systems for Small and Medium-sized enterprises (SMEs) enterprises in partially known and unstructured environments. The system will be validated in the non-automated production line of a factory which produces water-based paints. The main objective of this system is to optimize the picking task with respect to manual operation. Nowadays, this operation is carried out by an operator with a cart going to the different warehouse sections to pick-up the material in the right quantities for later insertion in the mixing tanks. One of the advantages of the system proposed will be to improve the carry of the materials with the AGV and avoid risky situations of the worker by reducing the worker exposure during the operation.

The main contribution of this paper is the development of a system able to perform all the automatic tasks without requiring major changes in manufacturing process. The warehouse

This work is partially supported by the ARCO experiment funded by the HORSE project (Grant number 680734) under the Horizon 2020 Framework Programme.

Fig. 1: ARCO robot used in the experiments (left) and current ARCO robot to use in a factory (right).

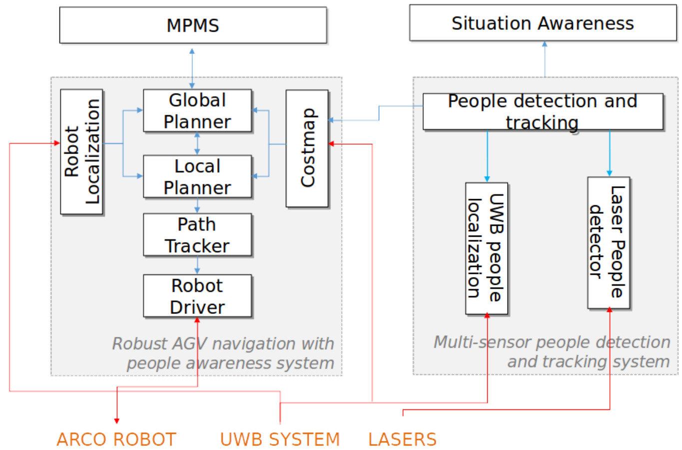

Fig. 2: ARCO ROS software architecture.

process is controlled by a Manufacturing Process Management System (MPMS) and it has been designed using the Business Process Model and Notation (BPMN v2.0) which, as the name suggests, it is a standard graphic notation for business processes modelling, maintained by the Object Management Group[1]. BPMN is generally used to model processes for administrative, financial or insurance businesses. The use of BPMN to model industry processes has been proposed by [11]. An implementation of BPMN on a smart industry 4.0 factory has been described in [12], while in [13] a BPMN solution has been applied for taking into account the human physical risks in manufacturing processes.

The autonomous co-worker robot used is known as ARCO robot (see Fig. 1). The robot will operate in conjunction with the workers to collect the raw materials associated with the production process, on the transportation of these materials and assisting on the scheduling of mixing quantities and timings. Also, an ultra-wide-band (UWB) localization system for people detection has been implemented. The inclusion of this localization system offers new capabilities for situation awareness in factories.

The main contributions of the paper are summarised bellow:

1) Implementation of an autonomous AGV navigation system able to operate in a factory without the need of cables, line-painting or RFID installation in the floor.
2) Implementation of ultra-wide-band localization system for people detection that will provide new capabilities for robot situation awareness.
3) Low computation time of the algorithms implemented in order to ensure safety when obstacles or unexpected events take place.
4) Experimental validation with a real platform.

The paper is organised into seven sections. Section II presents the software architecture of the system proposed. The navigation of the AGV is described in Section III. Section IV shows the UWB localization system and Section V the MPMS application. Experiments and results are presented in Section VI. Finally, conclusions are exposed in Section VII.

II. ARCO SOFTWARE ARCHITECTURE

The aim of the proposed architecture is to provide the required functionalities for AGV autonomous navigation in a factory, adapting to possible changes of the environment. Figure 2 shows the ROS architecture of the AGV navigation module and People detection and tracking module of the proposed system. It allows the AGV (henceforth ARCO) to execute transportation tasks in an intelligent warehouse-like environment.

To the software architecture has been designed following the guideline given by the HORSE Architecture Team (see [14] for more details) . The process is controlled by a Java application, the Manufacturing Process Management System (MPMS), which is running on a laptop acting as a server. The application has been developed over a Java Spring Boot framework [2], because it allows dependency injection, and it has an embedded Tomcat servlet. The MPMS has an embedded Camunda [3], an open source platform which gives to the BPMN model the ability of being executed and to control in real time the whole manufacturing process. Every task defined on the designed model can execute scripts or Java code. Furthermore, Camunda offers an embedded web application that allows monitoring and managing the process execution in real time, to watch variables values and to interact with the application through forms.

The manufacturing orders are received by email. They contain the list of raw materials, together with the amount that are needed for each of them. Then, the navigation goals are given one by one by the MPMS application to ARCO. The robot does not receive information about the manufacturing order or the sequence of goals it has to reach, all the process handling is performed by the MPMS application. Once a goal is received by ARCO, it should navigate efficiently and

[1] http://www.bpmn.org

[2] https://spring.io/projects/spring-boot

[3] https://docs.camunda.org/manual/latest/introduction/framework

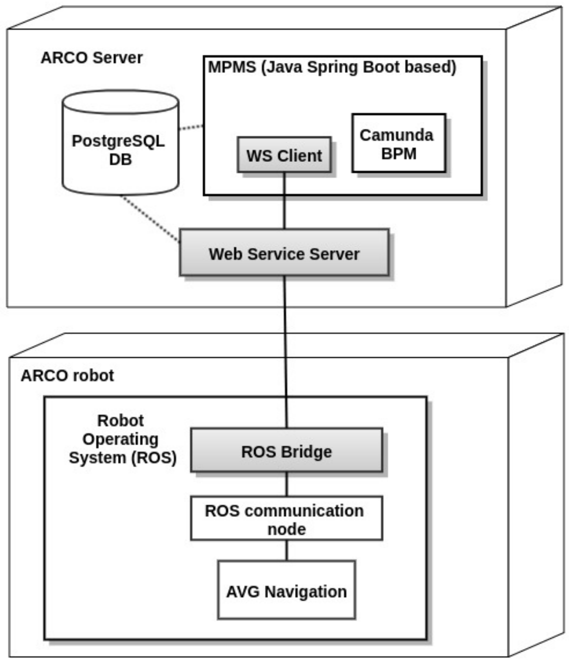

Fig. 3: ARCO communication architecture.

safely in the environment. During the process, the MPMS continuously checks the robot status.

The workers will interact with the system through a web application they can access in their phones or tablet. They can use forms to verify the amount of raw materials needed and they can interact with the system sending confirmations when their tasks are accomplished.

The communication between the MPMS and the ARCO robot has been set up through a web service server developed in Java which runs on the ARCO server (see Fig. 3). The robot uses two ROS modules for communication: the ROS bridge node acts as a web service client, and the ROS communication node acts as an interpreter that translates the web server messages into ROS topics, and vice versa.

III. ROBUST AGV NAVIGATION

This section presents the software developed to perform robust autonomous navigation in the environment considered. The basic ARCO's autonomy consists of localization, perception, path planning and path tracking. These modules are detailed in next paragraphs.

A. Robot Localization Module

In order to perform autonomous navigation, a map of the environment and a method to localize the robot in it are needed. Localization plays an important role in mobile robots. This module localizes the robot in a previously generated map of the environment and provides this information to the Global and Local Planner modules (see Fig. 2).

The sensors used are one laser placed in the font and another in the rear part of the robot. A static map of the environment should be computed during robot deployment. This laser-based map is generated by using the *gmapping* ROS package [15]. This package provides laser-based SLAM (Simultaneous Localization and Mapping), so a 2-D map is built from both lasers and pose data collected by ARCO.

The robot localization is based on the AMCL (Adaptive Monte Carlo Localization) which is a probabilistic localization system for a robot moving in 2D [16]. Given a map of the environment, this module estimates the position and orientation of a robot as it moves and senses the environment. The algorithm makes use of a particle filter to represent the distribution of likely states, with each particle representing a possible state, i.e., an hypothesis of where the robot is. The odometry estimates the pose of the robot by integrating the robots internal states such as speed and steering angle, which are computed from the wheel's optical encoders readings. The AMCL then compensates the error of this estimation, which is called odometry drift.

B. Global Planner

Once a reliable localization of the robot and a goal is provided, a trajectory is generated by the Global Planner. This global trajectory is defined by a sequence of points. Note that the Global Planner module will interact with the MPMS, moving the robot to the different places where to pick the materials. Therefore, inputs of the Global planner are the localization of the robot, goal to reach and costmap of the static map (global costmap).

The developed Global Planner is based on a variation of the any-angle path planning algorithm Theta* called Lazy Theta* [17]. Lazy Theta* is an optimization of Theta* which reduces the number of line of sight checks. An advantage of this algorithm is the optimization of the computational load. Other advantages of this graph-based algorithm with respect to sampling-based algorithms is that its result is closer to the shortest path and its solution is always the same. Furthermore, it gets a very low computation time and a high repeatability. The last advantages make this algorithm suitable for the proposed system by ensuring safety and robustness.

Algorithm 1 shows the pseudo-code of the Lazy Theta* algorithm. The algorithm is based on the A* algorithm, and this last one is basically the Dijkstra algorithm with Euclidean distance as heuristic. Theta* takes into account the line of sight between neighbours allowing any-angle paths, and finally Lazy Theta* reduces the line of sight checks improving computation times by keeping the same paths as Theta*.

Lazy Theta* starts expanding the first vertex and calculates the g and h values of its visible neighbours. After this step, it repeats the same procedure for every unexpanded visible neighbour s and checks if these neighbours have line of sight with the parent of his parent (start vertex at this moment). Instead of checking the line of sight between two vertexes s and s', Lazy Theta* assumes that they do have line of sight and at first it considers only Path 2 (ComputeCost in line 29), it checks later if there exists line of sight in the SetVertex procedure (line 34) and only changes the g-value and parent

Main()
open := closed := $\emptyset$;
g(s_{start}) := 0;
parent(s_{start}):=s_{start};
open.Insert(s_{start},$g(s_{start}) + h(s_{start})$);
while *open* $\neq \emptyset$ **do**
  s := open.Pop();
  SetVertex(s);
  If $s = s_{goal}$ **then**
    return *"path found"*;
  closed := closed $\cup s$;
  foreach $s' \in nghbr_{vis}(s)$ **do**
    If $s' \notin$ *closed* **then**
      If $s' \notin$ *open* **then**
        g(s') := ∞;
        parent(s') := NULL;
      UpdateVertex(s,s');
  return *"no path found"*;
end
UpdateVertex(s,s')
g_{old} := $g(s')$;
ComputeCost(s,s');
If $g(s') < g_{old}$ **then**
  If $s' \in$ *open* **then**
    open.Remove(s');
  open.Insert(s',$g(s') + h(s')$);
end
ComputeCost(s,s')
`/* Path 2 */`
If $g(parent(s)) + c(parent(s), s') < g(s')$ **then**
  parent(s') := parent(s);
  $g(s')$:= $g(parent(s)) + c(parent(s), s')$;
end
SetVertex(s)
If *NOT lineofsight(parent(s),s)* **then**
  `/* Path 1 */`
  parent(s) := *argmin* ($s' \in nghbr_{vis} \cap closed(g(s') + c(s, s'))$);
  g(s) := min ($s' \in nghbr_{vis} \cap closed(g(s') + c(s, s'))$);
end

Algorithm 1: Lazy Theta*

of s if there is no line of sight. The same procedure is repeated until the checked vertex is the goal vertex. For more detailed information see [17].

Figure 4 shows some of the advantages of Lazy Theta* algorithm in a complex scenario where the solution is difficult to find. In this case the maze shown has a lot of dead ends. For this non-realistic extreme case Lazy Theta* computes the optimal shortest path in 3 seconds.

C. Local Planner

The Local planner makes use of a segment of the global trajectory provided by the Global planner. This planner plays an important role in mobile robotics to re-plan quickly a safe trajectory when an unexpected event takes places such as unknown obstacles, persons appearing nearby the robot, etc. Moreover, this module should provide a free-collision trajectory to the Path tracker module.

A smarter AGV navigation should take into account the detection of unknown obstacles into the trajectory in advance for re-planning in real time thanks to a precise obstacle positioning knowledge. The detection is achieved from the measurements of the frontal and rear lasers, and the generation of the local costmap. Obstacle avoidance and the following of the global trajectory is performed by the Local Planner.

Therefore, inputs of the Local Planner are: a part of the global trajectory to follow, the costmap of a local map around the AGV generated from both lasers (local costmap), and localization of the robot. The output is the trajectory, defined as a sequence of waypoints, that the robot should follow. Note that the local costmap is centered in the ARCO robot and its dimension can be set up. In order to ensure safety and a quick reaction of the robot, the frequency of the Local Planner is high (about 20Hz).

The implemented Local planner is well adapted to changes of the environment thanks to the local costmap generated from the lasers and the efficient and fast path planning algorithm. This algorithm is also based on Lazy Theta*. It considers a reduced costmap and only a part of the global trajectory generated by the Global Planner. Figure 5 shows both global and local trajectory. Local trajectory is computed by considering the local costmap (square centered in the robot position). Note that the obstacles detected in the local costmap are inflated (in light blue) in order to consider a safety margin.

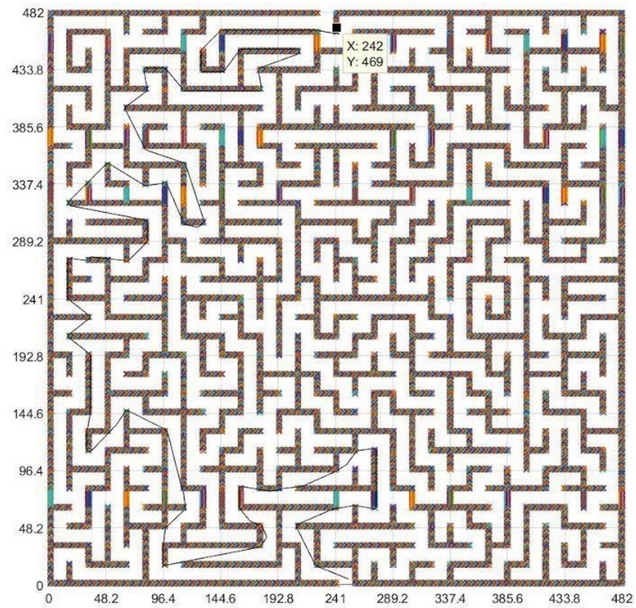

Fig. 4: Example of trajectory computed by the Lazy Theta* algorithm. The initial position (black square) is in the top and the final position at the bottom of the image.

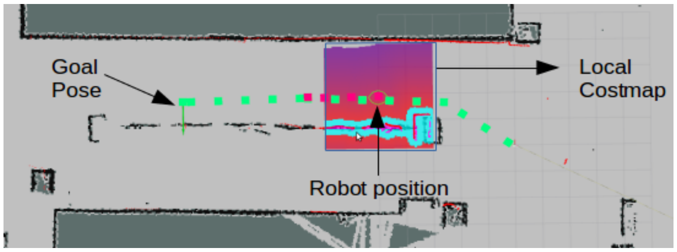

Fig. 5: Global trajectory (green square) and local trajectory (red square) computed by the Local Planner. Local costmap around the robot position is also shown.

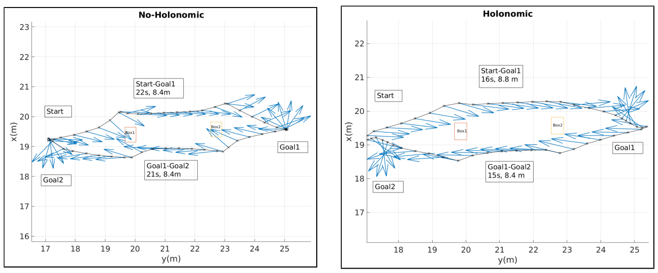

Fig. 6: Navigation modes implemented.

D. *Path tracker*

The Path tracker module carries out the navigation of the robot. It computes the linear and angular velocity commands to navigate and follow the trajectory computed by the Local planner. For the sake of safety, an emergency stop system has also been implemented in the Path tracker module. It is continuously checking if some obstacles (for example people crossing) whose distance is lower than a threshold is detected by the lasers. When this happens, ARCO immediately stops and it moves again only when the detected obstacle has moved away a distance greater than a second threshold.

There are two navigation modes implemented. The first one sends commands to move ARCO in straight line, straight line and slow rotation, and rotation in place, depending where the next point is. The second one considers the omnidirectional capabilities of ARCO. Both modes take into account the inertia of the robot to perform safe movements.

Figure 6 shows a test by using each navigation mode. The position provided by AMCL and the heading of ARCO robot are shown. Note that the final maneuver shows abrupt changes of the heading when the holonomic navigation is used. This is because ARCO turns to get the desired heading when the distance to the goal is lower than a threshold. The threshold is 1.5 m in the tests performed.

IV. UWB LOCALIZATION SYTEMS FOR PEOPLE DETECTION

One of the key features of the ARCO system is the direct integration of people into the robots decision making process. This way, the robot can act and decide considering the presence of humans in the surrounding area.

People detection is particularly interesting in order to build robot intelligence that treats persons differently, not just as obstacles. This information can be used to build more efficient solutions for safety and human acceptance.

Most of the approaches in the state of the art make use of cameras and LIDAR for automatic people detection. While these algorithms works well in general, they use to be limited to detections in the sensor vicinity, and occlusions have a great impact in the algorithm performance. In this sense, the use of an ultra-wide-band (UWB) system provides a cost-efficient and reliable localization system. Normally, this technology cannot be used for people detection in the wild because it requires the people to carry an UWB tag to be localized. However, the environment can be much more controlled in factories, so that workers can carry a tag, at least the ones working in the warehouse.

Thus, ARCO has developed a module for range-only localization, it was also integrated with a low-cost UWB system. This software makes use of the individual UWB range measurements and provides a position estimation of the tags carried by the operators. Based on previous authors works on range-only simultaneous localization and mapping (RO-SLAM) [18] [19], this module implements an Extended Kalman Filter (EKF) that integrates temporal constraints and range measurements to estimate the position of the tag. This module can be used to localize the robot and the people. While filter updates are identical in both cases, different prediction stages have been implemented in order to account for the robot odometry or the human motion model respectively.

The experiments presented in Section VI shows that persons can be localized with an averaged error of 20 cm approximatelly when the area is well covered by the UWB anchors. An interesting benefit of the developed approach for localization is that it does not need three measurements to estimate the tag position in 2D. Thanks to the temporal constraint imposed by the filter, two range measurements are enough to recover the tag pose.

V. MANUFACTURING PROCESS MANAGEMENT SYSTEM

The BPMN models described below have been designed with Camunda Modeler, and then they have been integrated into the MPMS application.

Camunda has a web application that can be accessed via login by the system administrator or by the workers. The application is employed to start and visualise the process status, and perform manual tasks with forms.

The main process begins when the orders handling process is started from the Camunda Tasklist (see Fig. 7). The orders handling process runs on two paths in parallel: the first one reads the email inbox folder checking whether there are new incoming orders, and it stores them into a PostgreSQL database. Then, one internal signal named ”email received” is thrown. On the second path, a task reads the database for unattended orders and if it finds any, it activates the warehouse process. If there are no new order on the database, the token stops until the ”email received” signal is caught. This signal has the role to avoid unnecessary readings from the database.

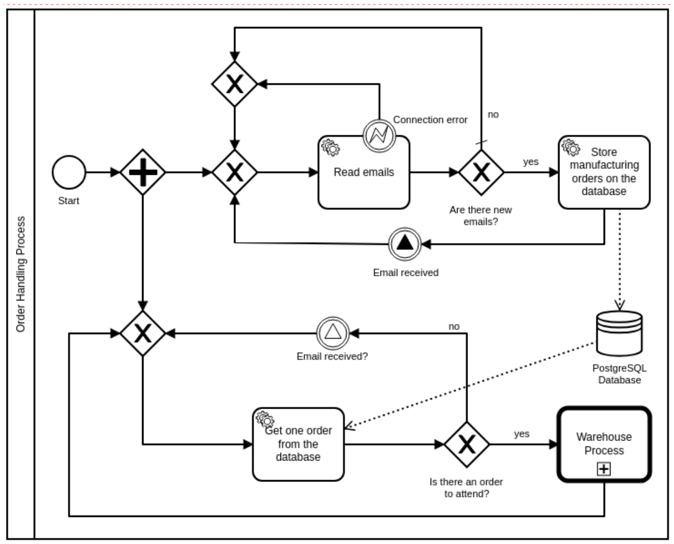

Fig. 7: BPMN model: orders handling process.

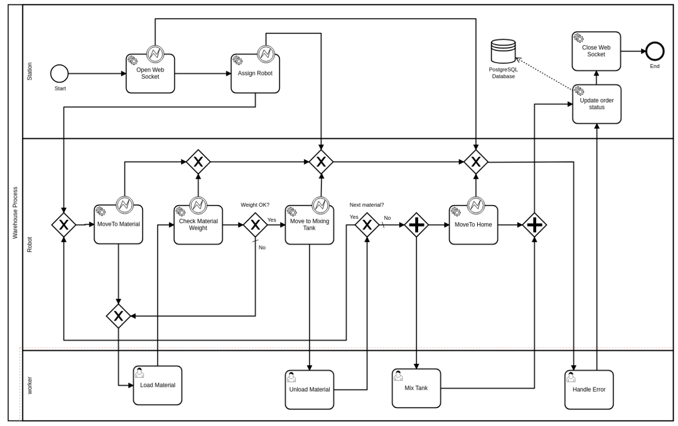

Fig. 8: BPMN model: warehouse process.

Fig. 9: Places where to pick the boxes and mixing tank in the testing environment considered.

The task "Read emails" has an error catcher that allows the system not to hang out if the email connection fails.

The warehouse process adds the use of lanes that represents which actor is going to perform each task. The tasks performed by the workers are manual tasks, which are performed through forms that can be accessed via the Camunda web app.

The warehouse process starts establishing the connection with the web server, then waits for the robot being ready (see Fig. 8). Once ARCO confirms the availability, then the "Move To Material" task sends to the robot the coordinate where the first raw material has to be picked up. One worker must go to the goal position and load the robot with the correct amount of raw material. Once the worker has loaded the material, it sends a confirmation through a form and then the MPMS checks if the weight of the raw material is correct. The robot then moves to the mixing tank. The material is unloaded manually and then, the robot goes to the next material location or, if all the materials have been added to the mixing tank, the robot goes back to the home position and the worker activates the mixing procedure. Finally, the order status is updated in the database and the connection with the web server is closed. The warehouse process ends and the orders handling process keeps querying the database for new incoming orders. If some error is caught while running the process, the worker must access a form and decide if the order must be cancelled, or completed manually.

VI. Experiments and Results

A testing environment that simulates the factory has been implemented at the University (see Fig. 9). There are two corridors and several places where to pick materials simulated with boxes (A, B and C). During the experiments, people will approach ARCO and some obstacles will appear in the corridors in order to test the re-planning capabilities of the robot.

The operating system used is Ubuntu 16.04. The code has been written in C++ language and has been integrated in ROS (Robot Operating System) framework. Particularly, with ROS Kinetic Distribution.

Experiments carried out show the whole mission. First, an order is received by the MPMS application. Then, the navigation goals are given one by one by the MPMS application to ARCO. In the testing environment considered, ARCO should reach A, B and C points to pick boxes and introduce each one in the mixing tank. Therefore, the order of the points is: A, tank, B, tank, C and tank.

A simplified BPMN model (Fig. 10) has been used in these initial experiments, as the weighing system was not yet available at that moment.

The tests performed evaluate the following characteristics of the proposed system:

1) Full mission given by the MPMS application: ARCO navigates efficiently and safely during the execution.
2) Detect and avoid: static obstacles (boxes) and people approaching to the ARCO are considered. ARCO should

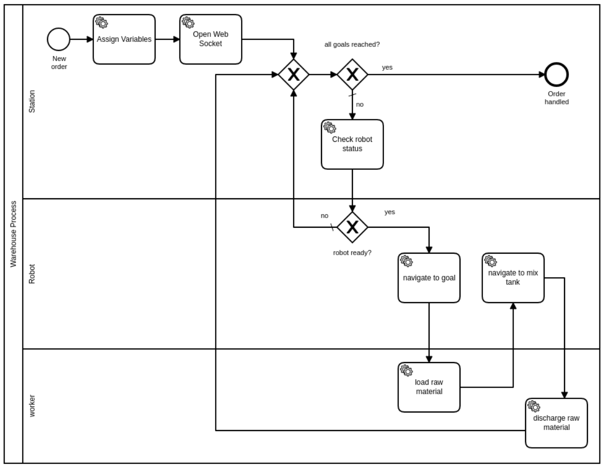

Fig. 10: BPMN model used during the experiments.

react correctly avoiding the obstacles or stopping when obstacles are very close (the emergency stop system is 20 cm).

The system has been tested for hours in the testing environment and all of them have been successful. A video of one of the tests performed can be seen in here[4]. It shows the reference system of ARCO (position of the robot), the static map generated of the environment (dark pink), the global costmap (it represents the inflated static map in light blue), the local costmap (red square centered in the ARCO in red) and the global path (green small square) and local paths (red small square).

Figure 11 depicts the robot trajectory of a performed mission (shown in the video). The Home position, wall, the places where to pick the boxes and the tank are shown. The distance travelled in this mission provided by the localization system is $62.7m$. The video shows a person approaching the robot at the time 1:45 and the ARCO reacts avoiding them. This manoeuvre can be seen in the segment Tank to B of the Fig. 11. ARCO moves to the right side in order to avoid the detected obstacle.

The UWB system has also been tested in real experiments. In order to evaluate the precision of the UWB system, an UWB tag was installed into the ARCO robot and the resulting information was compared to the information from AMCL laser based localization. Figure 12 shows the UWB and the AMCL localization. Figure 13 shows the relative error of the UWB localization with respect to the AMCL localization, it can be seen how the average localization error is below $0.2m$ and that the error in Y is larger than X, this is mainly produced by the lack of trilateration in Y in this experimental setup. A better distribution of the UWB anchors will definitely help to reduce such errors.

The computation time of the Global planner depends on the distance between the initial position and the goal. For example,

[4]https://drive.google.com/open?id=1sZj6EXf0y_3oj-LlRG498GurO9wtaAvL

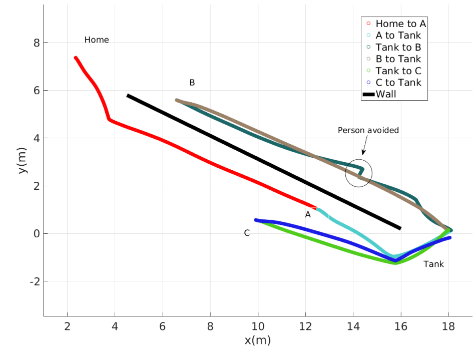

Fig. 11: ARCO trajectory in the mission recorded in the video Full_Mission_01.m4v.

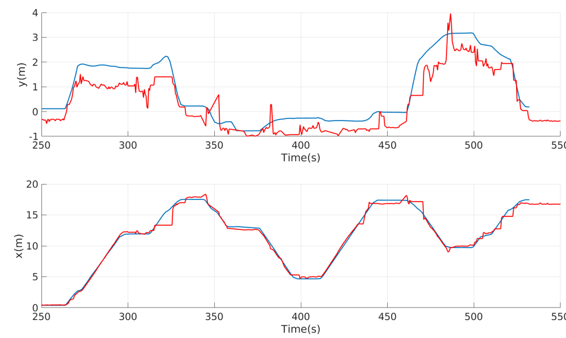

Fig. 12: ARCO localization from AMCL localization (blue line) and UWB estimation (red line).

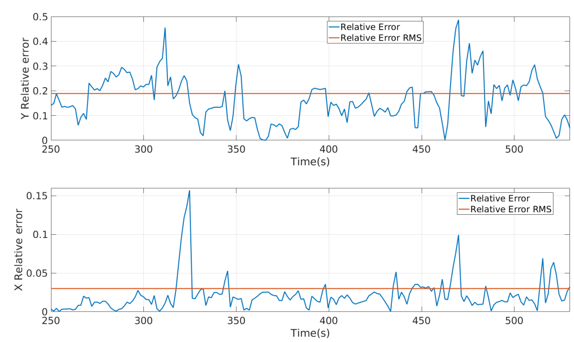

Fig. 13: Relative error between UBW localization and AMCL localization.

the computation time for distances greater than 10 m is 100-120 ms approximately. For distances shorter than 10 m, the global path can be computed in 40 ms approximately. On the other hand, the computation time of the Local planner depends on the number of obstacles detected. When ARCO moves in straight line without obstacles, the mean computation time is 1.13 ms, the standard deviation is 1.18 ms and the maximum computation time is 10 ms. In the cases where ARCO detects obstacles and the Local planner should re-plan the global path to avoid them, the mean computation time is 3.78 ms, the standard deviation is 4.43 ms and the maximum computation time in occasional cases is 42 ms. An important characteristic of the implementation of the Local planner is that its execution frequency is of 23Hz, so the this planner is very suitable for the application considered in this paper as it ensures a quick reaction when obstacles are detected.

Note that an emergency stop system is running at 40Hz, this provides an emergency response. If an emergency stop takes place, ARCO will move again when the distance to the obstacle or person is greater than 0.6 m.

VII. Conclusions

In this paper, a human-robot co-working system for warehouse automation has been presented. The system uses an automated guided vehicle (AGV) called ARCO and a Manufacturing Process Management System (MPMS) controls all the process.

The proposed system is able to carry a complete mission, navigating efficiently and safely into the testing environment. The system has been tested in a real environment. The duration of all the tests and the absence of errors in the performed tests validate the system.

The MPMS achieved the goal of controlling several processes at the same time, on an heterogeneous ensemble of connected devices. The order handling and the robot navigation were successfully managed, and the process execution was easily monitored in real time through the cockpit. Camunda BPM turned out being an effective tool for implementing processes into a connected industry 4.0 environment.

Future work will consider the integration of a weighting system for the raw materials. Also, kinodynamic constraints will be considered into the patch tracker or the planner so that ARCO can act considering the mass of the system.

References

[1] E. Commission, "Europe 2020: A strategy for smart, sustainable and inclusive growth," Brussels, Belgium, 2010.

[2] J. P. Holdren, T. Power, G. Tassey, A. Ratcliff, and L. Christodoulou, "A national strategic plan for advanced manufacturing," URL: http://www.docin.com/p-391856652.html, National Institute of Standards and Technology, Tech. Rep., 2012.

[3] J. Zhou, "Intelligent manufacturing-main direction of made in china 2025," in *China Mech. Eng.*, vol. 26, no. 17, 2015, pp. 2273–2284.

[4] I. F. of Robotics, "Robots double worldwide by 2020," URL: https://ifr.org/ifr-press-releases/news/robots-double-worldwide-by-2020, 2018.

[5] H. Lasi, P. Fettke, T. Feld, and M. Hoffmann, "Industry 4.0," in *Business & Information Systems Engineering*, vol. 6, no. 4, 2014, pp. 239–242.

[6] F. D. P. R. Wurman and M. Mountz, "Coordinating hundreds of cooperative, autonomous vehicles in warehouses," *AI Magazine*, vol. 29, no. 1, pp. 9–20, 2008.

[7] K. C. T. Vivaldini, J. P. M. Galdames, T. S. Bueno, R. C. Araujo, R. M. Sobral, M. Becker, and G. A. P. Caurin, "Robotic forklifts for intelligent warehouses: Routing, path planning, and auto-localization," in *Proceedings of the 2010 IEEE International Conference on Industrial Technology*, Viña del Mar, Chile, 14-17 March 2010, pp. 1463–1468.

[8] I. F. A. Vis, "Survey of research in the design and control of automated guided vehicle systems," *European Journal of Operational Research*, vol. 170, pp. 677–709, 2006.

[9] N. Pinkam, F. Bonnet, and N. Y. Chong, "Robot collaboration in warehouse," in *In Proc. ACCProceeding of the 16th International Conference on Control, Automation and Systems (ICCAS 2016)*, Gyeongju, Korea, 16-19 October 2016, pp. 269–272.

[10] H. Martinez-Barber and D. Herrero-Prez, "Autonomous navigation of an automated guided vehicle in industrial environments," *Robotics and Computer-Integrated Manufacturing*, vol. 26, pp. 296–311, 2010.

[11] S. Zor, F. Leymann, and D. Schumm, "A proposal of bpmn extensions for the manufacturing domain," in *Proceeding of the 13th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Khon Kaen (Thailand), 13-15 July 2016.

[12] R. Petrasch and R. Hentschke, "Process modeling for industry 4.0 applications. towards an industry 4.0 process modeling language and method," in *Proceeding of the 13th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Khon Kaen (Thailand), 13-15 July 2016.

[13] M. Polderdijk, I. Vanderfeesten, J. Erasmus, K. Traganos, T. Bosch, G. van Rhijn, and D. Fahland, "A visualization of human physical risks in manufacturing processes using bpmn," in *In Business Process Management Workshops - BPM 2017 International Workshops*, 2018, pp. 732–743.

[14] I. V. P. Grefen and G. Boultadakis, "D2.2a - complete system design," URL: http://horse-project.eu/sites/default/files/publications/HORSE-D2.2%20(Public%20Version).pdf, Eindhoven University of Technology and European Dynamics, Tech. Rep., 2015.

[15] C. S. Giorgio Grisetti and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, pp. 34–46, 2007.

[16] W. B. S. Thurn and D. Fox, *Probabilistic Robotics*. The MIT Press, 2005.

[17] A. Nash, S. Koenig, and C. Tovey, "Lazy theta*: Any-angle path planning and path length analysis in 3d," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, Atlanta, Georgia, USA, 11-15 July 2010, pp. 147–154.

[18] F. Fabresse, F. Caballero, and A. O. I. Maza, "An efficient approach for undelayed range-only slam based on gaussian mixtures expectation," *Robotics and Autonomous Systems*, vol. 104, pp. 40–55, 2018.

[19] F. Caballero, L. Merino, and A. Ollero, "A general gaussian-mixture approach for range-only mapping using multiple hypotheses," in *IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, AK, USA, 3-7 May 2010, pp. 4404–4409.