Proceedings of the ARW & OAGM Workshop 2019 DOI: 10.3217/978-3-85125-663-5-13

Adaptive Loading Station for High-Mix Production Systems

Alexander Raschendorfer[1], Michael Hofmann[2], Florian Pauker[3], Marcel Fuschlberger[1], Martin Kunz[1]

Abstract— **This paper presents a loading station for high-mix production systems in a production shift without factory workers. A UR10e is used to load and unload a pallet, fitted on an Autonomous Guided Vehicle (AGV). The robot is supported by the software XRob and cameras for visual detection of the raw parts as well as the raw part trays. Additionally the cameras are used to correct the position of the AGV. The process itself is orchestrated by the workflow engine centurio.work.**

I. INTRODUCTION

The motivation behind creating an adaptive, automated loading station lies in two main fields of research. Firstly creating a system that can handle a production during a factory shift without workers and a limited supply of trays for handling raw parts. Thus limiting and reducing the amount of trays for the part handling on the shop floor level. Reduced amounts of trays required for work piece handling lower the costs of the workflow as well as the environmental footprint. Less storage areas are required and the trays can easily be reused for similar raw parts. Pallets and handling equipment for raw parts is often highly specialized and only serves the single purpose to bring the piece to the first process stage in the manufacturing area. The trays used in this use case carry the raw parts as well as the finished product after machining. Secondly using the centurio.work engine to orchestrate the loading of a modular pallet with trays for individual raw parts in combination with a visual detection of the raw parts by cameras mounted onto the robot.

For demonstration reasons three parts were defined for the first stage of the use case. The three parts are chess pieces (Bishop, Rook and Pawn) and are all turned on an EMCO MaxxTurn 45. The raw parts are cylinder of varying diameter and height.

II. DESCRIPTION OF THE LOADING STATION

This section describes the hard- and software used for the loading station.

A. Layout and Hardware

The basic layout, as seen in Fig. 1 of the use-case consists of an UR10e-Robot by Universal Robots, a Neobotix AGV and Trays, that can be fitted on a special pallet, that is moved by the AGV and used within the production of the "Pilotfabrik 4.0" of the Vienna University of Technology. The UR10 of the e-Series is equipped with a range of built-in safety functions as well as various control signals to or from the electrical interface, to connect to other machines and devices [12]. The gripper used is a Robotiq 2F for Universal Robots. It is an electric, adaptive two-finger gripper with a maximum clamping force of 125 N.

The Neobotix MP-400 is an Autonomous Guided Vehicle (AGV) by the company Neobotix. Its maximum payload is up to 150 kg. For the AGV to work, no guidewires in the floor for steering are necessary. Instead the stations and roads between these stations are virtually designed within the Neobotix software. The whole environment can easily be designed within the software provided by Neobotix using the AGV's integrated vision system. Once everything is set up the AGV can move freely between predefined stations, utilizing the routes and maneuvering autonomously around obstacles in its way [5]. The AGV carries a pallet, that is used within the production of the "Pilotfabrik 4.0", and that can be equipped with individual trays for holding the raw parts as well as the finished product.

The trays are designed to fit several needs. At first the trays need to safely carry the raw parts and the finished product. Additionally the trays need a handle, so the UR10e can grip and position the trays on the pallet. Depending on the diameter of the raw part an individual tray, fitting that diameter, is used. Lastly the tray needs to contain information about the product as well as information about the positioning of the tray so the UR10 and the connected vision systems can locate the trays. The trays are created using the FDM 3D-printing method.

B. XRob

The XRob software framework[9] enables the creation of complex robot applications within fewer minutes. It builds on unique, easy-to-use features that significantly speed up commissioning and make the operation more costefficient and flexible than common programming methods. The special software architecture allows easy and intuitive creation of processes and configuration of the components of a robot system via a single user interface. Fig. 2 provides an overview on the software components within the XRob framework.

Object Recognition: The aim of 3D Object Recognition is to localize the pose and position of an object of interest in the scene. Given the 3D model of the object, the goal is to find a correct transformation (6DOF) of the 3D model in the point cloud reflecting the current scene. A 3D model can be obtained either by 3D reconstruction or based on

[1]Alexander Raschendorfer, Marcel Fuschlberger and Martin Kunz are with the Austrian Center for Digital Production, Seestadtstrasse 27, A-1220 Vienna, Austria a.raschendorfer@acdp.at

[2]Michael Hofmann is with PROFACTOR GmbH, A-4407 Steyr-Gleink, Austria michael.hofmann@profactor.at

[3]Florian Pauker is with University of Vienna, Faculty of Computer Science Workflow Systems and Technology Group, Waehringerstrasse 29, A-1090 Vienna, Austria florian.pauker@univie.ac.at

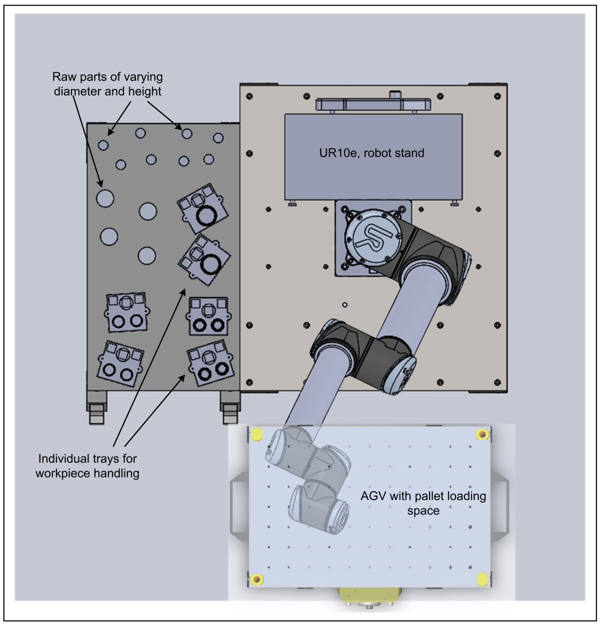

Fig. 1. Layout of the loading station

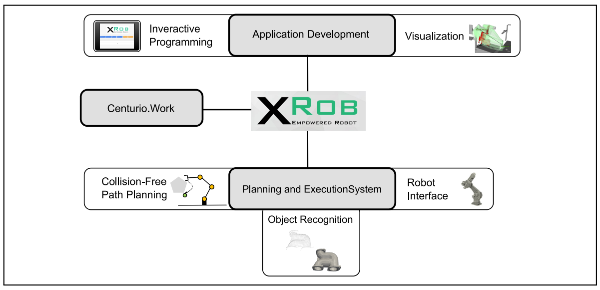

Fig. 2. Overview on the software components within the XRob framework.

the CAD model of the object, which is transformed into a point cloud during configuration. The 3D Object Recognition module is based on the Randomized Global Object Localization Algorithm (RANGO) [3], which extends the Random Sampling Algorithm [6]. Resulting object detections are used to plan collision-free robot movement paths for object manipulation. The accuracy of the 3D object recognition approach described above greatly depends on the sensor data quality and on the sizes of the objects of interest. Especially for very small objects, e.g. screws, 3D object recognition does not lead to reasonable detection results. To overcome these limitations, the XRob software framework supports 2D template matching and the detection of matrix markers [2] [1]. Matching templates can be configured on runtime, by positioning the sensor accordingly to record a template image.

Collision-Free Path Planning: The results of the Object Recognition and Localization system are used to plan and calculate collision-free robot manipulation paths [10] to enable handling of the detected objects. Based on predefined grasp as well as deposit points on the CAD model of the objects, the manipulation planner determines how the object can be grasped. All object localizations as well as the available workspace environment data are considered for collision checks. The path planning system is based on the Open Motion Planning Library (OMPL) [11] and follows rapidly-exploring random trees (RRT) [4] approach.

Robot Interfaces: To facilitate communication with the robotic system, the XRob framework provides a uniform communication interface, which can be extended in a Plug-In like fashion to support robotic systems of different vendors.

Application Development: The XRob software framework provides an intuitive user interface for application development, which includes an interactive programming environment, and software modules to simulate and visualize robotic movement paths as well as data acquisition via sensors.

C. Adaptive Loading Station for High-Mix Production Systems

The in XRob included process planner as described in II-B is one of the modules of XRob which are not easily reconfigured via programming. Instead it needs pre defined grasp points, the robot kinematics and a collision scene of the environment surrounding workcell. As a change of to be handled parts can not be easily programmed directly at the station a remote reconfiguration has to be done. The reconfiguration has to consider the consistency of the grasping points and the object model to be detected.

D. centurio.work

centurio.work is a process based framework for realizing a integration platform for manufacturing scenarios. Processes are widely used in companies for describing the workflow of business matters. In production environment they are not used explicit. The idea is using processes as unified context in a factory for orchestration bringing the flexibility from business down to shopfloor. Fig 3 shows the decomposition of the automation pyramid and how process can be used for connecting different functions on different layers.

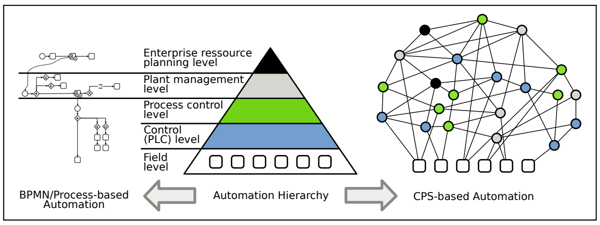

Fig. 3. Process based automation [8]

The architecture of centurio.work is shown in figure 4. It consists of several independent containers with defined information flow. Basically the framework differentiate between static and dynamic information. In the **Ressources** context all necessary information for process execution is stored. In this case the workflows and the robot programs, and information for XRob. The **Orchestration** context is important during runtime. The Process Execution as well as the Data Provisioner are located in this container. The Data Provisioner realizes the connectivity to Data Sources

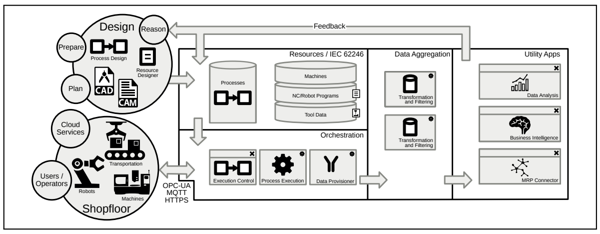

Fig. 4. centurio.work architecture [8]

like robots from different manufacturers. **Data Aggregation** container offers different solutions for storing the generated data during process execution in different forms. The idea is, that every application in the **Utility Apps** container has its own storage so no transformation is needed. Additionally a **Feedback** loop is integrated to improve static information during design.

III. Workflow and Communication

The workflows for orchestration are based on the approach presented in [7]. Atomic functions of the robot are modelled as single processes. The main processes are displayed in figure 5. The process itself is designed linear since the process steps necessary each require a successful step beforehand.

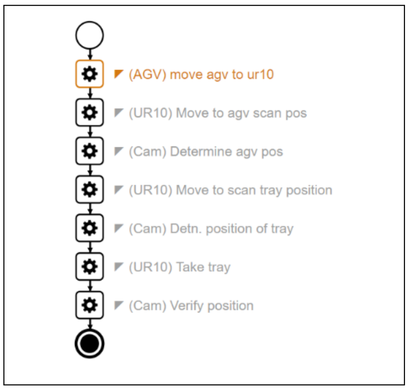

Fig. 5. centurio workflow of the loading process

Centurio.work starts with sending move orders to the AGV. The AGV has a fixed parking position right next to the loading station. The AGV needs to position itself within a small tolerance so the next processes can run on a stable basis. The process on the centurio.work level is finished once the AGV reports its final parking position. Next the UR10e is directed to move the mounted cameras over the pallet transported by the AGV. In the following step the camera determines the exact position of the pallet to get a matrix with correction values for the placing of the trays. This correction needs to be only done once for each loading or unloading process since the AGV does not move in the process. Once the position is determined the workflow engine moves the UR10 and its cameras to the area where the trays are located. The correct tray is selected and the exact position scanned. XRob provides the collision free paths so a safe picking of the tray is guaranteed. Centurio.work gives the order to execute the movements. The tray is placed on the is pallet. Once the trays are placed the same procedure starts with the raw parts. centurio.work saves the position of the trays on the pallet and so can easily place the right raw parts into the trays on the pallet.

In order to achieve a high flexibility, functions have been implemented that allow several raw parts to be handled by the softwarepieces. Centurio.work can send a container, containing a CAD file with predefined grip positions, to XRob.That way XRob can detect any form of raw part as long as it is within certain predefined boundaries. This allows for a very high flexibility on the shop floor and lot size one production. Changing raw parts can simply be pushed by centurio.work to the XRob visual recognition of the software with no worker interaction needed.

The communication between centurio.work and XRob is handled by REST (Representational State Transfer) calls. The corresponding module for communication in centurio.work is the Data Provisioner. One example is the ”GET /AGVPos” call. XRob returns the position of the AGV in Euler coordinates (rx, ry, rz, x, y, z) to the workflow engine.

IV. Conclusion and Future Work

The hughe flexibility in the system is a huge benefit of the presented use case. There are however a few design problmes of the whole system that shall be addressed. Currently the loading as well as the unloading process requires the AGV to be parked next to the loading station. This creates a dead time for the AGV itself without any productivity. The AGV cannot be used for any other tasks within the factory. Within a next iteration the loading station should be adapted to load and unload the pallet on a dedicated space without the AGV needed.

V. Acknowledgment

This work has been partially supported and funded by the Austrian Research Promotion Agency (FFG) via the Austrian Competence Center for Digital Production (CDP) under the contract number 854187.

References

[1] F. Bergamasco, A. Albarelli, and A. Torsello, “Pi-tag: a fast image-space marker design based on projective invariants,” *Machine vision and applications*, vol. 24, no. 6, pp. 1295–1310, 2013.

[2] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, “Automatic generation and detection of highly reliable fiducial markers under occlusion,” *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[3] P. Gmbh, “Candelor: A computer vision library for 3d scene interpretation,” 2017, http://candelor.com.

[4] J. J. Kuffner Jr and S. M. LaValle, “Rrt-connect: An efficient approach to single-query path planning,” in *ICRA*, vol. 2, 2000.

[5] *Neobotix Bedienungsanleitung MP_400*, NEOBOTIX, 04.2017.

[6] C. Papazov and D. Burschka, “An efficient ransac for 3d object recognition in noisy and occluded scenes,” in *Asian Conference on Computer Vision*. Springer, 2010, pp. 135–148.

[7] F. Pauker, I. Ayatollahi, and B. Kittl, “Service orchestration for flexible manufacturing systems using sequential functional charts and opc ua,” *Dubrovnik*, vol. 9, pp. 11–09, 2015.

[8] F. Pauker, J. Mangler, S. Rinderle-Ma, and C. Pollak, “centurio. work-modular secure manufacturing orchestration,” 2018.

[9] A. Pichler, S. C. Akkaladevi, M. Ikeda, M. Hofmann, M. Plasch, C. Wgerer, and G. Fritz, “Towards shared autonomy for robotic tasks in manufacturing,” *Procedia Manufacturing*, vol. 11, pp. 72 – 82, 2017, 27th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM2017, 27-30 June 2017, Modena, Italy. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2351978917303438

[10] A. Pichler and C. Wögerer, “Towards robot systems for small batch manufacturing,” in *2011 IEEE International Symposium on Assembly and Manufacturing (ISAM)*. IEEE, 2011, pp. 1–6.

[11] I. A. Sucan, M. Moll, and L. E. Kavraki, “The open motion planning library,” *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012.

[12] *Universal Robots e-Series User Manual*, Universal Robots, 2018.