Offline Programming of Collision Free Trajectories for Palletizing Robots

Ricardo Silva[1], Luís F. Rocha[1], Pedro Relvas[1], Pedro Costa[1,2], and Manuel F. Silva[1,3]

[1] INESC TEC - INESC Technology and Science, Porto, Portugal
rfsilva@inesctec.pt, luis.f.rocha@inesctec.pt, pedro.m.relvas@inesctec.pt
[2] FEUP - Faculty of Engineering of the University of Porto, Porto, Portugal
pedrogc@fe.up.pt
[3] ISEP - School of Engineering of the Porto Polytechnic, Porto, Portugal
mss@isep.ipp.pt

Abstract. The use of robotic palletizing systems has been increasing in the so-called Fast Moving Consumer Goods (FMCG) industry. However, because of the type of solutions developed, focused on high performance and efficiency, the degree of adaptability of packaging solutions from one type of product to another is extremely low. This is a relevant problem, since companies are changing their production processes from low variety / high volume to high variety / low volume. This environment requires companies to perform the setup of their robots more frequently, which has been leading to the need to use offline programming tools that decrease the required robot stop time. This work addresses these problems and, in this paper, is described the solution proposed for the automated offline development of collision free robot programs for palletizing applications.

Keywords: Palletizing, Industrial Robots, Simulation, Offline Robot Programming

1 Introduction

The use of robotic systems has been increasing in particular in the so-called Fast Moving Consumer Goods (FMCG) industry. Among the various applications of robotic systems in this industry, it should be highlighted the automated systems for the movement of raw materials and ingoing products and, fundamentally, the robotic systems for palletizing and packaging of finished products.

In particular, the palletizing activities are specially demanding not only in terms of operation speed but also due to the inherent complexity in the handling of different types of packaged products. Despite the common application of robotic technology solutions, these are generally designed to ensure high performance under well defined operating conditions (*i.e.* very specific in terms of operating standards). Therefore, a trade-off between the operational performance and the desired degree of flexibility is normally seen. In fact, because of

the type of solutions developed, focused on high performance and efficiency, the degree of adaptability of packaging solutions from one type of product to another is extremely low. Also, there is also high rigidity as regards a possible change in the configuration of the packaging system to new operating requirements, for example, integrated into a new layout of the manufacturing process.

From the perspective of the production process, this type of limitations significantly affects the flexibility of the production system itself, often preventing the development of production processes in a low volume/high variability environment [1]. In fact, after the efforts made in recent years by several industrial units to make production processes more flexible, allowing for a diversified and competitive response, it is clear that packaging/palletizing systems should follow this development: become more adaptable and flexible.

Following this set of identified problems, INESC TEC is developing a project having as its main objective, to investigate, study, explore and develop a framework that allows the design and development of adaptive palletizing solutions based on robotic technologies. The framework includes a set of innovative elements that efficiently create and evaluate palletizing solutions and, at a later stage, design and develop the solution to be produced, based on an innovative concept of modularity (modular functional elements) and advanced technology for offline programming of the robotic systems.

This project constitutes a unique opportunity for a qualitative leap in the industry equipment, which provides palletizing solutions, evolving from a paradigm based on the use of conventional design and development practices to a new paradigm of solution creation, based on technologically advanced modular architectures that are adaptive and intelligent (aligned with the principles and concepts of industry 4.0).

Bearing these ideas in mind, the next section describes the main ideas behind this project, followed by Section 3 that presents the related work to the problem addressed in this paper which is described in Section 4. Section 5 presents the conceptual architecture proposed for the solutions and, in the sequel, Section 6 introduces the main results achieved and its discussion. Finally, in Section 7 the main conclusions of this work are referred, as well as some ideas for future developments.

2 Proposed Solution Concept

The development of flexible and configurable products that can evolve in the future is a complex task. The limitations of conventional design and development approaches are recognized in this context.

Unlike conventional product development, where architectures/platforms with well-structured and detailed interfaces are not used, the concept underlying the proposed solution is based on the principles of integrating advanced modularity techniques into the design and development of products. In this way, it will be possible to make the integrated and modular development of new products

by drastically reducing the time of design, development and installation of new equipments.

The "modular product" project assumes to guarantee two main characteristics. First, each functional element of the product consists of "parts" which have well defined interactions with other parts of the same product. Second, modular product design allows to perform changes to be made in one part, or even replace it (ex: conveyor, robot, etc) without this change affecting other product functions.

As in the automotive industry, modularity should consider the entire design process. While standardization involves design-to-production optimization, the product architecture influences the use and the sale of the final product, as it contributes for the enhancement of its flexibility and adaptability.

The concepts of modularity considered here aim at the decomposition of what would be a robotized palletizing complex monolithic solution, in subsystems that constitute complete functional units, that can be designed and produced independently (allowing the construction of different products by means of a combination of subsystems) and with a reduced complexity, but keeping its integrated operation. This innovative concept will be materialized in a technological kit that will validate the application of the developed research in the design and development of robotic adaptive palletizers, and will be supported by a set of software and hardware prototypes (see Figure 1).

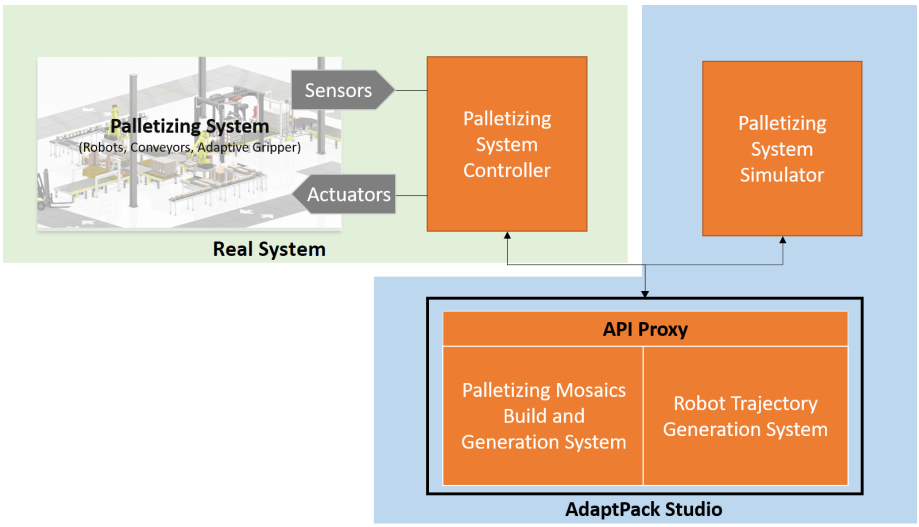


Fig. 1: Diagram of the proposed project solution modules

Considering the proposed project solution architecture depicted in Figure 1, this paper specifically focus on the "Palletizing System Simulator" and "Robot Trajectory Generation System" Modules. More in detail, is presented the simu-

lation environment, and how this simulator can be used for quickly designing a palletizing cell and programming the respective robots.

3 Related Work

Robot palletizing cells have captured a lot of attention in the last decade in the scientific sphere. Two essential issues affecting the performance of palletizing robots, that should be considered when designing a flexible automatic robot palletizing production line, are: the proper robotic cell layout arrangement and the path planning of the industrial robot.

In [2, 3] is presented an offline programming robot palletizing simulator that consist of: (*i*) a fast algorithm to generate the palletizing product mosaic, (*ii*) a 3D robot simulator, and (*iii*) an optimized trajectory generation algorithm. According to its authors, with this system it becomes possible to program the industrial robot without the need of using the teach pendant or writing a single line of code. The authors focus their work on the optimization of the computational time of their path planner, based on A*.

In [4] the authors deal with facility layout analysis and path planning of a robotic palletizing production line, using a 4-DOF partial closed-loop palletizing robot and two carton conveyors. The layout of the robotic cell is arranged considering the rechability of the same, but at the same time minimizing the robot footprint. In terms of path planning, the authors propose a straight-line and arc based motion planning approach under the constraints of joint velocity and acceleration to achieve a higher productivity and an easier implementation.

Ryosuke *et al.* [6] also deal with the design of a palletizing cell and with the robot path planning problem. To solve both, the authors propose a design method to improve the performance of a palletizing manipulator. The working environment is optimized regarding the base position of the manipulator and the shape and position of the pallet. To reduce the computational time, the parameters of an environment are quickly evaluated with the proposed method, in which are set passing points to reduce the computation time of path planning.

Focusing on the path planning problem, the authors in [5] present an algorithm that computes a collision free trajectory considering a 6-axis industrial robot. The target applications for the developed approach are palletizing and handling robotic cells. Given a 3D environment, the proposed algorithm decomposes the Cartesian workspace of the robot into 3 dimensional cylinder slices centered around the robot base.

For its turn, Nan Luan *et al.* [7] present a maximum speed algorithm for serial palletizing robots. As the authors state, operation speed is one of the most important performance index of a robotic palletizing cell. To meet the required performance, the authors propose an iterative control algorithm that optimizes the maximum robot speed given a certain positional tolerance. The authors refer that, with this algorithm it is possible to obtain maximum speed and acceleration for the robot manipulator regardless of manipulator position,

arm posture or joint couplings, thus improving palletizing efficiency and bringing more economic benefits into practical application.

4 Problem Description

The problem addressed in this paper is the identification, definition and development of the "Palletizing System Simulator" and "Robot Trajectory Generation System" modules. In Figure 1 is presented the project solution architecture.

As mentioned, this project foresees the inclusion of a 3D robotic cell simulator as an integral part of the proposed modular architecture. As its main requirements, this simulator should allow the design and simulation of the all palletizing cell, and also the offline programming and the download of the generated program code for robots from distinct brands.

The inclusion of this software in the project aims not only for the reduction in the design time of the palletizing cells, but also in the development and setup times of the robots programs. Firstly, this software allows the offline programming of the mentioned robots, as well as the simulation of its operation. Secondly, it makes possible to expand the software components library by designing new models of customized equipment or even whole palletizing cells. This way, their inclusion in each project can be performed in a simple way. Indeed, by simulating the robots virtual models and the palletizing systems in which they operate, the efficiency of design of a robotic cell can be significantly improved. Also, considering the accuracy and level of detail offered by the simulation software, it is possible to achieve models having a behaviour very similar to the actual equipment, enabling the study and test of a solution before its implementation, with very high reliability.

Moreover, using the simulator is also ensured the concept of adaptability proposed in the project. After the offline programming stage, through the software's own language, the goal is to translate this language into the ones used by the robot's controllers of different vendors (namely ABB, KUKA, FANUC and Yaskawa Motoman). For this purpose, the idea is to define a so-called neutral language, to which is translated the software's language. Then, the next step is to perform the translation of this neutral language to each considered robot's manufacturer language. The neutral language adopted in this Project is the Industrial Robot Language (IRL - DIN Standard 66312), since the software already allows the translation of its instructions set to this standard.

This simulator will be part of the "Palletizing System Simulator" module (see Figure 1), whose main goal is to make the programming stage of each palletizing cell's robot. Therefore, using the Python's API provided by the simulation software, it was developed a plug-in that generates the robot's program only by selecting the pick and place points of the products to palletize.

This module, on its turn, will exchange information with the "Robot Trajectory Generation System" module. It aims to program the movement of the robot between the points of pick and place, considering its operation in the projected

cell, in a way that results in collision free trajectories. Each of these two modules are detailed in the next section.

5 Modules Conceptual Architecture

The conceptual architecture of the two developed modules is presented in this section. Firstly, the "Palletizing System Simulator" module will be presented, followed by the "Robot Trajectory Generation System" module.

5.1 Palletizing System Simulator

The conceptual architecture of the simplified robot's programming module for a palletization task follows the scheme of Figure 2.

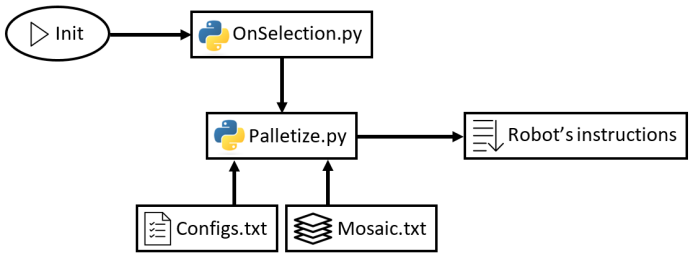


Fig. 2: Palletizing System Simulator architecture scheme

As the figure shows, an *Init* button is used to start the process. After that, the *OnSelection.py* function was developed to wait for the selection of the picking point, by clicking on the corresponding components (which must be a simulator predefined type of object - a product), and placing points (which must also be a simulator predefined type of object - a pallet). If the selected components are different than expected, the function rejects the selections.

After the selections are made successfully, the function stores the coordinates of the selected objects which are passed to the *Palletize.py* function. Besides the mentioned coordinates, this function has two input files: *Configs.txt*, which has information about the dimensions of the products to be transported and the dimension of the pallet on which the palletization will occur; and *Mosaic.txt*, which contains the points that define the products mosaic to be palletized. As output the function sets the program instructions for the robot's movements and gripper actuation, and starts the simulation.

5.2 Robot Trajectory Generation System

For the generation of collision free trajectories the work presented by [8] was considered. This planner is similar to the ones from the A* family and has associated the need of space discretization and the definition of the kinematics

equations that rule the robot's movement. Based on this algorithm, the implemented functionality follows the architecture scheme of Figure 3.

For the generation of the robot's collision free trajectory, the *A* Path* button is pressed to start planning. Thus, the *PathPlanning.py* function firstly receives the start and end point coordinates of the trajectory from the *OnSelection.py* function, resulting from the selection process referred in 5.1. These are defined as points above the product to be palletized and above the chosen pallet, respectively, meaning that the path planning is made between these two points of the palletizing cell. Then, the algorithm process starts. For its operation, the *RobotKinematics.py* function is first identified, which is responsible for handling the kinematics of the robot. For its turn, the *CollisionDetector.py* function finds information of collision occurrences in the 3D robot workspace using the simulator own collision detector. This way, it is possible to identify collisions between the set composed of the robot, gripper and transported product, and any other component part of the palletizing cell in each discretized space. After that, all the discretized joint space information required for A * algorithm's logic is stored in a data structure whose query functions are defined in *DataHandler.py*.

Finally, as Figure 3 shows, the output path points are handled by *Palletize.py*. This function, described in subsection 5.1, is responsible for its inclusion in the robot's movement instructions for the palletizing process.

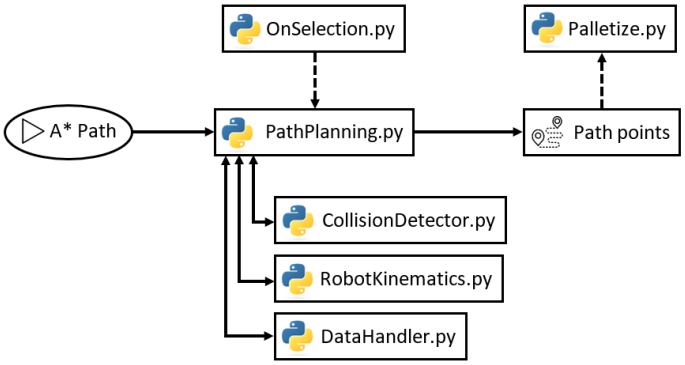
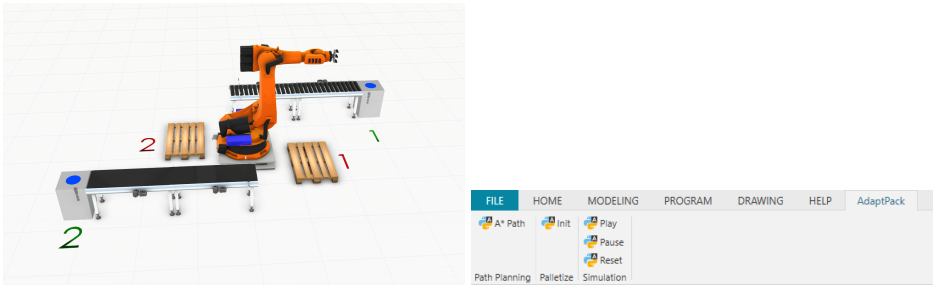


Fig. 3: Robot Trajectory Generation System architecture scheme

6 Tests and Results

This section presents the obtained results with the two developed modules. A simple palletizing cell, depicted in Figure 4(a), is used as an example. This cell has a 100 kg payload KUKA palletizing robot, two separate picking point feeders (identified by the green numbers) and two pallets representing two different placing points (identified by the red numbers). Also, a customized menu for

this work was defined and added to the simulator top tab menu to access the developed functions of the plugin (see Figure 4(b)).



(a) Palletizing cell (b) Customized project simulator menu

Fig. 4: Palletizing cell example for testing purposes

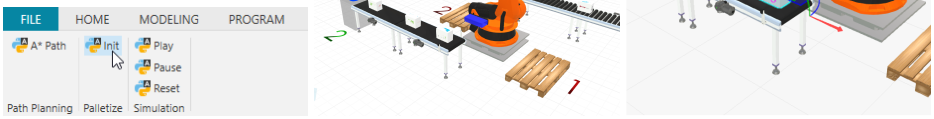
As in the previous section, this one is divided in order to present the results of the "Palletizing System Simulator" module in the first subsection, and the "Robot Trajectory Generation System" module in the second one.

6.1 Palletizing System Simulator

With the developed functions mentioned in subsection 5.1, it actually becomes possible to program the palletizing robot in a very simple way, just by selecting the pick and place points of the product to palletize. The process starts with the press of the *Init* button, located in the *Palletize* tab of the project simulator menu (Figure 5(a)). This action starts the simulation automatically until the products of both feeders reach the two picking points (Figure 5(b)). Selecting the product at the desired picking location (Figure 5(c)), sets the coordinates of the picking point for the robot's instruction generation (Figure 5(e)). Then, by clicking on the desired pallet, the coordinates of the placing point are also set (Figure 5(d)). As stated in 5.1, the exact coordinates of the placing points (Figure 5(f)) are defined in the *Mosaic.txt* file. After this last selection, the simulation resets and only the feeder corresponding to the defined picking point works and the robot starts the palletization process. In figures 5(g), 5(h) and 5(i) three moments of a three layer palletization process can be seen.

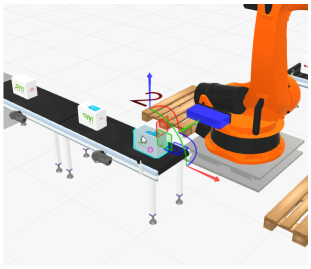
6.2 Robot Trajectory Generation System

Path planning comes as a result of the developed application described in subsection 5.2. The first test performed was the trajectory planning for the palletizing task shown in figure 5. After selecting the pick and place points and clicking the *A* Path* button located in the *Path Planning* tab of the project simulator

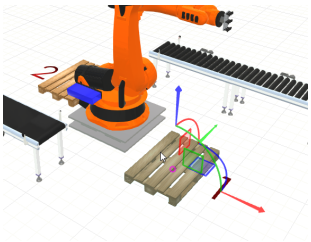


(a) *Init* button to start the process

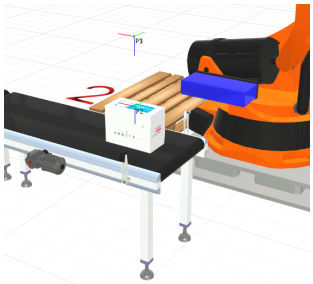
(b) Products stop at picking points



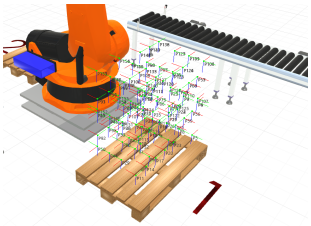
(c) Picking point selection by clicking in the product



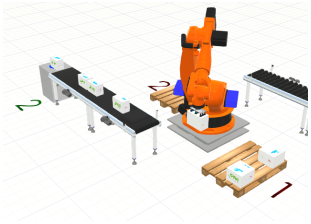
(d) Placing point selection by clicking in the pallet



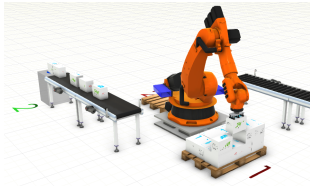
(e) Robot's generated picking points



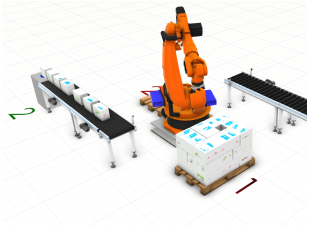
(f) Robot's generated placing points



(g) Palletization: first layer



(h) Palletization: second layer

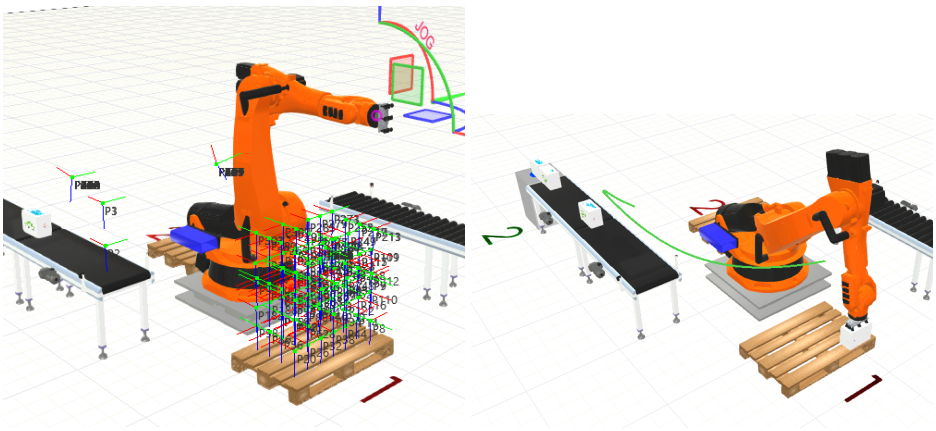


(i) Palletization: complete pallet

Fig. 5: Palletizing System Simulator steps

menu, the algorithm process starts. When it finishes, it returns the points of the discretized space that must be included in the trajectory. To illustrate the result of the planned trajectory for the given example, Figure 6(a) shows the points of the discretized space included in the resulting trajectory shown in 6(b).

The second test performed to the trajectory planning plugin was its behaviour in collision situations. Thus, metallic fences were located in the robot's workspace, resulting the layout depicted in Figure 7. After the algorithm application under these conditions, the obtained result is shown in Figure 8. Again, in



(a) Trajectory added points (b) Resulting trajectory

Fig. 6: Planned trajectory for the palletizing example of Figure 5.

8(a) are illustrated the points added to the trajectory and in 8(b) the resulting trajectory.

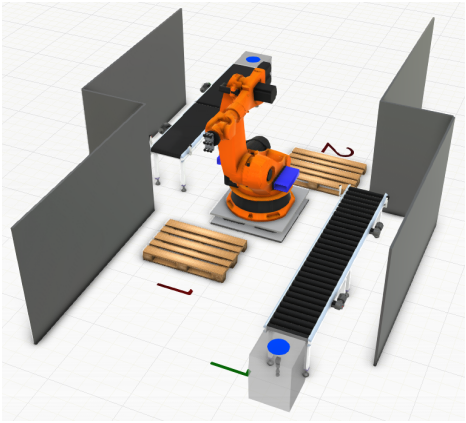
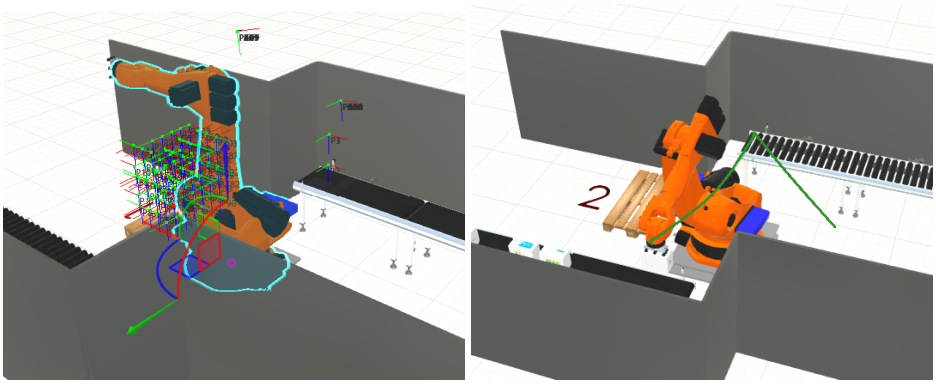


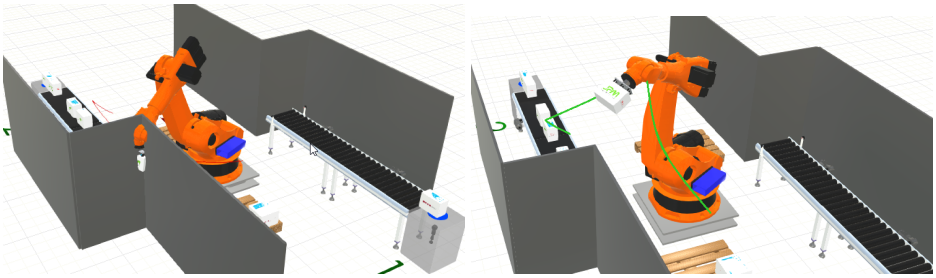
Fig. 7: Palletizing cell with fences as obstacles

Finally, in order to compare the two illustrated cases, Figure 9 shows the result of the collision's detector inclusion in the application architecture. As can be seen, the robot's movement is adjusted having a collision-free trajectory.



(a) Trajectory added points (b) Resulting trajectory

Fig. 8: Planned trajectory in a palletizing cell with obstacles



(a) Without collision detector (b) With collision detector

Fig. 9: Trajectory planning comparison

7 Conclusions

The use of robotic palletizing systems has been increasing in the FMCG industry. Due to marked demands, most of these companies are changing their production processes from low variety/high volume to high variety/low volume, which demands frequent reprogrammings of their equipments with the inherent production breakdowns and also the need to frequently customize and/or change the palletizing solutions. To overcome these problems, this work proposes a solution based on the principles of integrating advanced modularity techniques into the design and development of automated palletizing solutions.

This paper addressed the development of a system for the automated offline generation of collision free robot programs for palletizing applications in the scope of this project. It has been detailed the automatic generation of robot programs, based on the specification by the user of the picking and placing position coordinates and on the pallet mosaic configuration, and the robot collision

free trajectory generation system for the implementation of this program. Based on preliminary tests, the system works as predicted and has been considered user friendly by the people who have tested it. However, future improvements have to be performed in the collision free trajectory generation system in order to lower its computational burden and allow to decrease the computational time required to compute collision free trajectories.

Acknowledgements

This work is financed by the ERDF – European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation – COMPETE 2020 under the PORTUGAL 2020 Partnership Agreement, and through the Portuguese National Innovation Agency (ANI) as a part of project AdaptPack — POCI-01-0247-FEDER-017887.

References

1. Hande Yaman, Alper Sen: Manufacturer's Mixed Pallet Design Problem. European Journal of Operational Research, 2008, v. 186, pp. 826-840
2. SeungNam Yu, SungJin Lim, MaingKyu Kang, ChangSoo Han and SungRak Kim, "Off-line robot palletizing simulator using optimized pattern and trajectory generation algorithm," 2007 IEEE/ASME international conference on advanced intelligent mechatronics, Zurich, 2007, pp. 1-6.
3. SungJin Lim, SeungNam Yu, ChangSoo Han and MaingKyu Kang (2010). Palletizing Simulator Using Optimized Pattern and Trajectory Generation Algorithm, Mechatronic Systems Applications, Annalisa Milella Donato Di Paola and Grazia Cicirelli (Ed.), InTech.
4. Liangan Zhang et al., "Layout analysis and path planning of a robot palletizing production line," 2008 IEEE International Conference on Automation and Logistics, Qingdao, 2008, pp. 2420-2425.
5. C. Scheurer and U. E. Zimmermann, "Path planning method for palletizing tasks using workspace cell decomposition," 2011 IEEE International Conference on Robotics and Automation, Shanghai, 2011, pp. 1-4.
6. Ryosuke Chiba, Tamio Arai, Tsuyoshi Ueyama, Taiki Ogata, Jun Ota, Working Environment Design for Effective Palletizing with a 6-DOF Manipulator, International Journal of Advanced Robotic Systems, 2016, vol.13 , pp. 1-8
7. Nan Luan, Haiqing Zhang, Shanggao Tong, (2012) "Optimum motion control of palletizing robots based on iterative learning", Industrial Robot: An International Journal, Vol. 39 Issue: 2, pp.162-168, https://doi.org/10.1108/01439911211201627
8. Pedro Tavares, José Lima, Pedro Costa,"Double A* Path Planning for Industrial Manipulators",Robot 2015: Second Iberian Robotics Conference: Advances in Robotics, Lisbon, 2016, v. 2, pp. 119-130.