Industrial Robotics Platform for Simulation Design, Planning and Optimization based on Off-line CAD Programming

Khelifa Baizid[1, 2], Amal Meddahi[3], Ali Yousnadj[4], Saša Ćuković[5] and Ryad Chellali[6]

[1]*Mines Douai, IA, F-59508, Douai, France*
[2]*Univ. Lille, F-59000 Lille, France*
[3]*DIEI, University of Cassino and Southern Lazio, Cassino, Italy*
[4]*DGRSDT, Advanced Technology Development Center, Algeria*
[5]*FE, University of Kragujevac, Kragujevac, Serbia*
[6]*CEECS, Nanjing Robotics Institute, China*

Abstract This paper presents IRoSim: Industrial Robotics Simulation Design Planning and Optimization platform which we developed based on SolidWorks API. The main objective is to integrate features from mechanical and robotics CAD software into the same platform in order to facilitate the development process through a friendly interaction interface. The platform provides important steps to develop a given robotized task such as: defining a given task, CAD learning of the end-effectors' trajectory, checking the manipulator's reach-ability to perform a task, simulating the motion and preventing the trajectory from possible collisions. To assess the usability of the proposed platform, a car's doors painting task using a 6 Degree Of Freedom industrial manipulator has been developed.

1 Introduction

Nowadays, programming of industrial robots is of importance to many manufacturing industries, because changing and reinventing their production systems continuously appear within the manufacturing environment [1, 2]. However, industrial robots programming generally is a tedious and time-consuming task that demands significant technical expertise, and require a tremendous amount of programming to make them useful. Also, in order to accomplish such task skilled and experienced programmers are often a scarce resource. Nevertheless, new and effective programming approaches that are easier, faster and advanced with low cost are constantly sought such as conventional teaching pendant [3] and Off-Line Programming (OLP) environments. Usually, such environments are based on graphical simulation platforms, where programmers need only to learn the simulation language and not any of the robot programming languages. Moreover, OLP environments include various libraries such as tools for the simulation scenarios and some predefined applications of robotics tasks like spot-welding and painting. And the most important is that it can allow the kinematics control to enable users to plan collision-free trajectories. The simulation may also be used to determine the cycle time of execution of the robotics task.

In the last decades, OLP coupled with Computer Aided Design (CAD) leaded to learning trajectory through teach pendant [3, 4]. As a result of combining these technologies make it possible to reduce both the time and the cost of development process, and improve the quality of the product. Therefore, CAD systems have been in use for several industry process including, automobile, aircraft manufacture and shipbuilding. These fields, employ in their systems CAD and embedded knowledge to design the robot mechanism and to simulate its motion [5]. However, many of these systems do not rely on standard CAD packages such as SolidWorks and CATIA, which presents supplementary difficulties for novice users to model and to export their 3D models to simulate a given task. Whereas, standard CAD systems that are used for design do not provide non-free packages that can be used for robotics task simulation. Thus, our objective in this approach is to provide an easier process, to define a robotized task, integrated into a 3D CAD system.

In this paper, we present a novel Industrial Robotics Simulation (IRoSim) CAD robotics system that has been realized to assist users in designing and developing industrial robotized tasks. IRoSim is accessible to anyone with basic knowledge of CAD and robotics. As shown in Figure 1, the system offers users to simulate several steps to fulfill the desired objectives.

IRoSim is integrated into SolidWorks and can easily be upgraded to future versions. Unlike programming packages [3, 6], IRoSim provides several features such as 3D design of the robotic manipulator, definition of the task to be accomplished, accessibility verification, optimization of the task time execution [7], graphical simulation, detection of the eventual collisions and generation of a collision-free trajectory to be directly

© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (http://creativecommons.org/licenses/by/4.0/).

mapped to the real cell. In order to show the effectiveness of the proposed software we implemented a robotized task which is the painting (by dipping) using a 6-DOF robot manipulator.

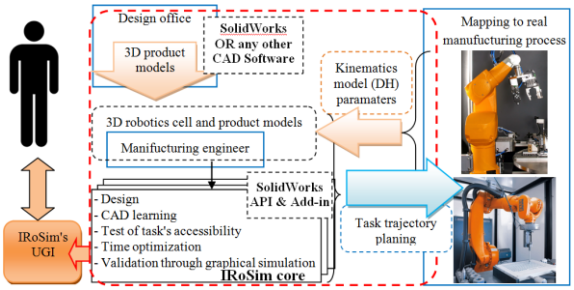

Figure 1. Offline-CAD programming schema of IRoSim.

2 CAD based approach

Nowadays, the CAD model databases are widely used in the industry to design, model and programming robotics products [3, 6] such us AutoCAD, Ideas, Cimatron, CATIA and SolidWorks API (Application Programming Interface). We used this last to offer the feature to calculate the robotics task parameters based on a virtual 3D model of the real robotics cell. Moreover, the modification and the redefinition of these parameters are easily possible with the developed user-friendly integration.

2.1 Data Acquisition

In order to evaluate the robot control through an OLP, data acquisition systems must be built into the control program. In our approach, it consists of the 3D virtual representation of the physical model of the robotized cell, followed by the application of several algorithms based on CAD learning to extract useful information from the virtual environments. This gives definition of the task and delimits some kinematics constraints between the robotic manipulator and the task. Figure 2 summarizes the process of data acquisition.

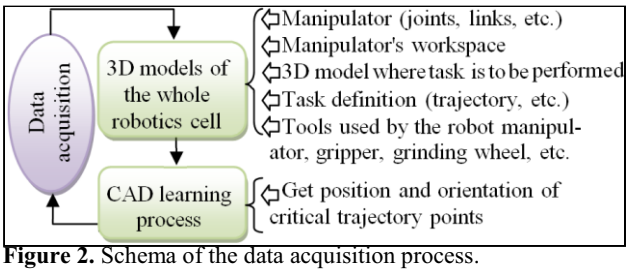

Figure 2. Schema of the data acquisition process.

2.2 3D Modelling of Robot and Environment

2.2.1 3D modeling

3D modeling of the robot cell [8] can be contacted based on a volume and/or surface. This can include the robots' libraries, End-EFfectors (EEF), the 3D model where the task is to be performed and the remaining additional parts. For instance, a robot's WorkSpace (WS) can be modeled only by volume. This allows having more geometrical information that can be used during the simulation process. However, other 3D objects can be modeled by surface if no need to a volume representation. Figure 3 shows some parts of a robotics cell which contains a robot manipulator, gripper, two conveyors and drying machine. The manipulator was designed in a way that allows possible kinematic joints representation of the corresponding links subject to certain numerical constraints (e.g., DH: Denavit–Hartenberg parameters).

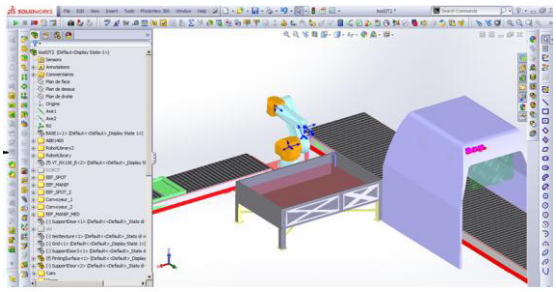

Figure 3. IRoSim development environment with 3D models of robot manipulator, conveyors, etc.

2.2.2 Task definition

Robotic task can be represented as a 2D/3D set of points to be visited by the manipulator [9]. We distinguish two kinds of trajectories: Point-to-Points (P2P) and Continuous. P2P trajectories may include spot welding and drilling tasks while arc-welding, laser cutting, painting, polishing [10] etc. are modeled using continuous trajectories where we used a 3D spline. Figure 4 presents critical points of grasping, painting and releasing car's doors.

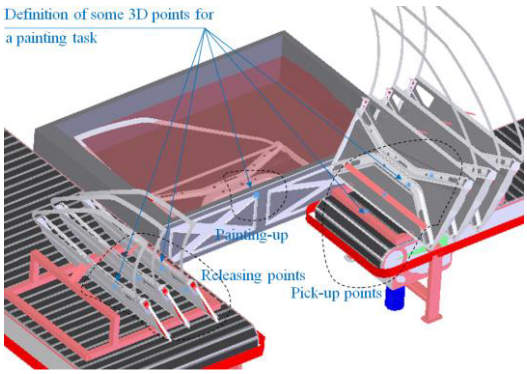

Figure 4. An example of asks defined in SolidWorks, 3D points are the picking-up, painting and releasing locations.

2.2.3 Robotics library

IRoSim platform includes a set of robot manipulators such as Staubli and KUKA KR6-2. 3D models of these manipulators can be integrated and assembled easily into SolidWorks; and the platform is open to integrate new manipulator models. Moreover, the library contains several other 3D tools that are needed for the simulation.

3 CAD programming stages

IRoSim platform is integrated as a plug-in with SolidWorks software based on Microsoft Visual Basic

Application (VBA). As shown in Figure 5, we present it as Product Lifecycle Management (PLM) software that develop a set of stages (Figure 5) in order to allow the user to define, simulate and validate a given industrial robotics task, without exporting all models from the design environment to the simulation software.

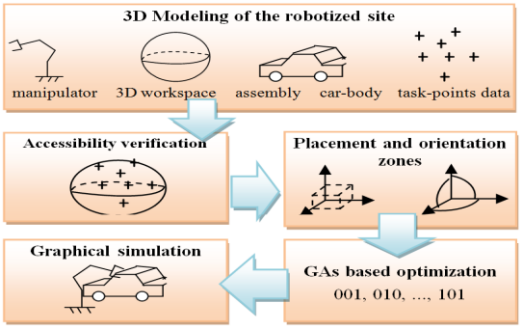

Figure 5. IRoSim offline-CAD programming steps.

3.1 CAD learning

The proposed learning process is provided by GUI of IRoSim platform. We select first: a 3D CAD model of the task (e.g., Figure 3 and 4) and a realistic 3D model of the EEF whose dimensions are specified in the main GUI of the IRoSim. User can move the 3D model of the EEF in the graphical area by translations and rotations around X, Y and Z axes using IRoSim's GUI. Then, we need to get the critical information that is helpful for learning more complex information from the graphical area. Figure 6 presents steps and pseudo-code to involve this process. Considering a P2P trajectory, we first get coordinates of the task points. From these point's coordinates we can extract the orientation at each point as the normal of the surface holding the point (i.e., a coincidence constraint between the surface and the point). So, we have 4-DOF (1 orientation and 3 positions) which will help in the next stage of learning.

The second step (Figure 7) involves tuning of the EEF 3D model in the virtual environment. The designed GUI of IRoSim platform offers features to displace and rotate the EEF around all possible axes. In case of a task involving a P2P trajectory e.g. grasping, this process is executed for two locations; the pre-grasping and the grasping itself.

3.2 Task accessibility checking

In order to accomplish the robotics task accurately and precisely, the robot should be able to reach all task points defined by the user. The task's accessibility check, in our platform, is based on: generating the 3D model of WS, representation of the task points and finally checking the interference between them.

The shape of 3D WS of a robotic manipulator is a function of its mechanism that includes DH parameters, link lengths, joints' range of motion etc. Since several industrial manipulators have the first three joints used for positioning, while the last three are mainly for orientation [11], we considered, in our approach, only the first three joints to generate the 3D model of the WS from DH parameters. Generally, we used the second and the third joint to generate a sketch in a vertical plane (parallel to the z axis of the first joint). So this joint is used for creating the volume function (e.g., revolution or extrusion according to the joint type revolute or prismatic, respectively). The algorithm that depicts the main sequence involved in this step is given in [5]. A set of different mechanisms for a manipulator and their corresponding 2D sketches and 3D WS are also given in [5] based on this algorithm.

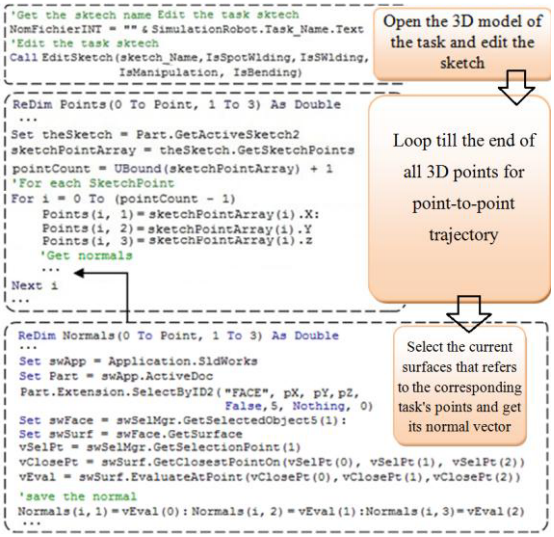

Figure 6. CAD learning of the critical task points with some pseudo-codes.

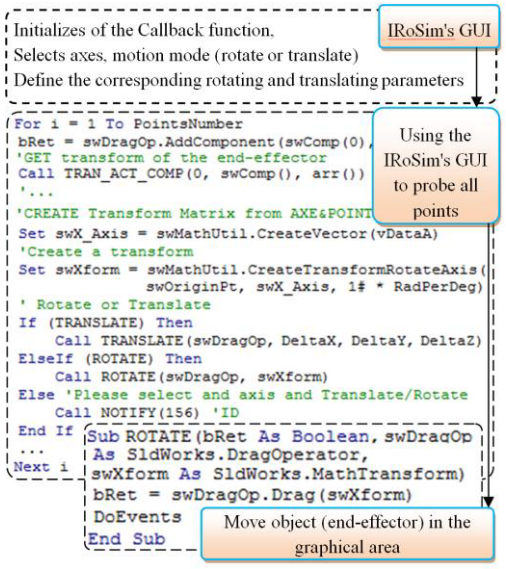

Figure 7. CAD learning of the critical transformation at each task point with some pseudo-codes, the objective here is to define the homogenous transformation matrix at each point.

In the proposed platform, tasks accessibility verification process is based on fully graphical. Using Oriented Programming Objects (OPO) of SolidWorks's API, we use geometrical features of the task points and the 3D WS model of the manipulator to check the interference between them. The interference is checked between parallelogram-shaped temporary 3D bodies that we create around each task points and the WS body of the

manipulator. In case the task points are not reachable, the IRoSim GUI offers possibility to redefine the task or to load another manipulator with different DH parameters if necessary to accomplish the task.

3.3 Placement and orientation zones

We developed platform that offers the possibility to optimize the best time of the task execution. In order to achieve this we need to find the shortest distance travelled by the manipulator in the coordinate space. It is reported previously in [12] that the placement and the orientation of the robotic manipulator are of the most importance in time optimization. In other words, the manipulator can be displaced on three axis coordinate system and can also rotate around these axes to obtain the suitable placement and orientation relative to the task by ensuring the task reachability.

3.4 Time optimization

The main motivation to have an industrial framework providing all necessary stages needed by a robotics task lead us to include time optimization algorithm in our IRoSim platform. This optimization depends, in the first case, on the distance traversed by the manipulator's joints, which is related to the sequence of the task-points visited by the EEF. Also, the manipulator executes a specific task in the operational space and performs the motion in the coordinate space, which makes the travelled distance a function of manipulator's IKM. In that sense, the optimization problem is concerned with finding the minimum distance between each two consecutive trajectory-points. Moreover, any solution of a IKM joint is strongly affected by the placement and the orientation of manipulator. Thus, in order to have a complete optimization method which performs the strategies presented by authors in [12, 7] we proposed an optimization method based on Genetic Algorithms (GAs) that incorporates the best cycle time of task execution as objective function, while the chromosome is composed of: order of achievement of the task-points, IKM at each task-point, relative placement and orientation between the manipulator and the task-points. The cycle time has been computed based on joints' displacement between each successive pair of points and the average velocity of the corresponding joint.

4 Graphical simulation

All graphical objects, modeled as 3D model can be displaced or moved by acting on its transform. During the motion of the 3D objects we solicit API futures to detect the collision with the surrounding. Figure 8, bellow, shows a simplified algorithm of motion of the robot manipulator's joint from one configuration to another. It start by adding the corresponding joints that are considered to be moved, gets the current joints axes (Z), creates an axis object, transforms and finally moves the joints by updating the 3D graphical area, in the main time obstacle avoidance algorithm is running on the fly.

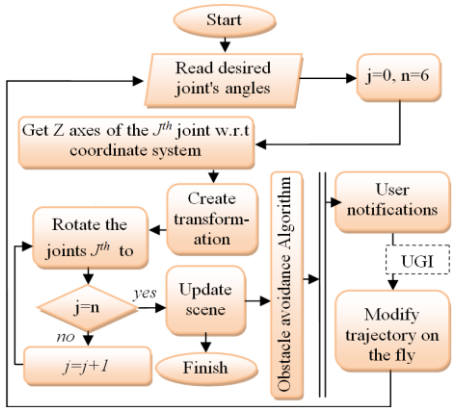

Figure 8. Algorithm of 6-DOF robot manipulator motion.

5 Case of study

Based on IRoSim, we present a simulation case study that involves an industrial task which is painting of a car's doors (by dipping).

5.1 Task definition, accessibility and CAD trajectory learning

As shown in Figure 4 each point of the task is constrained on the given surface of the corresponding car's door. Figure 9 shows snapshot of CAD trajectory learning GUI.

In details Figure 9a shows the IRoSim main GUI and the corresponding CAD learning trajectory. It is worth to note that Euler angles between the initial EEF configuration *Conf_0* and the first configuration *Conf_1* are given in [11]. As shown in Figure 9c user must rotate the EEF using the GUI to reach the grasping configuration *Conf_g*,. When this is done, user translates backward following Z axis to reach the pre-grasping configuration *Conf_p*. At the mean time he/she mast save the two matrices of *Conf_g* and *Conf_p* (Figure 9d) and finally he/she checks the task accessibility (Figure 9e) by moving the workspace manually around the task and checking if all the points are reachable, or automatically by running the corresponding algorithm.

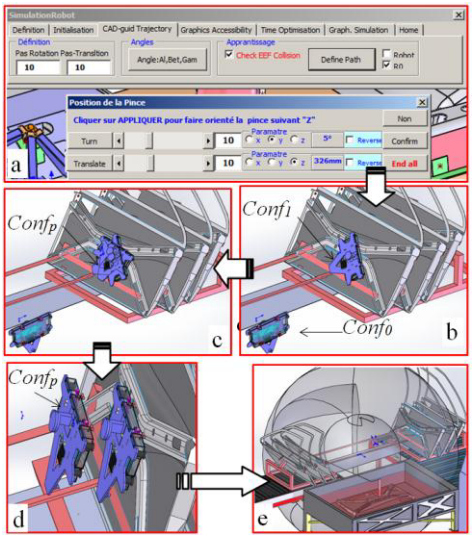

Figure 9. Snapshots of CAD learning trajectory and accessibility verification process.

5.2 Simulation

After optimizing the task's time, the last step is the graphical simulation and validation of the task trajectory.

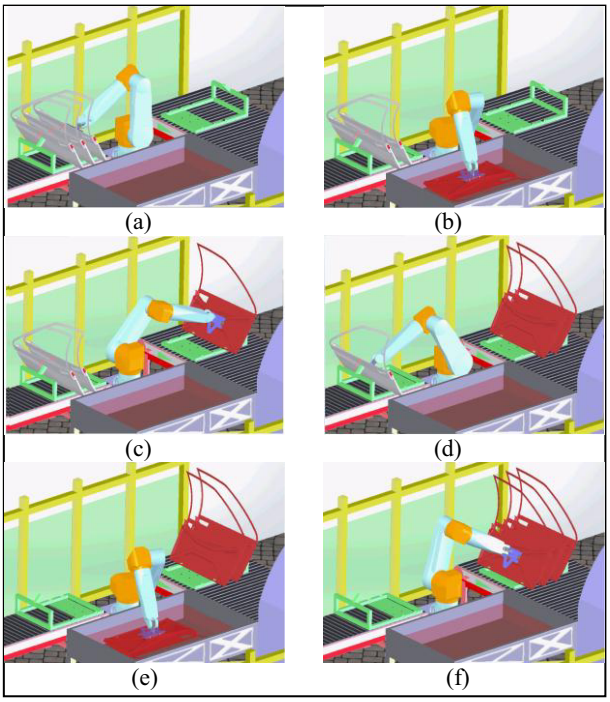

Figure 10. Snapshots of the painting simulation process.

It is worth to mention that the order of visiting points is fixed because of the highly constrained task, where we have only one sequence. For instance, the manipulator must grasps the support from the first conveyor and puts it on the second one, then starts transporting the first door (Figure 10a) and paints it in the basin (Figure 10b) and then dropping it on the second support posed, earlier, on the second (big) conveyor (green color model). Such operation is repeated till the last door (Figure 10c to Figure 10f). Finally, and after doors arrive to the drying machine the manipulator transports the last support to prepare for the next a round of the task. It is worth to note that collision avoidance is resolved by adding intermediate points in the trajectory of the robotic manipulator. A video can be found in this link: https://www.youtube.com/watch?v=im5qGO7Ky5Y

6 Conclusion

A novel CAD-based OLP platform IRoSim for industrial robots has been presented. A platform that is able to define, optimize, simulate and validate an industrial task and then map it into a real site. The proposed GUI of this platform is fully integrated with SolidWorks API and it provides many utilities that can cope with several industrial tasks including but not limited to; drilling, painting, spot welding, pick and place and other object manipulation tasks. Since the creation of the CAD models and robot programming task are performed in the same software the entire robot programming process becomes faster, easier and cheaper. In addition, the case of study of the painting task showed that the proposed platform is intuitive to novice use in industrial environments. In near future, we are looking forward to integrate some feedback control via cameras, sensors and other embedded systems.

References

1. P. Zengxi, Recent Progress on Programming Methods for Industrial Robots, *Robotics and Computer Integrated Manufacturing*, **28**(2), 87-94 (2012)
2. B. Hein, M. Hensel, H. Wrn, Intuitive and model-based on-line programming of industrial robots: a modular on-line programming environment, *IEEE International Conference on Robotics and Automation*, pp. 3952-3957 (2008)
3. P. Neto, N. Mendes, Direct off-line robot programming via a common CAD package, *Robotics and Autonomous Systems*, **61**, 896-910 (2013)
4. S. Mitsi, K. D. Bouzakis, G. Mansour, D. Sagris, G. Maliaris, Off-line programming of an industrial robot for manufacturing, *The International Journal of Advanced Manufacturing Technology*, **26**(3), 262-267 (2004)
5. A. Meddahi, K. Baizid, A. Yousnadj, J. Iqbal, API based graphical simulation of robotized sites, *14th IASTED Robotics and Applications Conference*, 458-492 (2009
6. H. Chen, W. Sheng, N. Xi, M. Song, Y. Chen, CAD-based automated robot trajectory planning for spray painting of free-form surfaces, *Industrial Robot: An International Journal*, **29**(5), 426-433 (2002)
7. K. Baizid, R. Chellali, A. Yousnadj, A. Meddahi, B. Toufik, Genetic Algorithms Based Method for Time Optimization in Robotized Site, *IEEE/RSJ International Conference on Intelligent Robots and System (IROS)*, 1359-1364 (2010)
8. Zeghloul, B. Blanchard, M. Ayrault, SMAR: A Robot Modeling and Simulation System, *Cambridge journal*, **15**(1), 63-73 (1997)
9. P. Th. Zacharia, N. A. Aspragathos, Optimal Robot Task Scheduling based on Genetic Algorithms, *Robotics and Computer-Integrated Manufacturing journal*, **21**(1), 67-79 (2005)
10. P. Neto, N. Mendes, R. Arajo, J.N. Pires, A.P. Moreira, High-level robot programming based on CAD: Dealing with unpredictable environments, *Industrial Robot*, **39**(3), 294-303(2012)
11. W. Khalil, E. Dombre, Modlisation identification et commande des robots, *HERMES Science Publications 75004*, (1999)
12. K. Baizid, A. Yousnadj, A. Meddahi, R. Chellali, J. Iqbal, Time scheduling and optimization of industrial robotized tasks based on genetic algorithms, *Robotics and Computer-Integrated Manufacturing Journal*, **34**, 140-150 (2015)