

An Integrated Software Package for Advanced Industrial Robot Applications

C. Liang, H. Yan, R. Li, I.-M. Chen, M.H. Ang Jr. and Z. Huang

Abstract This paper introduces an integrated robot software package for advanced industrial robot applications with the aims to make industrial robots easy-to-operation, easy-to-programming, much more flexible with advanced capabilities. This software package is developed based on ROS (Robot Operation System) middleware and ROS-Industrial open source packages by integrating different function modules. The module design implementation of a graphic user interface, visual programming, intermediate robot language and simulation environment are presented. Utility and flexibility of the software package are evaluated through two experiment demos: automated robot bin-picking and automated robot 3d object taping, which has a potential to be used for a wide of robot applications.

Keywords Industrial robot · Robot software development · Robot bin-picking · Robot taping

1 Introduction

For more than 30 years, industrial robots have been widely used for material handling, welding and dispensing in automobile industry, which are cost-effective for a high-volume and repetitive tasks [1]. However, low-volume, high-mixed parts

C. Liang (✉) · I.-M. Chen
School of Mechanical and Aerospace Engineering, Nanyang Technological University,
Singapore, Singapore
e-mail: chliang@ntu.edu.sg

H. Yan · M.H. Ang Jr.
Department of Mechanical Engineering, National University of Singapore, Singapore,
Singapore
e-mail: mpeyanh@nus.edu.sg

R. Li · Z. Huang
Institute for Infocomm Research, Agency for Science Technology and Research, Singapore,
Singapore
e-mail: li-r@i2r.a-star.edu.sg

production are the characteristics for next-generation manufacturing. Thus, smarter, faster and more flexible industrial robots are required. Advanced capabilities like real-time collision avoidance, mobile manipulation, 3d perception enabled path planning and grasp planning are preferred for accomplishing particular tasks that have never done before. These requirements greatly challenge capabilities of traditional industrial robots in both hardware and software aspects [2].

To meet these challenges for future markets, robot manufactures are developing next-generation robots with more advanced capabilities for different application areas. There are several commercial products available in the market. The KUKA LBR lightweight robot is a torque controlled 7 d.o.f.s (degrees of freedom) robot manipulator with intrinsic safety, which has been widely used in academic laboratories for research purposes [3]. ABB Frida® is a dual-arm industry robot for small parts assembly tasks in 3C industries (Computers, Communications and Consumer electronics). This robot is capable of working side by side with human workers [4]. A Danish robot company Universal Robots provides UR5 and UR10 light weight robots, which have a wide application in both research fields and practical industry applications [5]. Additionally, Nextage-Open is dual-arm industrial humanoid robot developed by Kawada industry Co. Ltd in Japan. About 150 units are currently working in factories together with human workers [6].

There are also many efforts on the development of robot software [7]. A comparison of robotic middleware is presented in [8]. ROS (Robot Operating System) is being used extensively by the research community for service robotics [9]. In particular, ROS has been applied to industrial robotics through ROS-Industrial, which is an open-source project that extends the advanced capabilities of ROS software to manufacturing, which contains libraries, tools and drivers for industrial hardware [10]. However, this software package is still for experienced developers. It takes a long time to build a robot application. A Japanese startup company Mujin released the Mujin controller, which claims it is the world's first commercial industrial robot motion controller for real-time applications like bin-picking and multi-robot manipulation planning. Mujin is capable of connecting with several different industrial robots to enhance their capabilities [11].

In this paper, a software package is introduced for advanced industrial robot applications. This software package is developed as a part of RADOE (Robot Application and Operating Environment) work package in the Singapore A*STAR industrial robotics research program. The rest of this paper is organized as follows: In Sect. 2, an architecture design of the software package is proposed. Functions of each module are briefly described. In Sect. 3, software modules design implementation issues are addressed in terms of simulations, graphic interface and visual programming, intermediate robot language, program manager and system integration. In Sect. 4, an automated robot bin-picking application and an automated robot 3d object taping application are demonstrated to evaluate practical operation performances of the software package.

2 Architecture Design

Figure 1 shows a scheme of the integrated software package architecture for advanced industrial robot applications, which consists of six modules. Functions of each module can be described in the following:

- (1) Simulation module is used to test developed control algorithms and for operation performances evaluation purposes. Simulation saves a lot of testing time for a complex system before the software deploying on a real robot. In particular, robot and devices can be conveniently enabled or disabled by configure macro parameters for a hardware in the loop simulation.
- (2) Intermediate Robot Language (IRL) module is to describe a task that can be implemented on different robotics platform. IRL releases people from managing different robot programs in different languages that depend on manufactures'. It also reduces the effort required to plan, apply and maintain core robotic programming techniques, saving time and money.
- (3) Advanced capabilities module is a cluster of open source software packages available from robotics research community. This module takes advantages of other people's work and advanced capabilities for industrial robot applications thus can be easily integrated.
- (4) Interface module consists of two sub-modules: comparing with tedious manual creates ROS components and coding, visual programming tool is to specify programs by manipulating function modules. It allows access to software libraries for rapid robot application development with appropriate APIs (Application Programming Interfaces). Visual user interface sub-module is used to configure the programming and operation environment, setting environment variables, as well as for parameters tuning purposes.

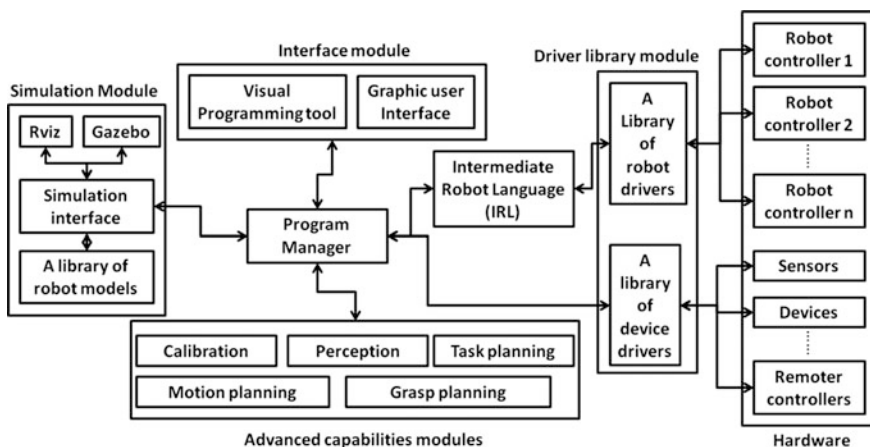


Fig. 1 A scheme illustrates an architecture design of the integrated software package

- (5) Driver library module contains two sub-libraries. One library includes device drivers for communicating and control of a variety of robots. In general, robot manufactures provide these ROS compatible robot hardware drivers. Another library is the device drivers for peripherals (such as conveyors, mobile bases, rotary tables, etc.), actuators and sensors.
- (6) Program manager module is the core of whole software package, which is responsible for setting up the whole program through parameters configuration files. It monitors operation sequence of each sub-module and plays a role of error recovering. In addition, program manager integrates other modules during the software package implementations.

3 Module Implementation

3.1 Simulation Module

Simulation module is developed based on Rviz and Gazebo in ROS environment. Rviz (ROS visualization) is used for displaying a virtual model of the industrial robot and live representations of sensor values coming over ROS Topics including camera data, laser scanner point cloud data and etc. Gazebo is a dynamic simulation environment, which can communicate with simulation interface via Gazebo-ROS API plugging. A library of robot models is included in the simulation module. Robot model are described in URDF (Unified Robot definition format) files. URDF files can be converted to SDF files via the use of XACRO for Gazebo simulations.

3.2 Intermediate Robot Language

Intermediate Robot Language (IRL) module enables users to conveniently program and design robotic applications in a generic and scalable way. The proposed IRL is completely independent to any robot manufactures' language. Development of the IRL commence in conjunction with the simulation platform. An IRL program consists of three components: .dat file-contains robot set up and target data;.prog file-contains individual robot programs; .task file-contains task level commands (for example call robot programs, timing, robot I/O etc.) These files can be individually categorised within an xml file as shown in Fig. 2.

3.3 Graphical Interface and Visual Programming

A user-friendly graphical interface and visual programming tool are created in C++ using QT creator. Visual programming tool allows a user to drag and drop different

```

<IRL_Program id="bk101">
  <data>
    ABB_2400 = {kin_model..}; rob_targ1 = {0,1,3,51,1...}; .....
  </data>
  <Programs>
    <prog id="Rob1_1"><Robot>ABB_2400</Robot>
      <motions>
        MoveJ{rob_targ1,100,100} MoveJ{rob_targ2,100,100} .....
      </motions>
    </prog>
    .....
    <prog id="Rob2_1"><Robot>ABB_4400_60</Robot>
      .....
    </prog>
    .....
  </programs>
  <tasks>
    Call Program "Rob1_1" {run_time="15" Max_Vel="1.1" ... ..}
    Wait SIG[1] then Wait 10 and Sig[2]="true"
    Start Concurrent : "Rob1_2" "Rob2_1"
    End Concurrent
    .....
  </tasks>
</IRL_Prog>

```

Fig. 2 An example xml file illustrates the intermediate robot language file structure

components onto canvas from a library. Each component is a ROS launch file and the connections between components represent ROS messages being passed through ROS topics. The output of the visual programming tool is automatically created ROS launch files with parameters, “remap”-ing and dependencies. This module has following functions:

- (1) Creating, saving, loading project files on a graphical interface
- (2) Adding components such as motion planning and grasping planning
- (3) Run simulations through corresponding launch files
- (4) Ability to drill down into a module to see the various sub-components and rewire them as necessary
- (5) Auto-populating all launch files available in a build ROS package

3.4 Program Manager and Integration

Program manager module bridges communications between different sub-modules by using ROS topics, services and actions. For each sub-module, different message types are created on the base of existing ROS messages format. These messages communication are base on the publishing and subscribing mechanism, which can

be defined in ROS msg files. For services and actions, message types are defined in .srv and .action files, respectively.

In particular, two messages ‘ErrorHandle.msg’ and ‘ModuleState.msg’ are defined. When an error in one sub-module occurs, error message is published through ‘ErrorHandle.msg’ for error recovering purposes. As for ‘ModuleState.msg’, it is published repeatedly in every 1 s in order to monitor the running status of each sub module. In addition, configuration parameters for each sub-module are stored in.xml files, which are included in launch files. Each sub-module runs as a node in ROS environment.

4 Example Demos

Two experiment demos have been built in the laboratory to evaluate operation performances of the developed software package. One demo is an automated robot bin-picking system with aims to pick different shaped items from a stationed shelf according to a wish list. This demo demonstrates an advanced capability of an industrial robot by integrating 3d object recognition, real-time obstacles avoidance motion planning, grasping and task planning. The second demo is an automated robot 3d object taping system. This demo aims to show flexibility of the software package, which is capable of generating and executing a taping motion trajectory after receiving point clouds data of a scanned 3d object.

4.1 Demo I: Automated Robot Bin-Picking

Picking and placing is one of the most common tasks for traditional industrial robots. In general, the robot is programmed to perform repetitive picking for a high-volume object with similar shapes in a simple constructed environment. Automated picking of different shaped objects in a cluttered environment is still very challenge for traditional industrial robots. Figure 3a shows an experiment setup of the automated robot items picking system, which is composed of a UR5 lightweight robot manipulator, a versatile robot gripper, a Kinect 3d sensor and a shelf with different shaped grocery products. One challenge is automatic recognition of the target object, thus a library including RGB and depth images, 3d point clouds model of all candidate objects are built. In the experiment, the robot is capable of generating a collision avoidance path and deploying a suitable grasping strategy to pick up the object.

Figure 3b shows a snapshot of the simulation. Collision avoidance motion planning algorithm and tasking planning strategy can be tested with real streaming data from Kinect sensor, which saves a lot of time for testing. Figure 4 shows snapshots of experiment tests for a bin-picking movement. The robot manipulator successfully picks a yellow-duck-bath-toy for one shelf bin and places it to an order bin.

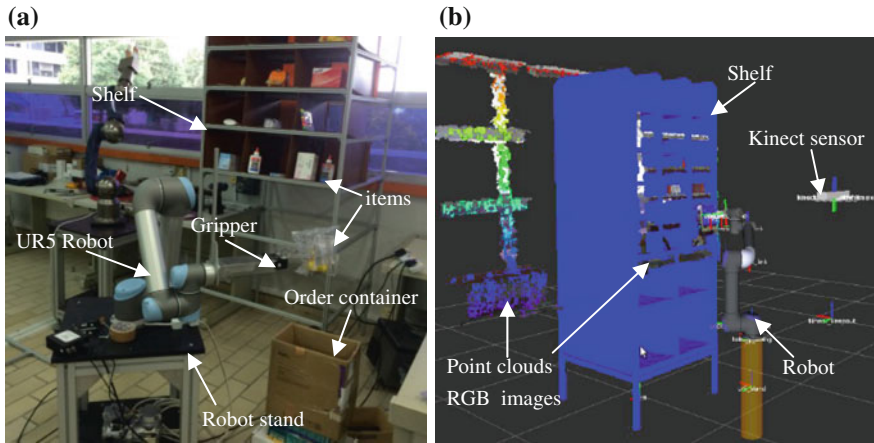


Fig. 3 Automated robot picking demo: a an experiment setup; b a snapshot of the simulation

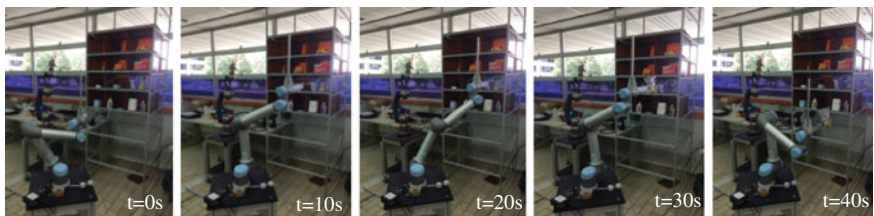


Fig. 4 Snapshots of experiment test for picking an object from the shelf

4.2 Demo II: Automated Robot 3d Object Taping

Another experiment demo is the automated robot 3d object taping as shown in Fig. 5a. The automated taping system is composed of a UR10 lightweight robot manipulator, a taping mechanism mounted on the robot end-effector, a Kinect 3d sensor and a rotation platform. A 3d model of the tapped object can be obtained by receiving scanned data from Kinect sensor. Figure 5b shows an example 3d model of the scanned object. A motion planning algorithm is used to generate a motion trajectory on the surface of the tapped object. In particular, the 3d model scanning algorithm has been built in the software package so that it is capable of taping different objects in a very short of setup time. Furthermore, taping movement can be simulated in a simulation environment to evaluated prescribed taping velocity and dimension of overlap between each layer.

Figure 6 shows snapshots of the robot taping experimental test. It can be noted that tape is pressed on the object surface. The compliance of the taping mechanism maintains certain of taping force. In the taping procedure, the robot moves up-down

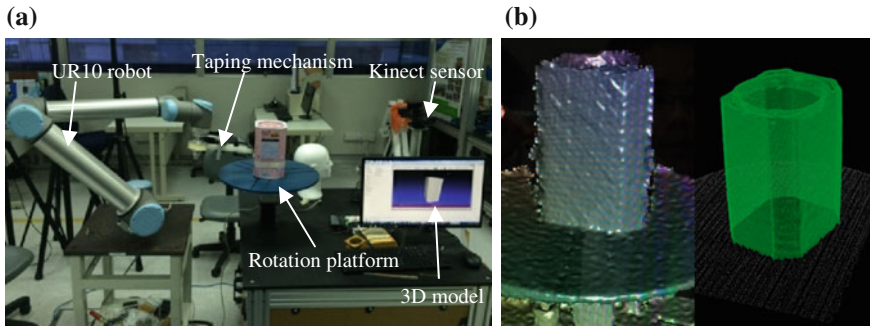


Fig. 5 Automated robot taping demo: **a** an experiment setup; **b** 3d models of the taped object



Fig. 6 Snapshots of experimental test for an automated robot taping

and keeps taping mechanism head normal to the object surface. The motion of rotation platform is synchronized with robot to ensure a good quality taping. Tapes are taped on the object surface for with prescribed velocity and overlap.

5 Conclusions

This paper introduces an integrated software package for advanced industrial robot applications. The open architecture of this software package makes it is very easy and flexible to integrate high-level robotic software modules and high value-added robot software library for diversified industrial robotic applications. Advantages of deploying this software package in industrial robots can be summarized as easy-to-use, easy-to-programming, and flexible with advanced capabilities. From two experiment demos, utility and flexibility of the developed software package are evaluated. This integrated software package has the potential for a wide of industrial robot applications.

Acknowledgments This work has been partially supported by the Singapore Agency for Science, Technology and Research (A*STAR) under SERC industrial robotics research program Project No. 1225100008.

References

1. Hägele, M., Nilsson, K., Pires, J.N.: *Industry Robotics*, (Springer Handbook of Robotics), pp. 963–986. Springer, New York (2008)
2. BMBF: *Industry 4.0*. <http://www.bmbf.de/en/19955.php> (2015)
3. Rainer, B., Kurth, J., Schreiber, G., Koeppel, R., Albu-Schäffer, A., Beyer, A., Eiberger, O.: The KUKA-DLR Lightweight Robot arm—a new reference platform for robotics research and manufacturing. In: *Proceedings 2010 6th German conference on robotics (ROBOTIK)*, pp. 1–8. VDE (2010)
4. Jingguo, G., Zhaofu, C., Qingwei, L.: Small part assembly with dual arm robot and smart camera. In: *Proceedings of International Symposium on Robotics, ISR/Robotik 2014*, pp. 1–6. VDE (2014)
5. Ostergaard, E.H.: Light weight robot for everybody, *IEEE Robot. Autom. Mag.* **19**(4) (2012)
6. Christian, S., Karayiannidis, Y., Nalpantidis, L., Gratal, X., Qi, P., Dimarogonas, V., Kragic, D.: Dual arm manipulation—a survey. *Robot. Auton. Syst.* **10**, 1340–1353 (2012)
7. Nader, M., Al-Jaroodi, J., Jawhar, I.: Middleware for robotics: a survey, robotics, In: *Proceedings of the 2008 IEEE Conference on Automation and Mechatronics AIM 2008*, pp. 736–742 (2008)
8. Gergely, M., Sinčák, P., Krizsán, Z.: Comparison Study of Robotic Middleware for Robotic Applications, *Emergent Trends in Robotics and Intelligent Systems*, pp. 121–128. Springer International Publishing, New York (2015)
9. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Andrew, YNg.: ROS: An open-source robot operating system. In: *ICRA workshop on open source software*, vol. 3.2, p. 5 (2009)
10. Edwards, S., Chris, L.: ROS-industrial—applying the robot operating system (ROS) to industrial applications. In: *IEEE International Conference on Robotics and Automation, ECHORD Workshop* (2012)
11. Diankov, R., James, K.: Openrave: A Planning Architecture for Autonomous Robotics, Robotics Institute Technical Report, CMU-RI-TR-08-34 79 (2008)