Study on Ubiquitous Robotic Systems for Smart Manufacturing Program*

Qixin Cao, Wenshan Wang, Xiaoxiao Zhu, Chuntao Leng, Masaru Adachi

***Abstract*—As the manufacturing tasks becoming more individualized and more flexible, the machines in smart factory are required to do variable tasks collaboratively without reprogramming. In this speech, we discuss the similarity between smart manufacturing systems and the ubiquitous robotic systems, and introduce the efforts we made on deploying ubiquitous robotic technology to the smart factory. Specifically, a component based framework is designed in order to enable the communication and cooperation of the heterogeneous robotic devices. Further, a planning method based on automated planning techniques is implemented to coordinate the devices for various tasks. We also use a test bed of smart factory to demonstrate the effectiveness of the proposed framework. Advances in planning technologies and cost reduction have brought the systems into the range of even small-to-medium enterprises.**

I. Introduction

The industrial robots have brought sustained productivity increases and manufacturing growth. However, the traditional program-by-teaching method, which takes considerable time and requires extensive expertise, has kept them out of low-volume, time-critical tasks [1, 2]. As the manufacturing tasks becoming more individualized and more flexible, it shows great prospect for the development of smart factories, where machines are not likely to be pre-configured for doing repeating jobs, but doing variable tasks collaboratively with each other and coping with a wide variety of unexpected environmental and operational changes.

This feature of doing various tasks utilizing collaboration of robotic devices shares common ideas with the ubiquitous robotic technology, which is mainly applied in service robots domain. In this perspective, the novel industrial manufacturing system could take advantages of the ubiquitous robotic technology.

Compared to the traditional industrial producing process, the smart factory encounter similar problems with the ubiquitous robotic systems [3, 4]. First, the distributed machines may be highly heterogeneous both with regard to hardware platform and software implementation - a state of affairs which presents considerable difficulty with regard to the communication and collaboration. The first problem is thus how to integrate these large amounts of heterogeneous robot devices while enabling painless modification, expansion and deletion. Second, there are a variety of customer orders and different situations for each order in the agile manufacturing domain. As a result, it requires up level task planning, that handle the various tasks and dynamic environment without recoding the robots.

Many existing researches focus on how to integrate RFID into the manufacturing system to collecting more data [5-7]. The manufacturing is smarter by tracking the processing information. We argue that it would achieve higher flexibility and intelligence if connecting not only the production but all the machinery processes, so that different robotic devices could collaborate into different groups according to different tasks. In the ubiquitous robotic systems, the most commonly employed techniques are based on Artificial Intelligence (AI). Young-Guk Ha et al. used SHOP2 planner to decompose services based on semantic knowledge [8]. Robert Lundh et al. implemented a configuration approach for their network robot system also based on SHOP planner [9]. Esra Erdem et al. presented an application of answer set programming (ASP) to housekeeping robotics [10]. Tim Niemueller et al. approached the task planning problem by deploying a rule engine [11]. In order to cope with the uncertainties brought by the human interference and environmental changes, some researchers have used probabilistic models in task planning problems. For example, Marco Barbosa et al. used Partially Observable Markov Decision Processes (POMDP) to model the tasks with uncertainty [12]. Marcello Cirillo et al. implemented RTLplan for probabilistic domains [13]. These AI based planning methods play an important part in the ubiquitous robotic systems, and could also be applied to the manufacturing problems in the smart factory, which could reach a higher level of flexibility and agility.

In view of the foregoing, we propose in this paper a framework of smart factory that takes advantages of ubiquitous robotic technology. We employ a component based method to abstract each machinery process as a module with standardized communication ports. So different machines are able to communicate and cooperate with each other upon these ports. Furthermore, a task planning method based on general purpose automated planning method is developed to coordinate these components according to customers' orders. A study case of the smart factory is implemented as a demonstration platform for our methods.

II. System Architecture

In contrast to traditional manufacturing processes, the smart manufacturing offers the advantage of distributed networked machines to complete different tasks through collaboration. The framework for smart factory is designed as Figure 1.

In the low layer, the robotic devices are developed into components that they can "plug and play" in the system and be reused and reconfigured according to different manufacturing process. These components are the foundation of the system.

* Resrach supported by Yaskawa Electric Corporation.

Qixin Cao, Wenshan Wang, Xiaoxiao Zhu and Chuntao Leng are with the Institute of Robotics, Shanghai Jiao Tong University, Shanghai, China (Chuntao Leng: 13816896878; e-mail: 1352481640 @ qq.com).

Masaru Adachi is with the Yaskawa Electric Corporation, Fukuoka, Japan

As mentioned, robotic components are highly heterogeneous with respect to platforms such as operating system, programming language and communication media. Middleware is thus employed to generalize the components into a uniform abstraction which enables dynamic communication and coordination between any two of the modules [14]. This also brings benefits to the modification of existing devices and the expansion of new ones.

In the upper layer, a number of functionalities are developed in the internal cloud, such as the human-system interface, storage management, task planning, virtual manufacturing and big data collection. The customer orders products through a human-system interface. The order includes customized requests, for instance the favorite color and shape of the parts and whether the parts being polished etc. These orders are sent to the task planning module, which also utilizing the information from the storage management module. The planner is the key part of the system's agility and intelligence. It turns customers' orders into sub-task sequences, which can be directly carried out by corresponding robotic components. It is a general purpose planner, which will be detailed later.

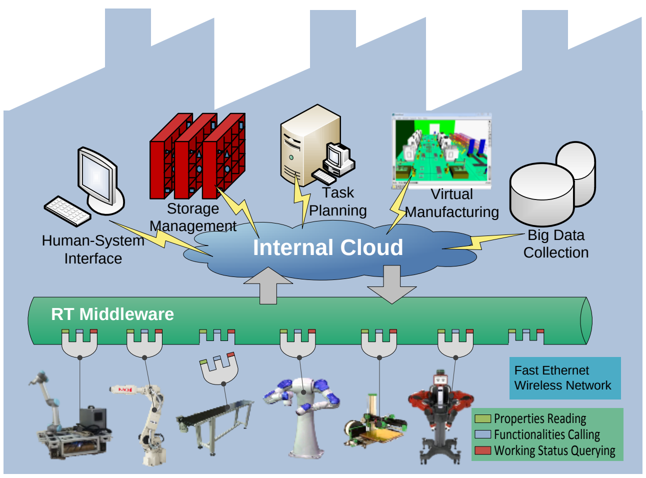


Figure 1. System architecture of the smart factory

III. COMPONENT-BASED MACHINERY PROCESS

Components use ports to communicate with each other and with high level controller. The ports are categorized into data ports and service ports [15]. The data port is responsible for the continuous exchange of data. Each component can have any number of data in-ports and out-ports. A data out-port sends the data to a corresponding in-port which receives the data. The service port provides the command based communication. The component with a service port, offering a set of services, listens for requests for those services via a connector.

Each component has three service ports, namely FuncGet, FuncSet and ExeStatusGet. The service port is responsible for the interaction with the upper layer. FuncGet port reports to the service layer about the components' state. For example, the polishing robot reports the available polishing configuration; the Autonomous Intelligent Mobile Manipulator (AIMM) feeds back its coordinates, etc. FuncSet port provides the functionality invoking, such as setting the target position for the AIMM, start polishing with certain configuration, etc. ExeStatusGet port returns the execution status, for example whether or not the AIMM has reached its destination, or whether the polishing robot succeed or fail in doing the task.

Each component may have any number of data ports for continuous data exchange between components. For instance, the localization information is transferred from the data out-port of laser component to the data in-port of the path planning component. Once two data ports are connected, those two components are able to perform real-time communication to accomplish the task collaboratively.

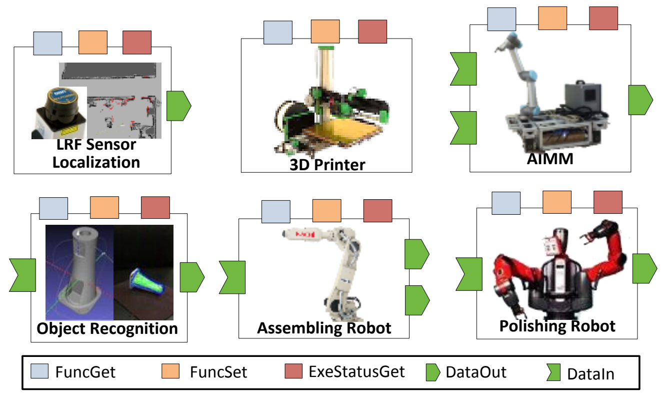


Figure 2. Some of the robotic components in our system. Each of the components has data ports and service ports.

A. Polishing Component with path auto-generation

Traditionally, the polishing path is taught by the expert engineers. This teaching process could be complex and tedious [16,17]. In our smart factory, the polishing path is automatically generated from the CAD data (Figure 3 (b-c)). Then, the robot follows this path by a motion planning algorithm with collision avoidance (Figure 3 (d)). Besides, the polishing area is easy to specify with a user-friendly GUI as Figure 3 (a).

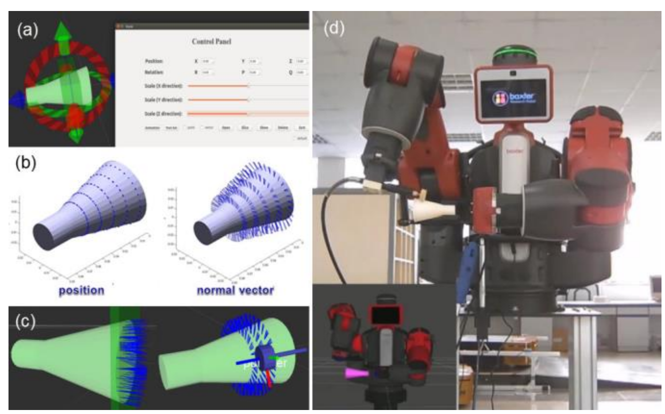


Figure 3. (a) Configure the polishing area, (b) Auto-generate the polishing path, (c) Path generation and tool simulation (d) Motion planning and polishing with dual-arm robot

The FuncSet service port of this component provides the polishing functionality calling. In the upper layer, the task planning module calls on this service port following the results generated by the planner. Every functionality of the service ports corresponds to one symbolic action of the planning domain. This polishing functionality is corresponds to the action: polish(polisher, object, configuration).

B. AIMM Component

AIMM is responsible for the transportation task that transports parts and work pieces between workstations and storages. Such transportation tasks contain physical separation larger than the workspace of the robot manipulator. This requires a lot of technologies such object recognition, grasp point generating, motion planning, localization, path planning and etc. It uses RGB-D camera for the object recognition and obstacle avoidance, and uses laser sensor for the localization.

This component provides three functionalities, picking up an object from working spots, putting down an object from working spots and moving itself between working spots. These introduce three actions of the planning domain, which are move(AIMM, location1, location2), pickup(AIMM, object, location), and putdown(AIMM, object, location).

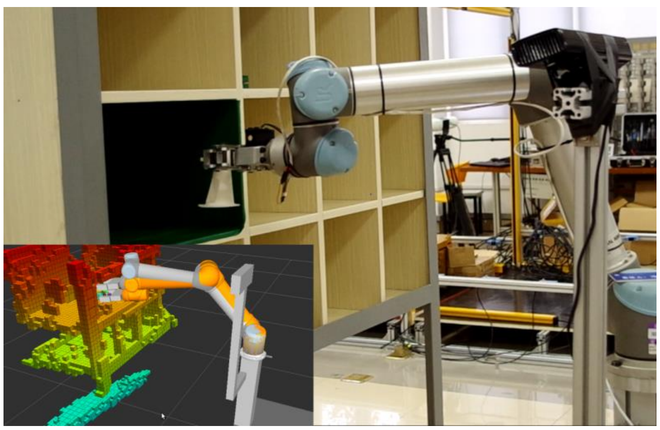


Figure 4. AIMM is picking up a working part form the warehouse. The left bottom scene is from the RGB-D camera. The motion planner convert the 3D data to grid obstacles.

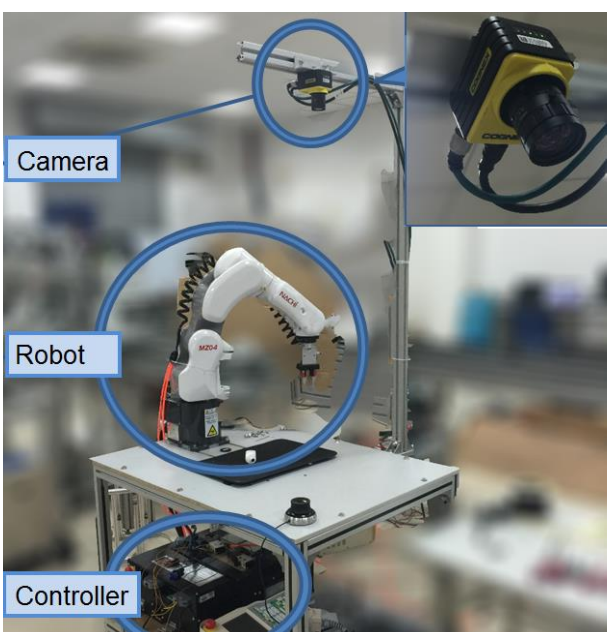


Figure 5. Assembling robot with visual detection. It consists a camera, a robot arm and a controller.

C. Assembling Component

The assembling robot also has the sensing capability. The working parts are detected and located online with a camera. The visual detecting method is able to recognize complex shape. The localization error is within 1mm.

The assembling component provides the assembling functionality. The corresponding action for planning module is assemble(assembler, part1, part2, configuration). The assembler could handle different types of assembling tasks. For example, it is able to assemble parts with different shapes, different orientations, or different joint shape. This information is also calculated by the task planner, and passed to the assembler through the 'configuration' parameter.

D. Object Recognition Component

Object Recognition is the foundation of different kinds of robot tasks such as polishing, assembling and transferring. The object recognition component is based on the RGB-D sensor. It detects the positions and orientations of target object, which is usually texture-less in manufacturing context. This study employs a combination of 2D template matching and 3D pose estimation techniques as Figure 6. shows. The composite of template consists of two parts, Gradient Orientation Map and 3D Orientation Coarse Estimation.

This component provides the recognition and localization of a number of predefined objects. It returns the object's localization and orientation. The corresponding action is objRecognize(camera, objName).

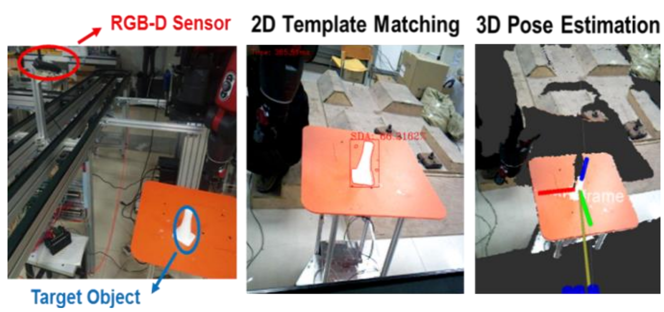


Figure 6. Texture-less object recognition using combination of 2D template matching and 3D pose estimation

E. Virtual manufacturing

A simulation environment is implemented using AutoMod® as Figure 7. shows. It empowers the designers to achieve a better layout of the machines, optimize the device configurations and fast adapt to change of the manufacturing task.

The simulation process plays an important role in the designing and implementation. It generates the statistic results on production, rate of capacity utilization and etc. It helps to improve the configuration of the production line. The Figure 7(c) shows the production number in 12 hours simulation. It is also reported that the cutting process of the CNCs on the right side is time consuming. Improve this process process is the key issue to improve the factory's production efficiency.

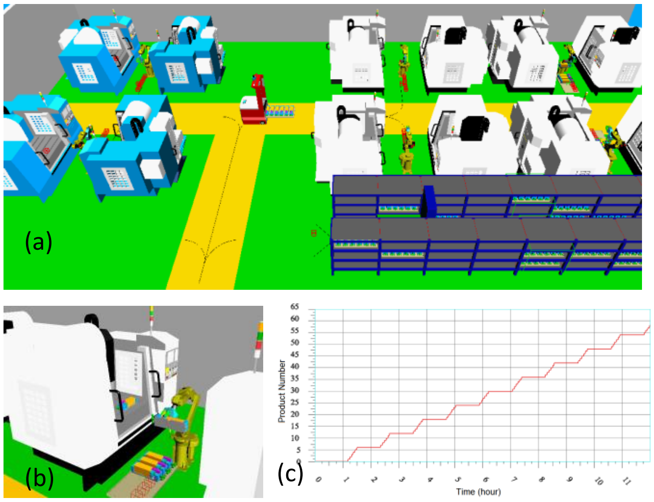


Figure 7. (a) The simulation environment of the virtual manufacturing system, (b) Simulation on CNC and industiral robot arm, (c) The manufacturing efficiency statistics

IV. General Task Planning Module Based On Automated Planning Technique

The task planning module is a crucial part in the smart manufacturing system. The problem of task planning is a hard open problem for distributed systems. In the industrial domain, the tasks are complicated and the situations are dynamic. It is unlikely to predefine all the possible states. As a result, a flexible and robust planning method is needed. What's more, it is supposed to be a domain-independent general approach for solving a variety of problems.

A. *Task Modeling*

Task modeling is the precondition of the task planning. The quality of the planning result is greatly depends on the expressivity of the task model. On the other hand, the more complicated of the model, the more difficult for the planner to solve the problem.

This paper follows the techniques in automated planning field. The Task planning problem is modeled as a state transition system. Formally, it is modeled as a five-tuple $\prod = (S, A, c, I, G)$, where:

- $S = \{s_1, s_2, \cdots\}$ is a finite set of world states;
- $A = \{a_1, a_2, \cdots\}$ is a finite set of actions, each $a \in A$ is a triple $(name_a, pre_a, eff_a)$ referred to the action's name, precondition and effects respectively.
- $c : A \mapsto \mathbb{R}_0^+$ is the cost function;
- $I \subseteq S$ is a set demotes the initial state;
- $G \subseteq S$ is a set denotes the goal state.

To further depict the planning domain and planning problem, the Planning Domain Definition Language (PDDL) [18] is employed. Some sample actions are shown below, representing the moving capability of the mobile robot and the grasping capability of a robot arm.

```
(:action drive
  :parameters (?r - mobile ?start - place ?dist - place)
  :precondition (and (at ?r ?start) (can-locate ?r))
  :effect (and (at ?r ?dist) (not (at ?r ?start))))
(:action pickup
  :parameters (?a - arm ?o - object ?p - plane)
  :precondition (and (beside ?a ?p) (on ?o ?p))
  :effect (and (in ?o ?a) (not (on ?o ?p))))
(:action putdown
  :parameters (?a - arm ?o - object ?p - plane)
  :precondition (and (beside ?a ?p) (in ?o ?a))
  :effect (and (on ?o ?p) (not (in ?o ?a))))
```

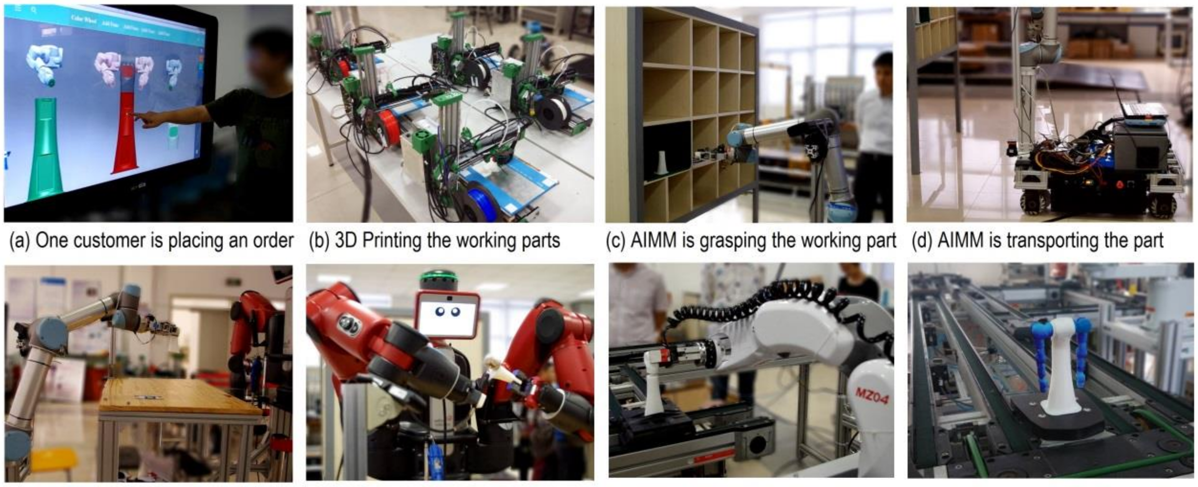
B. *Task Planning*

Inspired by International Planning Competition (IPC), the automated planning technology has been significantly improved these years. The increase was mainly due to three fundamental approaches in plan generation. First, the Graphplan approach [19] improved the planning efficiency by a relaxation method based on planning graphs. The second approach is the planning as satisfiability method [20], which uses propositional reasoning to solve the planning problem. The third is the heuristic searching [21] that accelerates the search speed with heuristic function.

This paper employs the heuristic search based algorithm to solve the planning problem we defined above, referring to the Fast Downward (FD) planner [21]. The PDDL files are translated to build a search space, which can be seen as a directed graph, where the node denotes the state of the system, and the link denotes the action that make the system transfer from one state to another. FD searches the shortest path that starts from the initial state and reaches the goal state. The links on the path compose an action sequence, which is the planning solution. We improve the FD planner by adapting it to the online planning system. The detailed algorithm is shown below.

Algorithm 1: Task planning

while exists task T uncompleted:
    for each alive component C_i :
        $s_i \leftarrow$ readState(C_i)
        if s_i is ERROR_STATE: reset(C_i) endif
    endfor
    $S_{init} \leftarrow$ analyzeState($s_0, s_1, \cdots$)
    $S_{goal} \leftarrow$ analyzeTask(T)
    $P_{task} \leftarrow$ taskModelPDDL(S_{init}, S_{goal})
    $T_{result} \leftarrow$ FDplanner(P_{task}, P_{domain})
    for each sub-task t_i in T_{result}:
        while($r_i \leftarrow$ execute(t_i) not complete) endwhile
        if r_i is SUCCESS: continue
        else: break with failure
        endif
    endfor
    if not failure: mark T as completed
endwhile



(a) One customer is placing an order (b) 3D Printing the working parts (c) AIMM is grasping the working part (d) AIMM is transporting the part

(e) Placing the part onto the polishing spot (f) Polishing the part (g) Assembling two working parts (h) The order has been successfully processed

Figure 8. The execution process of one manufacturing task

C. Combining middleware and task planner

As illustrates, the service layer and the device layer communicate through 3 kinds of service ports, namely FuncGet, FuncSet and ExeStatusGet. The Device Manager is developed as the bridge between these two layers.

Firstly, the FuncGet service ports are used by readState(C_i) function with respect to Algorithm 1. Each component C_i provides the functionality of reporting its own states. For instance, the object recognition component reports the name of objects that are currently in its view. All these states are translated by the Device Manager, and then form the initial state fed to the task planner.

Secondly, the planning result is in the form of action sequences, such as ‘drive AGV spot1 spot2’, ‘pickup AIMM object1 polisher_station’, etc. Notice that the first item is the action name, and the second item is configured as the component name, of who is in charge of this action. Device Manger compiles each action into a method-call through associated FuncSet service port. For example, the above two actions are compiled as AGV.move_to(spot2_x, spot2_y), AIMM.pickup(object1_id) respectively. These methods are defined in an interface definition language (IDL) file for each service port.

Thirdly, when executing each action, it is important to monitor its status. If it’s successful, it can move on to the next action, while if it’s failing, it can start that over again. The ExeStatusGet service ports are responsible for reporting the execution status. There are 4 types of status, namely idle, running, success and failure.

Another way to execute an action is to connect two components’ data ports. As the localization example of the video shows, every time the mobile robot switching localization component, it switches the data port, to which it connect its own data port.

Two major benefits of this approach are it allows an easy extension with new components and allows an easy transition to new task domains. Adding new components has little side effect on the existing ones and the planning module. All that need to take care is to define the three kinds of service ports or other necessary data ports. Besides, same set of components can be used in different task domains, as long as the PDDL files are provided.

V. Experiments and Results

A smart factory was implemented based on the ubiquitous robotic technology. It took in customers’ individualized order and arranged the producing process accordingly. Figure 8 shows one execution of the smart factory task. First, the customer made an order through the user interface. The order was then sent to the task planning module, which calculated the action sequence hierarchically. 3D printers started to make parts with specific color and shape as Figure 8 (b). Meanwhile, the AIMM transported the part from the storage to the polishing station as shown in Figure 8 (c-e). After that, the dual-arm polishing robot polished the part according to customer’s configuration as Figure 8 (f). At last, the parts were transported to the assembling spot after which the product was successfully processed as Figure 8 (g-h).

With the component-based framework, every machinery process is ready to cooperate with each other. For instance, the continuous localization data is transferred from the laser sensor to the AIMM’s path planning module through data port. And the polishing robot gets the location of object from the object recognition component. Further, this modular framework also facilitates the easy expansion of new devices and painless modification of the existing devices. For example, when we added new AGVs to the smart factory, no modification is needed for the system architecture, planning algorithm and any of other components. All that needed is to register the added AGVs to the planner, so that they can be called by the planner.

Compared to the traditional manufacturing systems, our system is more flexible and efficient. The industrial robots in our systems are all capable of sensing and planning techniques. Such as the picking, placing, polishing and assembling, none of these robots are setting by teaching methods. As a result, it is more accessible to dynamic tasks.

For example, the polishing robot in our system is capable of polish objects with different shapes and polishing areas; and the assembling robot is able to assemble working parts in in different locations and from different directions. We also upgrade the system with more AGVs for transferring the parts between the storage center and the working station. No modification is needed when deploying the existing components to the upgraded one. All that needed is adding some new components, and upgrading the domain description file. The components and the planning module are reusable for different domains.

VI. Summary and Future Work

Given the increasing popularity of smart manufacturing as a solution offering better autonomy, this paper discussed the similarity of the smart manufacturing with the ubiquitous robotic system. A component based framework has been proposed, and proved to be suitable for industrial domain. Further, since the manufacturing problems are often in large-scale with uncertainties, a planning method based on automated planning techniques is implemented to coordinate the devices for various tasks.

A smart factory was implemented as the testing bed of our framework and algorithms. The individualized orders were processed by the system that arranged the producing process accordingly. The results showed that the framework facilitates the communication and cooperation between the robotic components. Further the planning method has enabled the system to tackle various tasks in dynamic situation.

It is our view that the results obtained from this work represent a substantial improvement over the some of the more common approaches. Advances in planning technologies and cost reduction have brought the systems into the range of even small-to-medium enterprises.

Acknowledgment

The authors gratefully acknowledge YASKAWA Electric Corporation for supporting the collaborative research funds under the project of “Research and Development of Key Technologies for Smart Factory”.

References

[1] J. Lee, H.-A. Kao, S. Yang, Service innovation and smart analytics for industry 4.0 and big data environment, Procedia CIRP, 2014, 16: 3-8.

[2] Davis J, Edgar T, Porter J, et al. Smart manufacturing, manufacturing intelligence and demand-dynamic performance[J]. Computers & Chemical Engineering, 2012, 47: 145-156.

[3] A. Chibani, Y. Amirat, S. Mohammed, E. Matson, N. Hagita, M. Barreto, Ubiquitous robotics: Recent challenges and future trends, Robotics and Autonomous Systems, 2013, 61: 1162-1172.

[4] W. Wang, Q. Cao, X. Zhu, S. Liang, A Framework for Intelligent Service Environments Based on Middleware and General Purpose Task Planner, in: International Conference on Intelligent (IE), IEEE, 2015, 184-187.

[5] Huang G Q, Zhang Y F, Jiang P Y. RFID-based wireless manufacturing for real-time management of job shop WIP inventories[J]. The International Journal of Advanced Manufacturing Technology, 2008, 36(7-8): 752-764.

[6] Ngai E W T, Chau D C K, Poon J K L, et al. Implementing an RFID-based manufacturing process management system: Lessons learned and success factors[J]. Journal of Engineering and Technology Management, 2012, 29(1): 112-130.

[7] Zhong R Y, Dai Q Y, Qu T, et al. RFID-enabled real-time manufacturing execution system for mass-customization production[J]. Robotics and Computer-Integrated Manufacturing, 2013, 29(2): 283-292.

[8] Y.G. Ha, J.C. Sohn, Y.J. Cho, H. Yoon, A robotic service framework supporting automated integration of ubiquitous sensors and devices, Information Sciences, 177 (2007) 657-679.

[9] R. Lundh, L. Karlsson, A. Saffiotti, Autonomous functional configuration of a network robot system, Robotics and Autonomous Systems, 2008, 56: 819-830.

[10] E. Erdem, E. Aker, V. Patoglu, Answer set programming for collaborative housekeeping robotics: representation, reasoning, and execution, Intelligent Service Robotics, 2012, 5: 275-291.

[11] T. Niemueller, G. Lakemeyer, A. Ferrein, Incremental Task-Level Reasoning in a Competitive Factory Automation Scenario, in: AAAI Spring Symposium: Designing Intelligent Robots, 2013.

[12] M. Barbosa, A. Bernardino, D. Figueira, J. Gaspar, N. Gonçalves, P.U. Lima, P. Moreno, A. Pahliani, J. Santos-Victor, M.T.J. Spaan, ISRobotNet: A testbed for sensor and robot network systems, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, pp. 2827-2833.

[13] M. Cirillo, L. Karlsson, A. Saffiotti, A Human-Aware Robot Task Planner, in: Proceedings of the 19th International Conference on Automated Planning and Scheduling, AAAI press, 2009, 58-65.

[14] N. Ando, T. Suehiro, K. Kitagaki, T. Kotoku, W.-K. Yoon, RT-middleware: distributed component middleware for RT (robot technology), in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), IEEE, pp. 3933-3938.

[15] Wang W, Cao Q, Zhu X, et al. An automatic switching approach of robotic components for improving robot localization reliability in complicated environment[J]. Industrial Robot: An International Journal, 2014, 41(2): 135-144.

[16] Zhai M, McKenna G B. Surface energy of a polyurethane as a function of film thickness, Proceedings of the annual technical conference, Society of Plastics Engineers, ANTEC, Las Vegas. 2014.

[17] Zhai M, Yoon H, McKenna G. A Comparison of Particle Embedment and Nanoindentation: Probing the Surface Properties of Polymeric Materials, Bulletin of the American Physical Society, 2015, 60.

[18] D. McDermott, “PDDL–the planning domain definition language,” *the AIPS’98 Planning Competition Committee*, 1998.

[19] A. L. Blum, M. L. Furst, “Fast planning through planning graph analysis,” *Artificial Intelligence*, 2005, pp.279-298.

[20] H. A. Kautz, B. Selman, “Pushing the envelope: Planning, propositional logic, and stochastic search,” In *Proc. AAAI-96*, pp. 1194-1201.

[21] B. Bonet, H. Geffner, “Planning as heuristic search,” *Artificial Intelligence*, 2001.