Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia CIRP 7 (2013) 515 – 520

www.elsevier.com/locate/procedia

Forty Sixth CIRP Conference on Manufacturing Systems 2013

Developing concepts for improved efficiency of robot work preparation

M.S. Essers[a]*, T.H.J. Vaneker[a]

[a]Laboratory for Design, Production and Management, Dept. of Engineering Technology, University of Twente, Enschede, The Netherlands

* Corresponding author. *E-mail address*: m.s.essers@utwente.nl

Abstract

SInBot[1] is a large research project that focuses on maximizing the efficient use of mobile industrial robots during medium sized production runs. The system that will be described in this paper will focusses on the development and validation of concepts for efficient work preparation for cells of intelligent mobile robots that execute medium sized production runs. For a wide range of products, the machining tasks will be defined on an appropriate level, enabling control over the robots behavior and performance. When the system, system operator, and robots have more experience with a product the system can be controlled on a higher level (i.e. the subsystems or even robots can start allocating and executing tasks by themselves). Different test beds are used to test the diversity of aspects involved in the development of the SInBot system. The initial test bed used for this research is a combination of two Lynxmotion AL5D robots and a Samsung SUR40 multi-touch environment. In this paper, novel work preparation concepts will be described and an experiment setup is proposed to validate the model for definition and generation of tasks from a CAD file.

Keywords: Intelligent robots; automated manufacturing; work preparation; high level task description; task allocation

1. Introduction

Current industrial robots (IRs) play an important role in performing repetitive production tasks. They perform their tasks cost effectively and accurate over longer periods of time. The European production industry is moving toward higher added value production that must be lean and flexible in order to survive in a competitive market [1]. IRs need to get an even higher intelligence, collaborative and multipurpose deployment and effortless transfer of processes by which one-off or limited series can be produced.

SInBot is a project that specifically targets machining tasks in the manufacturing environment. Where these tasks are now performed by million euro lathe and milling machines, SInBot sees an important role for IRs. The focus of project SInBot will be on the development of methods and tools for simplifying definition, selection and use of intelligent, decentralized cooperation of IRs for manufacturing purposes. One of the areas in which a true simplification step must still be achieved is in work preparation. When the words 'work preparation' are used in this paper, it is used to describe the process to generate manufacturing data for some hardware to create a product from its technical product data as efficient as possible.

Multiple sources mention the qualitatively measured long and costly work preparation for IRs. For example; manually programming a robotic arc welding system for the manufacture of a large vehicle hull takes more than eight months, while the cycle of the welding process itself is only 16h [2]. In this particular case, the programming time is approximately 360 times the execution time. Roughly, programming accuracy can be inversely proportional to programming time [3]. The time consuming work preparation, complexity and diversity of programming languages remain a problem for IRs to be implemented in small and medium sized manufacturing enterprises (SM/MEs). A common need for European SM/MEs is found in publications of various authors; many of them describe the need for flexible manufacturing, mass customization and a decrease in setup and programming times [4]. Many of those authors connect these needs with either large or medium to small manufacturing enterprises, and intelligent IRs as a possible solution. Another view on the problem is described as the need for intuitive teaching methods. However, this need actually abstractly describes the need for efficient work preparation for manufacturing lines containing IRs [5]. In project SInBot, work preparation is approached as being too complex, where possible solutions lay in task-level programming.

Taking automated milling centers as an example of SM/MEs, the work preparation is far more automated than Industrial Robot (IR) programming. Milling stations are expensive machines with relatively small work envelopes, but ensure high accuracy and the ability to be programmed offline for 100%. Since programming files can be stored for later use, and the machine adheres to the offline programming (OLP) almost entirely, subsequent reorders are available for relatively low additional costs.

To obtain the foreseen simplification in work preparation, a flexible robot-based machining cell has to be able to translate CAD files relatively fluently into manufacturing tasks. Until now, robots are controlled through low level programming languages, although several higher level programming languages that are built around PLC systems and allow humans to give commands on a relatively high level are starting to emerge. SInBot will use these high level languages as task description. As task descriptions become more abstract, specific control is lost. Algorithms that deal with high level tasks, should provide ample input for the robot to perform its task efficiently.

SInBot has identified positioning and trajectory accuracy as the main problem with IRs. Many publications have identified the same problem, see for example [6]. SInBot seeks this solution in two accuracy improving prospects:

- Based on the CAD model, the robot predicts its own deviations under load and machining forces
- A short sensor loop alters the movement of the IR by predicting where the machining head should be, and where it actually is.

2. Literature Review / Context

There are three programming systems in the field of robotics; guiding systems, robot-level programming, and task-level programming systems [7]. The most common industrially applied method is robot-level programming, followed by the guiding systems. An interesting perspective is seen as some try to achieve intelligent robots by providing it with initial knowledge about a task, teaching the task (virtually) and let the robot interpret what the actual task should be [8, 9]. The research field of humanoid robots has some potential overlap with project SInBot [10]. The field that is encountered mostly in that particular overlap is service-robotics.

2.1. Related Research Projects

There are multiple projects trying to solve the IR implementation in the small batch manufacturing paradigm. The SMErobot initiative wants to develop in the topics programming and task description [4]. Minhas et al [6] see development opportunities in reconfigurable robots, task description and accuracy improvement. The Fraunhofer Institute seeks their development in the reintroduction of the pendant teaching method, by coupling voice recognition to robot guidance teaching [5].

2.2. Robot Programming and Allocation Strategies

In Guidance methods, there are two means of programming a robot: lead-through programming and walk-through programming (Lead by the nose) [7]. Today's robots used in industry, rely on lead-through programming with a teach panel, next to OLP with relatively complex tools [11]. Multiple techniques exist based on imitation [12, 13].

Advanced OLP software packages exist, in which the entire cell, the robot and all the machines or instruments in the workspace are mapped graphically. The technique has limited value because it relies on accurate measurement of the positions of the associated equipment, and also relies on the positional accuracy of the robot which may or may not conform to what is programmed. In practical industrial applications, there are two main categories of robotic programming methods; online and OLP. OLP is growing in potential, but Pan et al [2] state that OLP software packages have a major drawback: the price of software. Both commercial and open-source software packages are becoming more powerful and usable. More affordable packages use positional commands as the main tool for programming robots. Based on positional commands, multiple text-based robot programming languages have been developed. Among them, five classic robot languages are commonly used: RAIL, AML, VAL, AL, and RPL [14]. Next to these languages, proprietary languages have been developed [15].

Approaches to task level programming are emerging rapidly. There are multiple general task describing languages which may be of interest to project SInBot, for example [16, 17]. Some of these are intended for service robot use, and incorporate human speech into their concepts, like Ke Jia, and TPL-R [18, 19]. Others incorporate learning from 'seeing', or use cognition enabled control, like KnowRob, CRAM, [20, 21]. Drawbacks of task level programming languages are their dependence on static environments [3].

Distributed Computing and regular task allocation puts a strain on the processing power of individual units. If tasks are allocated without a proper strategy, the system will saturate and become sluggish. Using task allocation models is a first step towards faster intelligent systems. From a business management perspective, three main task allocation strategies are identified; 1) ad hoc, 2) Dedicated Task Executer, and 3) Task Swarming [22]. Strategies can also be found in not-so-obvious research topics; such as resource scheduling of a computational grid [23]. For the Strategic Task Allocation Model (STAM), auction solutions can be useful. There are numerous examples, such as [24, 25], from concepts to concept improvements, and on single robots to robot cooperation.

2.3. Literature Conclusion

Literature research revealed an emerging need for intuitive teaching methods, improved adherence to OLP and improved intuitive OLP. Dependent on the specific situation, IRs are programmed, tested, reprogrammed, and tested until it performs its tasks perfectly. OLP still requires real-time testing and reprogramming to ensure accuracy in task performance. Teaching is still by far the easiest and most accessible method of programming a robot, but lacks accuracy and reprogrammability.

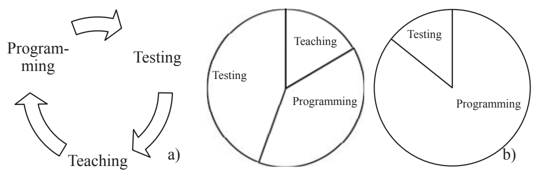

Fig. 1. (a) Programming iteration; (b) Relative time spent when programming a robot and CNC Milling machine respectively.

The software and hardware are far more advanced in automated CNC machine cells (A-CNC Cells), allowing a short iteration before production commences (Fig. 1 a). The IRs however, go through multiple iterations in an optimization process of task performance (Fig. 1 b).

Subsets of possible solution systems to the programming paradigm are easily found in literature. High level programming languages are useful, and where possible, used in the SInBot system. Similar to the CNC principles, SInBot will go towards the OLP direction, and improve on the existing projects by introducing a sensor network. This principle transforms the required approach, making it product oriented instead of process- or machine oriented. Auction systems are used in the Cost Model and, if required, in the Strategic Task Allocation Model. The manufacturing tasks can be created by using existing applications. Several strategic decision basics are found, which will find their implementation in the first test phases. The architecture will be based upon the automated CNC based machining cells, but incorporates the flexibility desired by the SM/MEs. Overall, several solutions of envisioned SInBot subsystems exist and will be incorporated in the concepts and models.

3. Concepts and Approach

As mentioned previously, SInBot is anticipating multiple theoretical models that will be the base of practical software modules in the SInBot system software framework. The development of the models is essential to show their individual and collaborative performance in test cases. In Fig. 2, the software framework is presented. The aim is to deliver a framework that encompasses multiple modules to aid the flexibility of IRs. The three main software layers are; 1) a processing layer, 2) a communication layer (DDS), and 3) an interface layer. The communication layer will be filled by a safety-certifiable data distribution service software package. The interface layer is the connecting layer to the framework, and contains interface modules, connected to the native software of the connected hardware components. The processing layer contains abstract modules that efficiently connect tasks to hardware; from cost models to strategic task allocation models. This architecture allows for the development of interface modules when new hardware is connected, and requires no adjustments to the modules in higher abstraction levels.

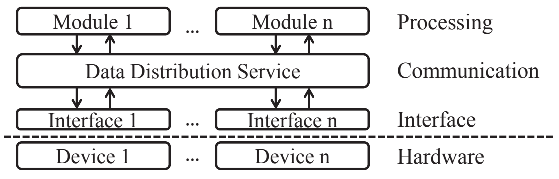

Fig. 2. SInBot Software Framework

The combination of an interface layer, communication layer, and distributed processing layer, has several advantages: 1) It allows multiple parties within project SInBot to develop models (and later; modules) individually, 2) modules can be tested separately, and 3) it allows for flexible implementation in which modules can be newly created or detached.

3.1. Task Creation Model

To obtain a more flexible manufacturing environment, the programming language (or manufacturing tasks) should no longer be hardware-specific. The programming language of robots should have an interface with higher placed software layers, in which communication exists on an higher (abstract) level. This higher level contains the task-oriented communication. Currently, OLP methods are machine-based, and rarely process-based. Automated CNC lathe and milling cells (A-CNC Cells) are located somewhere between process and product-based systems. The goal of SInBot is to move the OLP methods for IRs up the hierarchy towards product-based OLP methods (Fig. 3).

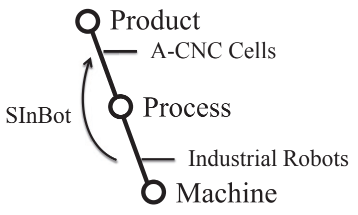

Fig. 3. Programming Orientation of IRs and A-CNC Cells

3.2. CAD to CAM

The SInBot system only functions if it receives specific manufacturing input. This input consists of information regarding incoming raw material or semi manufactured goods, and information regarding the required output. Currently, this is done primarily by the use of feature recognition, and manufacturing feature extraction. For the SInBot system modules to work properly, it needs input in the format of manufacturing tasks (Fig. 4). Dependent on the knowledge base, these tasks can be high level and low level.

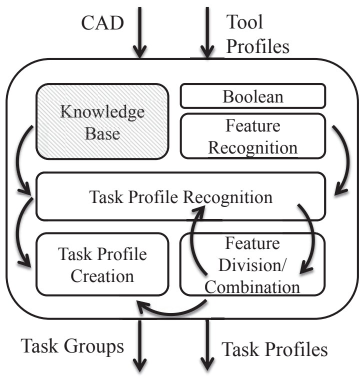

Fig. 4. CAD/CAM Model

By using this module, the SInBot system can perform tasks based on two Technical Product Data files; being the data of the raw material, and the data of the final product. This module is a precondition to automate work preparation to a high degree, giving the human operator the supervision task. SM/MEs with automated CNC lathe and milling cells work with software that is very similar to the envisioned CAD/CAM module. It is expected that an existing application can be adapted to provide the input for a proof of principle.

3.3. Capability model

In a flexible manufacturing environment, containing a finite amount of different IRs, and a finite amount of different manufacturing tasks, a potential is found to 'learn and repeat'. Manufacturing enterprises make their quotes on the basis of production planning. If they are unknown, the quote will be an estimate at best, or requires costly simulations. The flexibility in manufacturing products using the SInBot system should be kept, while introducing a steady forecast into this system. A model is build that supports the repeatability and estimation of production costs, while maintaining the flexibility that is gained from manufacturing with high level control (Fig. 5). The capability model has two main goals, being; the possibility to produce cost estimates on production without complex calculations or simulations, and freeing the cost/auction model for relatively unknown manufacturing tasks that require the computing power. Another way of putting the latter goal, is to emphasise that the capability model is a way to reduce the time between order scheduling and order execution.

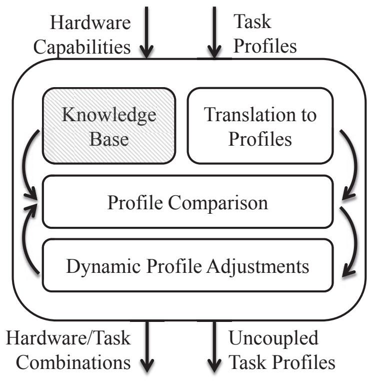

Fig. 5. Capability Model

Any piece of hardware has capabilities, such as strength, maximum speed, and accuracy. Combining pieces of hardware, such as a milling tool and a specific robot, creates a new set of capabilities, named a profile. If the current profile of a hardware combination is not required, the hardware, or assets, can be combined with other assets to create a profile that is. Practically, this could be a tool change, or an accuracy-improving software setting. Task profiles are generated by a CAD-to-CAM module, since the properties of such a task are what is needed to combine task to hardware profiles.

The incoming capabilities consist of classes of two categories: assets (or means) and goals. Goals are reachable with specific assets, while one specific set of assets unavoidably will be the most efficient to use. For SInBot, the assets are -for instance- robots and tools, and the goals are products (CAD) and tasks. For each specific object of an asset, different parameters may exist. A milling tool is an asset with the property 'flutes', but a robot asset obviously does not have a property 'flutes'. Properties are labelled high, medium or low impact, to facilitate mandatory task allocations and speed up profile combination.

The knowledge base is, as explained, dependent on the nature of the manufacturing strategy of the enterprise or department. For a high degree of control and/or secrecy, the knowledge base is kept as an internal library, with the content being created by the cost/auction model. Once certain hardware profiles are connected to specific task profiles, these profile combinations are stored and rated. When the manufacturing strategy is focused on throughput and accurate cost estimations, or the cost/auction model is to be avoided, the knowledge base could very well be a community in which multiple enterprises and research institutes drop data from profile combinations.

The profile comparison uses two types of comparison algorithms; the first locating profile combination matches between incoming profiles and knowledge base profiles, the second being mandatory task allocations. The tasks that are allocated are mandatory allocated when a task characteristic is of high impact. These mandatory allocations create a shortcut through the SInBot system software framework, bypassing the cost/auction module. The output of this model is either a robot/tool/task combination going directly to task allocation and scheduling, or tasks coupled with subsets of hardware, going to the cost/auction model to be decided upon.

Two clear deficiencies are found to be problematic and should be solved in further development; 1) exponential growth of profile storage generating a complex and costly process, and 2) tunnel vision due to the rigid nature of profile storage. A potential solution to both problems is found in using a system that continuously updates profile combinations and deletes 'obsolete' profiles in the manufacturing environment library.

3.4. Strategic Task Allocation Model

Early in the project, a need was unearthed to separate operational and strategic control. While the operational control can be fully automated, some influence on strategic level can be of great importance to manufacturing enterprises. Functionalities can contain 'evoke actions', limit means, task allocation based on a specific strategy, operate on the boundaries of calculations, divert task groups, etc. The reason and scope of the operational functionalities decide what the input should be, and what the output will be. This model will therefore be developed when the SInBot System is more mature.

3.5. Development/Testing Hardware Platform

To proof the validity of the software modules and the models that they are based on, an experimental setup of robots will perform several tasks using individual and combined modules. Since safety and costs dictate that actual IRs are not available for doing the experiments, the decision was made to use small scale articulated robots. The Lynxmotion AL5D is a small 5 DOF articulated robot [26]. It is ideal for educational and experimental purposes, but lacks accuracy, payload and reach for any industrial purpose (Fig. 6).

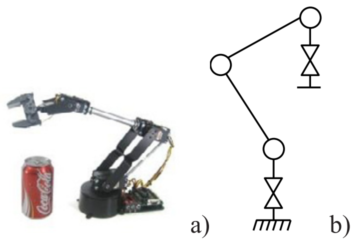

Fig. 6. (a) AL5D Robot arm; (b) AL5D Schematic

The final SInBot system contains sensor networks for the robots to be aware of their environment, trace the exact position of their end-effector, and find the exact location of the task at hand. However, until the final system is implemented, proof of principles have to operate without the sensor networks. In this experiment, the decision was made to emulate the sensor environment, and mimic the output from such a network. Since safety and costs dictated the robots to be small scale, the sensor environment can be relatively small too. The Samsung SUR40 [27] multi-touch table is an ideal sensor environment for this purpose.

3.6. Approach

The models will be developed to have a system that reacts to high level task descriptions. The first model to be developed into a software module and tested is therefore the Task Creation model. The capability model and strategic task allocation model are developed synchronously, while the cost model is emulated until completed. Both can be implemented in the experiment to test their influence on task performance. Connecting the cost model and testing the influence on dynamic locations of robots are considered the last steps in this experiment, and are also the input for further research and experimentation.

Currently, the dual AL5D and Samsung SUR40 test-bed is running, and can perform the a limited amount of high level 'pick and place' tasks based on sensory information. Among the short term goals for this test-bed are; to extend the understanding of high level tasks, and to optimize the information flow and processing in order to increase positional accuracy.

4. Conclusions

In theory, the presented models and envisioned software framework will enable SInBot to establish the basis of flexible task performance of IRs. The experiment setup enables a user to play out different scenarios and provide the robots with a variety of tasks, while introducing different errors. The model behavior can be tested and easily adapted because of the modular design of the different control algorithms. However, since the sensor network is emulated, testing will undoubtedly reveal that some errors cannot be resolved by the characteristics inherited by sensor emulation. The AL5D articulated robot arm is inaccurate and all errors and variables introduced that are related to arm accuracy cannot be resolved or viably tested until the system is ready for testing with a more accurate robot arm.

A next step is to install the robot arms on individual mobile platforms, swapping the Samsung SUR40 for overhead cameras. This transfer to mobile robots will yield important information regarding the SInBot system's responses to mobile robots.

References

[1] Intereg, SInBot. [cited Access 10-10 - 2011]; Available from: http://www.smartbot.eu/project-information/sinbot/.
[2] Pan, Z., et al., Recent progress on programming methods for industrial robots. Robotics and Computer-Integrated Manufacturing 2012; 28(2): 87-94
[3] Johansson, R., et al., Sensor integration in task-level programming and industrial robotic task execution control. Industrial Robot 2004; 31(3): 13
[4] Nilsson, K., et al., Productive robots and the SMErobot project. Third Swedish Workshop on Autonomous Robotics; 2005
[5] Schraft, R.D., The need for intuitive teaching method for small and medium enterprises. Proceedings of International Symposium on Robotics. Munich; 2006
[6] Minhas, S.H., et al., Reconfigurable strategies for manufacturing setups to confront mass customization challenges. 21st International Conference on Production Research (ICPR21). Stuttgart; 2011
[7] Biggs, G. and B. Macdonald, A survey of robot programming systems. in Proceedings of the Australasian Conference on Robotics and Automation (CSIRO). Brisbane, Australia; 2003
[8] Aleotti, J., S. Caselli, and M. Reggiani, Leveraging on a virtual environment for robot programming by demonstration. Robotics and Autonomous Systems 2004; 47(2–3): 153-161
[9] Chen, J.R., Constructing Task-Level Assembly Strategies in Robot Programming by Demonstration. The International Journal of Robotics Research 2005; 24(12): 1073-1085
[10] Calinon, S., F. Guenter, and A. Billard, On Learning, Representing, and Generalizing a Task in a Humanoid Robot. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 2007; 37(2): 286-298
[11] OSHA Technical Manual (OTM) - Section IV. O.o.S.a.T. Assessment, 01-00-015, 1999
[12] Nehaniv, C., H. Ab, and K. Dautenhahn, Of Hummingbirds And Helicopters: An Algebraic Framework For Interdisciplinary Studies Of Imitation And Its Applications. World Scientific Press, citeulike-article-id:4186342, 1999
[13] Eppner, C., et al., Imitation learning with generalized task descriptions. IEEE International Conference on Robotics and Automation. 2009
[14] Robot Programming Languages. [cited Access 10-11 - 2011]; Available from: http://www.roboticsbible.com/robot-programming-languages.html.
[15] Mühe, H., et al., On reverse-engineering the KUKA Robot Language. Unpublished; 2012
[16] Malosio, M., et al., High-Level Robot Programming in a PC-Based Control Environment. 8th IEEE International Conference on Intelligent Engineering Systems. Cluj-Napoca, Romania; 2004
[17] Jensen, B., et al., The interactive autonomous mobile system RoboX. IEEE/RSJ International Conference on Intelligent Robots and Systems. 2002
[18] Chen, X., et al., Developing high-level cognitive functions for service robots. Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems. Toronto, Canada; 2010
[19] Kim, R., et al., TPL-R: A High-Level Task Programming Language for Robots. Unpublished; 2009
[20] Tenorth, M., D. Jain, and M. Beetz, Knowledge Processing for Cognitive Robots. KI - Künstliche Intelligenz 2010; 24(3): 233-240
[21] Beetz, M., et al., CRAM - A Cognitive Robot Abstract Machine for everyday manipulation in human environments. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2010
[22] Williams, R., Three Strategies for Task Allocation. [cited Access - 2009]; Available from: http://agile.techwell.com/articles/weekly/three-strategies-task-allocation.
[23] Chandak, A.V., B. Sahoo, and A.K. Turuk, Heuristic Task Allocation Strategies for Computational Grid. Int. J. Advanced Networking and Applications 2011; 2(5): 7
[24] Dahl, T.S., M. Matarić, and G.S. Sukhatme, Multi-robot task allocation through vacancy chain scheduling. Robotics and Autonomous Systems 2009; 57(6–7): 674-687
[25] Tovey, C., et al., The Generation of Bidding Rules for Auction-Based Robot Coordination. Springer Netherlands, L. Parker, F. Schneider, and A. Schultz, 10.1007/1-4020-3389-3_1, 978-1-4020-3388-9; 2005
[26] Lynxmotion, AL5D. [cited Access 28-08 -; Available from: http://www.lynxmotion.com/c-130-al5d.aspx.
[27] SUR40 for Microsoft (r) Surface (r): User Manual. Samsung Electronics, 2011