Information requirements and interfaces for the programming of robots in flexible manufacturing

Arturo Bastidas-Cruz*, Oliver Heimann*, Kevin Haninger, and Jörg Krüger

Fraunhofer Institute for Production Systems and Design Technology (IPK)

Division of Automation Technology, Berlin, Germany

`firstname.lastname@ipk.fraunhofer.de`

* Equal contribution

Abstract. Flexible automation limits what information is known during robot commissioning, requiring new robot programming methodologies. To adapt robot behavior to product variation, an operator can supply missing information, controlling predefined robot behaviors via a user interface. This operator interface can abstract domain expertise (in robotic programming, assembly planning), allowing the efficient specification of changes to a robot program by a wide range of potential users. Towards designing such interfaces, this paper analyzes the requirements of flexible manufacturing, the required changes to the robot program, and the information needed to make those changes. Two user interfaces are presented, a drag and drop and a gesture interface, implemented on a robotic flexible manufacturing testbed.

Keywords: flexible manufacturing, robot programming, human-robot interaction, usability

1 Introduction

In standard industrial robot commissioning, as common in large-scale assembly lines, experts produce a fully-defined sequence of robot commands for a specific task during commissioning. Although providing higher quality and throughput, traditional automation reduces production agility with limited ability to accommodate product variation or new products without recommissioning. For small- and medium-sized companies in saturated markets with a faster product cycle, this is a major barrier. In comparison, on manual assembly lines new products can be quickly deployed (provided no hardware changes are required) and product variation is tolerated with marginal increases in cost or deployment time. Though difficult for standard automation, flexible manufacturing can address many growing business needs: individualized products, smaller lot sizes, shorter innovation cycles, and shorter production times. These capabilities support onshoring; local production at competitive costs. Flexible manufacturing can involve product variation, where the product varies between task iterations, or production changeover, where the assembly line is reconfigured for a new product. The manufacture of multiple products on an automated line can be

solved centrally, through methods like a production management system, which indexes all products and adjusts a cell as necessary. Decentralized flexibility is also possible, through the design of adaptive cells which detect and respond to product variation (e.g. with vision systems, or identifying tags). However, these approaches are complex and do not reduce the effort to introduce a new product. A promising approach to flexibility is human robot collaboration (HRC), where human operators can respond to product changes, adapting pre-programmed robot behaviors as appropriate [7]. Designing the robot such that an operator can later complete a task requires consideration of what decisions the human must make, how those decisions are communicated to the robot, and the contextual information required to make those decisions. Programming for flexible automation, moreso than traditional automation, occurs over time, where a variety of users contribute to the final sequence of robot commands. From the user perspective, the interface must include visibility of system status, understandable information presentation, a match between the system and real world, and help users recognize and recover from errors [2], [6]. Several modes of human-robot communication for modifying a robot program have been developed, including speech [9] and gestures [10], [11], [13]. However, more intuitive and compact modalities of communication can have ambiguity in the interpretation of the input. Therefore, the robot and human must have a common understanding of the environment to correctly ground an exchanged symbol (see [14] or [12]). This is a usability requirement for the design of intuitive human machine interfaces.

Human-robot communication can also be achieved through a graphical interface on a touchscreen in the workspace. Such graphical environments can allow for the simultaneous presentation of more complex information, allowing a user to see and adjust the parameters and program flow of a robot. This interaction can be further improved through the use of augmented reality (AR). AR allows GUIs to present information as an overlay to the actual environment, which enables a strong visual connection between parameters and their real world influence. Lambrecht [10] et al have shown that tablet based augmented reality can significantly reduce the programming time for free space programs. Guhl [4] et al as well as Ong [5] et al have further improved on the approach through a headworn AR device allowing for comprehensive virtual interaction with the robot program.

This paper explores HRC in flexible manufacturing, examining application requirements and usability perspectives. Programming for flexible manufacturing is presented as the partitioning of expertise and process information within a prior robot program, where the information required is reduced and easily supplied. While other GUI development considers defining a robot task from lower-level commands [3], the GUIs discussed here are targeting flexible manufacturing, where a subset of decisions or changes must be made via the GUI. Two interfaces for the flexible programming of a robot cell are presented, a gesture-based system for workspace communication, and a touchscreen-based graphical interface with predefined actions.

2 Flexible Manufacturing

What new design requirements does flexible manufacturing introduce? Consider a tier two supplier for the car manufacturing industry, where different subassemblies must be produced for just-in-time orders of tier one suppliers. The ordered variants will vary between the different customers and on a day-to-day basis, requiring fast change over times. Furthermore, some variants might only ever be produced a hundred times, rendering the programming of a robotic cell for all possible production unviable. Flexible manufacturing can also encompass artisan production, where highly individualized products are manufactured and robot assistants can be directed in response to material variation, artistic expression, or customer individualization. Table 1 presents different aspects of manufacturing which may vary in flexible production with examples.

Table 1: Flexible manufacturing examples, which may require on-/offline changes to the robot program

Variation	Examples	Offline	Online
Product Geometry	Screw locations	x	
Workspace Geometry	Material pickup location	x	
Inclusion/Omission of Operation	Optional subassembly	x	x
Change in Operation Ordering	Production process change	x	x
Artisan Process	Glue application location		x

There are two major types of variation considered here: geometric and plan. In geometric variation, the location(s) which determine a specific action may vary between cell cycles or product change-over. In plan variation, a change in the set or order of robot behaviors must be made. Other changes to the robot program may be necessary (dwell time, contact force, etc), but are typically process parameters which will require more expertise and are outside the scope of this paper.

3 Decision making

Product variation, as introduced in the last section, requires corresponding changes to the robot program. This section overviews available and necessary information to make those changes at the various stages of the flexible programming.

3.1 Information

Information and expertise is required to translate a product design into an appropriate set of robot instructions. As seen in Table 1 the product model changes

least frequently; it is a result of the design process and therefore is considered available as input to the assembly level planning. The assembly planning requires a very high level of expertise to translate a product design into a sequence of manufacturing processes in order to assemble the product. It involves knowledge about machine availability and resource planning. The assembly planning problem is well supported by tools (computer-aided manufacturing), more easily dealing with high-mix, low-volume production compared to lower-level planning layers (e.g. robot programming).

On the basis of a product-specific assembly plan, a corresponding robot task order - the sequence of robot actions - must be produced. Where the assembly plan includes general operation orders, the robot task order must specify the product-specific robot actions which achieve this result. With increasing variants and smaller production lots, this results in an exponential growth in required decisions, and tighter time constraints on every individual decision.

Therefore, a programming system for flexible manufacturing on the task level requires the ability to rearrange the order of processes or even substitute different processes in an efficient manner. An example is the production of gear boxes: while the overall process of interlocking gears, fitting shafts, bearings and seals is comparable between products, materials, sizes or even the number of overall parts might vary. Here, the operator needs to adapt the overall flow of control of a prior program. Furthermore, at the task level, the operator might specify specific process parameters. These include feed velocity, maximum torque for a screw or the slope of a thread to be cut. In order to make these decisions the operator needs expert knowledge regarding the machining processes involved as well as the capability to translate product specifications into the required processing parameters.

Decision speed becomes even more important on the individual action level as again, per task instance, multiple actions must be specified. Here craftsmanship is required to determine the correct instantiation of an action on a per product basis. As there may be no purpose made fixtures or computer vision systems for the individual production lot, the human operator needs to supply this information. Where the operation order is programmed in advance, the artisan process requires information in situ. The frequency of online decisions requires efficient in-situ programming to avoid a bottle neck in the manufacturing processes.

Figure 1 gives an overview on the different levels of programming and the corresponding informational requirements. It can be seen that the frequency of a decision increases as it comes closer to program execution. At the same time, the breadth of information required becomes narrower. This increasing specialization of the information provided helps to address the need for fast decisions.

While at the task planning level, an interface needs to provide means to program the entire process. This results in the necessity for the user to navigate a more complex interface. However, on the process level, the interface might just offer a very limited set of options specifically tailored to the individual process, allowing very fast navigation and user interaction.

Fig. 1: The various levels on which a flexible manufacturing task can be programmed, where a prior program can be specialized, with discrete or continuous inputs. The last two stages (task order and action list) are static in standard automation, but here can be adjusted offline or online.

3.2 Interface Usability

The prior section has shown that the frequency of required decision making varies with the level of detail of the process specification. For frequent shopfloor decisions, a user can make adjustments to the robot execution via an HRC interface. The means by which a user provides this information to the robotic system affects the ability to display contextual information, ergonomics, learnability, and flexibility [1]. HRC is very much defined by its context, requiring a common grounding to the symbols exchanged between robot and human.

The interface must provide feasibility, the ability to adjust the robot behavior as necessary for the appropriate task, and safety, such that for any supplied input of the operator the robot maintains safety. On top of the functional requirements, the interface should support usability [2], [6] in terms of:

Visibility: Information required to make a decision is readily available, including relevant states of the robot.

Predictability: The effects of an input are intuitive and repeatable, so that the user can choose the input required to achieve the desired robot behavior.

Efficiency: The necessary action can be quickly and ergonomically given to the robot. For online decisions, this should not significantly disrupt the operator's workflow.

4 Implementation

Fig. 2: Integrated flexible manufacturing demonstrator

To investigate feasibility and usability of various types of HRC for flexible manufacturing, the testbed seen in Figure 2 is built. The cobot used is a Universal Robot UR3. Above the table 2 RGB cameras and a projector are mounted. The current use case is the automatic placement of workpieces from a storage position onto the table followed by a simulated welding operation. This testbed can test variation in both geometric and plan information, where decisions must be made by an operator online or offline. Two interfaces are presented, which differ in how they present the available decisions and contextual information to the user, and how the user supplies their decision to the robot.

4.1 Graphical Programming

The graphical task-oriented robot programming system presented in this work was developed by researchers from the Process Automation and Robotics department of the Fraunhofer IPK institute in order to allow visitors without any programming experience to interact with the cobot, so that the user can make modifications to the robot program and execute an assembly task. The system is based on the task-oriented RP-i3 framework developed at Fraunhofer IPK [8] and aims to reach easy accessibility of complex systems by workers in industry. The task-level framework offers a visual structured interface of the assembly

process that allows an easy implementation of the two major types of variation discussed in Section 2: an easy rearrangement of the order of processes related to a specific assembly task (plan) as well as the easy specification or modification of specific action parameters (including geometric variations).

The user selects tasks that the robot can perform on the work pieces, where tasks and work pieces are offered as ready-to-use blocks. The blocks are combined to a program by dragging and attaching them to each other. New robot programs and new work pieces can be defined in that way. The graphical programming environment SCRATCH [17] visualizes the assembly environment abstraction of the RP-I3 framework as seen in Figure 3b.

A robot programming language based on C++ implements the RP-I3 abstraction of tasks and workpieces. A server that works as an interface receives the content of the GUI, organize the information in a data base and execute the programs using this information. In order to send the command to the robot, an external interface is used. The setup consist of a Cobot equipped with a two-finger gripper. The RP-i3 framework is located on a Linux-based PC. The visualization of the RP-i3 framework is based on a version of Scratch that was modified in order to allow the implementation of robot-based assembly tasks.

This interface offers a higher information-density, allowing the presentation of a complete robot plan which helps a user predict robot behavior. Although the actions and workpieces are abstracted from the physical workspace, graphics seen in Figure 3b can improve the operator's understanding. However, this interface does not as easily allow online monitoring or decisions, as the screen must be mounted to not obstruct the workspace.

Fig. 3: Programming of an assembly task using the graphical interface RP-i3 and universal robot UR3

4.2 Gesture based Programming using Augmented Reality

The programming interface for the action level aims to provide very fast and precise user input for a very limited breadth of decisions that allows in detail specification of the action at hand. Where the RP-i3 GUI enables the specification of the more abstract task and its parameters (e.g. weld together part A and B with feed velocity of 20mm/s), the gesture based interface allows to specify artisan-level decisions (e.g. exact position and length of a seam). This information is complex (therefore time consuming and expensive) to describe in advance at the task level but very easy to specify once the product is available in the manufacturing cell.

The interface allows machine operators to communicate these decisions in a very fast way through direct interaction with the program on the workpiece using augmented reality [15]. Geometric information (planar locations) can be communicated, as well as discrete decisions via virtual buttons in the workspace, allowing the omission of other inputs device such as a touch panel. Furthermore, the direct projection of paths onto the workpiece provides the user with understanding of the robot state and plan.

Fig. 4: Augmented Reality interface for gesture based programming. The user specifies the details of a given action through pointing gestures on the workpieces (left). The robot program is automatically generated based on the action specification (right).

To achieve the precision required for programming the artisan process, the system relies on 3D computer vision (stereo vision based on 2 RGB cameras with a pixel resolution of $35\mu\text{m}/\text{px}$). In an intermediate step the system then extracts relevant geometric features such as edges and planes. These are subsequently fed into an inference engine which determines probable process affordances[16] (areas of interest for a given process) based on pre-defined rules for the according

process. Using the detected affordances, the user interaction is elevated to a level of mutual understanding. The system presents possible actions to the user who then solves ambiguities and further details the action through pointing.

Figure 4 shows the programming process for a welding process. The two work pieces are placed on the table. The system automatically detects the common edge between the objects (as it presents seam affordance) and highlights it using the projection system. The user then specifies part of the edge for welding using his finger. The system combines the knowledge about the process affordance with the gesture input to compute a precise robot program (limited by the accuracy of the 3D object detection algorithm rather than the gesture tracking).

5 Conclusion

In this paper we introduced the application requirements of flexible manufacturing, corresponding programming requirements, and how those requirements can be met with two different user interfaces. Programming for flexible manufacturing introduces challenges as it requires the engagement of a number of stakeholders, with varying expertise and responsibilities. Two user interfaces are presented and analyzed according to their ability to address the various requirements of flexible manufacturing.

In future work, these two interfaces can be combined in new ways to offer new programming modalities - e.g. allowing re-programming of an action through gesture demonstrations. The unified testbed allows for future usability experiments which can quantitatively motivate best-practices for programming in flexible manufacturing.

References

1. Shackel, Brian and Simon J. Richardson, eds. *Human factors for informatics usability*. Cambridge university press, 1991.
2. Clarkson, Edward, and Ronald C. Arkin. "Applying Heuristic Evaluation to Human-Robot Interaction Systems." *Flairs Conference*. 2007.
3. Kraft, Martin, and Markus Rickert. "How to teach your robot in 5 minutes: Applying UX paradigms to human-robot-interaction." *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2017.
4. J. Guhl, S. Tung and J. Kruger, "Concept and architecture for programming industrial robots using augmented reality with mobile devices like microsoft HoloLens" *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Limassol, 2017, pp. 1-4.
5. S.K. Ong, A.W.W. Yew, N.K. Thanigaivel, A.Y.C. Nee "Augmented reality-assisted robot programming system for industrial applications, *Robotics and Computer-Integrated Manufacturing*, Volume 61, 2020, 101820, ISSN 0736-5845, <https://doi.org/10.1016/j.rcim.2019.101820>.
6. Yanco, Holly A., Jill L. Drury, and Jean Scholtz. "Beyond usability evaluation: Analysis of human-robot interaction at a major robotics competition." *Human-Computer Interaction* 19.1 (2004): 117-149.

7. Tsarouchi, P., Makris, S. and Chrysosolouris, G., 2016. Human–robot interaction review and challenges on task planning and programming. *International Journal of Computer Integrated Manufacturing*, 29(8), pp.916–931.
8. Surdilovic, D., Konz, B., Radke, M., 2019. Scratch robot programs - a task oriented RP-i3 framework for assembly programming. Under Review.
9. Forbes, Maxwell and Rao, Rajesh P. N. and Zettlemoyer, Luke and Cakmak, Maya (2015), Robot Programming by Demonstration with situated spatial language understanding. 2015 IEEE International Conference on Robotics and Automation (ICRA) (pp. 2014–2020) <https://doi.org/10.1109/ICRA.2015.7139462>
10. Lambrecht, J., Krüger, J. (2014), Spatial Programming for Industrial Robots: Efficient, Effective and User-Optimised through Natural Communication and Augmented Reality. *Advanced Materials Research*. Vol. 1018. Trans Tech Publications
11. Rudorfer, M., Guhl, J., Hoffmann, P., Krüger, J. (2018) Holo Pick'n'Place. IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA) (pp. 1219–1222.), <https://doi.org/10.1109/ETFA.2018.8502527>
12. Profanter, Stefan, Perzylo, Alexander, Somani, Nikhil, Rickert, Markus, Knoll, Alois (2015), Analysis and semantic modeling of modality preferences in industrial human-robot interaction. 2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD) (pp 1812–1818), <https://doi.org/10.1109/IROS.2015.7353613>
13. Tsarouchi, P., Athanasatos, A., Makris, S., Chatzigeorgiou, X., Chrysosolouris, G. (2016) High Level Robot Programming Using Body and Hand Gestures. *Procedia CIRP* vol. 55 (pp. 1–5)
14. Proctor, Frederick M., van der Hoorn, Gijs, Lipman, Robert (2016), Automating Robot Planning Using Product and Manufacturing Information. 14th CIRP CAT 2016 - CIRP Conference on Computer Aided Tolerancing (pp 208–213), <https://doi.org/10.1016/j.procir.2016.02.139>
15. Heimann, Oliver, Hüggle, Johannes, Krüger, Jörg (2017) Gesture based robot programming using process knowledge - An example for welding applications. 2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation(ETFA), <https://doi.org/10.1109/ETFA.2017.8247761>
16. Heimann, Oliver and Krüger, Jörg, Affordance Based Approach to Automatic Program Generation for Industrial Robots in Manufacturing, *Procedia CIRP* vol. 76 (pp. 133–137), Elsevier 2018
17. Resnick, Mitchel; Silverman, Brian; Kafai, Yasmin; Maloney, John; Monroy-Hernández, Andrés; Rusk, Natalie et al. (2009): Scratch. In *Commun. ACM* 52 (11), p. 60, <https://doi.org/10.1145/1592761.1592779>.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.