Mälardalen University Press Dissertations
No. 166

PLANNING AND SEQUENCING THROUGH MULTIMODAL INTERACTION FOR ROBOT PROGRAMMING

Batu Akan

2014

School of Innovation, Design and Engineering

ISBN 978-91-7485-175-5
ISSN 1651-4238
Printed by Arkitektkopia, Västerås, Sweden

Mälardalen University Press Dissertations
No. 166

PLANNING AND SEQUENCING THROUGH MULTIMODAL INTERACTION FOR ROBOT PROGRAMMING

Batu Akan

Akademisk avhandling

som för avläggande av teknologie doktorsexamen i datavetenskap vid Akademin för innovation, design och teknik kommer att offentligen försvaras måndagen den 8 december 2014, 09.15 i Gamma, Mälardalens högskola, Västerås.

Fakultetsopponent: Professor Bengt Lennartson, Chalmers University of Technology

Akademin för innovation, design och teknik

Abstract

Over the past few decades the use of industrial robots has increased the efficiency as well as the competitiveness of several sectors. Despite this fact, in many cases robot automation investments are considered to be technically challenging. In addition, for most small and medium-sized enterprises (SMEs) this process is associated with high costs. Due to their continuously changing product lines, reprogramming costs are likely to exceed installation costs by a large margin. Furthermore, traditional programming methods of industrial robots are too complex for most technicians or manufacturing engineers, and thus assistance from a robot programming expert is often needed. The hypothesis is that in order to make the use of industrial robots more common within the SME sector, the robots should be reprogrammable by technicians or manufacturing engineers rather than robot programming experts. In this thesis, a novel system for task-level programming is proposed. The user interacts with an industrial robot by giving instructions in a structured natural language and by selecting objects through an augmented reality interface. The proposed system consists of two parts: (i) a multimodal framework that provides a natural language interface for the user to interact in which the framework performs modality fusion and semantic analysis, (ii) a symbolic planner, POPStar, to create a time-efficient plan based on the user's instructions. The ultimate goal of this work in this thesis is to bring robot programming to a stage where it is as easy as working together with a colleague.This thesis mainly addresses two issues. The first issue is a general framework for designing and developing multimodal interfaces. The general framework proposed in this thesis is designed to perform natural language understanding, multimodal integration and semantic analysis with an incremental pipeline. The framework also includes a novel multimodal grammar language, which is used for multimodal presentation and semantic meaning generation. Such a framework helps us to make interaction with a robot easier and more natural. The proposed language architecture makes it possible to manipulate, pick or place objects in a scene through high-level commands. Interaction with simple voice commands and gestures enables the manufacturing engineer to focus on the task itself, rather than the programming issues of the robot. The second issue addressed is due to inherent characteristics of communication with the use of natural language; instructions given by a user are often vague and may require other actions to be taken before the conditions for applying the user's instructions are met. In order to solve this problem a symbolic planner, POPStar, based on a partial order planner (POP) is proposed. The system takes landmarks extracted from user instructions as input, and creates a sequence of actions to operate the robotic cell with minimal makespan. The proposed planner takes advantage of the partial order capabilities of POP to execute actions in parallel and employs a best-first search algorithm to seek the series of actions that lead to a minimal makespan. The proposed planner can also handle robots with multiple grippers, parallel machines as well as scheduling for multiple product types.

ISBN 978-91-7485-175-5
ISSN 1651-4238

Abstract

Over the past few decades the use of industrial robots has increased the efficiency as well as the competitiveness of several sectors. Despite this fact, in many cases robot automation investments are considered to be technically challenging. In addition, for most small and medium-sized enterprises (SMEs) this process is associated with high costs. Due to their continuously changing product lines, reprogramming costs are likely to exceed installation costs by a large margin. Furthermore, traditional programming methods of industrial robots are too complex for most technicians or manufacturing engineers, and thus assistance from a robot programming expert is often needed. The hypothesis is that in order to make the use of industrial robots more common within the SME sector, the robots should be reprogrammable by technicians or manufacturing engineers rather than robot programming experts.

In this thesis, a novel system for task-level programming is proposed. The user interacts with an industrial robot by giving instructions in a structured natural language and by selecting objects through an augmented reality interface. The proposed system consists of two parts: (i) a multimodal framework that provides a natural language interface for the user to interact in which the framework performs modality fusion and semantic analysis, (ii) a symbolic planner, POPStar, to create a time-efficient plan based on the user's instructions. The ultimate goal of this work in this thesis is to bring robot programming to a stage where it is as easy as working together with a colleague.

This thesis mainly addresses two issues. The first issue is a general framework for designing and developing multimodal interfaces. The general framework proposed in this thesis is designed to perform natural language understanding, multimodal integration and semantic analysis with an incremental pipeline. The framework also includes a novel multimodal grammar language, which is used for multimodal presentation and semantic meaning generation.

Such a framework helps us to make interaction with a robot easier and more natural. The proposed language architecture makes it possible to manipulate, pick or place objects in a scene through high-level commands. Interaction with simple voice commands and gestures enables the manufacturing engineer to focus on the task itself, rather than the programming issues of the robot.

The second issue addressed is due to inherent characteristics of communication with the use of natural language; instructions given by a user are often vague and may require other actions to be taken before the conditions for applying the user's instructions are met. In order to solve this problem a symbolic planner, POPStar, based on a partial order planner (POP) is proposed. The system takes landmarks extracted from user instructions as input, and creates a sequence of actions to operate the robotic cell with minimal makespan. The proposed planner takes advantage of the partial order capabilities of POP to execute actions in parallel and employs a best-first search algorithm to seek the series of actions that lead to a minimal makespan. The proposed planner can also handle robots with multiple grippers, parallel machines as well as scheduling for multiple product types.

Sammanfattning

De senaste decenniernas användning av industrirobotar har ökat effektiviteten och konkurrenskraften i flera sektorer. Trots detta faktum, anses i många fall investeringar i robotautomation vara tekniskt utmanande. Dessutom är denna process, för de flesta små och medelstora företag (SMF), förknippad med höga kostnader. På grund av företagens ständigt föränderliga produktlinjer kommer kostnaderna för omprogrammering sannolikt att överstiga installationskostnaderna med stor marginal. Det är också känt att traditionella programmeringsmetoder anses vara för komplexa för användare av dessa system, m.a.o. tekniker eller tillverkningsingenjörer. Hypotesen är den att för att göra industrirobotar vanligare inom SMF-sektorn, bör robotarna kunna omprogrammeras av tekniker eller tillverkningsingenjörer snarare än robotprogrammeringsexperter.

I denna avhandling föreslås ett nytt system som bygger på task-nivå programmering. Användaren interagerar med en industrirobot genom att ge instruktioner med ett strukturerat naturligt språk samt välja objekt genom ett augmented reality gränssnitt. Det föreslagna systemet består av två delar: (i) ett multimodalt ramverk som även innehåller ett naturligt språk gränssnitt för användaren att interagera i samt utföra fusion av olika modaliteter och semantisk analys, (ii) en symbolisk planeringsalgoritm, POPStar, för att skapa en tidseffektiv plan utifrån användarens instruktioner. Det främsta målet med denna avhandling är att föra robotprogrammering till ett stadium där det är lika enkelt att arbeta tillsammans med roboten som med en kollega.

Denna avhandling adresserar två frågor. Den första handlar om utveckling av ett ramverk för att designa och utveckla multimodala gränssnitt. Det generella ramverket som föreslås i denna avhandling är utformad för att utföra förståelse av naturligt språk, multimodal integration och semantisk analys med en inkrementell pipeline. Den inkluderar även ett nytt multimodalt språk som används för multimodal representation av information och generering

av semantiskt korrekta meningar. Det multimodala ramverket hjälper till att göra interaktionen med industriroboten enklare och mer naturlig. Den föreslagna språkarkitekturen gör det möjligt att manipulera, plocka upp eller placera föremål i en scen genom högnivåkommandon. Interaktion med enkla röstkommandon och gester gör att tekniker eller tillverkningsingenjörer kan fokusera på själva uppgiften, snarare än frågor kring programmering av industriroboten.

Den andra frågan som adresseras bygger på de inneboende egenskaperna hos kommunikation som sker genom naturligt språk; instruktionerna från användare är ofta vaga och kan kräva andra åtgärder som bör vidtas innan villkoren för tillämpning av användarens instruktioner uppfylls. För att lösa detta problem föreslås en symbolisk planerare, POPStar, som baseras på partial order planner (POP). Systemet tar landmärken som extraheras från det som användares säger, eller gestikulerar, som indata. Därefter skapas en sekvens av en plan för att styra robotcellen med minimal makespan. Den föreslagna planeringsalgoritmen utnyttjar POP:s förmåga att hantera partiella planer för att jobba parallellt och agerar som ett bäst-första sökalgoritm för att söka bland sekvenser som leder till en minimal makespan. Planeringsalgoritmen kan också hantera robotar med flera gripdon, celler son innehåller parallella maskiner samt schemaläggning för flera produkttyper.

Annem için

Acknowledgments

My journey in Sweden has been a long one, but I have known my co-supervisor Baran Çürüklü for even longer. We have discussed about many things, from cameras, guitars, whiskey, Japanese kitchen knives, to why the pipes of the buildings in Sweden are inside rather than outside, but most importantly lots and lots of research. Lots of questions and ideas going around the room in heated discussions, which I enjoyed very much (most of the time). I could not have written this thesis in fact I wouldn't even be here writing these lines without his support.

Many thanks go to my supervisors Lars Asplund and Baran Çürüklü for teaching me a lot of new stuff, for guidance and support, for all the fruitful discussions, and for the company during the conference trips. Last but not least i would like to thank Mikael Ekström for his feedback on this thesis as well as Stefan Cedergren and Daniel Sundmark for reviewing the PhD proposal.

Many thanks go to, Fredrik Ekstrand, Carl Ahlberg, Jörgen Lidholm, Leo Hatvani, Nikola Petrovič and Stefan (Bob) Bygde for all the funny stuff, the humor, the support and for sharing the office space with me where working is both fruitful and fun. I owe many thanks to Afshin Ameri for helping me as co-author, co-developer and as friend, so thank you Afshin. I wish to thank the people at IDT; Carola Ryttersson, Malin Åshuvud, Jenny Hägglund, Ingrid Andersson, Susanne Fronnå and Sofia Jäderń for making life at the department easier for all of us. I would like to thank many more people at this department, Adnan and Aida Čaušević, Aneta Vulgarakis, Antonio Ciccheti, Cristina Seceleanu, Dag Nyström (Now I know why the birds sing), Farhang Nemati, Giacomo Spampinato, Hüseyin Aysan, Jagadish Suryadeva, Josip Maraš, Juraj Feljan, Kathrin Dannmann, Luka Lednički, Mikael Åsberg, Daniel Kade, Saad Mubeen, Moris Benham, Radu Dobrin, Séverine Sentilles, Svetlana Girs, Thomas Nolte, Tiberiu Seceleanu, and Yue Lu for all the fun coffee breaks, lunches, parties, whispering sessions and the crazy ideas such as

having meta printers that could print printers for printing anything.

I dont know where I would be if it was not for Ingemar Reyier, Johan Ernlund and Anders Thunell. Thank you for helping me with many technical and theoretical challenges that I have had.

Along the way I picked up lots of new and precious friends both in and outside the university environment and without whom I believe I could not have continued further. Thank you Burak Tunca, Cihan Kökler and Cem Hizli. Thank you to Fanny Ängvall and Anton Janhager for keep me from going insane in Västerås.

Finally, I would like to express my gratitude to my parents Nimet Ersoy and Mehmet Akan as well as to my sister Banu Akan for their unconditional love and support through out my life.

This project is funded by Robotdalen, VINNOVA, Sparbanksstiftelsen Nya, EU European Regional Development Fund.

Thank you all!!

Batu Akan
Västerås, December, 2014

List of Publications

Papers included in the thesis [1]

Paper A *Object Selection Using a Spatial Language for Flexible Assembly*, Batu Akan, Baran Çürüklü, Giacomo Spampinato, Lars Asplund, In Proceedings of the 14th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'09), p 1-6, Mallorca, Spain, September, 2009.

Paper B *A General Framework for Incremental Processing of Multimodal Inputs*, Afshin Ameri E., Batu Akan, Baran Çürüklü, Lars Asplund, In Proceedings of the 13th International Conference on Multimodal Interaction (ICMI'11), p 225-228, Alicante, Spain, November, 2011.

Paper C *Intuitive Industrial Robot Programming Through Incremental Multimodal Language and Augmented Reality*, Batu Akan, Afshin Ameri E., Baran Çürüklü, Lars Asplund, In proceedings of the IEEE International Conference on Robotics and Automation (ICRA'11), p 3934-3939, Shanghai, China, May, 2011.

Paper D *Scheduling for Multiple Type Objects Using POPStar Planner*, Batu Akan, Afshin Ameri E., Baran Çürüklü, In Proceedings of the 19th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'14), p 1-7, Barcelona, Spain, September, 2014

Paper E *Towards Creation of Robot Programs Through User Interaction*, Batu Akan, Afshin Ameri E., Baran Çürüklü, To be submitted as a journal paper

[1] The included articles are reformatted to comply with the PhD thesis layout

Other relevant publications

Licentiate Thesis

- *Human Robot Interaction Solutions for Intuitive Industrial Robot Programming, Batu Akan, Licentiate Thesis, ISBN 978-91-7485-060-4, Mälardalen University Press, March, 2012.*

Conferences and Workshops

- *Scheduling POP-Star for Automatic Creation of Robot Cell Programs,* ***Batu Akan****, Afshin Ameri E., Baran Çürüklü, Lars Asplund, 18th IEEE International Conference on Emerging Technologies and Factory Automation - ETFA 2013, IEEE, Cagliari, Italy, September, 2013*

- *Augmented Reality Meets Industry: Interactive Robot Programming, Afshin Ameri E.,* ***Batu Akan****, Baran Çürüklü, SIGRAD, Svenska Lokalavdelningen av Eurographics, p 55-58 Västerås, Sweden, 2010*

- *Incremental Multimodal Interface for Human-Robot Interaction, Afshin Ameri E.,* ***Batu Akan****, Baran Çürüklü, 15th IEEE International Conference on Emerging Technologies and Factory Automation, p 1-4, Bilbao, Spain, September, 2010*

- *Towards Industrial Robots with Human Like Moral Responsibilities, Baran Çürüklü, Gordana Dodig-Crnkovic,* ***Batu Akan****, 5th ACM/IEEE International Conference on Human-Robot Interaction, p 85-86, Osaka, Japan, March, 2010*

- *Towards Robust Human Robot Collaboration in Industrial Environments,* ***Batu Akan****, Baran Çürüklü, Giacomo Spampinato, Lars Asplund, 5th ACM/IEEE International Conference on Human-Robot Interaction, p 71-72 , Osaka, Japan, March, 2010*

- *Object Selection Using a Spatial Language for Flexible Assembly,* ***Batu Akan****, Baran Çürüklü, Giacomo Spampinato, Lars Asplund, SWAR, p 1-2, Västerås, September, 2009*

- *Interacting with Industrial Robots Through a Multimodal Language and Sensory Systems, Batu Akan, Baran Çürüklü, Lars Asplund, 39th International Symposium on Robotics, p 66-69, Seoul, Korea, October, 2008*

- *Gesture Recognition Using Evolution Strategy Neural Network, Johan Hägg,* ***Batu Akan****, Baran Çürüklü, Lars Asplund, ETFA 2008, p 245-248, IEEE, Hamburg, Germany, September, 2008*

Contents

List of Figures

List of Tables

I

Thesis

Chapter 1

Introduction

Robots have become more powerful and intelligent over the last decades. Companies concentrated on large scale production such as car industries have been using industrial robots for machine tending, joining and welding metal sheets for several decades now. Thus, in many cases an investment in industrial robots is seen as a vital step that will strengthen a company's position in the market through increased productivity. However, in small and medium enterprises (SMEs) robots are not commonly found due to a number of key factor, e.g. low volume production, necessity of continuous reprogramming, and layout of SME shop floors.

Even though the hardware costs of industrial robots have decreased, integration as well as programming costs make them unfavorable among many SMEs. Unlike large scale production industries, many SMEs deal with small volume production, with continuously changing product types. From the programming point of the view, no matter how simple the production process is, one has to rely on a professional programmer to integrate and reprogram the robot cell for a new product. Either the company will have to setup a dedicated software department responsible for programming the robots or outsource this need. Maintaining a software department or hiring a consultant from an integrator company is costly for SMEs as well as larger companies. Furthermore, reconfiguring a robot cell is a time consuming process even for skilled engineers. Also it may be difficult to find experts when needed. More specifically, an SME may be forced to rely on a limited number of experts or integrators that are located in the close proximity of the company. This is a real challenge since in Sweden many integrators try to serve a limited geographic

region in their close proximity, e.g. they try to avoid project which may force them to have experts staying over night. It is plausible to assume that similar problems exist in other countries that have high labor costs.

There are additional challenges as well. An industrial robot must be placed in a cell that will occupy valuable workspace and maybe operate only a couple of hours a day. In comparison to large industries, SME shop floors are more often less structured, therefore it is even more challenging to deploy robots to various SMEs. Under these circumstances, it is hard to motivate a SME, which is constantly under pressure, to carry out a risky investment in robot automation. Obviously, these issues result in challenges with regard to high costs, limited flexibility, and reduced productivity.

In order to make industrial robots more favorable in the SME sector, the issues described have to be resolved. Typically for those SMEs, that have frequently changing applications, it is quite expensive to afford a professional programmer or technician, therefore, in our view a human-robot interaction solution is demanded. Using a high-level language, which hides the low-level programming details from the user, will enable a technician or a manufacturing engineer who has knowledge about the manufacturing process to easily program the robot at task-level and to let the robot switch between previously programmed tasks.

The goal of this thesis is to provide tools and methods to make task level robot programming easier and more available to a wider range of users. The primary goal can be divided into several subgoals:

1. Understanding the user's intentions by the system

2. Giving proper feedback to the user confirming that the user's intentions are understood properly

3. Generating a complete robot program based on user's instructions that would utilize the resources in a robot cell optimally.

The driving idea behind this thesis is to bring robot programming to a higher level. Rather than mapping instruction to individual commands, the goal is to provide a complete framework where most of the planning and scheduling tasks are taken care of by the proposed robot programming system. Ideally, the users' part in programming is to give out a brief summary of order of machines to be used in natural language, and all the low-level details to be determined by the system. The whole process of programming would be as easy as teaching the task to a new member of the work team.

This doctoral thesis presents a novel system to support for easy high-level programming of industrial robots. The proposed system includes:

- A multi-modal, incremental framework for rapid development of multi modal interfaces;
- A simulator which checks and verifies robot code, and also acts as visual feedback to the user;
- POPStar planner which is based on partial order planner, to plan and schedule the operations of machines and robots in the cell, based on users instructions

1.1 Outline of thesis

The remainder of this thesis consists of two main parts. The first part contains six chapters: Chapter 2 introduces human-robot interaction (HRI) and methods used for programming robots. It also introduces technical concepts which are used throughout the thesis. Chapter 3 formulates the main research goal, derives research subgoals, and describes the research method that is used. Chapter 4 surveys related work. Chapter 5 presents the research results in line with the research goals and finally Chapter 6 concludes and summarizes the thesis, and gives directions for possible future work. The second part of the thesis is a collection of four peer-reviewed conference and workshop papers.

Chapter 2

Background

This chapter introduces important technical concepts that are used thoughout the thesis. It provides a general introduction to human-robot interaction, a general overview of methods for programming robots. Followed by multi modal interaction and symbolic planning.

2.1 Human-Robot Interaction (HRI)

Robots are artificial agents with capacities of perception and action in the physical world. As robot technology develops and the robots start moving out of the research laboratories in to the real world, the interaction between robots and humans becomes more important. Human-robot interaction (HRI) is the field of study that tries to understand, design and evaluate robot systems for use by or with humans [2].

Communication of any sort between humans and robots can be regarded as interaction. Communication can be of many different forms. However, the distance between the human and the robot alters the nature of communication. Communication, and thus interaction, can be divided into the following two categories: proximate interaction and remote interaction [2]. In proximate interaction, the user and the robot share the same environment, e.g., the user may be located in the robot cell during the programming phase. In remote interaction, the user and the robot can be spatially and temporally separated from each other, e.g., controlling Mars rovers implies that the user and the rovers are both temporally and spatially separated from each other. In most

cases remote interaction is solely limited to spatial separation, e.g. in the case with teleoperated surgery robots. This division helps to distinguish between applications that require mobility, physical manipulation or social interaction. As an example teleoperation and telemanipulations use remote interaction to control mobile remote robot and manipulate objects that are not in the immediate surrounding of the user, whereas proximate interaction, lets say with a mobile service robot, requires social interaction [2].

In social interactions, the robots and the humans interact as peers or companions, however the important factor is that social interaction often requires close proximity.

While the distance between the robot and the user alters the nature of communication, it doesn't define the level or quality of the interaction. The designer, attempts to understand and shape this interaction process in the hope of making it more beneficial for the user. From the designers point of view, the following five attributes can be altered to affect the interaction process [2]:

- Level and behavior of autonomy
- Nature of the information exchange
- Structure of the team
- Adaptation, learning and training of users and the robot
- Shape of the task

2.1.1 Levels of Autonomy

Robots that can perform the desired tasks in an unstructured environment without human intervention are autonomous [2]. From an operational point of view, the amount of time during which a robot can be left without supervision is an important characteristic of autonomy. A robot with high autonomy can be left alone for longer periods of time, whereas a robot with lower autonomy needs continuous supervision and user control. Autonomy, however, is not the highest achievable goal in the field of HRI, but only a means to support productive interaction. Therefore in a human centered applications the notion of *levels of autonomy (LOA)* gains more importance. Even though there are many scales for LOA the following one proposed by Sheridan and Verplank [3] is the most cited one [2]:

1. Computer offers no assistance; human does it all.

2. Computer offers a complete set of action alternatives.

3. Computer narrows the selection down to a few choices.

4. Computer suggests a single action.

5. Computer executes that action if the user approves.

6. Computer allows the human limited time to veto before automatic execution.

7. Computer executes automatically then necessarily informs the human.

8. Computer informs human after automatic execution only if human asks.

9. Computer informs human after automatic execution only if it decides too.

10. Computer decides everything and acts autonomously, ignoring the human.

Note that, these scales may not always be applicable to the whole problem domain but are more beneficial when applied to the subtasks.

The scale proposed by Sheridan helps to determine how autonomous a robot is under certain circumstances, however it does not help to evaluate the level of interaction between the user and the robot from an HRI point of view. This can be illustrated by the following example: A service robot should exhibit different levels of autonomy during the programming phase and the execution phase. A different perspective of autonomy regarding the level of interaction is presented in Figure 2.1. It should be noted the ends of the scale do not indicate less versus more autonomy, thus on the direct control side of the scale, the challenge is to develop a user interface that minimizes the operator's cognitive load. At the other end of the scale, the issue is to create robots with the appropriate cognitive skills in order to interact naturally and efficiently to achieve peer-to-peer collaboration with a human [2]. In addition to fully autonomy at sub-level tasks peer-to-peer collaboration also requires robot with social skills for seamless HRI, therefore it is often considered more difficult to achieve than full autonomy alone.

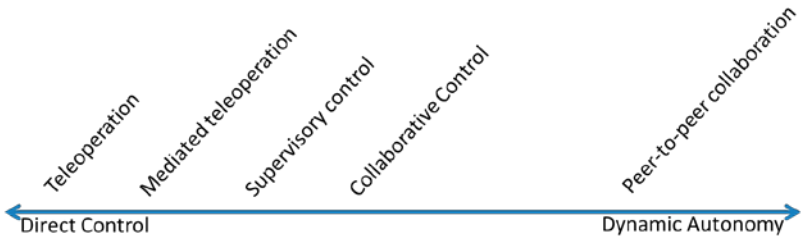

Figure 2.1: Levels of autonomy with emphasis on human interaction.

2.1.2 Nature of Information Exchange

Autonomy is only one aspect that governs the interaction between a human and a robot. The second component defines how the information is exchanged. Input modality defines the nature of the interaction between the robot and the user. Different modalities carry different types of information. While interacting with computers or robots often three of the five senses we use are utilized: audio, visual and touch. However, the same message can be carried over two different channels that address two different senses, e.g. text and speech may carry the same information but appeal to two different senses. Both represent verbal information exchange, but speech carries additional channels such as tonality so the information exchange really has two modalities: verbal and non-verbal. Verbal communication could be better suited for passing commands to the robot and non-verbal communication through gestures is more suitable for conveying spatial information. However, combining these two modalities would yield a more complete and richer communication between the robot and the user.

Speech is an important modality for exchanging information between a robot and a user. The user can give speech commands to the robot to make it interact with the objects in a scene. These commands can be like: “Pick up the blue object” or “Put it next to the green object”. It is also possible to adjust the settings for the task. The user may command the robot to go “faster”, or “slower”, etc. These commands will enable the user to fine-tune the tasks and the skills. Also, any skills or tasks that have been taught to the robot, can be executed through these speech commands.

A gesture is a form of non-verbal communication where visible bodily

actions communicate particular messages. Gestures include movement of the hands, face, or other parts of the body. Gestures differ from physical non-verbal communication in that they do not communicate a specific messages. Gestures can be static or dynamic meaning that, the gesture can be a certain pose of the hand or the body or a movement in certain predetermined patterns. As an example, a pointing gesture is a static gesture while drawing a circle figure is a dynamic gesture. Gestures can be used to program or control robots [4]. Voyles and Khosia integrated a gesture based set of commands into a programming by demonstration framework [5]. Strobel *et al.* use static gestures to direct the attention of a robot to a specific part of the scene [6].

2.1.3 Structure of the Team

It could be that interaction is not limited to one user and one robot. There can be cases in which a person needs to command and interact with multiple robots, or multiple users with different roles interact with a single robot, or multiple users interact with multiple robots. Robots used in search and rescue operations are often operated by two humans, with special roles in the team [7] is an example of many-to-one interaction. On the other hand many unmanned/uninhabited air vehicles (UAVs) can be controlled simultaneously by a single operator [8].

Designing the structure of the team is another aspect of HRI. There are several questions that arise in this respect:

- Who has the authority to make certain decisions: the human or the robot?
- Who has the authority to instruct the robot and at which level?
- How are conflicts solved?
- How are the roles of the robot and the user defined?

The question of what is the role of the human has recently gained importance [9]. Often robots may need to interact with humans who are bystanders with no training at all, e.g., a health-assistant robot must help patients and interact with visitors.

2.1.4 Adaptation, Learning and Training

Designing robots that have the ability to adapt and learn has been widely researched in academia. Even though HRI researchers often want to create robot systems, that can be used with little to no training on the users side, it is sometimes necessary to give training to the users or even HRI designers. This section addresses the issues regarding the training of operators, designers,

as well as robots. The training is often given in the hope of understanding and improving the user interface, interpreting video feedback, controlling the robot, coordinating with other team members and staying safe while operating the robot in a hostile environment [10].

Efforts to Train Users

Even though one of the goals of successful HRI is to minimize the training of the users, certain applications require careful training given to the users in cases where the operator workload or risk is too high. Examples of such cases are military and law enforcement applications, space applications, and search and rescue operations. On the other hand, robots that interact with humans socially are often designed to change the behavior of, educate or train their users, especially if longterm interaction is assumed [11].

Training Designers

The training of designers has received little attention in the HRI literature; however, it is important that they do receive training in the procedures and practices in the fields they seek to help. There are workshops and tutorials for search and rescue robotics [7] as well as tutorials on metrics and experiment design for robot applications [12].

Training Robots

It is often the case that robots need to learn and adapt to the environment or to the user once they leave the factory or the laboratory where they are preprogrammed regarding certain skills and behaviors. However, a well-designed robot, that is beneficial to its user, continues to learn and adapt by improving its reasoning capabilities through interaction. Approaches to robot learning include teaching/programming by demonstration (PbD), as well as task and skill learning. Methods for programming robots are discussed in more detail in Section 2.2.

2.1.5 Task Shaping

As new technology is introduced to our lives, the way we do certain things changes. Similarly, introduction of new robot technologies allow a human to perform things that they were not capable of doing before, or it eases the physical or cognitive workload by making the task easier or more pleasant

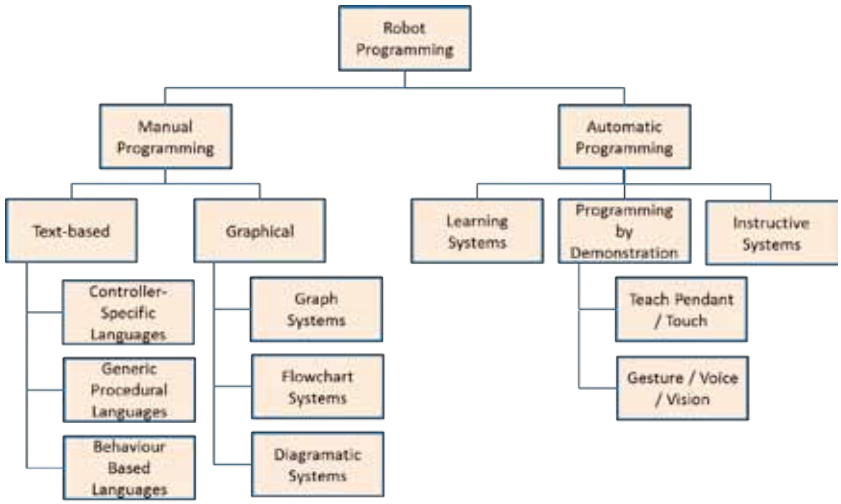

Figure 2.2: Categories of robot programming systems

perform. This means that introducing new technologies fundamentally change the way humans perform the task. Task shaping is the term that emphasizes the importance of considering how the task should be done or will be done after a new technology is introduced [2].

2.2 Robot Programming Systems

This section gives a brief overview of how robots are programmed. This overview is in line with the classification in Biggs and Macdonald [13]. The field of robot programming is divided in to two: manual programming and automatic programming (Fig 2.2). In manual programming the code is created by hand, and this is done through either text-based programming or graphical programming. In automatic mode the robot code is automatically generated and the user has little or no direct control over the code.

2.2.1 Manual Programming Systems

Manual programming systems require the user to create the program by hand often without the actual robot. Once the program is finished, it is loaded into the robot and tested. Manual programming systems are offline methods for programming robots, because the code is created either without using the robot, or with the robot disconnected from the programming environment. However when there are no safety concerns, for example while programming toy robots, it might be possible to control the robot online through an interpreted language where line-by-line execution is possible while creating the code.

Manual programming systems can be divided into two groups: (i) text-based systems, and (ii) graphical programming environments.

Text-Based Systems

Text-based programming systems makes use of conventional programming. It is also the most common mean to program industrial robots together with lead-through programming. Text-based systems can be grouped depending on the type of language used, i.e., controller-specific languages, generic-procedural languages, and behavior based languages.

Controller-specific languages are the most common methods for programing industrial robots. Ever since the invention of industrial robots and robot controllers there has been a machine language, and often a programming language to go with it that can be used to create robot programs. These languages often consists of simple commands for controlling the robot, input/output (I/O), and program flow. ABB's RAPID programming language [14] is an example of controller specific languages (Fig. 2.3). Similarly, all other major industrial robot manufacturer have their own languages dedicated to their own robots and/or systems, e.g., KUKA [15], Motoman [16], and Comau Robotics [17].

In fact, there are as many languages as there are manufacturers. The major disadvantage of controller specific languages is the lack of collaboration, e.g. international standards, in order to develop common languages that can used by several manufacturers. If a company owns robots from several manufacturers, either in-house programmers need to be trained for each type of robot or the company will need to outsource robot programming.

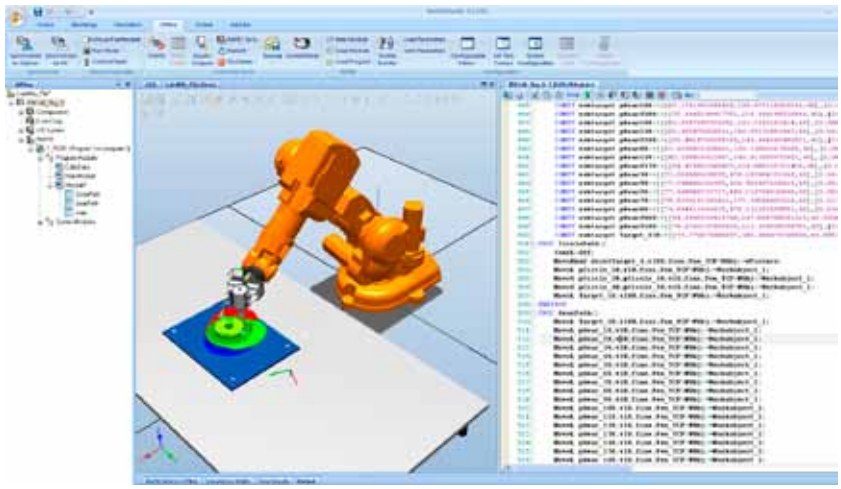

Figure 2.3: A screenshot of the ABB Robot Studio [1]

Generic procedural languages provide an alternative to controller specific languages. Generic programming languages for robots extends standard high-level procedural languages such as C/C++ or Java in a way to provide functionality for the target robot platform. Such an approach is beneficial in research environments, where generic languages are extended to meet the needs of the research project. The extended generic language can be used for system programming or application level programming.

The abstraction provided by the generic language, which consists of a set of functions and/or classes, can facilitate the programming process. Abstraction provides an easy mean to control the robot, while hiding low-level issues, such as handling I/O's or raw sensor data, e.g. moving the robot to a particular position.

Behavior-based languages provide an alternative to the procedural languages. When using these languages one can define how a robot should react to some stimulus or event rather than following a procedural description. The idea behind behavior-based programming is to supply a set of behaviors that independently work to accomplish their goals, while allowing the robot to accomplish high-level tasks. Behavior-based programming employs a

hierarchical system of behaviors specifically written to perform an action based on a set of triggers (cruise, bumper escape, avoid, home, etc..). As the complexity of the overall system increases, new behaviors can be added without changing existing ones. As an example to behaviur based programing is Functional Reactive Programming (FRP) which reacts to both analog and discrete signals. Yampa [18] and Frob [19] are two recent extensions of the FRP architecture. The advantage of FRP is that it is much more code efficient in comparison with procedural languages. In Yampa, for example, it is possible to write a wall-following algorithm with just 8 lines of code.

Graphical Programming Environments

Graphical programming environments provide an alternative to their text-based counterparts. Even though graphical and automatic programming environments have common features, the former are still regarded as manual programming environments. This is because, the user still needs to manually design the program flow and actions. Graphical programming environments utilize graphs, flow charts or diagrams to provide means for programming a robot. In short, small interdependent modules are connected to each other to create procedural flow or behaviors.

Lego's Mindstorm NXT [20] products provide an intuitive and easy to use flowchart based programming environment (Fig. 2.4). Its design is very simple, since its primary target is children. In the programming environment iconic building blocks representing low-level functions are stacked together to produce a sequence of actions. It is also possible to create macros within the programming environment. The generated sequence of commands can either be executed as the main process of the robot, or mapped as a behavior when a certain sensor is triggered. A similar approach developed by Bischoff *et al.* [21] has been used to program industrial robots. In their system the user joins iconic low-level functions to reconfigure the robot to perform the required tasks. Usability tests show that both experts and novice users found the graphical system easier for handling robots.

2.2.2 Automatic Programming Systems

Automatic programming systems can be divided into there categories: (i) learning systems, (ii) programming by demonstration and (iii) instructive systems.

In learning systems, the robot learns by inductive inference from user

Figure 2.4: Lego Midstorm programming environment

provided examples and self exploration [13]. First the robot watches and observes the user through a range of sensors and then tries to imitate the user. Billard and Schaal created a hierarchy of neural networks developed for learning the motion of human arm in 3D space [22]. Weng and Zhang proposed a robot that can learn simple tasks and chain them together to form larger and complex behaviors [23].

Programming by Demonstration (PbD) is a common method for programming robots for trajectory oriented tasks such as arc welding or gluing [24]. Lead-through programing, which can be categorized as PbD, started about 30 years ago with the development of industrial robots and has grown importantly in the last decade with the advances in computer science and sensor technology [13]. Traditional PbD systems use a teach-pendant to jog the robot to the desired position. This position is recorded and a sequence of these positions is used to generate a robot program that will move the robot through a certain path. This method has been the industry standard for many

years. In research this traditional ways of guiding/teleoperating the robot was progressively replaced by more user-friendly and intuitive means [25], such as vision [26, 27], data gloves [28], laser range finder [27] or kinesthetic teaching (ie. by physical guiding the robot's arm through motion) [29, 30, 31]. Kinesthetics provide a rapid way of teaching new paths to robots, especially when used in assembly. Myers *et al.* [32] used programming by demonstration to teach the robot subtasks which are then grouped into sequential tasks by the programmer.

Over the years, research and applications has moved from simply copying or imitating the demonstrations to generalizing across a set of demonstrations. Münch *et al.* [33] suggested the use of machine learning (ML) techniques to recognize elementary operators, thus defining a discrete set of basic motor skills. In their work, they have also issued how to (i) generalize a task, (ii) reproduce a skill in a completely novel situation, (iii) refine the reproduction attempt, and (iv) better define the role of the teacher during learning. There are two different approaches for skill representation. A low-level representation of the skill can be seen as nonlinear mapping between sensory and motor inputs. Trajectory encoding is an example of low-level skills. By contrast, high-level representation of a skill decomposes the skill into a sequence of elementary actions and perception units also referred as symbolic encoding [25]. A significant portion of the work done in the PbD field uses symbolic representation of both the learning and the encoding of skills and tasks [33, 34].

In the case of instructive systems, series of instructions are given to the robot. This type of programming is best suited for executing series of tasks that the robot is already trained for. Using speech to instruct a robot provides a natural and intuitive way. Lauria *et al.* [35] use a speech based natural language input to navigate a mobile robot to different locations via specified routes. Brick and Scheutz [36] provide an incremental framework where the robot can act upon sufficient information to distinguish the intended referent from perceivable alternatives, even when this information occurs before the end of the syntactic constituent. Hand gestures are also used as input by Voyles and Khosia [37] who integrated a gesture based set of commands into a programming by demonstration framework. Strobel *et al.* [6] use static gestures to direct the attention of the robot to a specific part of the scene. Combining the two modalities can even provide improved robustness [38].

2.3 Multimodal Interaction

In-person communication between humans is a multimodal and incremental process [36]. Multimodality allows us to use different, and independent, information channels while communicating with each other. In addition, the incremental nature of this process means that humans start processing of inputs from different channels as soon as the signals are initiated. Thus semantic meanings of the inputs are integrated in real time. Multimodality and incremental processing benefit the interaction between humans in several ways:

Firstly some modalities can transfer certain types of information more precisely while others are more error-prone to special data types. In a multimodal communication, humans use the modality which is more reliable for the information to be transferred. As an example when someone asks about directions to a location, a good way of informing them is using a map, the visual channel, rather than explaining it with words. This is due to the fact that in this context the audio channel is more error-prone to spatial data.

Secondly in a multimodal interaction, different modalities are complementary to each other. This helps to reduce the ambiguity in perceived information and removal of vague data [39]. For example, instructions for assembling newly bought furniture would be more ambiguous and vague without any pictures and only with plain text.

Lastly, the incremental nature of communication in humans helps to start processing of perceived inputs from different modalities as they are being received and build up the semantic meaning of them [40]. The incremental process also applies at context resolving and planning levels. It means that humans build up their responses or reactions as they perceive the inputs [40, 36]. In the HCI/HRI domain, incremental processing helps to improve response times of computer systems. This is specially beneficial for audio inputs, since the computer can use the speech time for performing most of its calculations, thus resulting a much more improved response time [41, 42]. A fast response time is an important feature for designing a multimodal system. Otherwise they may lead to repetition of commands from the user, causing more ambiguity in the recognition process, and user annoyance [43].

Apart from helping in addressing ambiguities in error-prone inputs, multimodal interfaces have other benefits including, interface robustness, reliability, error recovery, alternate communication methods on different situations and increased communication bandwidth [44, 45]. Due to these qualities multimodal interfaces are highly desired in HCI/HRI applications

2.4 Symbolic Planning

Planning in artificial intelligence is a process of composing a sequence of actions to achieve a certain set of goals. In general planning problems can be defined in a formal manner using three entities, i.e. states, actions and goals.

States. Planners use logical conditions to define facts about the world and use conjunctions of these facts to represent states. Propositional literals (e.g., *nothing*, *empty*) or first order literals (e.g., *at(robot1, machine1)*, *processed(obj1, machine1)*) can be used to represent true statements in a robot cell. Conditions that are not mentioned in the state are assumed to be false.

Goals. A goal is the specification of the desired state in the planning problem. A state s satisfies a goal g if s contains all the atoms in g.

Actions. An action consists of preconditions, which must be true before the action can be executed, and effects, which describes how the state changes when the action is executed. A sample action schema to define the movements of a robot from one machine to the next one can be defined as:

```
jog(?robot, ?from, ?to)
  pre: at(?robot, ?from)
  add: at(?robot, ?to)
  del: at(?robot, ?from)
```

jog(?robot, ?from, ?to) serves to identify the action. The action requires the *?robot* to be at location *?from* as a precondition. When the action is executed, the atom *at(?robot, ?to)* will be added to the state, and the atom *at(?robot, ?from)* will be removed from state. From this action schema different actions are created by assigning applicable values to variables *?robot*, *?from*, and *?to*.

2.4.1 Planning with State-space Search

Forward Chaining Planning with forward state-space search, sometimes called progression planning, starts from the initial state and applicable actions are applied to the state at every iteration to get the successor state. Any series of actions that achieve reaches the goal state is a solution. Forward search considers all possible actions therefore without efficient heuristics, the algorithm slows down quickly due to high branching factor.

Backward Chaining Planning with backward chaining starts from the goal state and progresses backwards by only adding actions that achieve the desired goal. An action is relevant to a conjunctive goal if it achieves one of the

conjuncts of the goal. Limiting the search algorithm to use only relevant actions which leads to a lower branching factor than forward search.

2.4.2 Partially Ordered Planners

Totally ordered planners, which search through the state space forwards or backwards, explore only linear sequences of actions by connecting initial state and goal state. Instead of working on subgoals separately, these planners tries to sequence actions from all subgoals. Even a simple planning problem may have many solutions solely because of different ordering of actions. On the other hand partial order planners work on different subgoals independently and then combines the subplans when necessary. This feature is called least commitment strategy. It lessens the search space since the search is carried out in plan space rather than state space. Another important feature of partial order planners is that two or more actions can be added into a plan without specifying which one comes first as the algorithm offers the possibility to execute these actions simultaneously.

A partial order plan can be represented by a set of actions $\mathcal{A}$, a set of constraints $\mathcal{O}$ and a set of causal links $\mathcal{L}$, together forming a tuple $< \mathcal{A}, \mathcal{O}, \mathcal{L} >$. Each action a in $\mathcal{A}$ is an instantiation of an action A scheme which is defined in the planning domain. A plan may contain multiple instances of an action [46]. Every action is represented using the STRIPS [47] representation, where each actions has a list of preconditions and set of achieved conditions and removed conditions.

A partial ordering is a transitive and asymmetric less-than relation between the actions. A partial constraint $a_i < a_j$ forces the action a_i to occur before action a_j [48]. The planner starts with a set of initial conditions $\mathcal{I}$, and a set of goals $\mathcal{G}$. For uniformity $\mathcal{I}$ and $\mathcal{G}$ can be treated as actions $a_\mathcal{I}$ and $a_\mathcal{G}$, where the effects of $a_\mathcal{I}$ represents the initial conditions, of the planning problem and the preconditions of $a_\mathcal{G}$ represent the goals of the problem [46]. When the preconditions of action $a_\mathcal{G}$ hold true, the problem is solved [48]. Any total ordering of actions that is consistent with the partial ordering will be able to solve the planning problem from initial state to the goal state.

A causal link, $a_i \xrightarrow{q} a_j$, represents a commitment by the planner that precondition q of action a_j is to be fulfilled by an effect of action a_i. An *open condition*, $\xrightarrow{q} a_i$, is a precondition q of action a_i that has not yet been linked to an effect of another action [46]. If there exists an action a_t whose effects unifies with the negation q, then that action forms a threat against the causal link $a_i \xrightarrow{q} a_j$. In such cases the causal link $a_i \xrightarrow{q} a_j$ must be protected

against action a_k and new constraints must be added to satisfy either that action a_k comes before a_i (promotion), or a_k must come after a_j (demotion).

A partial-order planner starts with the actions $a_{\mathcal{I}}$ and $a_{\mathcal{G}}$ and the partial ordering constraint $a_{\mathcal{I}} < a_{\mathcal{G}}$. Every action that is added to the plan must come after $a_{\mathcal{I}}$ and before $a_{\mathcal{G}}$. The planner maintains an agenda that is a set of $< q, a >$ pairs, where a is an action in the plan and q is one of the preconditions of a that must be achieved. Initially the agenda contains pairs $< g, a_{\mathcal{G}} >$, where g is one of the atoms that must be true in the goal state [48].

At each iteration, in the planning process, a pair $< q, a_j >$ is selected from the agenda. Then an action, a_i, is chosen to achieve q. That action is either already in the plan, e.g. the start action, or it is a new action that must be added to the plan. If action a_i already exists then a constraint is added such that $a_i < a_j$. If the action a_i is newly added than the following two constraints $a_{\mathcal{I}} < a_i$ and $a_i < a_{\mathcal{G}}$ are also added. A new causal link is also added to $\mathcal{L}$ which states a_i achieves q for action a_j. This causal link is protected by adding extra constraints if it is threatened by other actions. If a_i is a newly added action, its preconditions are added to the agenda, and the process continues until the agenda is empty [48].

2.5 Summary of Verification for Growth Process

As a direct consequence of the research results, the plausibility of the proposed solutions in the market place is investigated through VINNOVAs VFT process (results described below are from the VFT-0 step) [49]. The primary goal was to find out if there is a concrete interest in the technology that is developed, in addition to finding out how such a system is perceived by various types of players. In this context the players are robot manufacturers, integrators that sell complete robot cell, sometimes including after market services, to the end customer, and finally the end customer itself.

Within all these three categories there are different companies. Thus, all robot manufacturers have the same service portfolio meaning that other players are needed in order to fulfill even the basic requirements of end customers. ABB Robotics seems to an exception in this context, i.e. the company prefers to provide robots to integrators in other words they do not provide complete robot cell to all their customers. Thus, the integrators are needed in the chain. It is then the role of the integrators to provide a complete robot cell to the customer. This is at least the case for minor customers, such as most SMEs. This means that an average SME is highly dependent on the integrators if they

have chosen to populate their workshops with robots made by ABB.

Some integrators have a large after market portfolio whereas others prefer to sell a complete system and let other integrators to take over after that the cell is delivered. The end customers, whether a large company or an SME, are heterogeneous as a group. However, in most cases it is unlikely that there are in-house expert robot programmers that can provide the necessary service. This is thus true for all companies, although it creates a rather delicate situation for an average SME.

The result of the interviews made by A-Focus AB [50] indicate that not only the SMEs and the robot manufacturers, but also the integrators welcome a tool that help SMEs in their everyday activities with respect to robot programming. Previously it was assumed that the integrators would feel threatened by such a system, since it would make SMEs address a number of problems that solely integrators can solve. The rationale behind the integrators argument is that many times there is no economical gain in selling maintenance services. In addition, the integrators have dense tight schedule, thus they actually lack the time needed to help with minor issues.

To conclude, it is obvious that this independent study made by A-focus verify our main assumption on the type of solution that is needed by the SMEs in order to facilitate their industrial robot investments. The interviews clearly pointed our SMEs and short series of production. Note that this case is the stating point of this project from the perspective of an innovation potential.

Chapter 3

Research Goals and Methodology

In this chapter the scope of the thesis is presented by formulating the main research goal and employed research methodology. The main research goal is furthermore divided into sub research goals that are required to achieve the main research goal.

3.1 Research Goal

As previously described in Section 1 introducing industrial robots to SME's is a challenging task due to low volume production, crowded and less organized shop floors and lack of skilled technicians that can program robots. The main effort put into this thesis covers the issues regarding lack of skilled technicians in SME's.

For an industrial robot to be used in the manufacturing process, it has to be programmed. This is typically done by either an in house robot programmer, or through an integrator company. An in house robot programmer is an expensive resource for an SME, and in most cases buying the programming need as service from an integrator is not rewarding given the short life cycles of the produced products.

The hypothesis is that in order to address these issues, industrial robots should easily be (re)programmable by engineers that work in the production line at a manufacturing plant. The intention of this thesis is to give an

industrial robot the ability to communicate with its human colleagues in the way that humans communicate with each other, thus making the programming of industrial robots more intuitive and simple. Consequently, a human-like interaction interface for robots will lead to a richer communication between humans and robots.

Based on the arguments above the objective of the thesis, thus the research goal, can be formulated as follows:

> ***Provide tools and methods to make task-level robot programming easier and more available to a wider range of users.***

The research goal presented above points out a complex problem which raises several questions and admits many possible answers. In order to refine this general problem, we define three smaller subgoals that are addressed in the thesis.

3.2 Research Subgoals

This section presents our motivation for the three research subgoals, followed by the iteration of the research subgoal.

3.2.1 Research subgoal 1.

In simplest terms, programming a robot can be seen as a user instructing a robot, to perform some actions related to the robot itself, the machines in the cell or materials to be processed. The commands given by the user need to be parsed and semantically analyzed. Inherent characteristics of natural language, the instructions that are given by the user are either incomplete or can be missing entire chain of actions. The missing information in the users instructions should be filled in by a planner. Later, the actions of the robot should be scheduled to optimize the cell. Finally, this schedule needs to be converted into robot code while paying attention to errors in the code, such as singularities or out of reach commands. Accordingly, the first research subgoal can be formulated as follows.

> *Identify the basic elementary features of human-robot interaction to support seamless interaction between a human user and an industrial robot.*
>
> *(RSG1)*

3.2.2 Research subgoal 2.

The interaction language between the robot and the user is important since the information flows through this language. Since the overall goal of this thesis is to make robot programming easier for people who lack background in programming or robot programming, the choice of language should be readily available, such as natural language.

Nature of information exchange is an important part of natural language interaction between a robot and a human. In-person communications between humans is a multimodal and incremental process [36]. Multimodality allows us to use different information channels while communicating with each other. Incremental nature of this process means (1) humans start processing inputs from different channel as soon as the signals are received, (2) they build up the semantic meaning of the inputs on the fly. These two features benefit the interaction between humans in several ways:

First, multimodal inputs form a basis for a more robust communication scheme as some modalities are more error-prone to some special types of information [39]. They can transfer other data types more precisely. Second, in a multimodal interaction, different modalities are complementary to each other. This helps to reduce the ambiguity in perceived information and removal of vague data [39]. Third, the incremental nature of communication in humans helps to start processing of perceived inputs from different modalities as they are being received and build up the semantic meaning of them. The incremental process also applies at context resolving and planning levels. It means that humans build up their response or reaction as they perceive the inputs [40, 36].

Apart from helping in addressing ambiguities in error-prone inputs, multimodal interfaces have other benefits including: interface robustness, reliability, error recovery, alternate communication methods on different situations and more communication bandwidth [44, 45]. These qualities make multimodal interfaces a good choice for most HCI/HRI applications.

All these benefits of using multimodal natural language helps us to motivate our next research subgoal.

Developing a multimodal and incremental interaction interface to support seamless interaction between the robot and the user

(RSG2)

3.2.3 Research subgoal 3.

From a different perspective, programming a robot using natural language interaction is not only semantically decoding users instructions, but it is also a planning problem. Inherent characteristics of natural language imply that user instructions might be incomplete, e.g. in a machine tending application the user may not explicitly specify the gripper or which machine to use if there are parallel machines. The users' instructions might as well be missing entire chains of actions, e.g. a user might instruct the robot to load machine M_i with raw material and then load the object into machine M_j. In this case several instructions might be absent: starting M_i, jogging the robot to M_i, picking the partially processed object from M_i and jogging to M_j. Thus, the missing information in the user's instructions should be filled by a planner.

From the performance point of view, the robot program generated by the system should be optimal or near optimal. Machine tending, where the robot is responsible for loading and unloading raw materials into machinery for processing, is one of the typical usages of industrial robots. Sequencing, scheduling and optimizing robot's movements in machine tending cells is normally considered as a flow shop problem. The complexity of the robot cell scheduling problem increases if the cell has more complex structure, e.g. cells with multiple robots, dual-gripper robots and/or parallel machines.

There are other logical constraints as well, which are related to the planning problem: (i) a gripper cannot hold more than one object, (ii) the robot cannot load a machine which is already in use, (iii) the robot cannot unload a machine which is empty and (iv) the grippers of the same robot cannot be at different machines at the same time.

This justifies our third and final research subgoal:

Generating a complete production process from user instructions based on a single object.

(RSG3)

3.3 Research Methodology

Research is a process that begins with defining a problem and ends with proposing a solution to the problem. It is a systematic, methodical and ethical process for solving practical and/or problems and generates new knowledge within the domain [51]. During the research process, it is vital to adopt an appropriate research methodology and research methods that are proper for the given research domain.

The research presented in this thesis is motivated by a practical industrial problem - *providing tools and methods to make task level robot programming easier and more available to a wider range of users* - therefore it falls into the category of applied research. More specifically the research proposes solutions that fall in to human-robot interaction and symbolic AI planning.

The basic research methodology was to identify and formulate a general research problem by studying the related work and try to propose better solutions to the problem. Through this work new tools have been iteratively developed, measured, analyzed and validated until no more improvements seemed possible. Essentially the following steps were performed iteratively.

1. Perform literature review on the current research problem.
2. Propose a solution based on the state of the art.
3. Construct a prototype that implements the proposed solution.
4. Verify and validate the prototype through experiments.

In this thesis in order to understand the given research problem, first we performed information gathering and state-of-the-art investigation covering and familiarizing ourselves with previous work conducted on the research problem. Based on the findings from the state-of-the-art investigation, research subgoals were formulated as described in Section 3.2. In the following phases of research, each research subgoal were addressed individually.

Paper A initially address the issues on instructing and receiving feedback from the robot. Papers B and C enriches this communication by adding extra information channels. Paper D is based on taking the input from procedures described in papers A, B, C and tries to generate a complete plan to operate the robot cell.

Chapter 4

Related Work

This chapter relates the work presented in this thesis to work of others. The chapter is divided into subsections in which comparisons are provided for each area respectively.

4.1 Programming Industrial Robots

Activities within the scope of industrial robot programming can be divided into two groups, i.e. low-level programming and high-level programming. The former consists of path creation and programming the logic aspects, whereas the latter focuses on a group of operations or tasks. As an example low-level programming approach would define the joint angles of the robot while picking up an object. On the other hand high-level programming approach would provide instructions like ”pick object” [52].

In further division low-level programming systems can be grouped into two groups as academic research work and commercially available products. In their survey, Rossano *et al.* [52], provides an industrial perspective by presenting commercially available low-level solutions. They further divide low-level programming into five sub categories, including icon-based programming [53], programming via data flow diagrams [54], CAD-based programming [55, 56], wizard-based programming [57] and lead through programming [58].

Relevant research projects also focus on high, low and combination of both. In addition, combination of different methods are also used to perform robot programming. Pires [59] uses a combination of voice and lead through

programming where the robot arm is manipulated by the user from point A to point B while the user speaks out the type of path the robot should follow.

On a higher level of abstraction, speech modality alone has been used by Pires [60] to command an industrial robot through switching between preprogrammed asks. In their work, an automatic speech recognition system is used to convert structured spoken commands into text in order to control the robot.

Marin *et al.* [61, 62] presents a user interface and a system architecture of an Internet based telelaboratory that allow remote control and programming of two online robots. An augmented reality and nonimmersive virtual reality interfaces gives the remote operator information about the objects in the robot cell. The system can receive high-level voice commands such as "pick up an object" or "release object".

4.2 Multimodal Approach

Designing and implementing a multimodal interface is a high cost process, thus multimodal application frameworks are needed [44, 43]. Several researchers have proposed multimodal frameworks to address this shortcoming [63, 64, 65, 66].

W3C has introduced a web-specific multimodal framework [67]. It is a markup language for multimodal systems and does not define system architecture. It is designed to be used as a guideline for implementing multimodal systems. As a part of this framework, W3C has introduced EMMA (Extensible Multi-Modal Annotation markup language) which has later been used by other researches for implementing multimodal interfaces [68]. The framework is limited in the sense that it can not handle new innovative ways of interactions easily, mostly due to the fact that it does not support programming languages and most of the programming is only available through Javascript [69].

Multimodal grammars were proposed by Johnston *et al.* [70, 71] for definition of mixed inputs from different modalities. In his work unification-based approaches and later, finite-state transducers (FST) to parse and fuse different modality inputs are used [71, 72]. A useful feature of multimodal grammars is that they represent the input from different modalities in a single file which also can represent semantic data simultaneously. This allows rapid development of multimodal applications and also increases their maintainability.

A more dynamic approach compared to FST model is presented by Stiefelhagen *et al.* [73]. The model uses a graph with terminal and non-terminal nodes for grammar representation. This approach allows for real time expansion of a graph during the recognition [73].

A framework which uses spatial ontologies and user context for multimodal integration is described by Iriwati *et al.* [74]. Object ontologies are defined in OpenCyc, and Microsoft's Speech API (SAPI) is used for speech recognition. The grammar is defined as a part of SAPI, thus object and other modalities' anchors are defined as a part of the speech grammar. This means that non-speech modalities need a specific speech anchor in order to be integrated into the multimodal interaction. As an example, the phrase "put it here" has two words, "it" and "here", which act as objects anchors. The multimodal integration engine then uses these two anchors and object ontologies to perform the integration. This means that without the anchor words, other modalities will be ignored.

Another multimodal system with an incremental pipeline is presented by Brick and Scheutz [36]. This system which is called Robotic Incremental Semantic Engine (RISE), can process syntactic and semantic information from audio/visual inputs incrementally and generates the feedback in real time. RISE's architecture is based on incremental parsing of speech input with different sub-systems that try to resolve the references in the speech through visual data or the context of dialog. Therefore RISE is not a fully multimodal system, since non-speech modalities are complementary to the speech, thus are not part of the interaction.

An abstract model for implementing an incremental dialog manager is proposed in [75]. Although the model is not designed for multimodal interactions, but it can easily be modified for that purpose.

Similar to the works of Irawati *et al.* [74] and Brick *et al.* [36], the solution proposed in this thesis also use object references during fusion. The difference is that in order to dismiss the problem of using speech as the main modality, we have defined a multimodal language which allows for definition of all possibles mixtures of different modality inputs. Through this language (which we call it COLD) any modality or any mixture of modalities can be used in the interaction. Pure non-speech communications are easily achievable through COLD. We also take a similar approach as [73] for grammar representation and parsing. Our contribution is in expanding this model to a more general one which can represent several modalities in the same graph.

4.3 Augmented Reality

Augmented reality (AR) is a term used for overlaying computer generated graphics, text, two dimensional (2D) and three dimensional (3D) models over real video stream. Virtual information is embedded into the real world, thereby augmenting the real scene with additional information. Augmented reality is proved to be useful in several industrial cases, especially for visualizations. Olwal *et al.* [76] used 3D optical visualization techniques to visualize the process of a CNC machine in order to support a machine operator. AR also provides great opportunities for HRI, and has been widely used in telerobotics since AR allows the operator to work as if he is present at the remote working environment [77, 61, 78]. However AR can be beneficial for programming industrial robots as well whether it is remote or local. Through wearable computers and head mounted displays it is possible to visualize and generate paths through a pointing device [79]. In their work Chong *et al.* [80] visually tracked marker to define collision-free paths for the industrial robot to follow. Once the path is generated a virtual robot simulates the behavior of the robot on the screen.

4.4 Scheduling

Robotic flow shop problems, in which travel times between machines are not negligible, have received fair amount of interest from industry. After all, scheduling the robot's movements between the machines will influence the throughput of a robot cell. A detailed classification of types of robot cell scheduling problems is given by Dawande *et al.* [81].

Configuration of a robot cell is also an influential factor when it comes to deciding on a scheduling strategy for robot cells. A configuration consisting of parallel machines or a dual/multiple gripper robot adds new challenges to the scheduling problem [82].

In their work, Fan and Winley [83] use a heuristic search algorithm for the flow shop scheduling problem based on best-first search. Even though the work presented in the paper is not for a robot cell, the best-first search guided by heuristic can be used to guide a search in a robot cell application.

Other solutions for the general problem include mathematical methods such as mixed integer programming. In their work Koné *et al.* [84] present an event based mixed integer linear programming method for resource-constrained project scheduling problem.

4.5 Planning

As discussed earlier in the Introduction, the problem presented in this thesis is also a symbolic AI planning problem. Classical symbolic planning algorithms such as STRIPS [47] and Graphplan [85] exist in literature as totally ordered planners.

LAMA planner [86] uses heuristic guided best-first search to find a solution with minimum action cost. It pre-processes the problem and extracts landmarks, which are propositions that must be true in every plan, to use as subgoals. LAMA employs a forward state-space search by imposing a total-order on the actions. This avoids the need for explicit search on conflicts in the plan. In a totally ordered plan, a new action can start only after the previous finishes. However, solving a flow shop problem requires planning many actions that should be executed in parallel. Partial order planners [87, 46] can order actions in parallel with each other, if the required resources for the actions are mutually exclusive. Partial order planners often use backward chaining, starting from a goal state adding actions to support the goal. Backward chaining reduces the search space as it explores only the possibilities that would lead to the goal, with the cost of constantly searching for conflicts between actions in the plan.

In their work, Coles *et al.* [88] present partial order planning with forward chaining with the advantage of reducing searches for conflicts.

In comparison to the work presented above, the solution that is proposed in this paper deals with the robotic flow shop problem. The proposed algorithm can handle dual gripper robots, parallel machines and machines with complex configurations. Best-first search strategy is used similar to the works of Fan *et al.* [83] and the LAMA planner [89]. In order to schedule actions which can run in parallel with each other a modified version of the partial order planning algorithm was used. Similar to the LAMA planner [89], landmarks were used to narrow the search space. However, the landmarks are used in this work are the partial commands uttered by the user.

Chapter 5

Results

This chapter, which is divided into two sections, presents the results achieved within this thesis. The first is a summary of the research done in relation to the research subgoals presented in Section 3.2. The second part presents a short summary, specific scientific contribution, and the author's contribution of the Papers A-E.

5.1 Contributions

The contributions of this thesis can be divided into four parts.

5.1.1 Object-Based Programming Scheme

Programming industrial robots is not an easy task for a person who doesn't have previous experience. One of the reasons is that, even though we occupy the same physical space as the robots, the intrinsic representation of the world is different for humans and robots. Humans represent the world around them by describing objects and spatial relations between these objects in a natural language. Robots, by contrast, work in abstract numeric coordinate systems which are not intuitive to us. Papers A and C aims to find a mapping between our object-based representation of the world and the robots' numeric representation.

Paper A explores the use of speech only natural language that is supported by spatial relations in an industrial environment. In the paper, a high-level language in order to command an industrial robot for simple pick-and-place

applications is demonstrated. The proposed high-level language allows the user to give high-level commands to the robot to manipulate the objects. The proposed language can handle attributes of the objects in the environment, such as shapes, colors, and other features. It is equipped with functions for handling spatial information, enabling the user to be able to relate objects spatially to other objects. Using the features about the objects in the scene and their spatial relations to one another it is possible to give commands like "pick the round object to the left of the red one". Which helps to make the communication natural and human like, and frees the user from think about the working coordinate system of the robot.

Paper C improves the work presented in Paper A by introducing the multimodal framework and augmented reality into the system. This allows the user to sequence basic skills e.g. picking and placing objects to accomplish more complex tasks. A camera mounted on the gripper of the robot, gives the ability to see through the eyes of the robot and select objects and drop locations for these objects and give high-level speech commands to manipulate the objects. The commands that the user has given is overlaid on the augmented reality window showing the path of the robot, pick and place operations and the final positions of the objects. This gives confidence to the user that the commands were understood correctly removing any vagueness.

The proposed system demonstrates an alternative method for interaction between industrial robots and humans. Using natural means of communication is definitely an interesting alternative to well-established robot programming methods. These methods require considerable larger amount of time, and perhaps more importantly, a programming expert. In the experiments it is demonstrated that efficient collaboration between the robot and its human peer can help to clarify vague situations easily. Multimodal language allows the system to understand the user's intentions in a faster and more robust way.

Finally Paper E brings together the components that were developed earlier for creating a novel system for task-level programming of industrial robots.

The proposed system has also received positive attention for its innovative approach in programming from VINNOVA - Swedish Governmental Agency for Innovation Systems. Based on an independent survey done by A-Focus AB [50] where interviews with 8 companies is performed, the prosed system has been granted VFT-0 [49]. According to the interviews such a system is demanded by the industry and is believed to increase the companies competitiveness and decrease programming costs.

5.1.2 General Multimodal Framework

There have been numerous efforts to implement multimodal interfaces for computers and robots [90, 41, 91, 75]. Yet there is no general standard framework for developing multimodal interfaces. In order to design and implement such interfaces efficiently, we propose a general framework. The proposed framework in Paper B is designed to perform natural language understanding, modality fusion and semantic analysis through an incremental pipeline.

The framework also employs a new grammar definition language which is called COactive Language Definition (COLD) which is responsible for multimodal grammar definition and semantic analysis representation. COLD is used to (i) generate separate grammars for each modality, (ii) define the fusion pattern, (iii) define the semantic variables and calculations and (iv) define dialog patterns and dialog turns. This means that COLD affects the whole process from the processing of inputs to modality fusion, semantic analysis and dialog management.

The framework is also an incremental system which allows to start processing of input words or signals from other modalities as they are received before the sentence is complete. Parsing, modality fusion, semantic meaning generation and execution are all performed through this incremental pipeline. With an incremental system it is easy to build a response to user inputs even before the sentence is completed. In the HCI/HRI domain, incremental processing helps to improve the response times of computer systems. Multimodal systems should be responsive, because they would otherwise require the repetition of commands from the user, more ambiguity in the recognition process as well as user annoyance.

5.1.3 POPStar Planner

The third contribution of this thesis, presented in Paper D, is a symbolic planner based on a partial order planner (POP). The proposed planner, POPStar, is used for planning and scheduling the robot's motions as well as operation of other machinery in the cell. The challenges of this part of the research is associated with RSG3. The system takes the landmarks extracted from user instructions as input, and tries to create a sequence of actions to operate the robot cell with the shortest possible makespan. The proposed system takes advantage of partial order capabilities of POP to execute actions in parallel and employs a best-first search algorithm to seek the series of actions that lead to the shortest makespan.

Since POPStar uses first order logic to define the rules of the operating world, new constraints can be added easily. As an example gripping patterns or which gripper to use can be added as constraints for some machines. Our proposed planner can handle robots with multiple grippers, as well as parallel machines in deterministic free-pickup robot cells. Using different topologies in the landmark graph, one can generate schedules for processing multiple type objects simultaneously or with smooth transition from one type to another. Using POPStar planner it is possible to generate complete production process based on users' instructions.

5.1.4 Simulation Environment

A simulation environment which can simulate a general 6 Degree of Freedom (6DoF) industrial robot has been created in an OpenGL environment. Besides providing a quick test environment, it also acts as a visual feedback to the user. The simulator wraps around basic movement functions of ABB RAPID [14] language. Basically the simulator provides a similar programming interface as the RAPID language provides, making it possible to program them through C# [92]. With an extended version of this API it is also possible to use Prolog [93], which is a general-purpose logic programming language, to program the robot. Calls from C# are tested for reachability along a path. Collision free path planning from point A to point B can be performed. The simulator can work as a augmented reality engine as well. Overlying virtual objects and robot paths over live camera feed, it is possible to create a rich user interface to be used with incremental multimodal framework as shown in Paper C. Through the simulator it is possible to do both offline and online programming.

5.2 Overview of Papers

5.2.1 Paper A

Object Selection using a Spatial Language for Flexible Assembly, Batu Akan, Baran Çürüklü, Giacomo Spampinato, Lars Asplund, In Proceedings of the 14th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'09), Mallorca, Spain, September, 2009.

Summary: In this paper we propose a limited natural language that utilizes spatial terms to hide the complexities of robots programming. We use Gaussian kernels to represent spatial regions such as "left" or "above". We also

introduce our robot simulation environment where we check for reachability and collisions. The simulation environment also provides application programmers interfaces for procedural and behavioral programming in C# and Prolog languages.

Author's contribution: I was the main author of this paper contributing with the development of the system, together with the programming interfaces. The co-authors contributed with technical aspects, such as inverse kinematics for the robot and valuable feedback on the overall paper.

5.2.2 Paper B

A General Framework for Incremental Processing of Multimodal Inputs, Afshin Ameri E., Batu Akan, Baran Çürüklü, Lars Asplund, In Proceedings of the 13th International Conference on Multimodal Interaction (ICMI'11), Alicante, Spain, November, 2011.

Summary: In this paper we propose a framework for the rapid development of multimodal interfaces. The framework is designed to perform modality fusion and semantic analysis through an incremental pipeline. The incremental pipeline allows for semantic analysis and meaning generation as the inputs are being received. The framework also employs a new grammar definition language which is called COactive Language Definition (COLD). COLD is responsible for multimodal grammar definition and semantic analysis representation. This makes it easy for multimodal application developers to view and edit all the required resources for representation and analysis of multimodal inputs at one place and through one language.

Author's contribution: As the second author of this paper, I contributed with the idea and formalization of COLD, the partial development of the system, experiments and the editing of the paper

5.2.3 Paper C

Intuitive Industrial Robot Programming Through Incremental Multimodal Language and Augmented Reality, Batu Akan, Afshin Ameri E., Baran Çürüklü, Lars Asplund, In proceedings of the IEEE International Conference on Robotics and Automation (ICRA'11), Shanghai, China, May, 2011.

Summary: In this paper, we propose to use an incremental and multimodal natural language, which we developed in Paper B, in combination with our simulation environment and augmented reality. A view of the working environment is presented to the user through a unified system. The system overlays visuals through augmented reality to the user and also receives inputs and voice commands through our high-level multimodal language. The proposed system architecture makes it possible to manipulate, pick or place the objects in the scene. Such a language shifts the focus of industrial robot programming from a coordinate-based programming paradigm to an object-based programming scheme.

Author's contribution: I am the main author of this paper and contributed with the development of the AR module, the experimental setup and the writing of the paper. The co-authors contributed with technical aspects and valuable feedback on the overall paper.

5.2.4 Paper D

Scheduling for Multiple Type Objects Using POPStar Planner, Batu Akan, Afshin Ameri E., Baran Çürüklü, In proceedings of the IEEE Emerging Technologies and Factory Automation (ETFA'14), p 1-7, Barcelona, Spain, September, 2014

Summary: This paper presents scheduling algorithm for robot cells that produce multiple object types in low volume productions. The main challenge is to adopt the schedule for changing object types while maximizing the number of objects produced in a given time window. Proposed algorithm, POPStar, is based on a partial order planner which is guided by best-first search algorithm and landmarks. The best-first search, uses heuristics to help the planner to create complete plans while minimizing the makespan. The algorithm takes landmarks, which are extracted from user's instructions as input. Using the POPStar algorithm, it is possible to create schedules for changing object types using different topologies for the landmarks graph.

Author's contribution: I was the main author of this paper contributing with the idea, software development and prepared most of the manuscript. The other co-authors contributed with technical aspects of the paper

5.2.5 Paper E

Towards Creation of Robot Programs Through User Interaction, Batu Akan, Afshin Ameri E., Baran Çürüklü, To be submitted as a journal paper

Summary: This paper brings together the components that were previously developed to create a novel system for task-level programming. The user interacts with an industrial robot by giving instructions in a structured natural language and by selecting objects through an augmented reality interface. The proposed system consists of two parts. The first component is a multimodal framework which provides a natural language interface to the user to interact in which the framework performs modality fusion and semantic analysis. The second component is the POPStar planner, which creates a sequence of actions to operate the robotic cell with minimal makespan. Results show that the proposed system can create and adapt schedules for robot cells with changing product types in low volume production based on user's instructions. The work presented in this paper deals with all three research subgoals simultaneously.

Author's contribution: I am the main author of this paper and I am responsible for software development and the manuscript. The co-authors contributes with technical aspects and valuable feedback on the overall paper.

Chapter 6

Conclusions and Future Work

This chapter provides a brief summary of the thesis and conclude with possible directions for further research within this topic.

6.1 Conclusions

The objective of the research presented in this thesis is to develop tools and methods to make task level robot programming easier and more available to a wider range of users. The motivation behind issues related to programming industrial robots is presented in Chapter 3 and scientific contribution of included papers are presented in Chapter 5.

Within the context of the proposed methods and tools, robot programming process can be divided into 3 parts, (i) user communicating with the robot, (ii) planning and sequencing actions and (iii) generating code to operate the robot cell. The work presented in the thesis touches upon all aspects described.

The communication is an important aspect of the whole process. We propose that any method for communication, should be clear and easily be used by an untrained operator, thus the user needs to give high-level user commands as well as should easily describe which object to be manipulated by the robot. Communication is also a two way process. As the user gives the commands, the system needs to give feedback to make sure that the user understands the system's status, e.g. that the system has interpreted the given

command in the correct way, or if any information is missing. Papers A, B and C address the communication issues. Paper A addresses the problem by trying to refer to objects in the scene by using their spatial relation with one each other. Papers B and C proposes a multimodal framework where the user can directly point out to an object in the scene by selecting through an augmented reality interface using the simulation environment. This is important as it allow the user to communicate with the system based on references to objects rather than abstract coordinate systems.

The proposed multimodal framework is designed to perform natural language understanding, modality fusion and semantic analysis through an incremental pipeline. It also provides a novel grammar definition language, COLD which is used to define the multimodal grammar and semantic analysis procedures. The incremental nature of the framework allows to for in-time interaction between the robot and its user. Apart from programming industrial robots, the proposed multimodal framework can be used in areas of computer games, human-computer interaction or human-robot interaction. The common ground found in all these application domains is the need of rich information exchange.

The second issue that is addressed in the thesis is the planning and scheduling phase of programming. This need rises from the inherent characteristics of natural language where the users instructions are either incomplete or missing hence needs to be filled in by a planner. Furthermore, to run the robot cell optimally the order of actions of the robot needs to be carefully scheduled. The solution that we have proposed is the POPStar algorithm that uses best-first search with guidance of heuristics to generate schedules and hence programs to run the robot cell. POPStar uses first order logic to define possible actions and state of the cell.

The advantage of POPStar against other solutions is that it uses first-order logic to create the constraints in the search space through the definitions of actions, therefore it is simpler to define and extend the problem to similar types of robot cells where different constraints must be kept. POPStar algorithm is evaluated with a flow shop scheduling problem with parallel machines and multi-gripper robots. This algorithm suits well with task level programming as it takes incomplete instructions as input and generates plans for accomplishing the task while optimizing the resource usage and minimizing the makespan. This feature provides an advantage to the user since he/she can focus on the core aspects of the program that has to be composed.

Overall the multimodal framework and the POPStar planner helps us achieve our main goal by enabling the system to receive commands from

the user based on ordinary objects in the scene and the planner helps to fill in the missing information to achieve the given task. The proposed system, which is easy to learn and use, enables large numbers of users to benefit from that technology, since it lowers the threshold associated with industrial robot programming. Minimizing the need of expert knowledge is beneficial for the industry, especially for SMEs. Providing such a solution to this segment might encourage them to invest in robot automation, which in return might help to boost productivity. It is not only SMEs that can benefit from the easy programming of industrial robots; integrators, who develop robotic automation solutions for various companies can do so too. Developing and delivering solutions to their customers at the fraction of the time it took before would increase their competitiveness in the market.

6.2 Future Work

This thesis opens up possibilities to conduct further research in certain areas that needs additional attention.

6.2.1 Multimodal Framework

The current work addresses the interaction between the robot and the user. In a way the robot understands the user's intentions, but it has no sense of the environment. Object recognition and localization are essential abilities for robots in order to work in unstructured environments. From the point of view of this thesis, object localization abilities will also enrich the interaction process between the robot and the user. In the future we plan to add a vision system capable of object recognition and object localization.

In our initial experiments, users have found the system easy to learn and to use. The application domain is solely pick-and-place applications. We have not touched upon, e.g. spot/arc welding applications or assembly applications. It is plausible to assume that widening of the application domain will help to test the system further, and find shortcomings that are hidden in the current system.

The current natural grammar is limited in size, As the grammar grows upon requests from the industry, it is necessary to analyze the impact of this growth and how it effects the multimodal framework. Another extension would be to introduce a context analyzer, which switches on and off some of the rules defined in COLD to improve the robustness of the framework as the size of the

grammar gets bigger.

6.2.2 POPStar

In terms of flow shop scheduling problem, our implementation currently supports free-pickup cells, although a new heuristic can be introduced to support no-wait and interval-pickup cells as well.

In the case of producing multiple types of objects simultaneously a load balancer can be added to the planner so that different production rates can be assigned to different object types. This would increase the user's possibility to control the operation of the robot cell even further. It is also possible to control the planner in an interactive manner so that it would merge new plans with already existing ones while the robot cell is operating. This will give the user the ability to handle changes which might be required in a robot cell, interactively. This offers the flexibility that is of great importance to an SME as well as their larger counter parts.

Bibliography

[1] "ABB RobotStudio," accessed: 29 August 2014. [Online]. Available: http://new.abb.com/products/robotics/robotstudio

[2] M. a. Goodrich and A. C. Schultz, "Human-Robot Interaction: A Survey," *Foundations and Trends in Human-Computer Interaction*, vol. 1, no. 3, pp. 203–275, 2007.

[3] T. B. Sheridan and W. L. Verplank, "Human and Computer Control of Undersea Teleoperators," Tech. Rep., 1978.

[4] T. Zhang, V. Ampornaramveth, M. Bhuiyan, Y. Shirai, and H. Ueno, "Face and gesture recognition using subspace method for human-robot interaction," *Advances in Multimedia*, pp. 369–376, 2005.

[5] R. Voyles and P. Khosla, "Gesture-based programming: a preliminary demonstration," *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, no. May, pp. 708–713, 1999.

[6] M. Strobel, J. Illmann, B. Kluge, and F. Marrone, "Using spatial context knowledge in gesture recognition for commanding a domestic service robot." Robot and Human Interactive Communication, 2002, 2002, pp. 468–473.

[7] R. R. Murphy, "HumanRobot Interaction in Rescue Robotics," *IEEE Transactions on Systems Man and Cybernetics Part C Applications and Reviews*, vol. 34, no. 2, pp. 138–153, 2004.

[8] J. S. McCarley and C. D. Wickens, "Human Factors Implications of UAVs in the National Airspace," *Human Factors*, no. April, 2005.

[9] J. Scholtz, “Theory and evaluation of human robot interactions,” *System Sciences, 2003. Proceedings of the 36th*, 2003.

[10] Z. Pronk and M. Schoonmade, *Mission preparation and training facility for the European Robotic Arm (ERA)*, ser. NLR TP // Nationaal Lucht- en Ruimtevaartlaboratorium. Nationaal Lucht- en Ruimtevaartlaboratorium, 1999.

[11] T. Kanda, T. Hirano, D. Eaton, and H. Ishiguro, “Interactive Robots as Social Partners and Peer Tutors for Children: A Field Trial,” *Human-Computer Interaction*, vol. 19, no. 1, pp. 61–84, 2004.

[12] G. Trafton, “Experimental Desing in HRI,” 2007.

[13] G. Biggs and B. MacDonald, “A Survey of Robot Programming Systems,” in *Proceedings of the Australasian conference on robotics and automation*, Australasian Conference on Robotics and Automation. Citeseer, Dec. 2003.

[14] *RAPID Reference Manual 4.0*, ABB Flexible Automation, 72168, Vasteras, SWEDEN.

[15] “KUKA Roboter GmbH,” accessed: 21 October 2014. [Online]. Available: http://www.kuka-robotics.com/en/

[16] “Yaskawa America, Inc.” accessed: 21 October 2014. [Online]. Available: http://www.motoman.com/

[17] “Comau Robotics,” accessed: 21 October 2014. [Online]. Available: http://www.robotics.comau.com/

[18] P. Hudak, A. Courtney, H. Nilsson, and J. Peterson, “Arrows, Robots, and Functional Reactive Programming,” in *Advanced Functional Programming, 4th International School, volume 2638 of LNCS*. Springer-Verlag, 2002, pp. 159–187.

[19] J. Peterson, G. D. Hager, and P. Hudak, “A language for declarative robotic programming,” *Proceedings 1999 IEEE International Conference on Robotics and Automation Cat No99CH36288C*, pp. 1144–1151, 1999.

[20] “Lego Mindstorms,” http://mindstorms.lego.com/en-us/Default.aspx, accessed: 21 October 2014.

[21] R. Bischoff, A. Kazi, M. Seyfarth, A. K.-r. De, and I. D. B, "The MORPHA Style Guide for Icon-Based Programming," *Robotics*, 2002.

[22] A. Billard and S.Schaal, "Robust learning of arm trajectories through human demonstration." Intelligent Robots and Systems, 2001, pp. 734–739.

[23] Y. Zhang and J. Weng, "Action chaining by a developmental robot with value system." International Conference on Development and Learning, 2002, pp. 53–60.

[24] R. D. Schraft and C. Meyer, "The Need For an Intuitive Teaching Method For Small and Medium Enterprises." International Symposium on Robotics, May 2006.

[25] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, *Handbook of Robotics : Robot Programming by Demonstration*. Springer, Jan. 2008, ch. 59.

[26] Y. Kuniyoshi, M. Inaba, and H. Inoue, "Teaching by showing: Generating robot programs by visual observation of human performance," *International Symposium on Industrial Robots*, 1989.

[27] S. B. Kang and K. Ikeuchi, "A Robot System that Observes and Replicates Grasping Tasks," in *The Fifth International Conference on Computer Vision*, Jun. 1995, pp. 1093–1099.

[28] C. P. Tung and A. C. Kak, "Automatic learning of assembly tasks using a DataGlove system," in *IROS '95: Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*. Washington, DC, USA: IEEE Computer Society, 1995, pp. 1–8.

[29] S. Calinon, F. Guenter, and A. Billard, "On learning, representing and generalizing a task in a humanoid robot," *IEEE Transactions on Systems,Man and Cybernetics, Part B. Special issue on robot learning by observation, demonstration and imitation*, pp. 286–298, 2007.

[30] M. Ito, K. Noda, Y.Hoshino, and J. Tani, "Dynamic and interactive generation of object handling behaviors by a small humanoid robot using a dynamic neural network model," *Neural Networks*, vol. 19, no. 3, pp. 323–337, 2006.

[31] T. Inamura, N. Kojo, and M. Inaba, "Situation recognition and behavior induction based on geometric symbol representation of multimodal sensorimotor patterns," in *Intelligent Robots and Systems,*, Beijing, China, Oct. 2006, pp. 5147–5152.

[32] D. R. Myers, M. J. Pritchard, and M. D. J. Brown, "Automated programming of an industrial robot through teach-by showing." International Conference on Robotics and Automation, 2001, pp. 4078–4083.

[33] J. Kreuziger, M. Kaiser, and R. Dillmann, "Robot programming by demonstration (rpd) - using machine learning and user interaction methods for the development of easy and comfortable robot programming systems," in *In Proceedings of the 24th International Symposium on Industrial Robots*, 1994, pp. 685–693.

[34] M. Pardowitz, S. Knoop, and R. Dillmann, "Incremental Learning of Tasks From User Demonstrations, Past Experiences, and Vocal Comments," *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICSPART B: CYBERNETICS*, vol. 37, no. 2, pp. 322–332, 2007.

[35] S. Lauria, G. Bugmann, T. Kyriacou, and E. Klein, "Mobile robot programming using natural language," *Robotics and Autonomous Systems*, vol. 38, no. 3-4, pp. 171–181, Mar. 2002.

[36] T. Brick and M. Scheutz, "Incremental natural language processing for HRI," in *Proceeding of the ACM/IEEE international conference on Human-robot interaction - HRI '07*, vol. 07pages. New York, New York, USA: ACM Press, 2007, p. 263.

[37] R. M. Voyles and P. K. Khosia, "Gesture-Based Programming: A Preliminary Approach."

[38] S. Iba, C. J. J. Paredis, and P. Khosla, "Interactive Multi-Modal Robot Programming."

[39] S. Oviatt, "Ten myths of multimodal interaction," *Communications of the ACM*, vol. 42, no. 11, pp. 74–81, 1999.

[40] Y. Kamide, G. T. M. Altmann, and S. L. Haywood, "The time-course of prediction in incremental sentence processing: Evidence from anticipatory eye movements," *Journal of Memory and Language*, vol. 49, no. 1, pp. 133–156, 2003.

[41] D. Mori, S. Matsubara, and Y. Inagaki, *Incremental parsing for interactive natural language interface*. IEEE, 2001.

[42] C. P. Rose, A. Roque, and S. Francisco, "An Efficient Incremental Architecture for Robust Interpretation," in *In Proceedings of the Human Languages Technologies Conference*, 2002, pp. 307–312.

[43] F. Flippo, A. Krebs, and I. Marsic, "A framework for rapid development of multimodal interfaces," *Proceedings of the 5th international conference on Multimodal interfaces - ICMI '03*, p. 109, 2003.

[44] A. Jaimes and N. Sebe, "Multimodal human-computer interaction: A survey," *Computer Vision and Image Understanding*, vol. 108, no. 1-2, pp. 116–134, 2007.

[45] B. Dumas, D. Lalanne, and S. Oviatt, "Multimodal interfaces: A survey of principles, models and frameworks," *Human Machine Interaction*, vol. 5440 LNCS, pp. 1–25, 2009.

[46] R. G. Simmons, "VHPOP : Versatile Heuristic Partial Order Planner," vol. 20, pp. 405–430, 2003.

[47] R. E. Fikes and N. J. Nhsson, "STRIPS : A New Approach to the Application of . Theorem Proving to Problem Solving '," vol. 2, 1971.

[48] D. Poole and A. K. Mackworth, *Artificial Intelligence - Foundations of Computational Agents*. Cambridge University Press, 2010.

[49] "VINNOVA - Verification for Growth," accessed: 1 October 2014. [Online]. Available: http://www.vinnova.se/en/Our-acitivities/Innovativeness-of-specific-target-groups/The-Knowledge-Triangle/VINN-Verification/

[50] "A-Focus AB," accessed: 1 October 2014. [Online]. Available: http://http//www.a-focus.se/

[51] C. Neville, "Introduction to research and research methods," July 2007.

[52] G. F. Rossano, C. Martinez, M. Hedelind, S. Murphy, and T. a. Fuhlbrigge, "Easy robot programming concepts: An industrial perspective," *2013 IEEE International Conference on Automation Science and Engineering (CASE)*, pp. 1119–1126, Aug. 2013.

[53] “Morpha,” accessed: 21 October 2014. [Online]. Available: http://www.morpha.de/download/flyer_englisch_Stand_Feb2002.pdf

[54] “LabVIEW,” accessed: 21 October 2014. [Online]. Available: http://www.ni.com/labview

[55] “MotoSim EG,” accessed: 21 October 2014. [Online]. Available: http://www.motoman.com/datasheets/MotoSim%20EG.pdf

[56] “ROBOTGUIDE FANUC Robotics Simulation Software,” accessed 21 October 2014. [Online]. Available: http://www.fanucrobotics.de/en/products/software/simulation%20and%20development/roboguide

[57] “Universal Robots,” accessed: 21 October 2014. [Online]. Available: http://media1.limitless.dk/UR_Manual/UR5_User_Manual/UR5_User_Manual_GB.pdf

[58] “IEEE Spectrum, How Rethink Robotics Built Its New Baxter Robot Worker,” accessed: 21 October 2014. [Online]. Available: http://spectrum.ieee.org/robotics/industrial-robots/rethink-robotics-baxter-robot-factory-worker

[59] J. N. Pires, G. Veiga, and R. Araújo, “Programming-by-demonstration in the coworker scenario for SMEs,” *Industrial Robot: An International Journal*, vol. 36, no. 1, pp. 73–83, 2009.

[60] J. N. Pires, “Robot-by-voice : Experiments on commanding an industrial robot using the human voice,” vol. 32, no. 6, 2005.

[61] R. Marin, P. Sanz, and J. Sanchez, “A very high level interface to teleoperate a robot via Web including augmented reality,” *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, no. May, pp. 2725–2730, 2002.

[62] R. Marin, P. Sanz, P. Nebot, and R. Wirz, “A Multimodal Interface to Control a Robot Arm via the Web: A Case Study on Remote Programming,” *IEEE Transactions on Industrial Electronics*, vol. 52, no. 6, pp. 1506–1520, Dec. 2005.

[63] C. Elting, S. Rapp, G. Möhler, and M. Strube, “Architecture and implementation of multimodal plug and play,” *Proceedings of the 5th international conference on Multimodal interfaces - ICMI ’03*, p. 93, 2003.

[64] M. Turunen and J. Hakulinen, "JASPIS 2 - An Architecture for Supporting Distributed Spoken Dialogues," in *Eurospeech*, 2003.

[65] K. Katsurada, Y. Nakamura, H. Yamada, and T. Nitta, "XISL: a language for describing multimodal interaction scenarios," in *Proceedings of the 5th international conference on Multimodal interfaces*. ACM, 2003, pp. 281–284.

[66] M. Araki and K. Tachibana, "Multimodal dialog description language for rapid system development," *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue - SigDIAL '06*, no. July, p. 109, 2006.

[67] J. A. Larson and T. V. Raman, "W3C Multimodal Interaction Framework," http://www.w3.org/TR/mmi-framework, Dec., 2002, Accessed: 21 October 2014. [Online]. Available: http://www.w3.org/TR/mmi-framework

[68] M. Johnston, "Building multimodal applications with EMMA," *Proceedings of the 2009 international conference on Multimodal interfaces - ICMI-MLMI '09*, p. 47, 2009.

[69] S. Sire and S. Chatty, "The Markup Way to Multimodal Toolkits," in *W3C Multimodal Interaction Workshop*, Sophia Antipolis, France, 2004.

[70] M. Johnston, P. R. Cohen, D. McGee, S. L. Oviatt, J. a. Pittman, and I. Smith, "Unification-based multimodal integration," *Proceedings of the 35th annual meeting on Association for Computational Linguistics -*, pp. 281–288, 1997.

[71] M. Johnston, "Unification-based multimodal parsing," *Annual Meeting of the ACL*, p. 624, 1998.

[72] M. Johnston and S. Bangalore, "Finite-state multimodal integration and understanding," *Natural Language Engineering*, vol. 11, no. 02, pp. 159–187, 2005.

[73] R. Stiefelhagen, C. Fogen, P. Gieselmann, H. Holzapfel, K. Nickel, and a. Waibel, "Natural human-robot interaction using speech, head pose and gestures," *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, pp. 2422–2427, 2004.

[74] S. Irawati, D. Calderón, and H. Ko, "Spatial ontology for semantic integration in 3D multimodal interaction framework," in *Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*, vol. 1, no. June. New York, New York, USA: ACM, 2006, pp. 129–135.

[75] D. Schlangen and G. Skantze, "A general, abstract model of incremental dialogue processing," in *European Chapter Meeting of the ACL*, 2009, pp. 710–718.

[76] A. Olwal, J. Gustafsson, and C. Lindfors, "Spatial augmented reality on industrial CNC-machines," *Proceedings of SPIE*, vol. 6804, pp. 680 409–680 409–9, 2008.

[77] H. Fang, S. K. Ong, and A. Y.-C. Nee, *Robot Programming Using Augmented Reality*. IEEE, Sep. 2009.

[78] A. de Robótica and A. de La Producción, "An augmented reality interface for training robotics through the web," *Communication*, pp. 189–194, 2009.

[79] T. Pettersen, J. Pretlove, C. Skourup, T. Engedal, and T. Lokstad, "Augmented reality for programming industrial robots," *The Second IEEE and ACM International Symposium on Mixed and Augmented Reality, 2003. Proceedings.*, pp. 319–320, 2006.

[80] J. Chong, S. Ong, a.Y.C. Nee, and K. Youcef-Youmi, "Robot programming using augmented reality: An interactive method for planning collision-free paths," *Robotics and Computer-Integrated Manufacturing*, vol. 25, no. 3, pp. 689–701, Jun. 2009.

[81] M. Dawande, H. N. Geismar, S. P. Sethi, and C. Sriskandarajah, "Sequencing and Scheduling in Robotic Cells: Recent Developments," *Journal of Scheduling*, vol. 8, no. 5, pp. 387–426, Oct. 2005.

[82] H. N. Geismar, M. Pinedo, and C. Sriskandarajah, "Robotic cells with parallel machines and multiple dual gripper robots: a comparative overview," *IIE Transactions*, vol. 40, no. 12, pp. 1211–1227, Oct. 2008.

[83] J. P.-o. Fan and G. K. Winley, "A Heuristic Search Algorithm for Flow-Shop Scheduling," vol. 32, pp. 453–464, 2008.

[84] O. Kon, "Event-based MILP models for resource-constrained project scheduling problems," vol. 1, pp. 3–13, 2011.

[85] A. L. Blum and M. L. Furst, "Fast Planning Through Planning Graph Analysis ," pp. 1–20, 1997.

[86] S. Richter, M. Westphal, and M. Helmert, "LAMA 2008 and 2011," *The 2011 International . . .* , 2011.

[87] I. B. M. T. J. Watson, Y. Heights, J. S. Penberthy, and D. S. Weld, "UCPOP : A Sound , Complete , Partial Order Planner for ADL," 1991.

[88] A. Coles, A. Coles, M. Fox, and D. Long, "Forward-Chaining Partial-Order Planning," 2010.

[89] S. Richter and M. Westphal, "The LAMA Planner : Guiding Cost-Based Anytime Planning with Landmarks," vol. 39, pp. 127–177, 2010.

[90] K.-y. Hsiao, S. Vosoughi, S. Tellex, R. Kubat, and D. Roy, "Object schemas for responsive robotic language use," *Proceedings of the 3rd international conference on Human robot interaction - HRI '08*, p. 233, 2008.

[91] M. Johnston and S. Bangalore, "Finite-state multimodal parsing and understanding," *International Conference On Computational Linguistics*, p. 369, 2000.

[92] "C# Language specification," http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-334.pdf, accessed: 21 October 2014.

[93] J. Wielemaker, "An overview of the {SWI-Prolog} Programming Environment," in *Proceedings of the 13th International Workshop on Logic Programming Environments*, F. Mesnard and A. Serebenik, Eds. Heverlee, Belgium: Katholieke Universiteit Leuven, Dec. 2003, pp. 1–16.