Technische Universität München
Fakultät für Maschinenwesen
Lehrstuhl für Automatisierung und Informationssysteme
Prof. Dr.-Ing. Birgit Vogel-Heuser

Bachelor's Thesis

Concept and development of a human robot interface using parametrised Augmented Reality

Konzept und Entwicklung einer Mensch-Roboter Schnittstelle mit parametrisierbarer Augmented Reality (GERMAN)

Author:	Eric Vollenweider
Supervisor:	Prof. Dr.-Ing. Birgit Vogel-Heuser
Advisor:	Dr. Pantförder Dorothea
Issue date:	Februar 26, 2019
Submission date:	August 26, 2019

Table of Contents

1 Introduction 1
1.1 Reasons for- and problems with Human-Robot collaboration 1
1.2 Parametrised development as an approach to solve AR-developer shortage . . 2

2 State of the Art 3
2.1 Augmented Reality in a nutshell 3
2.1.1 AR-Standardisation 3
2.2 AR in Robotic applications 4
2.2.1 AR during development and testing 4
2.2.2 Operating robotic systems with AR support 5
2.3 Parametrised Development in other disciplines 5
2.4 Current methods programming industrial robots 6
2.5 Parametrised AR in Robotic applications 7

3 Concept 9
3.1 Goal 9
3.2 User Requirements 9
3.3 System Requirements 10
3.4 PARRHI Concept 11
3.4.1 Input Data 13
3.4.1.1 Variables 13
3.4.1.2 Points 13
3.4.1.3 Holograms 14
3.4.1.4 Events 15
3.4.2 Interpreter 17

4 Bibliography 19

1 Introduction

1.1 Reasons for- and problems with Human-Robot collaboration

Today's world is highly automated in each and every aspect of our lives. One of the most important advancements within the last 30 years where robots. They allow us to minimize costs, while maximizing output, quality and efficiency. With the rising need of agile production plants, classic robotic systems do not seem to fulfil all requirements anymore. Industrial robots do have numerous abilities but adapting to new situations easily is certainly not one of them. In contrast, the human's strength always was exactly that - to learn quickly. Thus, Human-Robot (HR) collaboration is an inevitable step towards the future of humanity.

Robotics needed to adapt to the new situation where humans and robots share a perimeter, thus compliant robots where developed. For machines to feel natural it is not enough to be compliant and not hurt humans. As Gary Klein et al. stated, communicating intent is a key issue in effective collaboration within teams [1].

Augmented Reality (AR) can help us to communicate with robots in three dimensions. With rapid advancements in the field of robotics during the last few years [2], innovative ways to program, develop and operate these highly complex systems need to be researched. Both robotics and AR have been heavily worked on for the last 20 years separately, but their combination has mostly remained untouched. Lately though, it has become a new focus area in research. While it is a known fact that AR can improve communication in certain use cases, the way how we develop these applications is not as clear. Most projects involving AR re-do most of the same steps over and over again, requiring software engineers along the way to do everything. Thus, the time-to-market is long and expenses are high.

According to the U.S. Department of Labour, there is a massive hunt for talent in the software industry [3]. The department reported an expected growth of over 30% within the next 10

years. One possible reason could be, that acquiring the necessary skills to develop software applications takes a long time and is not viewed as an easy path. Specifically AR applications currently require a wide variety of skills. On top of that, many processes and steps have to be repeated for each new AR-application and it is increasingly hard to build upon other peoples work.

The lack of programming talent is also seen in the robotics industry, which requires its developers to posses an even wider set of skills, including software development, mechanical and the basics of electrical engineering. Due to the lack of developers the speed of innovation and time-to-market is slowed as stated by multiple companies worldwide. [Citation maybe?]

1.2 Parametrised development as an approach to solve AR-developer shortage

Today, most industrial robots are only parametrised and not actually programmed, which results in lower requirements for employees and thus in less time and money spent on training. Employees only set certain locations, points and actions utilizing an easy to understand User-Interface. Applying a similar methodology to the development of Augmented Reality applications could lead to similar effects. If non-programmers could setup, configure and thus develop Human-Robot interfaces involving AR and robot controlling all by themselves, without the need to write a single line of code, the cost efficiency and speed of innovation can be improved.

This bachelor thesis aims to research the feasibility and practicability of such an abstract development environment. First there will be a ”State of the Art” section, where current research and progress is displayed. From there some gaps will be identified, a concept proposed, implemented and tested in a real application.

2 State of the Art

2.1 Augmented Reality in a nutshell

AR technology must not be confused with Virtual Reality (VR). Whereas VR completely immerses users, the former still allows the real world to be seen together with superimposed 3D objects projected into the user's view. T. Azuma crafted a very general definition of AR-Systems very early on [4]. He wrote that, in order for a system to be classified as an AR system, it has to fulfil three characteristics. Namely, the system has to combine real and virtual objects, be interactive in real time and spatially map physical and virtual objects to each other. The term Augmented Reality is often used as a synonym for the also commonly used expression "Mixed Reality".

There are various types of AR devices and principles. In this bachelor's thesis the term AR should be understood as head-mounted superimposing Augmented Reality devices, such as the Microsoft HoloLens, that actually create the illusion of three dimensional holograms being projected into the real world.

2.1.1 AR-Standardisation

C. Perey et. al [5] examined existing standards and some best-practice methods for AR. They clearly identified some gaps in the AR value-chain. The researchers lay out numerous standards for low level implementations such as geo-graphic location tracking, image tracking, network data transmission etc.

Figueroa et. al [6] researched about a "Conceptual Model and Specification Language for Mixed Reality Interface Components". They want to lay a foundation for other developers to standardize 3D interface assets, for others to build upon. The team developed an xml based data structure called "3DIC" that defines the look and some smaller behaviour actions of UI control elements. Although their work goes into the right direction, it lacks certain features

that robot-human AR interfaces need, and contains unwanted components like pseudo-code. ”3DIC” is a specification language that cannot be fully processed automatically, since pseudo-code is not executable.

Concluding from this general overview, there does not seem to be an existing standard for HR Interfaces utilizing parametrised Augmented Reality, that allow the development of complete HR-AR-applications that can actually be processed, executed and used.

2.2 AR in Robotic applications

2.2.1 AR during development and testing

Wolfgang Hönig et al. examined three different use cases of Augmented Reality [7]. The research team primarily focused on the benefits offered by the co-existence of virtual and real objects during the development and testing phase. They documented three individual projects where the implementation of AR as a main feature resulted in lowered safety risks, simplified debugging and the possibility of easily modifying the actual, physical setup.

Another important dimension of development and testing is of financial nature: upfront investments and operating costs. For example, robotic aerial swarms tend to be quite expensive due to the high cost of drones. Hönig et al. successfully scaled up the number of objects in their swarms without adding physical hardware, thus, saving money and space [7]. However, it is stated that this approach might not be applicable to all experiments since simulations are never perfect replicas of actual systems. This small delta in physical behaviour might be enough to raise doubts regarding the correctness of the experiment results.

All projects by Hönig et al. focus on isolating certain aspects of the system to analyse and test them more flexibly, less expensively or with improved safety for all participants involved (humans and machines). Similarly, Wünsche et al. have created a software framework for simulating certain parts of robotic systems [8]. These researchers from Auckland worked on methods to combine real world and simulated sensor data and navigate a real-world robot in the combined environment.

2.2.2 Operating robotic systems with AR support

A well-known bottleneck in robotics is the controlling of and thus, the communication with robots. In all previously cited cases, Augmented Reality was not used to enhance the interaction between humans and robots but to mitigate the current challenges in developing robotic systems. Very early work by Milgram et al. [9] shows that even the most basic implementations of AR technology, with the objective to improve the information exchange, enhance the bidirectional communication in multiple ways. The team proposed means to relieve human operators by releasing them from the direct control loop and using virtually placed objects as controlling input parameters. This replaces direct control with a more general command process.

As Gary Klein et al. stated, communicating intent is a key issue in effective collaboration within teams [1]. Whenever robots and humans collaborate in a close manner, it is critically important to know each other's plans or strategies in order to align and coordinate joint actions. For machines lacking anthropomorphic and zoomorphic features, such as aerial and industrial robots, it is unclear how to communicate the before-mentioned information in natural ways.

In order to solve this problem, Walker et al. [10] explored numerous methods to utilize Augmented Reality to improve both efficiency and acceptance of robot-human collaborations via conveying the robot's intent. The group of researchers defined four methods of doing so, with varying importance being put on "information conveyed, information precision, generalizability and possibility for distraction" [10]. The conclusion was that spatial Augmented Reality holograms are received much more intuitively than simple 2D projected interfaces.

2.3 Parametrised Development in other disciplines

Numerous disciplines utilize parametrised development environments heavily. For example the CAD software *CADENCE* (integrated electrical circuits) offers parametrised cells to optimize the development process [11]. Users can place these cells and adjust certain parameters. Generally there is no coding skill required.

In architecture Parametric Design is taking overhand since the late 2000s. New capabilities in computer rendering and modelling opened up a whole new world for architects at the time. Using parametrised mathematical formulas with boundary constrains allows architects to generate building structures easier and more efficient [12]. Adapting to a changing

environment during the design process is much easier, since a substantial portion of work can be completed by algorithms and software applications. Additionally the reusability of components increases massively due to the very formal way of representation. Architecture studios also begin to include Virtual and Augmented Reality technologies actively in their design processes to further incorporate and better understand these new design styles called *Parametric Design*.

2.4 Current methods programming industrial robots

Almost all industrial robots are mainly programmed in three distinct ways.

1.) Teaching Pendant

2.) Simulation / Offline Programming

3.) Teaching by Demonstration

According to the British Automation & Robot Association over 90% of all industrial robots are programmed using the Teach Pendant method [13]. In the case of Fanuc controllers they basically are touch-tablets fitted for industrial use. They allow the operator to jog the robot into certain configurations, save the positions and offer a number of other possibilities. For easier operations Fanuc offers specific User Interfaces where the operator simply enters parameters and positions. More complex tasks can also be completed with the Teach Pendant, but I will not go into detail here.

Teach Pendants are great for trivial and simple tasks, that do not require lots of collaboration between factory components. Reprogramming the robot leads to downtime, since the actual real robot has to be used. Important to note is, that parametrised programming already is an industry standard in industrial robotics.

The other two methods are used for more complex tasks and production lines and each have their Pro's and Con's. While Offline Programming is very precise and powerful, it does require a substantial amount of schooling time for the operator. Teaching by Demonstration is the exact opposite. Everybody could succeed quickly, but the complexity of achievable tasks and the precision is limited.

2.5 Parametrised AR in Robotic applications

Many industries adopted Augmented Reality in their daily process already. It's advantages and challenges are clearly known and being worked on. Developing AR applications is generally speaking still a very costly task. On the other hand there is the design principle of "Parametrisation" that may involve increased initial investment but can lower the cost of changes and adaptions to changing environments. Combining the perks of both AR technology and parametrised architecture may bear great potential.

3 Concept

For further reading the to be built system will be called ”Parametrised Augmented Reality Robot Human Interface” (*PARRHI*).

3.1 Goal

As described in section 1.2 this thesis presents a possible approach to solve the previously described problems (see section 1.1). It is the main target of this bachelor's thesis to remove the necessity of high software engineering skills to develop reasonably complex Augmented Reality Robot Human Interface applications, maintaining the quality of the outcome and even increasing the degree of resuability. This might result in lower development costs, lower struggle to gather software engineering talent and even in a shorter time to market.

To succeed in this goal, a specific set of requirements has to be defined and documented in a formal way. To gather these requirements the V-Model developed by the Federal Republic of Germany was used [14].

3.2 User Requirements

Before defining the User Requirements the system's end user has to be defined. The characteristics of the actual enduser of *PARRHI* might be someone who:

- Knows the basics of text editing software,
- has no qualifications in software engineering,
- wants to develop an Augmented Reality application for professionals that collaborate with industrial robots in a shared perimeter.

Having an idea of the end-user, the user-requirements can be defined. The user wants to:

- Develop a AR-applications without software engineering skills
- Have the tools necessary to create medium complex applications for use cases such as tutorials, maintenance instructions or other teaching purposes
- Build upon other people's work or projects
- Launch the AR-application on a suitable device
- Possibly use the same system on different types and brands of robots

3.3 System Requirements

Deriving from the previous chapter the System should:

1.) have input data in a non-code format.

2.) have readable and intuitive feedback on the input data.

3.) achieve reusablity by having an input in non-binary text format that allows copy and paste reproduction.

4.) support building for hand held mobile devices and head mounted Augmented Reality glasses.

5.) allow bidirectional communication (Sending commands and receiving data) with the robot's controller

6.) allow documentation of the applications workflow

7.) be as platform independent as possible

3.4 PARRHI Concept

As discussed parametrising workflows grants a multitude of benefits (see section 2.3). Maintainability and Reusability are just some of them. By giving a system knowledge about the real world, users are relieved from some portion of the workload. In *PARRHI's* case, this could mean include information about the robot's kinematics. Both, the developer of PARRHI-Applications and the end user do not have to understand robot kinematics in order to build and use costly AR applications.

First, *PARRHI* possesses a predefined set of model knowledge about the real world and the Fanuc Robot. Since interfaces are meant to interact with the contexts it connects, the system furthermore has ways to receive information from and act on the augmented and real world. Its information input during runtime is the robot's momentary configuration and the AR-Device's coordinates. *PARRHI* can use the received information in combination with all its knowledge for its parametrised definitions. The defined behaviour uses its parametrised objects to generate some output, which can either be some augmented holograms presented to the user, or actual physical acts like directly commanding the robot.

Of course the instructions to the user and into the physical world my directly influence the augmented reality space due to the feedback loops in the system. If something triggers a command that moves the robot, its movement can be tracked by the systems ability to receive external information.

How do I mean parametrised? How is the system built (basic level)? What is parametrised in my specific case? How does the system interact with the real world?

PARRHI basically consists of three components (see Fig. 3.1):

1.) Input Data

2.) PARRHI Engine

3.) the "real world"

Input Data file, which is created by the application's developer, represents *PARRHI's* only input. It contains a set of definitions, commands and behaviour rules that are necessary to create AR Interfaces that is able to fulfil the defined requirements in 3.3.

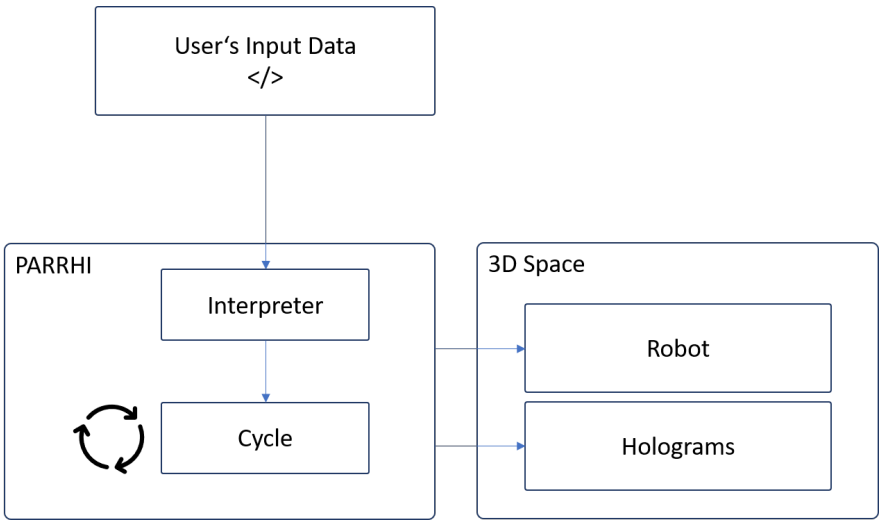

Figure 3.1: PARRHI general concept

The *PARRHI Engine* interprets the input data and acts on its behalf. After validating the input data against certain schemes the engine imports all contents and sets up its internal states, before entering a cyclic mode - ready for operation.

3.4.1 Input Data

The Input data is the document, which the end user of the *PARRHI* system constructs. It contains parametrised, hierarchically structured data, that defines the behaviour, feel and look of the final AR-HR-application. The following chapters explain the set of tools that are available to the developer, how the work and interconnect.

Table 3.1 displays the top level structure and its main parts, where Variables, Points and Holograms define Utilities that Events can work with.

Table 3.1: *Input Data* structure

Name	Section	Explanation
Variables	3.4.1.1	Integer variables to create state machines
Points	3.4.1.2	Different kinds of 3D Point definitions (fix, relative to the robot, relative to the user)
Holograms	3.4.1.3	Holograms can be mounted onto points and have a set of properties
Events	3.4.1.4	Events have certain triggers and carry two Actions as a payload

3.4.1.1 Variables

PARRHI's variables can be used to create different steps in one's application. Variables can be the source of an event's trigger or the target of an event's action (see section 3.4.1.4).

3.4.1.2 Points

Points are a three dimensional vectors that can be defined in three different ways.

1.) Fix-Point

2.) Robot-Point

3.) Camera-Point

The **Fix-Point** has static coordinates. It could be used to setup holograms that visualize certain spacial environmental constraints or for different steps in a *PARRHI* AR-HR-Interface application (see figure 3.2).

To interact with the robot, *PARRHI* needs a way to read and display data from the robot. For that, there are **Robot-Points** that are defined by two indexes of the robot's joints and one scalar value that defines the exact location between them (see figure 3.3). The position is calculated as follows (with s being the scalar value and J_n the position vector of Joint n):

$$\boldsymbol{P} = \boldsymbol{J_1} + (\boldsymbol{J_2} - \boldsymbol{J_1}) * s \tag{3.1}$$

Camera-Points are a way to involve the user's position into the application. Its coordinates are periodically updated with the AR-Device's spacial position in each cycle.

3.4.1.3 Holograms

To actually augment the reality *PARRHI* needs holograms. These elements have a number of attributes. Holograms can be active (respectively also inactive), have a render-mode, a geometric definition and of course a location. The locations are defined by a certain number of points - depending on the hologram's geometry. Currently the *PARRHI* supports two types of holograms. There are spheres and cylinders, respectively taking one or two points and a radius as input values to define its size and position.

A Sphere's centre is always set to the point it was defined with, whereas a cylinder always connects the two points of its definition. As input points, all described types in section 3.4.1.2 can be used. With a given radius and point the three dimensional figure is completely defined.

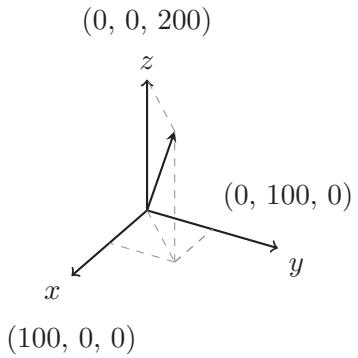

Figure 3.2: Fix-Point example

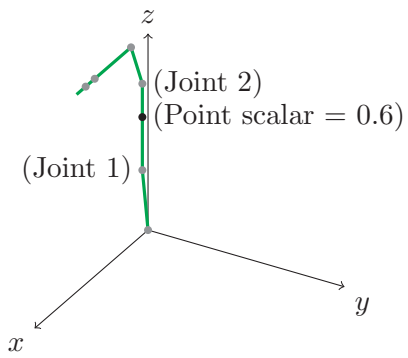

Figure 3.3: Robot-Point example

Holograms have an attribute called *renderMode*, which if set to "transparent", renders the hologram in a half transparent way, allowing holograms to be used for boundary or zone visualizations. Furthermore the visibility of holograms can be changed by actions as described in section 3.4.1.4.

3.4.1.4 Events

All previous elements (variables, points and holograms) exist to define the scene and to set up assets that can be utilized by events, which now actually describe the application's workflow. To do so, there are two subtypes in this category. There are event-triggers and event-actions (or short *triggers* and *actions*). Triggers have a boolean expression, which is checked periodically. As soon as the boolean expression evaluates to *true*, the attached actions will be executed and the trigger will be disabled, avoiding multiple executions. One could say that if triggers the *PARRHI's* sensors, actions are its tools to act on the augmentation.

Triggers can be enabled and disabled, either from the beginning on, or toggled by an action. Every trigger has at least one action as a payload. To reach a reasonably capability numerous different but easy to understand triggers are available for the application's developer. In table 3.2 is a complete list of all defined triggers. *(Note, that the enabled/disabled flag and actions are omitted in this table)*

Distance trigger can be used for two main purposes. First the application can use the robot's movement as an input using a *Robot-Point*. An action could be triggered as soon as the user jogged the robot's TCP into wished position by using a *Fix-Point* an waiting for them to come close to each other or to monitor the robot's configuration in Joint-Space by using two *Robot-Points* as input parameters. Second the user's movement can be monitored by utilizing a *Camera-Point* as an input. The application can thus ask the user to move to a specific location.

Table 3.2: Event Triggers

Name	Input Parameter	Trigger expression
Distance trigger	Two Points $\boldsymbol{P_1}, \boldsymbol{P_2}$, distance d	$\lvert\boldsymbol{P_2} - \boldsymbol{P_1}\rvert \leq d$
Variable trigger	Variable v, trigger value v_{tr}	$v = v_{\text{tr}}$
Time trigger	trigger time t_{tr}, time since enabling t_{enabled}	$t_{\text{tr}} \geq t_{\text{enabled}}$

With **Variable triggers** the application is able to use defined variables as triggers. One could implement a counter for certain events, and trigger an action when a threshold value is reached. It can also be used for workflows that need states or steps. Finally the **Time trigger** allows the application to involve timers. The user could be given a maximum time for a task or holograms can be hidden after a few seconds.

Whenever a trigger's boolean expression evaluates to true, its actions are invoked and the trigger gets disabled. Since actions are the only way *PARRHI* can influence the augmentation, there are numerous different types of actions - each serving a general purpose to fulfil the defined requirements (see section 3.3).

As with triggers, actions have a set of input parameters they need to fulfil their task. As with all other input data objects discussed so far, actions have a unique ID. The table 3.3 gives a quick overview about all actions that *PARRHI* currently supports.

Increment-counter actions increment their integer variable by 1. If a developer wanted to count the number of times a user jogged the robot into a specific region, an Increment-counter action could be used as a payload of a Distance trigger. After a threshold value is reached, a Variable trigger could change the UI text and display a hint.

The **Set-Hologram-State action** enables hiding and showing holograms at runtime. If a hologram represents a region for a tutorial step, it can be hidden after the user's task is completed. The new scene can then be setup by displaying new holograms that guide the user's way. Another possible use would be, to display a warning boundary, if the user moves into a forbidden zone. This can be achieved by combining Distance trigger and Set-Hologram-State actions.

Table 3.3: Event Actions

Action Name	Input Parameter	Explanation
Increment Counter	Variable v	Increments the value of v by 1
Set Hologram State	Hologram-IDs, State to set	Enables/disables all specified holograms
Set Trigger State	Trigger-IDs, State to set	Enables/disables all specified triggers
Change UI Text	Text to set	Sets the UI Text
Move Robot	Point P	Moves the robot to P
Set robot-hand State	State to set (open/close)	Opens or closes the robot's gripper

When using *PARRHI* the user is presented a GUI that shows text and some other few options. The **Change-UI-Text action** allows to change this displayed text. There are numerous obvious use-cases where this is useful. Whenever it is of value to inform the user about something that cannot be achieved by holograms, this is a simple way to do so.

To create meaningful and longer applications, enabling and disabling triggers is an essential tool. This is what the **Set-Trigger-State action** is for. Triggers can only invoke their payload actions, if they are active. Triggers can either be defined as disabled from the beginning on, or get disabled by triggering as described above. The Set-Trigger-State action has the ability to (re)activate disabled triggers. There is a speciality in the case of *Time triggers*. Their inner timer starts ticking, whenever they get enabled. This allows for timers to be used in the middle of applications, relative to other events.

3.4.2 Interpreter

4 Bibliography

[1] G. Klein, P. J. Feltovich, J. M. Bradshaw, and D. D. Woods, "Common ground and coordination in joint activity", *Organizational simulation*, vol. 53, pp. 139–184, 2005.

[2] C. Laschi, B. Mazzolai, and M. Cianchetti, "Soft robotics: Technologies and systems pushing the boundaries of robot abilities", *Sci. Robot.*, vol. 1, no. 1, eaah3690, 2016.

[3] U. D. o. L. Bureau of Labor Statistics. (2018). Occupational outlook handbook, software developers, [Online]. Available: `https://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm` (visited on 03/28/2019).

[4] R. T. Azuma, "A survey of augmented reality", *Presence: Teleoperators & Virtual Environments*, vol. 6, no. 4, pp. 355–385, 1997.

[5] C. Perey, T. Engelke, and C. Reed, "Current status of standards for augmented reality", in *Recent Trends of Mobile Collaborative Augmented Reality Systems*, Springer, 2011, pp. 21–38.

[6] P. Figueroa, R. Dachselt, and I. Lindt, "A conceptual model and specification language for mixed reality interface components", in *VR 2006, In Proc. of the Workshop "Specification of Mixed Reality User Interfaces: Approaches, Languages, Standardization*, 2006, pp. 4–11.

[7] W. Hoenig, C. Milanes, L. Scaria, T. Phan, M. Bolas, and N. Ayanian, "Mixed reality for robotics", in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2015, pp. 5382–5387.

[8] I. Y.-H. Chen, B. MacDonald, and B. Wunsche, "Mixed reality simulation for mobile robots", in *2009 IEEE International Conference on Robotics and Automation*, IEEE, 2009, pp. 232–237.

[9] P. Milgram, S. Zhai, D. Drascic, and J. Grodski, "Applications of augmented reality for human-robot communication", in *Proceedings of 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'93)*, IEEE, vol. 3, 1993, pp. 1467–1472.

[10] M. Walker, H. Hedayati, J. Lee, and D. Szafir, "Communicating robot motion intent with augmented reality", in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, ACM, 2018, pp. 316–324.

[11] V. Grozdanov, D. Pukneva, and M Hristov, "Development of parameterized cell of spiral inductor using skill language", Apr. 2019.

[12] M. Stavric and O. Marina, "Parametric modeling for advanced architecture", *International journal of applied mathematics and informatics*, vol. 5, no. 1, pp. 9–16, 2011.

[13] B. A. R. Associtation. (2018). Robot programming methods, [Online]. Available: `http://www.bara.org.uk/robots/robot-programming-methods.html` (visited on 04/07/2019).

[14] M. Broy and A. Rausch, "Das neue v-modell® xt", *Informatik-Spektrum*, vol. 28, no. 3, pp. 220–229, 2005.

Declaration

The submitted thesis was supervised by Prof. Dr.-Ing. Birgit Vogel-Heuser.

Affirmation
Hereby, I affirm that I am the sole author of this thesis. To the best of my knowledge, I affirm that this thesis does not infringe upon anyone's copyright nor violate any proprietary rights. I affirm that any ideas, techniques, quotations, or any other material, are in accordance with standard referencing practices.

Moreover, I affirm that, so far, the thesis is not forwarded to a third party nor is it published. I obeyed all study regulations of the Technical University of Munich.

Remarks about the internet
Throughout the work, the internet was used for research and verification. Many of the keywords provided herein, references and other information can be verified on the internet. However, no sources are given, because all statements made in this work are fully covered by the cited literature sources.

Munich, December 1, 2016

YOUR NAME