Robot Programming: Background and Classification of Robot Programming Techniques

Hassen Smaoui
hsmaoui@mail.uni-paderborn.de

University of Paderborn

Abstract. In modern industries, robots are increasingly needed to achieve industry competitiveness. The integration of robots allows the automation of production, an idea certainly intriguing and attractive for any company. However, this automation requires the programming of robots. Over the years, several robot programming techniques have emerged. However, most of these techniques are complex and time-consuming, making them unsuitable for small and medium-sized enterprises (SMEs) whose product lines are constantly changing. Consequently, this paper reviews the literature on the background of robot programming as well as on existing robot programming methods and contributes to the classification of these methods according to certain classification criteria derived from the literature. This classification allows a clearer vision of the programming techniques which are best suited to a company, according to its processes, its robots and its needs/requirements. This paper also highlights both novel robot programming techniques of augmented reality-assisted robot programming and virtual reality-assisted robot programming.

Keywords: Robot Programming · Robot Programming Techniques · Classification

1 Introduction

In today's demanding and competitive international marketplace, industries are facing increasing dynamics of innovation, shortened product life cycles and continuously changing product lines. Besides, industries' success relies heavily on the quality of its products. On the other hand, industries are under the pressure of finding skilled workers at reasonable costs.
In recent years, robots have become key elements in achieving that competitiveness of the industry. They are used in a wide variety and almost all fields. They are mainly employed for carrying out repetitive and/or hazardous tasks in order to reduce human effort and allow workers to focus on higher value tasks. Due to their high speed and precision that surpasses that of human workers, they are also able to increase the productivity and quality of the products.
Robot integration involves automating the relevant industries' processes, a certainly intriguing and appealing idea for any enterprise in current global market

situation. In order to achieve that automation, robots need to be programmed. Over the years, several robot programming techniques emerged. Nonetheless, most of these techniques are complex, time-consuming and result in high costs. Therefore, they are mostly only suitable for large enterprises.
Small and medium-sized enterprises (SMEs), on the other hand, are characterized by their rapidly and continuously changing product lines. The complexity of some robot programming techniques prevents robot automation from spreading in SMEs, where mid/small productions does not allow for such costly and time consuming setups. Therefore, SMEs require techniques that enable operators to quickly and intuitively create robot programs for small batch production. Note that SMEs represent 99 per cent of all enterprises [20].
Recently, during the VARobot (Virtual and Augmented Reality Assisted Robot Programming) PG (Project Group), the questions of what robot programming is, what robot programming techniques exist, whether they can be classified and what the advantages and disadvantages of these programming methods are, have arisen. Therefore, to answer these questions, this paper reviews the literature on the background of robot programming as well as on existing robot programming techniques and the different types to which each technique may belong.
This paper therefore contributes to the understanding of robot programming, robot programming techniques, its historical background and the progress made over the years. This paper also contributes to the classification of robot programming techniques according to certain classification criteria derived from the literature. This classification and analysis of each robot programming category allows a clearer vision of the robot programming technique to be used for which process, robot, situation and requirements. This paper also highlights the importance of robot programming techniques based on novel interaction techniques, namely augmented reality and virtual reality.
Taking all these points into consideration, in Chapter 2, this paper first defines robots and reviews their main areas of application. We also provide an overview of the VARobot PG and its main objectives. Note that the VARobot PG perspective is taken into account in all the chapters of this paper.
In the next chapter, we review literature on which robot programming techniques exist and whether these can be classified. We investigate and discuss the main categories of robot programming techniques considered in literature to finally derive our own classification criteria.
After having understood the main existing robot programming approaches, including their advantages, their drawbacks, and when they are best suitable. In Chapter 4, we then focus on robot programming techniques that are more likely suitable for SMEs, namely augmented reality-assisted robot programming and virtual reality-assisted robot programming, as these are the ones considered in the VARobot PG. Finally, in Chapter 5, we present possible future work.

2 Background

2.1 Robots and Their Applications

The Oxford English Dictionary definition of a robot is "a machine capable of carrying out a complex series of actions automatically, especially one programmable by a computer." [2]
There exist different types of robots. Mordechai Ben-Ari et al. (2018) [10] classify these in two possible ways.
Firstly, as described in Figure 1, the classification can be based on the environment in which the robot operates. In this case, Mordechai Ben-Ari et al. especially accentuate the difference between fixed and mobile robots. Fixed robots are often industrial robots that perform certain process-specific repetitive tasks, such as welding or painting in a well-defined work cell. On the other hand, mobile robots perform tasks in different, not well-defined environments. They therefore have different capabilities and application domains than those of fixed robots.
Secondly, they classify robots according to their application domain, as described in Figure 2. Here again we consider industrial robots, that perform production tasks in well-defined environments. By contrast, service robots assist human beings in their tasks excluding manufacturing operations and industrial automation applications. Industrial robots are mainly the robots considered in this paper.

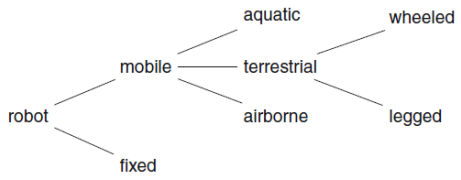

Fig. 1. Classification of robots by environment and mechanism of interaction [10]

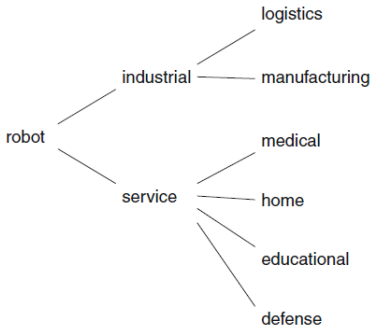

Fig. 2. Classification of robots by application field [10]

2.2 Virtual and Augmented Reality Assisted Robot Programming (VARobot) Project Group (PG)

The VARobot PG is a group of around ten master students in computer science supervised by Dr. Enes Yigitbas and Ivan Jovanovikj (M.Sc.) at the University of Paderborn. The main goal of the VARobot PG is to design and implement VR/AR interfaces and environments to assist robot programming regarding different aspects such as programming, debugging or documentation [3].
As explained in the previous two sections, robots are used in a wide variety of application areas. To make the most of robots, for example to eliminate tedious tasks and allow workers to focus on higher value tasks, the need for robotic automation has arisen. The operation and programming of robots is a difficult task that requires a deep field as well as programming knowledge. It is often a difficult and time-consuming task, which is not suitable for small and medium-sized enterprises (SMEs), because their product line is constantly changing and the robots reprogramming times and costs make of robotic automation an inappropriate solution.
The aim of this PG is to investigate programming methods, which are mainly based on novel interaction techniques, which can facilitate the task of operating and programming the robot. At the center of these new interaction techniques, augmented reality (AR) and virtual reality (VR), as well as AR / VR collaboration are mainly considered.

3 Classification Criteria

In this chapter, we review literature on different possible classification criteria for robot programming techniques. We investigate and discuss the main categories considered in literature to finally derive our own classification tree depicted in Figure 8.
In many literature sources [26, 12, 17, 19], robot programming methods are mainly classified to online robot programming methods, and Offline Robot Programming (OLRP) methods. Both techniques are first explained, analyzed, and compared in the following two sections. Besides, in the third section we derive an additional classification criterion. We finally summary and conclude in the last section.

3.1 Online Robot Programming

The online robot programming method mainly refers to, using the actual robot in order to develop the robot control program. This method requires therefore no direct control over the program code to be run on the robot. Alternatively, the robot control program is automatically generated based on information entered in the robot system by the robot operator in different possible approaches [11].
Two main approaches are often discussed in literature [26, 15, 8, 30] when investigating the online robot programming topic, namely, lead-through programming and walk-through programming.
In the following, both categories are explained.

Lead-through Programming

Method Description

Conventionally for lead-through programming -the most common way of robot programming [28, 14]-, as described in Figure 3, a skilled robot operator manually manipulates the robot movements and orientations and guides it through a desired path using a teach pendant at all stages of the task to be performed by the robot until the task is successfully completed. The teach pendant is considered the main human machine interface device for a robot system in this setting, [28] and often contains a joystick with two degrees of freedom or keypad in order to move the robot [14]. This process of manually moving the robot through a desired path via the teach pendant is often called "jogging". During this process, the robot controller records relevant robot configurations and specific points to later use these to generate the robot code that commands the robot to playback certain movements relative to the learned task [26].
The operator in this method is the one responsible for programming the robot, i.e. guiding the robot and maintaining the desired robot orientation and position in six degree-of-freedoms (DOFs). Figure 4 presents these DOFs.

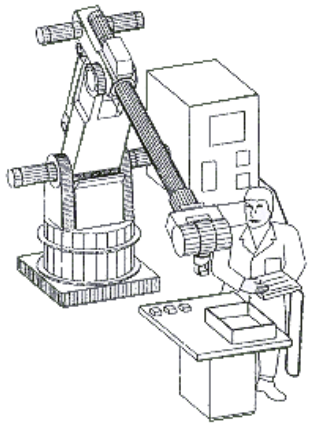

Fig. 3. Lead-through robot programming [1]

Discussion

Lead-through programming is a simple, intuitive, widely used robot programming method, that doesn't require neither high programming skills nor high initial costs, which makes them possibly the only robot programming choice for most SMEs [26]. Note that SMEs are characterized by their continuously changing product lines.
Although the concept of lead-through programming is simple, it has several

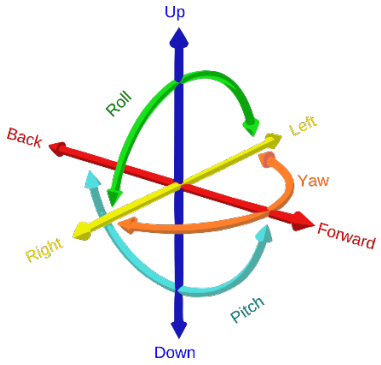

Fig. 4. The six degrees of freedom: forward/back, up/down, left/right, yaw, pitch, roll [4]

drawbacks.
Despite it being an efficient and cost effective solution for programming simple robotic system, the suitability of this method reduces as the process becomes more complex. If the process is complicated or the workpiece has a complex geometry, it becomes more difficult and time consuming to guide the robot through the desired path accurately while avoiding collisions with other objects in the workspace [26].
Since the robot operator in lead-through robot programming approaches is considered as the robot programmer, the quality of the program is directly influenced by the operator and is therefore limited by his skills. The safety of the operator is also to be considered, as during the teaching process, he is exposed to a hostile environment [26].
Lead-through programming lacks flexibility, as once the robot program is generated, it has to pass a lot of tests in order to assure its reliability. And once a fault in the generated robot program is detected, it is very difficult to make changes. Instead, the robot has to be reprogrammed again from scratch [26].
Reusability is also a big problem relative to lead-through programming, since the same robot for the same process has to be reprogrammed again from scratch for a slightly different workpiece [26].
The increased robot downtime is also a problem, as the robot cannot be used for production during the teaching/programming stage [26].
Another problem is guiding the robot and maintaining the desired robot orientation and position in six DOFs using a joystick with only two DOFs. This makes robot teaching a cumbersome and time consuming task [27].
To overcome these lead-through programming drawbacks and limitations, new lead-through programming approaches have been developed by researches from both academia and industries [14]. These are based on technological advancements regarding sensor and control technologies that may assist the operator performing complex robot motions more easily [26]. Although these new approaches may slightly differ from the system explained in the previous section,

all lead-through programming approaches share a similar concept as in Section 3.1.1, which explains their classification under this same category of robot programming techniques.

Walk-through Programming

Method Description

A typical walk-through programming approach consists of, as described in Figure 5, an operator that teaches the robot by guiding it once through the required process motions. Walk-through programming is in a way similar to the previously described lead-through robot programming method. However, it is based on the physical human-robot interaction. Instead of using a teach pendant, the operator physically "walks" the robot through the desired positions, as if he is doing the process himself [7].
More specifically, during the teaching phase where the operator manually moves the robot through the desired positions, the robot records these process specific positions and motions to then play them back autonomously. Due to a Force/Torque (F/T) sensor mounted on the robot, the robot also records the forces/torques executed by the operator on the robot during the teaching process [22]. This kind of programming is often also referred to as teaching.

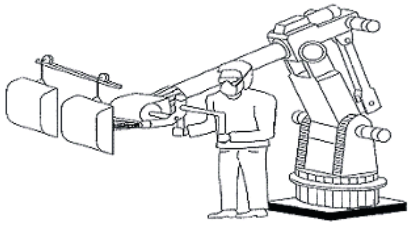

Fig. 5. Walk-through robot programming [1]

Discussion

Walk-through robot programming is a simple, intuitive and faster robot programming method that is based on a physical interaction between the human operator and the robot rather than other complex programming paradigms [9]. However, this programming method has several drawbacks.
As walk-through robot programming techniques require the operator to physically move the robot, this method is limited to relatively small-sized robots. Typically, during the teaching phase, the joint motors of the robot are not powered and left free with brakes turned off in order for the operator to be able to physically move the actual robot. Therefore, this method is not feasible with

big-sized robots or robots that can't allow joints to be released and powered down. Such robots are also difficult to move physically and one has to consider gravity force that requires constant force to maintain a certain desired position [7].
Walk-through robot programming as for lead-through methods also entail safety issues, since operators work in very close proximity to robots [12].
As for lead-through programming approaches, researches developed several walk-through approaches that may overcome one or several drawbacks of this programming method while still exploiting its benefits [17]. However, in this paper, we are interested in understanding the common main concept of this programming method that all newly developed approaches share. This allows for a classification of these approaches under the walk-through robot programming category.

3.2 Offline Robot Programming

The OLRP method refers to, as its name suggests and as described in Figure 6, developing the robot control program without using or accessing the robot during the development phase. However, robots are still needed for the final test and validation phase of the program [19]. The finished program is loaded into the robot afterwards.
G. Biggs and B. MacDonald (2003), researchers from the University of Auckland, New Zealand, further divide offline robot programming methods into two other categories, namely text-based and graphical techniques [11]. Both of these categories are explained in detail in the following.

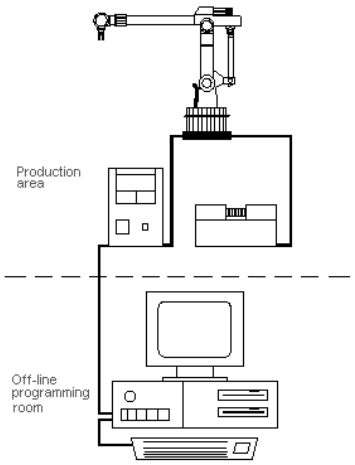

Fig. 6. Offline robot programming [1]

Text-based Techniques

Text-based offline robot programming methods, as its name suggests, are based on the use of traditional programming languages [11]. The user writes the program offline in a programming language then downloads it to the robot when it is ready. These programming languages vary from controller-specific languages, to generic procedural languages, and behavior-based languages. This adds additional criteria to further classify robot programming techniques. The difference between these three subcategories is explained in the following.

Controller-Specific Languages

Controller-specific languages refer to the use of the machine programming language provided by the robot manufacturer for that particular robot so that users can develop the corresponding robot programs.
These programming languages are often easy to use due to their basic syntax and simple commands. However, their biggest drawback lies in the fact that, for a company that is using different robots from different manufacturers, they have to deal with the programming overhead coming from learning to program with the different programming languages relative to the different robots, since there exists no universal standard between those programming languages. An alternative would be to pay the robot manufacturer to develop the required robot programs, which results in significantly increased development time and costs, if several robots from different manufacturers are used. Although, to overcome this problem, robot manufacturers try to provide predefined instructions that users can use in order to reduce development time, these instructions come with a loss in flexibility, since users are bound by specific applications [11].

Generic Procedural Languages

An alternative to controller-specific languages is the use of generic, high-level, multi-purpose programming languages, such as C++. These programming languages are generally extended with a robot abstraction, such as classes or methods that allow and simplify the access to common robot functions, such us commanding the robot to move a workpiece to a certain position. These programming languages can also provide higher-level abstractions, such as methods that use path planning to move the robot from one position to another [11].

Behavior-based Languages

Another alternative to both methods explained in the previous two sections is to program the robot based on descriptions that are based on behaviors and events. Instead of providing a procedural description, the aim of this method is to program and describe the way a robot reacts to different situations, or in other words, to specify actions to be triggered by certain events. These behavior-based languages often require less code than generic procedural languages. They are

also often used by robot developers rather than end users. The goal is to define robot functionalities that the end user would use to perform some tasks [11].

Graphical Techniques

Although G. Biggs and B. MacDonald (2003) limit graphical techniques to techniques using a graph, flow-chart, or diagram view of the robot system in order to provide a graphical medium for programming. Many literature sources [26, 12, 7, 17, 32, 19] consider graphical offline programming as offline programming methods that are based on the 3D Computer-aided design (CAD) model of the robot and the workpiece on which it operates, in order to generate and simulate robot programs, rather than using the actual robot as in online robot programming, or typing the program offline relying solely on a programming language. Z. Pan et al. (2010) [26] summarize in Figure 7 the key steps of offline programming that is based on the 3D CAD model of the complete robot work cell.
The starting point to virtually recreate the robot work cell is to load the robot model and the 3D CAD models that form the work cell, such as devices that the robot will handle. After loading these models, the OLRP method consists of extracting relevant information, such as robot position tags, from the CAD models features, such as corners and edges. These data are later used to strategize trajectories, without the physical presence of the robot and without the need of being in the working environment. These trajectories must nonetheless take into consideration many other details, such as assuring the reachability and avoiding collision. The user can then handle process related information (such as the cooperation of multiple robots to minimize cycle time), before generating a program that can be simulated, tested, and if need be fine-tuned for it to finally be ready to be downloaded to the robot controller.

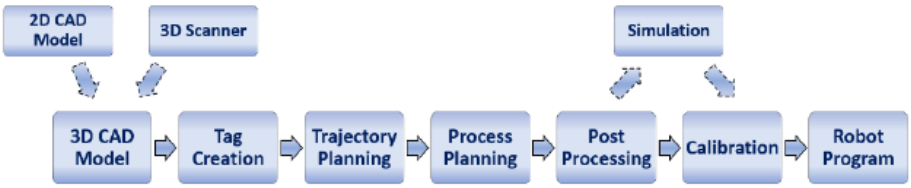

Fig. 7. Key steps of offline programming [26]

Discussion

OLRP has its strengths on programming complex systems and offers many advantages over the online robot programming method. It is proved to be more time-efficient and cost-effective. The production robot down time may be reduced by as much as 85 per cent [32], since the programming process does not require the physical presence of the actual robot and it can be carried out at

early design phase or even in parallel with production [26].
OLRP is more reliable than online robot programming methods as it often incorporates simulation, making it possible to test multiple scenarios of a robot cell, and if need be, adjust and fine-tune some properties before setting up the production. Hence, it reduces error as these can be predicted in time and it therefore also increases accuracy [26].
OLRP also provides more flexibility to changes, as these can be directly integrated by just replacing the corresponding program parts [26].
However, OLRP still have its drawbacks, especially for small and medium-sized enterprises (SMEs). Despite OLRP providers' efforts to develop more powerful, modular and compatible OLRP packages, these packages still require programming effort on the part of the users in order to customize the software for a particular application. This programming overhead and the high OLRP packages prices make them not possible, and possibly not worth for SMEs to introduce, especially for small batch production [26].

3.3 Novel Robot Programming Techniques & VARobot PG Perspective

Traditionally, robot programming is performed either online (Section 3.1) or offline (Section 3.2). Both these methods, however, lack intuitiveness and efficiency [25] and suffer from several drawbacks.
Online robot programming methods often lack amongst other things flexibility, reusability, accuracy and pose safety concerns. OLRP methods are often too complex and require skilled operators with a thorough understanding of programming and programming languages, in case of text-based OLRP techniques, or a thorough understanding of CAD combined with process knowledge, in case of graphical OLRP techniques, in order to produce good robot program.
Today robots are perceived as indisposable and the technology behind them is getting better and better for every new generation. Advancements in robotics have gained much momentum in recent years [25], leading the way for new, innovative robot programming techniques.
The purpose of these novel robot programming techniques is to propose a different approach that is safer, easy to use, speeds up the programming process, utilizes the intuitive process knowledge of the operator and increases the quality of the finished robot program without the need of a CAD model as in offline robot programming methods, nor the physical presence of the robot as in online robot programming approaches.
From the VARobot PG perspective, we are investigating exactly such methods that may ease the task of robot operation and programming through novel technology and interaction techniques. These methods would make it possible and reasonable for SMEs to integrate robots, as they reduce the robot integration and programming/reprogramming costs and difficulty.
The aim of the above-mentioned interaction techniques that are at the center of these newly emerged robot programming techniques is to mainly make the robot able to communicate with humans in a safe environment and in a human-like

manner, i.e. in the way humans communicate with each other. Thus, making the programming of industrial robots more intuitive, easier and safer.
Amongst others, Augmented Reality (AR) and Virtual Reality (VR) are such potential beneficial novel interaction techniques that may facilitate the robot programming process. Both these techniques, namely AR-assisted robot programming and VR-assisted robot programming are discussed in detail respectively in Sections 4.1 and 4.2. Note that other techniques may exist, however, these are out of the scope of this paper, as we are mostly interested in the techniques considered in the VARobot PG, namely AR and VR.
These novel robot programming methods based on innovative interaction techniques cannot fall neither exclusively under the online robot programming category, nor exclusively under the offline robot programming category. However, they quite blur the boundaries between online and offline robot programming methods to exploit the benefits of both methods while overcoming their limitations and avoiding their drawbacks. Therefore, in this paper, we derived a new classification criterion, namely, Novel Robot Programming Techniques, based on which robot programming techniques may also be classified.

3.4 Summary

Figure 8 depicts the classification tree derived in this paper from the previous sections of this chapter. The left and middle branches of the tree represent respectively online and offline robot programming methods. These are the main two methods considered in literature.
Online robot programming mainly refers to the use of an actual physical robot by an operator, that may interact with the robot system in different possible ways. As a result of this interaction, the robot control program is automatically generated. Two main approaches exist for the operator to program the robot in this way. The first approach is referred to as "lead-through programming", where the operator uses a teach pendant in order to manipulate and teach the robot certain process specific movements and orientations. The second approach is called "walk-through programming". In this approach, the operator "walks" the robot through the desired positions, as if he is doing the process himself. Online robot programming is a simple, intuitive and widely used programming method. It is especially suitable and beneficial for simple robotic systems. However, it often lacks flexibility, reusability, and accuracy. It also poses safety concerns and accounts considerable robot downtime.
An alternative to online robot programming is offline robot programming (OLRP), that is represented by the middle branch of our classification tree. OLRP mainly refers to developing the robot control program without using or accessing the actual physical robot. OLRP can be divided into two main sub-categories, namely, text-based OLRP techniques and graphical OLRP techniques. The first subcategory solely relies on the use of different programming languages, namely controller-specific, generic procedural, and behavior-based languages, in order to produce the robot program code that is later downloaded into the robot. On the other hand, the second sub-category is based on the 3D CAD model of the

robot and the workpiece on which it operates, in order to generate and simulate robot programs. OLRP is proved to reduce the robot downtime that comes with previously mentioned online robot programming methods. It is also more reliable and provides amongst other things more flexibility to changes. However, OLRP methods are often more complex due to the need to model the physical entities in the actual robot environment and the need for calibration and fine-tuning the simulated environments.

To conclude, there is no one programming method that is perfect for every situation. The best robot programming method will depend highly on the actual process, the robot, and the requirements. The advantages and disadvantages listed in Sections 3.1 and 3.2 can help the decision-making process. For example, online robot programming is more suitable for simple robotic systems. As the process becomes more complex, the suitability of online programming reduces and OLRP proves to be more beneficial.

As for the right, and last branch of our classification tree, namely novel robot programming techniques. This category represents methods that are not exclusively considered as online programming methods nor can be exclusively considered as OLRP methods, but rather in-between. These novel techniques propose a different approach that doesn't require the CAD model of the OLRP methods nor the physical presence of the robot as for online robot programming. At the same time they enhance the robot programming process in several ways.

Advancements in these methods are driven by the technological advancements in novel human–computer interaction techniques. Although further interaction techniques may exist, in this paper, we focus exclusively on both augmented reality and virtual reality, as these are the ones considered in the VARobot PG.

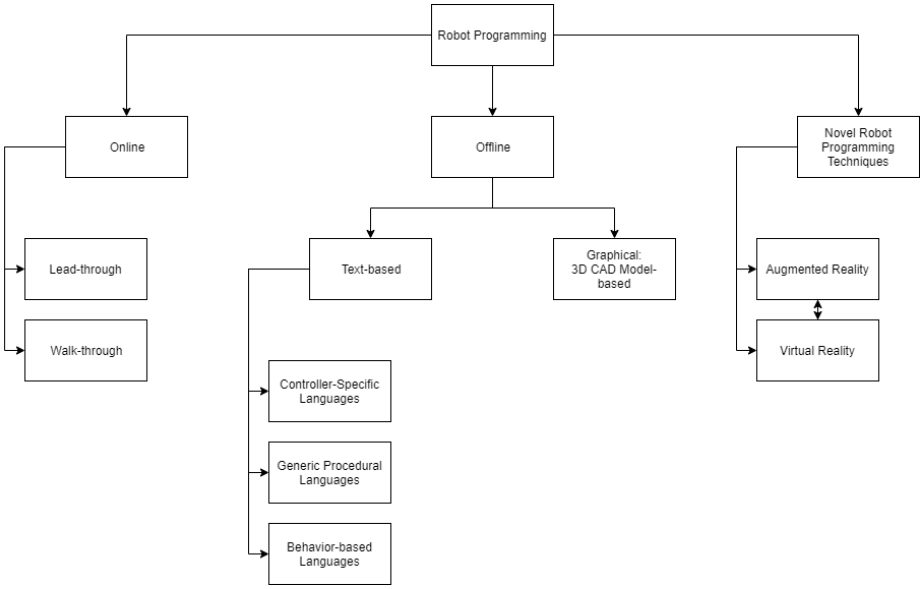

Fig. 8. Classification tree for robot programming techniques

4 Novel Robot Programming Techniques

4.1 Augmented Reality

Augmented Reality (AR) refers to a real-time view of physical real-world environment that has been augmented by merging its elements with virtual computer-generated information, such as texts, images, animations, etc., creating a mixed reality, where real and virtual objects are combined.
Since the virtual objects added to the real-world environment provide the user with information that he cannot directly detect with his senses, augmented reality aims to assist the users in their daily work [18]. In this paper, we are mainly interested in the use of augmented reality to assist users program robots.
The general approach to realize augmented reality is to use fast video processing, precise tracking and computer graphics to merge the real scene viewed by the user with a virtual scene generated by a computer. A typical AR system includes a video camera to capture the actual scene viewed by the user and an AR software. Since the location of the camera and the user is known, the AR software uses fast image-processing techniques to identify one or more markers placed in the viewed scene. The position and orientation of these markers are calculated precisely using the optical properties of the cameras. Next, the system combines the video image captured by the camera with computer-generated images of virtual objects in positions relative to the physical markers. From a user's perspective, the resulting view is no longer the raw video captured by the camera, but rather a representation of the physical world that has been augmented with virtual objects [29].
Entrepreneurs and investors can't ignore the AR's forecasted market revenues of more than 80 billion dollars by 2021. Therefore, many companies such as Microsoft, Google, Apple and DAQRI are taking an interest in this technology [16]. Consequently, due to the increasing efforts of several reputable universities and companies such as IBM, HP, Sony and Google in the field of augmented reality, this innovative technology has been successfully applied to solve a wide range of problems in many fields [31].
Head mounted displays (HMD), handheld displays and spatial displays are the three major types of displays used in Augmented Reality. HMD is a display device worn on the head or as part of a helmet that places images of the real and virtual environment over the user's view of the world. Handheld displays use small computing devices with a screen that users can hold in their hands [18]. Unlike the other two body-attached displays, spatial displays detach the technology from users and integrate it into the environment. Spatial augmented reality use data projectors to superimpose computed generated virtual objects directly onto physical objects' surfaces [33].
In the next section, we will examine the use of augmented reality to assist the robot programming process.

Augmented Reality-Assisted Robot Programming

Taking into consideration both drawbacks of online and offline robot programming techniques explained respectively in Sections 3.1 and 3.2, many researchers intend to combine the benefits of both these techniques while minimizing their drawbacks. Robot programming using augmented reality (RPAR) is a revolutionary concept that has been at the center of research in recent years [26] and that has proven to be very beneficial for programming robots [5].
RPAR systems may differ from a project to another, for example, some systems may use head mounted displays, while others eventually use handheld displays, but the concept remains the same. Through computers and AR displays it is possible to visualize motions and trajectories of a virtual robot, which is a replicate of a real robot, amongst real entities in a real environment. This enables users to intuitively interact with, control, teach, and simulate the behavior of these robots, for example through a pointing device, before executing the tasks using the real robot.
T. Pettersen et al. (2003) [27] demonstrate, as illustrated in Figure 9, an example RPAR system that allows operators to program robot waypoints and process specific events related to paint applications in a sequential manner. Through a tracking system that comprises a camera worn by the operator and fiducials with unique identities and known 3D positions, the operator is able to specify the robot path and control and simulate its actions using the input device. The system then presents visual feedback of the simulated paint result for the operator through a head mounted AR display overlaid on the real physical target. A wearable computer connected to the HMD, camera, and input device, is responsible for the automatic generation of the robot program to be installed on the actual robot, as well as for the execution of the different necessary processing algorithms. In addition to it being intuitive and much easier to generate robot programs, T. Pettersen et al. (2003) [27] report that this RPAR technique is at least five times faster than traditional robot programming techniques.

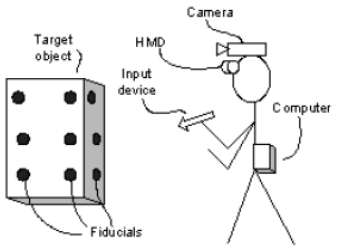

Fig. 9. RPAR system configuration [27]

J.W.S. Chong et al. (2009) [15] also present, as illustrated in Figure 10, another example RPAR system that allows controlling and simulating a virtual airplane washing robot over a scaled-down model of an airplane in order to generate a robot sequence that can later be calibrated and installed for a real airplane washing robot.

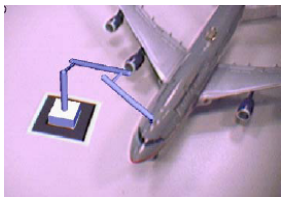

Fig. 10. Virtual airplane washing robot in real environment over a scaled-down model of an airplane [15]

Discussion

As explained in Section 3.3, RPAR carries through benefits of both online and offline robot programming, while overcoming their limitations.
RPAR retains the OLRP advantages of not having to take a physical robot out of production, including the safety and operational benefits that comes with it. On the other hand, RPAR also overcomes OLRP limitations caused by having to virtually model the different entities in the working environment, by augmenting the virtual robot directly into the real-world work cell, providing the flexibility and adaptability for different environments. RPAR therefore also eliminates the need for careful calibration between the different entities of both the real and virtual words, as in OLRP [26].
Since models in RPAR can be scaled down, as for the airplane model explained in the previous section, RPAR can also overcome the online robot programming limitation of being able to only program small robots, that the operator can directly teach [26]. It is possible to program robots such as large pick-and-place robots using augmented reality [25].
Virtual robots in RPAR are also no longer subject to mechanical and physical constraints as in online robot programming, resulting in a much safer programming process [25].
RPAR allows the operator to get instant real-time visual feedback of the simulated process overlaid on the real objects, which reduces the number of iterations needed in traditional robot programming techniques to assure a good program quality, and hence robot programming time and costs are reduced. The quality of the resulting robot program is also proved to be better than that of traditional robot programming methods. Operators also report that RPAR is intuitive and much easier than traditional methods [27]. It is not as complex as traditional methods for inexperienced robot programmers, such that often neither external

assistance nor rigorous training in robot programming are needed [5].
In addition, since the physical presence of the robot in the work cell is not required, robots can be programmed and evaluated even during the shipment period, before the physical robot is shipped to the actual work cell [25].
From the VARobot PG perspective, the RPAR benefits explained above, resulting from the introduction of AR, a novel interaction technique, make RPAR an appropriate solution for SMEs to benefit from robotic automation. This is not possible with traditional online and offline robot programming methods, which are not suitable for the continuously changing SMEs product lines, resulting in reprogramming costs that exceed installation costs by a large margin [5].

4.2 Virtual Reality

Virtual Reality-Assisted Robot Programming

Robot programming using virtual reality (RPVR) refers to the use of a VR-based, high-end human-computer interface, a mean where the operator is fully immersed in the simulated virtual environment of the robotic system, and by which the operator can operate complex robotic systems in an intuitive, natural way [24].
As for RPAR systems, the settings of a RPVR system may differ from a project to another. However, these different systems adopt the similar concept explained in this section.
The operator is immersed in a virtual environmental simulation of the robot work cell through a stereo viewer, e.g. VR headsets, that keeps track of the operator's head position and orientation and presents visual feedback to the operator. To achieve the feeling of immersion inside the virtual environment, the graphics are updated as soon as the operator's view changes [24]. This is usually done via a multiprocessor graphics workstation [13]. The virtual robotic cell in RPAR systems usually consists of a virtual robot, an avatar representing the human operator, eventually some other digital objects referencing objects in the real work cell and the computational element [23]. The operator can move and navigate through this virtual environment as they might with the real world. By simply looking at a desired specific location, the operator can "walk" or "fly", "teleport" to it. The operator can also directly interact with the virtual objects constituting the robotic system, instead of the interaction through a flat screen display and a mouse as in offline robot programming methods [24]. Input/output (I/O) devices, such as joysticks or 3D trackers for simple tasks and sensing gloves for more dexterous tasks, capture these interactions [13]. The RPVR system allows the operator to specify the dynamic behavior of components, such as their paths, accelerations, velocities, interpolations, etc. [13] The RPVR system captures, for example from the operator's hand movements, a sequence of high level actions and translates them into a sequence of commands for the robot [6]. The RPVR system then allows these robot commands to be captured, previewed in a simulated environment for validation, and once completed, downloaded to the real robot [24].

Discussion

The RPVR approach provides a user-friendly human-robot interface that enables operators to operate complex robotic systems intuitively [24]. Performing the robot demonstration in the virtual environment of the robotic system not only increases the safety, since all objects are digital and cannot cause any harm in the event of collisions [23], and the operator no longer has to be physically present in a hazardous environment [21], but also decreases the time and fatigue required for demonstration [6]. Specifying a trajectory can be as simple as a hand gesture [13]. The operator can also look at the virtual scenes from any angle, and thus see details that may not be visible in real life [13].
Another advantage of RPVR systems is that the virtual robotic cell can be augmented with operator aids and visual feedback, that help the operator while demonstrating the task to the robot [6]. VR also enables the simulation of the task before executing it with the real robot, and this, as many times as they want with no risk or additional costs. It therefore also reduces robot downtime and don't disturb production lines during the teaching/programming stage [21]. RPVR reduces the required skills of a programmer and allows non-experts to program robots without interacting with the real robots, with reduced programming time and errors. Eventual damage to the actual robot is also reduced to a minimum level, as robot commands are virtually tested beforehand [21].
However, these advantages must be weighed against the main disadvantage of RPVR systems, namely the need for advanced explicit encoding of a priori knowledge about the task, which then limits the applicability of RPVR [6].

5 Future Work

Due to lack of time, some concepts have been left for the future.
Future work mainly concerns a broader analysis of additional literature and robot programming methods, in particular regarding novel robot programming techniques. As these techniques are based on technological advancements and technological advancements continue to grow, paving the way for new methods.
Another point left for future work is a more in-depth analysis of each of the robot programming techniques. There are many variations of approaches in each of the robot programming categories. Future papers can further investigate each category separately. Note that this article provides an overview, reviews the main concepts of each of the robot programming methods and classifies them. However, in practice and theory, many variations may exist for each category.
It would have been better if we could also interview experts in the field to potentially get new information on possible new categories. As such a classification cannot be closed, new categories may emerge and certain other categories may have been excluded from the scope of this paper, due to lack of time, and/or sources of information, and/or due to the restriction of the number of pages.

References

1. Industrial Robots and Robot System Safety. https://www.osha.gov/dts/osta/otm/otm$_i v/otm_i v_4.html, accessed : 2020−06−05
2. Oxford English Dictionary. https://www.lexico.com/definition/robot, accessed: 2020-06-05
3. Project Group: VARobot. https://cs.uni-paderborn.de/en/dbis/lehre/veranstaltungen/project-group-varobot/information, accessed: 2020-06-05
4. Six degrees of freedom. https://en.wikipedia.org/wiki/Six$_d egrees_o f_f reedom, accessed$: $2020 - 06 - 05$
5. Akan, B., Ameri, A., Çürüklü, B., Asplund, L.: Intuitive industrial robot programming through incremental multimodal language and augmented reality. In: 2011 IEEE International Conference on Robotics and Automation. pp. 3934–3939. IEEE (2011)
6. Aleotti, J., Caselli, S., Reggiani, M.: Leveraging on a virtual environment for robot programming by demonstration. Robotics and Autonomous Systems **47**(2-3), 153–161 (2004)
7. Ang, M.H., Lin, W., Lim, S.Y.: A walk-through programmed robot for welding in shipyards. Industrial Robot: An International Journal (1999)
8. Ang, M.H., Wei, L., Yong, L.S.: An industrial application of control of dynamic behavior of robots-a walk-through programmed welding robot. In: Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065). vol. 3, pp. 2352–2357. IEEE (2000)
9. Bascetta, L., Ferretti, G., Magnani, G., Rocco, P.: Walk-through programming for robotic manipulators based on admittance control. Robotica **31**(7), 1143–1153 (2013)
10. Ben-Ari, M., Mondada, F.: Elements of robotics. Springer Nature (2017)
11. Biggs, G., MacDonald, B.: A survey of robot programming systems. In: Proceedings of the Australasian conference on robotics and automation. pp. 1–3 (2003)
12. Brunete, A., Mateo, C., Gambao, E., Hernando, M., Koskinen, J., Ahola, J.M., Seppälä, T., Heikkila, T.: User-friendly task level programming based on an online walk-through teaching approach. Industrial Robot: An International Journal **43**(2), 153–163 (2016)
13. Burdea, G.C.: Invited review: the synergy between virtual reality and robotics. IEEE Transactions on Robotics and Automation **15**(3), 400–410 (1999)
14. Choi, S., Eakins, W., Rossano, G., Fuhlbrigge, T.: Lead-through robot teaching. In: 2013 IEEE Conference on Technologies for Practical Robot Applications (TePRA). pp. 1–4. IEEE (2013)
15. Chong, J.W.S., Ong, S., Nee, A.Y., Youcef-Youmi, K.: Robot programming using augmented reality: An interactive method for planning collision-free paths. Robotics and Computer-Integrated Manufacturing **25**(3), 689–701 (2009)
16. Evans, G., Miller, J., Pena, M.I., MacAllister, A., Winer, E.: Evaluating the microsoft hololens through an augmented reality assembly application. In: Degraded Environments: Sensing, Processing, and Display 2017. vol. 10197, p. 101970V. International Society for Optics and Photonics (2017)
17. Ferraguti, F., Landi, C.T., Secchi, C., Fantuzzi, C., Nolli, M., Pesamosca, M.: Walk-through programming for industrial applications. Procedia Manufacturing **11**, 31–38 (2017)
18. Furht, B.: Handbook of augmented reality. Springer Science & Business Media (2011)
19. Holubek, R., Delgado Sobrino, D.R., Košt'ál, P., Ružarovskỳ, R.: Offline programming of an abb robot using imported cad models in the robotstudio software environment. In: Applied Mechanics and Materials. vol. 693, pp. 62–67. Trans Tech Publ (2014)

20. Horváth, C.M., Korondi, P.: Supportive robotic welding system for heavy, small series production with non-uniform welding grooves. Acta Polytechnica Hungarica **15**(8) (2018)
21. Jen, Y.H., Taha, Z., Vui, L.J.: Vr-based robot programming and simulation system for an industrial robot. International Journal of Industrial Engineering **15**(3), 314–322 (2008)
22. Landi, C.T., Ferraguti, F., Secchi, C., Fantuzzi, C.: Tool compensation in walk-through programming for admittance-controlled robots. In: IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society. pp. 5335–5340. IEEE (2016)
23. Maragkos, C., Vosniakos, G.C., Matsas, E.: Virtual reality assisted robot programming for human collaboration. Procedia Manufacturing **38**, 1697–1704 (2019)
24. Miner, N.E., Stansfield, S.A.: An interactive virtual reality simulation system for robot control and operator training. In: Proceedings of the 1994 IEEE International Conference on Robotics and Automation. pp. 1428–1435. IEEE (1994)
25. Ong, S.K., Chong, J., Nee, A.Y.: Methodologies for immersive robot programming in an augmented reality environment. In: Proceedings of the 4th international conference on computer graphics and interactive techniques in Australasia and Southeast Asia. pp. 237–244 (2006)
26. Pan, Z., Polden, J., Larkin, N., Van Duin, S., Norrish, J.: Recent progress on programming methods for industrial robots. In: ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics). pp. 1–8. VDE (2010)
27. Pettersen, T., Pretlove, J., Skourup, C., Engedal, T., Lokstad, T.: Augmented reality for programming industrial robots. In: The Second IEEE and ACM International Symposium on Mixed and Augmented Reality, 2003. Proceedings. pp. 319–320. IEEE (2003)
28. Qi, L., Zhang, D., Zhang, J., Li, J.: A lead-through robot programming approach using a 6-dof wire-based motion tracking device. In: 2009 IEEE International Conference on Robotics and Biomimetics (ROBIO). pp. 1773–1777. IEEE (2009)
29. Radkowski, R., Herrema, J., Oliver, J.: Augmented reality-based manual assembly support with visual features for different degrees of difficulty. International Journal of Human-Computer Interaction **31**(5), 337–349 (2015)
30. Stolt, A., Carlson, F.B., Ardakani, M.M.G., Lundberg, I., Robertsson, A., Johansson, R.: Sensorless friction-compensated passive lead-through programming for industrial robots. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 3530–3537. IEEE (2015)
31. Wang, X., Ong, S.K., Nee, A.Y.: A comprehensive survey of augmented reality assembly research. Advances in Manufacturing **4**(1), 1–22 (2016)
32. Wittenberg, G.: Developments in offline programming: an overview. Industrial Robot: An International Journal **22**(3), 21–23 (1995)
33. Zhou, J., Lee, I., Thomas, B., Menassa, R., Farrant, A., Sansome, A.: Applying spatial augmented reality to facilitate in-situ support for automotive spot welding inspection. In: Proceedings of the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry. pp. 195–200 (2011)