25th IEEE International Symposium on
Robot and Human Interactive Communication (RO-MAN)
August 26-31, 2016. Columbia University, NY, USA

Augmented Reality for Industrial Robot Programmers: Workload Analysis for Task-based, Augmented Reality-supported Robot Control

Susanne Stadler[1], Kevin Kain[1], Manuel Giuliani[1], Nicole Mirnig[1], Gerald Stollnberger[1], and Manfred Tscheligi[1]

Abstract— **Augmented reality (AR) can serve as a tool to provide helpful information in a direct way to industrial robot programmers throughout the teaching process. It seems obvious that AR support eases the programming process and increases the programmer's productivity and programming accuracy. However, additional information can also potentially increase the programmer's perceived workload. To explore the impact of augmented reality on robot teaching, as a first step we have chosen a Sphero robot control scenario and conducted a within-subject user study with 19 professional industrial robot programmers, including novices and experts. We focused on the perceived workload of industrial robot programmers and their task completion time when using a tablet-based AR approach with visualization of task-based information for controlling a robot. Each participant had to execute three typical robot programming tasks: tool center point teaching, trajectory teaching, and overlap teaching. We measured the programmers' workload in the dimensions of mental demand, physical demand, temporal demand, frustration, effort, and performance. The study results show that the presentation of task-based information in the tablet-based AR interface decreases the mental demand of the industrial robot programmers during the robot control process. At the same time, however, the programmers' task completion time increases.**

I. INTRODUCTION

A reshaping of robot control and programming interfaces is taking place in the industrial robotics industry and in robotics research. This is closely connected to economical changes (e.g., the re-industrialisation of western countries), technological progress (e.g., the internet of things), and emerging new robot types (e.g., light-weight robots). Apart from larger companies, also small and medium sized enterprises (SMEs) will use production automation and low cost robotics solutions to produce goods more competitively. The fact that SMEs extend the requirement profile for industrial robots, Schraft and Meyer [1], is of particular interest for us. SMEs typically have small lot size, short production cyles, unstructured environments and do not employ experts with robotic knowledge [2]. We use augmented reality (AR) as an approach to enhance industrial robot programming.Industrial robot programming and maintainance is not done on a daily basis. It is needed throughout the implementation of a new robot line or for a single robot in a factory. The complexity of industrial robot programming comes from the integration of the robot with other robots and machines. Industrial robot control is needed when a robot stops in production (e.g., singularity, axis limits) and is done by maintainers via manually guiding the robot to a safe/working position. Industrial robot programming is often divided into offline and online programming. The overall coordination between the robots and machines involved in the production process is programmed in a digital simulation called offline programming. The fine-tuning of robot tasks for single robots done by hand in the analog world is called online programming. AR has the potential to bridge the digital and analog worlds of robot programming for programmers and maintainers.For example, robot programmers could receive additional task-based information from the digital simulation on a tablet-based interface during the fine-tuning of single robots.

However, the work by Wurhofer et al. [3] shows that the introduction of additional information to human-machine interfaces in factory environments can lead to an increased perceived workload, which adds to the stress of factory workers. Therefore, providing additional information to robot programmers, who are factory workers, should be introduced carefully in AR interfaces. Within this work, we answer the question how the workload of industrial robot programmers and their task completion time is affected by using an AR interface that overlays industrial robot programming tasks with task-based information. We conducted a user study in which robot programmers used a tablet-based AR interface for task-based robot programming. We analyzed the impact of augmented reality on robot programmers throughout the teaching process. Specifically, we measured the workload of the user in all its dimensions: mental demand, physical demand, temporal demand, frustration, effort, and performance as well as the task completion time. In this paper, we review related work, present our user study setup, and report and discuss the study results.

II. BACKGROUND

For our work, we have to take two main topics into account. The first is robot programming in the factory context. Industrial robot programming includes on- and offline programming. Within the process of industrial robot programming, the robot always has to be taught online: either to check an offline program in the real cell (workspace of robots) with a real robot or the robot has to be taught from scratch for the given task. For online programming, a so

We gratefully acknowledge the financial support by the Austrian Federal Ministry of Economy, Family and Youth and the National Foundation for Research, Technology and Development (Christian Doppler Laboratory for "Contextual Interfaces").

[1]S. Stadler, K. Kain, M. Giuliani, N. Mirnig, G. Stollnberger, and M. Tscheligi are with the Center for Human-Computer Interaction, University of Salzburg, Austria. firstname.lastname@sbg.ac.at

called teach pendant – a hand-held control and programming unit – with restricted capabilities to visualize information is used. This process is time consuming, the robot is idle, and a person with spatial abilities and experience is needed (e.g., mental rotation, kinematics). More time and cost saving is offline programming; the programmer uses an external computer to program a robot in the simulated robot cell.

In the following paragraphs, we shortly describe three highly repetitive tasks throughout the online teaching process, which we used as programming tasks for our study.

Tool center point (TCP) teaching: A prerequisite for successful and efficient online robot programming is an accurately configured coordinate system (e.g., base, world, tool). The tool coordinate system belongs to the most often used coordinate systems, because tools, such as grippers, change often in an industrial context. TCP teaching is done by moving the robot, making the TCP brush against a fixed point in the close surrounding to the robot. The fixed point is e.g., the tip of a nail. By brushing against the tip of the nail from at least four different orientations, the coordinate of the TCP in relation to the robot base frame is calculated [4].

Trajectory teaching: The online programmer has to prepare the program of the robot by programming the robot trajectory points. The task of the online programmer is to prepare a collision free program and adapt it to an overall production sequence (several robots are working on one object, cycle times have to be respected).

Overlap teaching: The online programmer has to prepare a program by programming trajectory points. Often, vertices have to be rounded, which is done by overlapping. Two criteria of overlapping exist: velocity-based and position-based overlapping. Velocity-based overlapping starts when the velocity becomes lower than a fixed minimal value (drawback: dependent on velocity profile). Position-based overlapping starts when the TCP enters the overlapping zone. Outside the overlapping zone the TCP keeps the cartesian path (only the overlapping zone is reached, not the exact trajectory point – Fly-By-Point).

The second major topic in this section is augmented reality.AR enhances the user perception by overlaying the real world information with computer-generated information [5]. Bischoff and Kazi [6] discussed the potential of augmented reality-based human robot interaction and assume the extension of the teach pendant is a promising approach for robot teaching. Abbas et al. [7] proposed the usage of mobile devices in combination with AR for industrial robot online programming. Mateo et al. [8] state that the use of Android tablets in industrial robot working environments is growing and provides new means for small batch industrial applications that need fast and easy to use tools to program the robots. Tang et al. [9] conducted a user study regarding the relative effectiveness of AR instructions in an assembly task. Regarding remote control with AR, Hasimoto et al. [10] introduced a touchscreen AR prototype.

The study results show that the usage of AR does not lead to faster task completion time than displaying instructions on traditional media (paper plan, display). The results also show that assistance in mental transformation and the minimizing of attention switching does result in an improvement of performance. However, Tang et al. found that the possible advantage of overlaying information on the workspace may have been negated by the cost of visual interference. This is supported by the work of Wurhofer et al. [3]. With our tablet-based approach, we follow Abbas et al. and Mateo et al. and focus on the workload of task completion as Tang et al. Our contribution is that we investigate three highly relevant industrial tasks executed by professional robot programmers.

III. EXPERIMENTAL DESCRIPTION

To investigate the impact of augmented reality support on industrial robotic end-users, we performed a within-subject (AR vs. No AR) study with 19 participants. This section includes a description of the participants' characteristics (demographics including AR background, competence level), an overview of the study setup (robot platform, augmented reality interface, test environment), a description of the study procedure, as well as a short description of the methods used to analyze the gathered data (workload, time on tasks).

A. Participants

In total, 19 male participants with an average age of 33.53, SD=1.75 (22 to 47 years) took part in the study. This reflects the gender situation within the factory and is due to the requirement to recruit real robot programmers. All participants had a strong link to robot programming. As can be seen in Table I, the participants belong to four job families. The job family of robot programmers consisted of on- and offline programmers for industrial applications. The job family of operators/maintainers were workers with little practical on-/offline programming experience. They manually control robots in error situations during day-to-day operation. The software engineers job family consisted of engineers who developed applications for robot control interfaces, ranging from the system level to interface level. The manager job family consisted of product and project managers for robotics applications.

TABLE I
CHARACTERISTICS OF PARTICIPANTS: AR BACKGROUND AND COMPETENCE LEVEL REGARDING ONLINE/OFFLINE PROGRAMMING.

		Background AR		Competence Level	
Job Family	**#**	**Fictional**	**First-hand**	**Novice**	**Expert**
Robot Programmer	8	6	2	1	7
Operator/Maintainer	2	-	-	2	0
Software Engineer	4	2	2	2	2
Manager	5	2	3	1	4

Regarding AR background, we distinguish between two groups of participants: the first being participants who have only fictional understanding of AR in terms of having heard of or having seen AR (e.g., movies, media, TV, discussions) and the second being participants who have first hand experience with AR (used AR at least once). Table I shows that the participants mainly had a fictional understanding of

AR (10 subjects); a few had already used an AR system (7 subjects). Two participants had never heard of AR.

Participants rated their competence level regarding on- and offline programming on a four point Likert scale with the values “none”, “beginner”, “advanced”, “expert”. Based on this self-assessment, we divided the participants into two groups: *Novice* (6 participants) and *Expert* (13 participants). Participants who rated themselves at least as “advanced” or “expert” in on- and offline programming are summarized in the group *Expert*. Participants who rated their competence level as “beginner” or lower are included in the *Novice* group.

B. Study Setup

As illustrated in Fig. 1, the study setup consisted of three components: the robot, the AR interface, and the testbed.

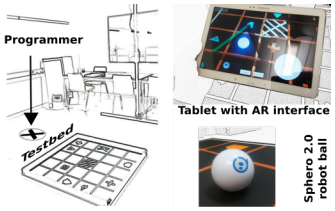

Fig. 1. Study setup components (Testbed, Sphero, tablet)

The requirements for the **robotic platform** were to be easy to operate, easy to program, transportable, and cost-effective. We used the Sphero 2.0 robot ball[1] in this study because it met these requirements. The Sphero is a spherical robot, capable of omnidirectional movement on a 2D plane. The simple design of the robot allows our study participants to focus on the study tasks. For this study we decided that the Sphero robot, with its many limitations, provides a good platform for the workload analysis, disregarding industrial aspects. For the **AR interface**, we used a tablet-based (see-through and robot control) AR approach to evaluate task-based information. Therefore, we followed the taxonomy of Tönnis et al. [11]; the tablet-based see-through approach implements the presentation scheme of a direct overlay. The task-related information was directly, environmentally mounted on the testbed in a discrete way. The frame of reference (viewpoint reference frame) was egocentric, but from the device’s perspective. The whole scenery was shown from the point of view of the hand-held device, and not that of the user (as when wearing AR glasses). The AR tablet-based interface was developed with Unity 5.1.3f1 Personal and built for Android OS 5.0.2 Lollipop. We decided to use Unity for development based on the beneficial handling of 3D graphics and the existing integration capability of the Sphero (control) and Vuforia library[2] (AR). The hardware was a Samsung Galaxy Tab 4 (SM-T530, 10.1 inches display). The **testbed** was the environment where the Sphero robot was controlled. It was built from a 1×1 meter baseplate, colored adhesive foil, a bounding frame, and prepared to serve as AR marker.

[1] http://www.sphero.com/sphero
[2] https://developer.vuforia.com/

C. Procedure

As depicted in Fig. 2, the study procedure was divided into several parts. Represented as dots, the beginning included

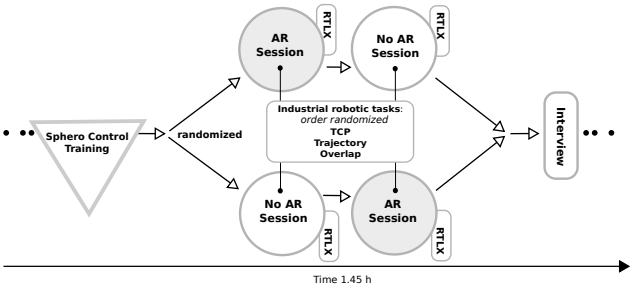

Fig. 2. Study procedure

organizational parts (welcome, introduction, data usage consent). Shown as a triangle, the Sphero Control Training, served as the initial training for the movement and control of the robot. In the center of the study were two robot programming sessions, shown as circles, followed by an interview. At the end of the study, also represented as dots, was a short debriefing. All in all, the study took approximately 1hr45min.

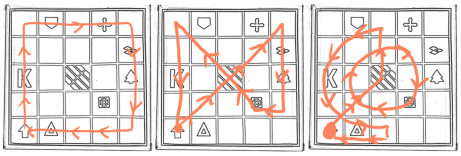

Fig. 3. *Sphero Control Training* steps

In the Sphero Control Training (no augmented reality support), the participants familiarized themselves with controlling the Sphero using a tablet. All participants had to complete three training steps, see Fig. 3. In the first step, the participants trained to move the robot forward, left, and right. In the second training step, moving the robot diagonally was added (needed for trajectory teaching). The third training served as practice to stop the Sphero robot in a controlled way and added circular motion (needed for overlap teaching).

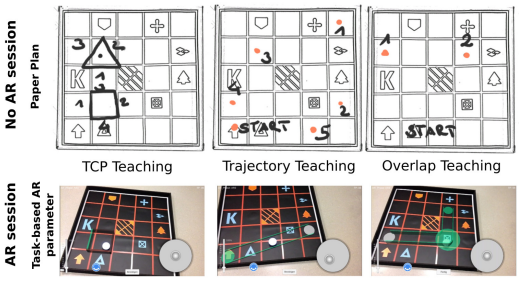

Fig. 4. Visualization of task-based parameters per task (TCP, trajectory, overlap) in the No AR session (paper plan, top) and AR session (bottom)

As shown in Fig. 4, the next two sessions (AR and No AR) included in each case the same three tasks: TCP teaching, trajectory teaching, and overlap teaching. The difference between the sessions was that, in the AR session, the tablet

showed task-based support parameters, whereas in the No AR session, the tablet displayed no support parameters. In the No AR session, all task-based information was provided on paper as graphical sketches describing the required robot motion. We call this information the paper plan. This plan was approximately two meters away from the teaching position of the participant and, therefore, not in the participant's area of view. Each task (TCP, trajectory, overlap) was shown on a separate paper plan. In the AR session, all task-based information was shown within the AR interface on the tablet. Participants themselves confirmed the end of each completed task and started the next task using a button on the tablet interface. In the following a detailed description of the programming tasks used in our study and shown in Fig. 4 for the AR session and No AR session is given.

TCP teaching required moving the robot ball in several directions to a reference point. The task-based information for TCP teaching were the directions to reach the reference point, visualized as geometric shapes. When participants had to teach from three directions, the shape was a triangle and in its center was the reference point. Likewise, when four directions were given, the shape was a rectangle. The procedure for both cases was the same: Move the robot in the first given direction to the reference point, stop the robot, and continue over the opposite vertex or adjacent edge of the shape. Repeat this procedure until no direction remains. In the No AR session, the order of edges to reach the reference point was provided by increasing numbers on paper. The user had to remember the edges and was allowed to go to the paper plan as often as needed to complete the task. In the AR session, the edge to reach the reference point was colored in green, as opposed to gray for inactive directions.

Trajectory teaching required moving the robot ball along a predefined trajectory path. Without AR, the trajectory points were shown as numbered points on the paper plan in the session. The participants had to remember the seven points (including return to start) and were allowed to refer to the paper plan. In the AR session, only path fragments were shown with a start point in gray, an end point in green, and a hose connecting the points. The participants' task was to start at the gray point and reach the green end point. Then they confirmed and got the next path fragment.

Overlapping required moving the robot ball along a predefined trajectory including the overlapping of one point. This was the shortest task. The task-based information was the start, end, and fly-by point, as well as the edge. The task of the participants was to teach the overlap trajectory. In the No AR session, the participants looked at the paper plan and were instructed that the overlap radius is 10 cm. In the AR session, the start point, end point, and overlap radius of the fly-by point were marked in the AR interface.

To avoid side effects, we randomized the AR sessions and the tasks within the sessions. Subsequent to each session, the participants had to complete a questionnaire to assess the workload. After the sessions, we obtained further details about the participants' attitudes towards using AR in an industrial context via a semi-structured interview.

D. Measurements

To obtain participants' characteristics, such as general demographics and robotics/AR background (see Table I) we used a questionnaire. To investigate the workload of the participants immediately after each session, we used the NASA-RTLX questionnaire [12]. We implemented a logging function on the tablet (robot acceleration, tablet orientation, interaction data) to record the time spent on each task. We summed up the tasks (task completion time) within the sessions to compare the average time spent on tasks in the AR and No AR session.

Independent variables: We considered augmented reality support (AR/No AR), the participants' competence level (*Novice*/*Expert*), and the augmented reality background (fictional understanding/first-hand experience) as independent variables. Augmented reality support was manipulated as repeated measure. AR background and the competence level were manipulated as between-group variables.

Dependent variables: We considered the RTLX workload ratings and time spent on tasks as dependent variables.

IV. RESULTS

We report on differences regarding the workload of robot programmers throughout the sessions, AR and No AR, and the task completion times related to the sessions. For the group without AR knowledge, we did no analysis because only two participants were included in this group.

All mean comparisons for workload and task completion time started with checking the assumptions for parametric analysis: normal distribution (W, p) and homogeneity of variance (Levene test, F, p).The means of the outcome variable regarding *augmented reality support* came from the same entities, so we used the dependent t-test (t) or, if assumptions were not met, a robust version (Yuend, T_y). To compare means, based on the treatments *competence level* and *AR background* as subcategories of *augmented reality support*, we used the independent t-test (t) or, if assumptions were not met, a robust version (Yuen, T_y).

a) Workload: Overall six NASA-RTLX factors [13], mental demand (MD), physical demand (PD), temporal demand (TD), performance (P), effort (E), and frustration (F) were rated from 0 (low) up to 100 (high). The ratings were averaged to calculate an estimate of the overall workload.

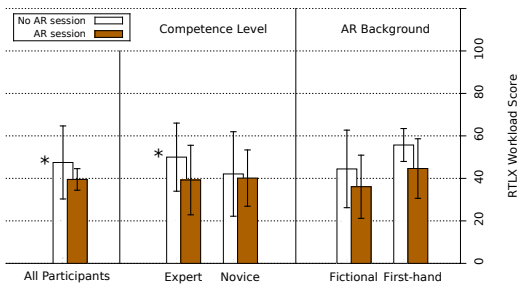

Fig. 5. Mean and SD of RTLX workload ratings: all participants, competence level and AR background, * – statistically significant

Fig. 5 shows the results of the RTLX workload ratings. Regarding the overall workload score the dependent t-test significantly shows ($t(18)$=2.81, p=.01) that the participants' workload is lower in the AR session (M=39.52, SD=15.08) than without displayed task-based AR parameters in the No AR session (M=47.50, SD=17.18). Taking into account the participants' *competence level*, participants at the *Expert* level perceived significantly less ($T_y(8)$=2.89, p=.02) workload in the AR session (M=39.23, SD=16.36), compared to no augmented reality support in the No AR session (M=50.00, SD=16.02). *Novices* showed no significant difference (AR session: M=40.14, SD=13.25, No AR session: M=42.08, SD=19.88). Regarding *AR background*, results show a trend $T_y(4)$=2.37, p=.07 that participants with first-hand experience perceive less workload $T_y(4)$=2.37, p=.07 in the AR session. The workload of participants with a fictional understanding remains unaffected by augmented reality support $T_y(4)$=1.36, p=.24. As can be observed in Fig. 6, we

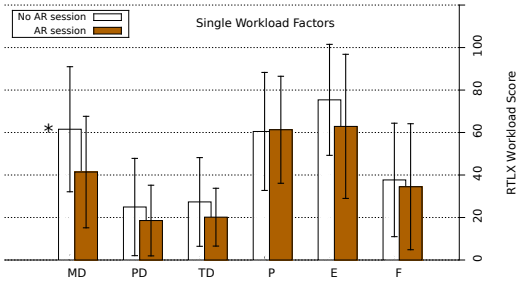

Fig. 6. Mean and SD of single workload factors , * – statistically significant

found no significant differences within the single workload factors except for the factor mental demand (MD). The users' mental demand was significantly lower ($t(18)$=−3.75, p=.001) in the AR session (M=41.05, SD=26.01) than in the No AR session (M=61.05, SD=29.18).

From Fig. 7, it can be seen that participants at *competence level Expert* perceived a lower mental demand throughout programming in the AR session (M=44.62, SD=28.61) than in the No AR session (M=64.62, SD=28.97). This effect was significant $t(12)$=3.68, p=.01. Although the participants at the *Novice* level had also a lower perceived mental demand with augmented reality support, the difference was not significant $t(5)$=1.53, p=.19 (normal distribution was met for session AR/No AR, variance $F(1,10)$=0.68, p=.43). Also illustrated in Fig. 7, having a fictional understanding as well as having previously used an AR system significantly decreases the mental demand in the AR session – fictional understanding ($t(6)$=2.70, p=.04): No AR (M=60.71, SD=29.36), AR (M=45.71, SD=25.89), first-hand experience ($t(6)$=2.70, p=.03): No AR (M=75.71, SD=23.70), AR (M=23.75, SD=11.81).

b) Task completion time: Inspection of Fig. 8 shows that only first-hand experience with augmented reality has a significant impact on teaching times ($t(6)$=−3.61, p=.01). When having first-hand experience with AR, participants needed more time on the tasks in the AR session (M=158.96,

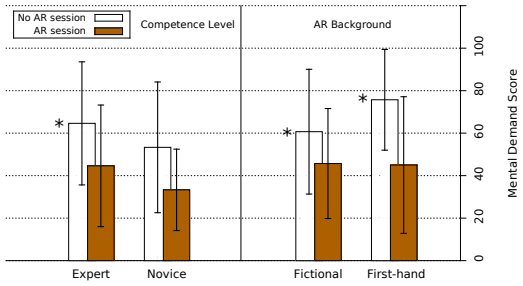

Fig. 7. Mean and SD of mental demand including competence level, and AR background, * – stat. significant

SD=35.26), than as in the No AR session (M=107.81, SD=17.04). Further investigation of tasks showed that only trajectory teaching took significantly longer ($T_y(11)$=3.07, p=0.01) with AR (M=151.28, SD=71.52) than without (AR M=109.89, SD=29.08).

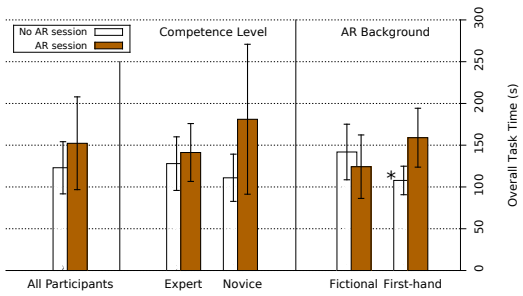

Fig. 8. Overall time spent on tasks (seconds) per session (AR vs. No AR) regarding competence level and AR background, * – stat. significant

V. DISCUSSION

Our results indicate that the support of industrial robotic online programming tasks with augmented reality task-based parameters decreases the mental workload of the participants, but increases the task completion time. This finding is partly consistent with the results of Tang et al. [9]. They found, based on an object assembly task with Lego bricks, that AR systems can relieve mental demand, without having an effect on task times.

Regarding workload, we suppose that with the superimposed task-based information shown on the tablet, the working memory is relieved by not having to remember the teaching points and directions. Furthermore, the simultaneous attention on two different information sources (paper plan and robot) is avoided (split attention effect, Kalyuga et al. [14]). These arguments do not explain why the AR support has a beneficial impact on the workload ratings of expert robot programmers only. An explanation could be the expertise reversal effect: Instructional techniques that are highly effective with inexperienced learners can lose their effectiveness and even have negative consequences when used with more experienced learners (Kalyuga [15], [16]). It seems that the presentation (no text, only colors, shapes and paths) of the task-based information plays into the hands of robot programming experts. During the interview, all

participants confirmed the simplicity of the representation of the AR information. On the question if the interface should be changed or improved, novices had more suggestions than experts. Due to the expertise reversal effect, the simplistic interface may have been advantageous for experts because it avoided the processing of redundant information in the working memory. In contrast, the interface may not have integrated enough information for novice robot programmers. This may have led to a uniform overall workload and not significant decreased mental demand of novices in our experiment.

Regarding completion time, in our robot teaching setup, especially during the trajectory teaching and tool center point teaching task, the task completion time was increased. We have to caution that our interface prototype was sometimes lagging, but in the same way for all participants.Several participants mentioned this in the interview as an area for improvement. Apart from that, one reason for the longer task completion time could be that participants had no official spoken time limit. We observed that participants' motivation to fulfill the task more accurately increased with AR. At the same time, however, they stated during the interview that they were more stressed by the direct task-related overlay because they saw when they were not accurate. We assume further that the participants were mainly prevention motivated (sacrificing overall speed for the sake of accuracy, Förster et al. [17]) and tried to fulfill the task as accurate as possible with AR. The minimalistic AR presentation also seems to be better adapted to the expert with regards to task completion time. Towne found in [18] that cognitive time (time not engaged with devices or instruments) can account for about 50 percent of task time. While completion time was only slightly increased for experts, novices were much slower with AR support, but not significantly so. Regarding first-hand experience with AR, the mental workload significantly decreases, but at the same time leads to a significant longer teaching time when using AR. We assume that users with first-hand AR experience were motivated to explore the capabilities of our setup.

VI. CONCLUSIONS

The success of AR interfaces will be closely related to the expected benefit which may be tangible with AR. Our results show that by supporting the user with helpful AR information, the workload throughout robot control can be decreased.To generate a positive experience with AR and an increase in task performance, however, it is inevitable to include knowledge about the target user group, the nature of the task, and a clearly defined goal in the design process of the AR interface. Within this experiment with a Sphero robot, we found that the workload of domain experts is decreased by AR support. The results of the experiment repeated with ordinary programmers are not published yet. A further study with professional robot programmers and an industrial robot platform is in preparation. This future study will focus on two main topics. First, deepening workload research by extending workload investigation (e.g., measurement of spatial ability, which was not done in this experiment) and the improvement of the prototype to allow more solid claims how AR affects the task performance (task accuracy in combination with task completion time – manual time/cognitive time, Towne [18]). Second, we want to address the question if it is necessary to develop industrial AR robotic interfaces with actual industrial robots when focusing on workload aspects, or whether this can also be researched with simpler robot platforms.

ACKNOWLEDGMENT

We thank Philipp Wimmer for his assistance with recruiting participants from cooperating companies.

REFERENCES

[1] R. D. Schraft and C. Meyer, "The need for an intuitive teaching method for small and medium enterprises," *VDI BERICHTE*, vol. 1956, p. 95, 2006.

[2] A. Gaschler, M. Springer, M. Rickert, and A. Knoll, "Intuitive robot tasks with augmented reality and virtual obstacles," in *IEEE Int. Conf. on Robotics and Automation, ICRA '14*, May 2014, pp. 6026–6031.

[3] D. Wurhofer, V. Fuchsberger, T. Meneweger, C. Moser, and M. Tscheligi, "Insights from UX Research in the Factory: What to Consider in Interaction Design," in *Proc. of Human Work Interaction Design*, 2015.

[4] J. Hallenberg, "Robot tool center point calibration using computer vision," Master's thesis, Dep. of Electrical Engineering, Linköpings University, Sweden, 2007.

[5] D. Perritaz, C. Salzmann, and D. Gillet, "Quality of experience for adaptation in augmented reality," in *IEEE Int. Conf. on Systems, Man and Cybernetics, SMC '09*, Oct 2009, pp. 888–893.

[6] R. Bischoff and A. Kazi, "Perspectives on augmented reality based human-robot interaction with industrial robots," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS '04*, vol. 4, Sept 2004, pp. 3226–3231.

[7] S. Abbas, S. Hassan, and J. Yun, "Augmented reality based teaching pendant for industrial robot," in *12th Int. Conf. on Control, Automation and Systems, ICCAS '12*, Oct 2012, pp. 2210–2213.

[8] C. Mateo, A. Brunete, E. Gambao, and M. Hernando, "Hammer: An Android based application for end-user industrial robot programming," in *IEEE/ASME 10th Int. Conf. on Mechatronic and Embedded Systems and Applications, MESA '14*, Sept 2014, pp. 1–6.

[9] A. Tang, C. Owen, F. Biocca, and W. Mou, "Comparative effectiveness of augmented reality in object assembly," in *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems, CHI '03*. New York, NY, USA: ACM, 2003, pp. 73–80.

[10] S. Hashimoto, A. Ishida, M. Inami, and T. Igarashi, "Touchme: an augmented reality interface for remote robot control," *Journal of Robotics and Mechatronics*, vol. 25, no. 3, pp. 529–537, 2013.

[11] M. Tönnis, D. A. Plecher, and G. Klinker, "Representing information – classifying the augmented reality presentation space," *Computers & Graphics*, vol. 37, no. 8, pp. 997–1011, 2013.

[12] S. G. Hart, "NASA-Task Load Index (NASA-TLX); 20 years later," in *Proc. of the Human Factors and Ergonomics Society*, 2006, pp. 904–908.

[13] NASA Human Performance Research Group and others, "Task Load Index (NASA-TLX) v1.0 computerised version," *NASA Ames Research Centre*, 1987.

[14] S. Kalyuga, P. Chandler, and J. Sweller, "Managing split-attention and redundancy in multimedia instruction," *Applied Cognitive Psychology*, vol. 13, no. 4, pp. 351–371, 1999.

[15] S. Kalyuga, P. Ayres, P. Chandler, and J. Sweller, "The expertise reversal effect," *Educational Psychologist*, vol. 38, no. 1, pp. 23–31, 2003.

[16] S. Kalyuga, "Expertise reversal effect and its implications for learner-tailored instruction," *Educational Psychology Review*, vol. 19, no. 4, pp. 509–539, 2007.

[17] J. Förster, E. Higgins, and A. T. Bianco, "Speed/accuracy decisions in task performance: Built-in trade-off or separate strategic concerns?" *Organizational Behavior and Human Decision Processes*, vol. 90, no. 1, pp. 148–164, 2003.

[18] D. M. Towne, "Cognitive workload in fault diagnosis." DTIC Document, Tech. Rep., 1985.