Geek Toys for Non-Techies? Using Robots in Introductory Programming Courses for Computer Science Non-Majors

Erica Weilemann
University of Applied Sciences
Neu-Ulm
erica.weilemann@hs-neu-ulm.de

Philipp Brune
University of Applied Sciences
Neu-Ulm
philipp.brune@hs-neu-ulm.de

Dany Meyer
University of Applied Sciences
Neu-Ulm
dany.meyer@hs-neu-ulm.de

Abstract

While LEGO® MINDSTORMS® robots and Arduino boards are widely used today in high school education to stimulate pupils' interest for technology-related subjects or to introduce beginner computer science students to programming, it is interesting how these tools may be successfully used also in programming education of technology-agnostic non-computer science majors.

A possible lack of motivation as well as of practical IT skills to handle the development environments for these tools here form a specific challenge.

Therefore, in this paper a learning environment and toolchain especially tailored for this target group is proposed and empirically evaluated in a classroom setting. Results indicate that with a proper setup and development environment these "geek toys" may be successfully used also for more technology-agnostic audiences.

1. Introduction

To stimulate the interest of kids in technology-oriented subjects in high school or as a didactic tool in computer science education at universities, "geek toys" like i.e. LEGO® MINDSTORMS® robots and Arduino boards are frequently used today ([1-4]). These tools provide effective means to visualize and learn complex topics in introductory programming courses as well as in advanced courses for real-time operating systems or embedded systems in a playful way ([3-6]). E.g. the concepts of object orientation (OO) and object oriented programming (OOP) can be learned more easily with the help of such physical visualizations or similar simulation approaches to visualize the program execution and make programming tangible ([4, 7, 8]). This suggests that it would be promising to use these concepts also in introductory OOP courses (in this case using the Java language) for non-computer science major students.

However, specific toolchains and development environments are required for programming and running these devices [9]. While this is no problem for technologically oriented students in computer-science majors, installation and usage of these tools to the authors' experience may form a serious challenge for technology-agnostic non-computer science major students (i.e., to establish the communication between a PC and a robot, specific network drivers may need to be installed and configured, a task which these students typically are not used to).

Students who learn IT-related topics only as non-majors frequently show a lack of intrinsic motivation and interest in these subjects [10]. The challenge therefore is not to discourage them from the very beginning by too complicated "technical" issues which are not primarily linked to the learning objectives.

One possibility to address these problems is the use of virtualization, providing the students with a pre-configured guest operating system as a virtual machine (VM) image [11].

However, few results have been published so far on how these approaches may be combined and put into action in a classroom setting in case of introductory OOP courses for non-computer science majors. In particular, the students' perception of using these tools and the respective benefits and problems from the students' perspective have not been systematically analyzed.

Therefore, in the present paper a learning environment and toolchain is proposed and evaluated for enabling 1st year non-computer science major Bachelor students to program and use the LEGO® MINDSTORMS® robots in a simplified way using the Java language with the Eclipse IDE. As hypervisor for the virtualized environment, VirtualBox[1] was selected due to its compatibility with various operating systems

[1] https://www.virtualbox.org/

1530-1605/16 $31.00 © 2016 IEEE
DOI 10.1109/HICSS.2016.13

and the fact of being open source. Linux Mint[2] distribution is used for the guest operating system, since its desktop is similar to Windows or Mac desktops and therefore suitable for beginners.

The proposed approach is evaluated from the students' perspective by an exploratory empirical study in the context of a teaching experiment: Is the effort of installing the toolchain reasonable from the students' point of view? Is the environment easy enough to use? And what was the overall perception of the approach regarding motivation, interest for the subject and learning outcomes?

The evaluated teaching experiment has been carried out in the 1st year of the Bachelor programs Information Management and Corporate Communications (IMCC) and Information Management Automotive (IMA) at Neu-Ulm University of Applied Sciences (Germany).

In their first study year these students should get acquainted with the object-oriented paradigm using Java. For this purpose, they had to work on a project assignment for two weeks using LEGO® MINDSTORMS® robots.

The results indicate that the present approach in principle is suitable to support an effective usage of these robots for OOP education of mainly technology-agnostic students in a classroom environment being satisfactory and motivating for the participants.

The rest of the paper is organized as follows, based upon the structure suggested in [12]: In section 2 the related work is analyzed in detail. Section 3 describes the design of the teaching experiment and the proposed development toolchain. Section 4 and 5 explain the execution and analysis of the learning experiment, respectively. In section 6, the obtained results are discussed and interpreted. We conclude with a summary of our findings.

2. Related Work

The use of LEGO® MINDSTORMS® robots and similar devices in programming education is rather common and has been studied by various authors in recent years ([13, 10, 14, 3]). Barnes discusses reasons why students prefer physical devices over graphical simulations. He argues that "controlling a physical model is likely to have much more appeal than controlling a graphical representation of what is, after all, meant to be a simulation of a physical model." [4].

Other approaches aiming to reduce the fear of programming and generate a more immediate impression of achievement have been discussed, i.e. tangible programs. Van Elten and Graef invented simple plastic bricks which enable a very easy introduction to programming by just connecting these bricks [15]. Horn created a tangible programming language – Tern [16]. The aim of Tern is "a painless introduction to computer programming for children in educational settings" [16].

Cliburn describes how he uses LEGO® MINDSTORMS® in a non-major Computer Science course. The aim is to teach the students algorithm design. For this purpose he uses the visual programming environment as provided with the robot sets. To increase participants' motivation, the described project consists of two challenges. His conclusion is that this was "Perhaps the author's most successful implementation of the Mindstorms in the classroom" [17]. So the usage of these robots is reported to be in principle suitable for non-major computer science courses [18, 19].

The usage of LEGO® MINDSTORMS® was also examined regarding their suitability for attracting more females to technology-oriented subjects ([20-23]). Robots used in the right way can raise the interest of female students for jobs related to information technology [22]. Müllerburg et al. report that female students are attracted by robots since they represent physical tools [24]. Xu argues that female students are more attracted when there is a "real life impact or human content" [25].

For introducing the Java language, Barnes uses the leJOS environment and the associated Java API3. Unfortunately, he does not describe all the tools around the usage of leJOS, e.g. if he uses an integrated development environment and in case which one. He also does not describe if the students have problems with the usage of the robots [4].

Lawhead et al. recommend the use of LEGO® MINDSTORMS® for teaching Java programming. They also use leJOS [9]. However, for loading the Java-program on the brick, they use a command-line interface, which may be problematic for non-computer science students.

Heuer et al. describe how LEGO® robots could be used by larger project teams. They also used Java the leJOS API. The robots were introduced in later semesters with students having already programming experience and thus were familiar with a development environment. The authors chose leJOS because of its compatibility with Windows, Macintosh and Linux systems. Students did not have to get accustomed to a new environment [13]. This paper supports the use of leJOS, but does not give any hint how to simplify the setup of leJOS.

[2] http://www.linuxmint.com

[3] http://www.lejos.org/

"The school computer lab is no longer the primary place of student computer use. Instead, students increasingly expect to use their own hardware to complete their school assignments." [26]. In the context of a LEGO® MINDSTORMS® project in combination with the leJOS API, this strongly increases the maintenance effort for the lecturer.

One solution could be to use virtualization, i.e. by providing the students with a pre-configured VM guest operating system image. Delman et al. report as one of the few the usage of a Virtual Machine and explain the reasons as follows. "The IDE and robotic software executes on a virtual machine running under the freely available software, Sun™ VirtualBox. This allows for a uniform programming platform for Windows, MacOS, and Unix/Linux." [27].

While the described related work illustrates that the application of LEGO® MINDSTORMS® robots in programming education has been studied with respect to different aspects, the particular challenge of how to apply these robots in an effective way for teaching OOP and Java to technology-agnostic non-computer science majors way has not been studied so far.

Therefore, in the following the questions are addressed how a learning setup and development toolchain for programming the robots with Java and Eclipse IDE for this particular target group may be implemented, and how this setup is perceived by the students.

3. Teaching Experiment Design

3.1. Goals, Hypotheses, Parameters, and Variables

One of our primary objectives is to make programming more attractive and interesting for our students. They consider themselves usually more as creative designers rather than software developers. Thus the idea of using robots which also can be physically designed looked promising. It was assumed that students would lose their fear of programming if they first could design and build an individual personal robot.

The purpose of the teaching experiment was to explore how the understanding of OOP and fundamental programming concepts like e.g. control structures and algorithms for our 1st year students could be improved by using robots. Thus, the robots should be usable without previous knowledge about computer science or embedded programming.

We assumed that our students would better understand how every single programming statement works by visualizing it using the robot, thus providing them a visual and haptic experience of their programs.

We also wanted to find out how to increase the identification and motivation of the students with their work and in best case create an emotional bond between the students and their robots.

For this reason and considering that our students are often attracted by visual creations and art, we arranged the programming tasks with a certain creative part, e.g. the students should design their own robot constructions or challenge course. Additionally, the students worked in self-organized groups and were allowed take home the robot equipment during the project time.

Finally, the students should realize that programming is a tool to implement cool applications and can help to solve realistic and exciting problems.

3.2. Experiment Design

We chose a setting for visualizing program sequences using an easy to use robot both in a simulated software environment as well as a real version.

For the robot simulator we used the NXTSim Java library, which simulates the leJOS environment moving around a little robot image in a display window.[4]

Both versions can be programmed to read sensors (light, touch and distance) and move in the same way. As the programming exercise for the students the following ficticious setting was chosen:

In a factory, a transport robot should deliver a part to three different work places. The transport robot has a suitable container for the load, which contains a sensor detecting its occupation state. After loading, the robot should move on a defined track to the work places to deliver its load. This setup consists of a sequence of following tasks:

a) Analyzing of the given problem and finding a first description of the algorithm (using UML diagrams). In this part the students should train their ability to abstract problems and formulate algorithms.
b) Implementation of the algorithm using the robot simulation. In this step students should apply their knowledge about programming especially using control structures like loops. Using the simulator, students receive a direct feedback how their algorithm works.
c) Switch to real robots: the teacher defines a robot challenge (competition) as an enhancement of the first task. Now the robot should not travel a fixed

[4] http://www.aplu.ch/home/apluhomex.jsp?site=45

track but find a given target using his sensors avoiding collisions with obstacles. To solve this challenge, groups are formed and every group receive a Lego Mindstorm robot, a construction set and the VM with the programming environment. To program the real robot, the students are using classes with exactly the same interface as the classes to program the simulated robot. This project phase takes two weeks.

d) Finally a competition between the student groups is organized to find the best solution, judging the solutions according to the criteria "creativity of the robot design" and "correctness and performance of implementation".

3.3. Participants

In our study, 1st year students from both study programs IMCC and IMA took part. They all attend the course "Programming Technique", which provides an introduction to object orientation and Java programming. Whereas the IMCC program has a remarkably high percentage of female students (over 75%), the vast majority (approx. 80%) of the IMA students are male.

Most of these students are absolute programming novices. Computer science-related topics are non-majors in these study programs. We encouraged all those students to take part in the survey. The participation was voluntary.

3.4. Objects

For using the robots, a stable, easy to use and portable development environment is required. To enable the students to work and learn self-organized and independent of the university's computer labs, the complete toolchain is provided to the participants as pre-installed VM guest operating system image. This VM images as a compressed file is approximately 2 GByte in size and is available for download to every student via the university's intranet.

As the virtualization hypervisor Oracle VM - VirtualBox[5] is used (version 4.3.8 or higher, including the extension packs), since it is available for free and supports most current host operating systems. We set up a VM image with a pre-configured Linux operating system (OS) (Linux Mint Version Maya desktop mate, 32-bit) including JRE (release 1.6) and Eclipse (Kepler Service Release 2) as integrated development environment (IDE). A screenshot of the guest OS is shown in figure 1.

[5] https://www.virtualbox.org/

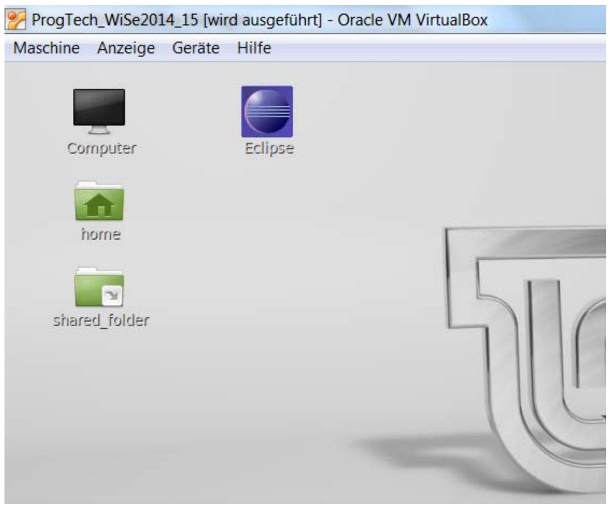

ProgTech_WiSe2014_15 [wird ausgeführt] - Oracle VM VirtualBox
Maschine Anzeige Geräte Hilfe
Computer
Eclipse
home
shared_folder

Figure 1. Screenshot of the Linux desktop in the pre-configured VM image provided to the students.

We installed leJOS for NXT (version 0.9.1 beta) and EV3 (version 0.8.1) and the corresponding Eclipse leJOS plugins for MINDSTORMS® NXT and EV3 as our students used both versions of the robot. The USB filter of the VM is configured to automatically detect connection options to a robot via USB and Bluetooth.

Corresponding to the leJOS versions we flashed the firmware of the NXT respectively prepared the SD-card with the image for the EV3 provided by the leJOS project.

The differences in the leJOS API for the NXT and EV3 bricks and the simulator are leveraged by a simplified and unified Java wrapper class provided to the students, thus hiding the API differences between the simulated and the NXT or EV3 robots and further unnecessary implementation details. All the wrapper classes implement a common interface (see Figures 2-4). This interface is very easy and contains only the basic operations to read sensors or move the robot.

The project mission for the students was to program only the class Robot using the operations given by the interface (see listing in figure 5). It shows a program to move forward a robot until an obstacle is detected, after that rotate 90° and travel 30 cm.

However, the simulator project could be used also without outside the VM by simply importing it as an existing project in an Eclipse on the host operating system. This is especially important during the early stage of the students' work. Starting with the simulator less frightening for the students compared to more complicated programing environments like the VM.

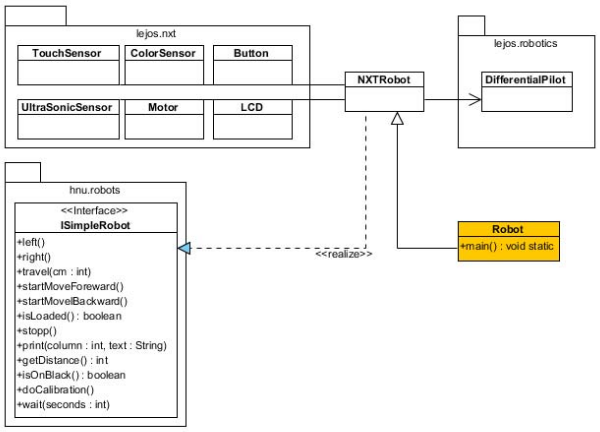


Figure 2. Class diagram of the NXT project

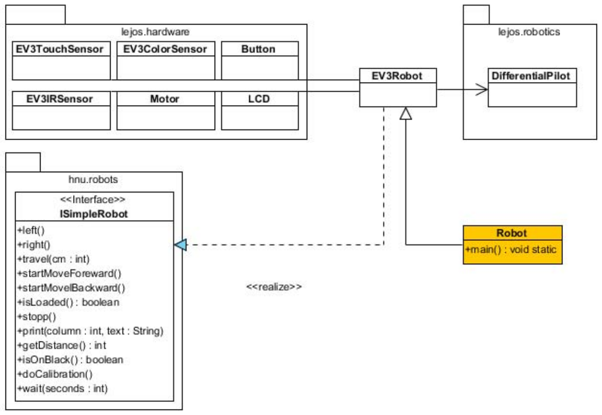


Figure 3. Class diagram of the EV3 project

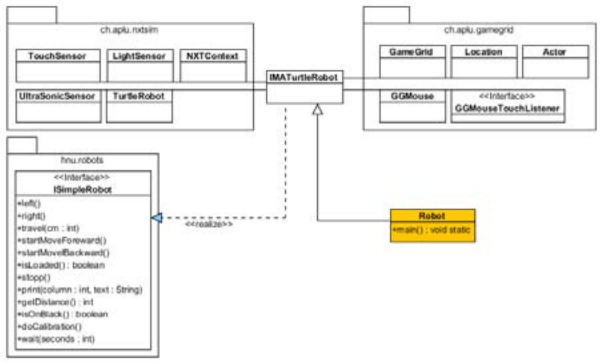
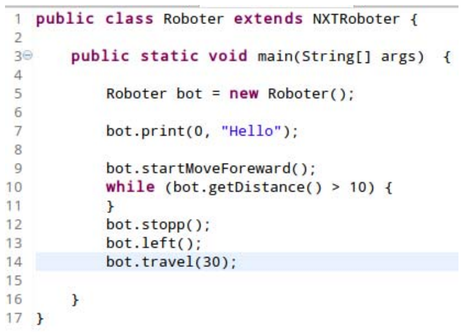


Figure 4. Class diagram of the robot simulator project

But nevertheless the simulator was including also in the VM Linux image to support students also in later stages of their project work.



```
public class Roboter extends NXTRoboter {

    public static void main(String[] args)  {

        Roboter bot = new Roboter();

        bot.print(0, "Hello");

        bot.startMoveForeward();
        while (bot.getDistance() > 10) {
        }
        bot.stopp();
        bot.left();
        bot.travel(30);

    }
}
```

Figure 5. Program listing of the robot class (NXT simulator version)

Our tests revealed that the detection of colors using the NXT color sensor is very inaccurate depending on different light conditions. Therefore we decided to use the color sensors only to detect light (white) or dark (black) environments. Therefore, the robot needs to be calibrated calling our procedure "doCalibration()" as first statement in the classes main method.

3.5. Instrumentation

The students were provided with a step-by-step instruction including screenshot images and pictures. Alessio Colombo from University of Trento was very helpful regarding this. He already created a similar instruction [28] and made it available to us together with helpful hints.

These step-by-step instructions describe how and where to download VirtualBox and the Extension Pack which is needed to enable the use of USB connections. It illustrates where to download the VM image, how to unpack it and how to open and start it in the VirtualBox as well as the needed VM configurations. It shows how to check the status of the USB connection to the LEGO brick.

Additionally, it gives the students hints what they should do if the USB connection could not be established and how to establish the connection manually.

As the students do not work with the VM all the time but also use their host PC, it describes how to transfer files between the VM and their host OS.

Then it explains the students how to create a Java class and how to upload it on the brick. Further instructions describe the specific methods of the robot-class.

3.6. Data Collection Procedure

The data collection was performed by a survey using a questionnaire and by a semi-structured in-depth interview with selected participants. The participants were the student group with the best result in the interview.

For the survey we handed out a paper questionnaire to all students who took part in the lecture. This questionnaire could be answered anonymously. The questionnaire included questions which concerned the handling of the VM and the Linux operating system. We asked the following questions:

1) How did you get along with VirtualBox?
2) Did there occur any problems in working with VirtualBox?
3) If yes, which ones?
4) How did you get along with the Linux operating system?
5) Did there occur any problems in loading the program on the brick of the robot?
6) If yes, which ones?
7) Which advantages and/or disadvantages does working with VirtualBox have from your point of view?

Questions 1) and 4) could be answered on a Likert scale from 1 to 6, with 1 corresponding to "very good" and 6 to "very bad". The scale was chosen like this because German students are used to it; in German schools grades range from 1 (very good) to 6 (fail). So they have an intuitive feeling for the scale. Questions 2) and 5) could be answered with yes or no. Questions 3), 6) and 7) required free text answers.

The two lecturers of the course handed out the paper questionnaires after all students handed in their solution to the task they had to work on for the previous two weeks. The students were given 10 minutes at the end of the lecture to answer the questions of the questionnaire. With these questionnaires we gathered immediate feedback.

For the in-depth interview, the interviewees were selected after the students submitted their solutions. We interviewed the group with the best result to avoid an immanent bias. By interviewing a group not being able to solve the project task, the students presumably would have reported mainly problems and not successes and solutions they did find.

On the other hand, results of the interviewed group also could have been biased in a positive way, in particular if we would have interviewed the students right after the end of the project. Students might have felt a very emotional sense of achievement then. However, our objective was to analyze the positive and negative aspects of the project. Thus, we conducted the interview one semester after the project was finished. So the students had the possibility to reflect on several aspects of the project. In addition, one semester was not too long so that the students had not yet forgotten details. They reported on positive experiences but also reported about the problems arose while working on the assignment.

We conducted a problem-centered interview [29]. As the main focus of this interview method lies on experiences, reflections and perceptions of the interviewee on a special problem, this method seemed perfect for our purposes.

The interview was recorded with the help of a dictaphone. For the coding of the open questions and the interview, the MaxQDA[6] software was used as a qualitative data analysis tool.

3.7. Analysis Procedure

We analyzed the data collected from the questionnaires using descriptive statistics. We computed the mean and median of the answers given on a Likert scale. We consciously also computed the mean because our students grow up with the grading system at school with grades from 1 to 6. During the school year, not only integers serve as grades but also intermediate stages of them. Averages are also not restricted to integers. So students perceive the given Likert scale rather continuous. This is why we did not restrict our analysis to medians.

From the paper based survey we also gathered impressions expressed in free text. We coded them as well, just like the interviews but we defined separate categories.

We transcribed the interview and coded it axial and selective in order to gather information not only about problems which occurred working with the VM but also about things which could be improved during the workflow.

3.8. Validity Evaluation

All paper questionnaires were handed out at the end of the lecture when all students handed in their results of the project work. The point of time was chosen this way so that we could gather all the "fresh" impressions from the students. In this results, we wished to include also the emotions of the students – whether they were angry or relaxed or had fun.

The point of time of the interview was chosen in a way that students already had enough time to reflect on the project. We did not aim to gather their emotions but first of all their elaborated impressions of the project.

[6] http://www.maxqda.de/

Students should tell us unemotionally where problems occurred and what we should change in the process or tools to simplify the work and avoid problems.

4. Execution

4.1. Sample

In the paper-based survey, 41 students took part. 22 of them are students of the study program IMCC, 16 are of the study program IMA, the remaining students did not give any information about their study program. Participants are aged between 18 and 27 years, where more than 75% are aged between 18 and 22 years. There were 21 female and 20 male participants.

The interview was conducted with 3 female students of the group which handed in the best solution for the project task. All of them were in their 2nd semester.

4.2. Preparation

The participants did not receive any special training. We offered the students help in form of a step-for-step instruction how to install and use the VM. And students had the possibility to visit us in our office hours or before or after a lecture in case they had questions, also concerning technical problems.

4.3. Data Collection Performed

The data collection was performed as described. We handed out the project tasks and students had three weeks of time – two weeks of Pentecost break included – for working on the project task and thus working with the VM. At the end of the three weeks, students handed in their solutions to the project task via Moodle (our learning management system) and in the following lecture we handed out the questionnaires.

In addition, we interviewed three of the participants one semester after the project work.

4.4. Validity Procedure

We managed to hand out the questionnaires in the scheduled time. So we were able to gather "fresh impressions" from the students.

Moreover, we interviewed three participants of the course one semester later to gather reflected feedback.

5. Analysis

In the survey, 41 students participated. 22 of them were from the study program IMCC, 16 from the study program IMA, the remaining 3 did not give any information. The overall average of all female participants was 51%. In IMCC, 73% of the participants were female, in IMA the amount was 25%.

All students were in their first year, 94% of them were in their first semester, 6% in the second semester. Those students were probably repeating the course due to previously failing the exam. 72% of the students enjoyed the project.

The following questions related to the usage of the VM were answered by 35 students:

1) How did you get along with the VirtualBox?
   57% of the students answered in the range of 1 (very good) to 3 (satisfactory). 31% answered in the range of 5 (fail) to 6 (inadequate). The median of answers was 3, the mean 3.4 and the standard deviation 1.6.
2) Did there occur any problems in working with the VirtualBox?
   49% of the students answered with yes, 51% with no.
3) If yes, which ones?
   Students mentioned that installation took a very long time. Also students stated that starting the VM was sometimes not possible or that the starting process took a long time. The VM crashed several times. A connection to the robot was not possible. Eclipse did not save the workspace. And the public folder did not work.
4) How did you get along with the Linux operating system?
   51% of the students answered in the range of 1 to 3. 17% answered in the range of 5 to 6. The median of the answers was 3, the mean 3.4 and the standard deviation 1.2.
5) Did there occur any problems in loading the program on the brick of the robot?
   28% of the students answered with "yes", 72% answered with "no".
6) If yes, which ones?
   Some students encountered the problem that the robot hang. It took a long time till the program was loaded. And some students could even not manage to load the program on the brick.
7) Which advantages and/or disadvantages does working with the VirtualBox have from your point of view?
   Answers here were sparse, especially concerning the advantages. The mentioned advantages were a new user interface and the possibility to start right

there with the work where you stopped before you switched off the VM.

The named disadvantages were that an extra installation is required. The installation was very elaborate. It took too much time and students did not enjoy it. Students had the solutions stored on only one PC so the other participants in the group could not test their individual solutions. If there occurred problems, the new environment was not helpful. Students were not able to handle the problems on their own.

Feedback of the students concerning the learning success was as follows:

Many students said that they saw the impact of loops live. They saw how action changes when they change the loop. This is probably the reason why some students said that the project was a very good preparation for the exam.

On the question what students think they learned with the help of the project, they answered that they learned teamwork, not to give up too early, logical thinking and the realization of a theoretical thing, a better understanding of programming and what can be achieved with programming.

But there were also some students who answered that the project was a huge loss of time. This results from having problems with technical things like constructing the robot or installing and working with the VM.

Results obtained from the coded in-depth interview were in agreement with these observations (coded phrases are italic):

Students also reported on *problems* which occurred during the *installation of the VM*. All of them agreed, that the installation was the hardest part of the whole project. The *consequence* was a *work overload* for the particular student with the working VM. Students felt *awkward* and *desperate*. They *communicated* via WhatsApp[7] – they used *voice mail* as well as *videos* which showed as interim result the working robot. According to the students this *raise their motivation*.

They suggested several *improvements*: A *video tutorial* and a *more detailed step-to-step instruction*. This should be accompanied by a *seminar* where problems with the installation are solved with the help of an instructor. This could also happen in *individual consultation hours*.

The three students also reported that initially they were absolutely not attracted by the task to construct a robot. A robot was a *too technical* thing for them. They reported that they were motivated by the *creative* part of the task – to use their creativity in constructing the robot. They concluded that this part of the task made the robot attractive and they enjoyed the project.

[7] https://web.whatsapp.com/

6. Interpretation

6.1. Evaluation of Results and Implications

Regarding the feasibility of the approach, the biggest challenge turned out to be the installation of the VM hypervisor and VM image. The subsequent handling of the provided VM guest OS itself was considered manageable by many of the participants, even though most of them had no previous experience with Linux. Uploading the software to the brick was no problem for the majority. So the toolchain in general seems to be helpful and practically feasible if the VM installation procedure could be improved.

In contrast to previous results [8], the in-depth interview revealed that the robots are not perceived as motivating and helpful by female students just because they are physical objects. One reason might be, that females see "the robot as more machine-like" and are "not socially facilitated by the robot while engaged in the arithmetic tasks" [30]. Additional measures like using a stimulating and interesting task and enclosing "story" for the design of the robot need to be taken by the lecturers to turn the design of the robot into a stimulating experience.

6.2. Limitation of the Study

The validity of the study is limited mainly by the number of participants and the fact that the empirical study was performed only in one university and within the two Bachelor study programs IMA and IMCC.

Further research is needed to repeat and enlarge the survey and to perform a comparative evaluation with other universities and study programs.

6.3. Inferences

Since the robots were used mainly in the last weeks of the course, the previous overall perception of the course, the lectures and the teachers by the individual student might affect the perception of the robots and the task to do with them and therefore the answers given in the survey.

6.4. Lessons Learned

The installation problems of the VM hypervisor should be addressed in the future by the help of video tutorials provided by the lecturers.

7. Conclusions and Future Work

In conclusion, we presented a learning setup and development toolchain for beginners OOP education in non-computer science majors using LEGO® MINDSTORMS® robots, the Java language and Eclipse IDE. The approach was evaluated in by an explorative empirical study in the context of a teaching experiment carried out a German University of Applied Sciences.

7.1. Relation to Existing Evidence

The findings from the described teaching experiment are in agreement with those made in previous studies, i.e. regarding the need for also providing the students with a robot simulator to shorten the edit-compile-upload-run cycle: “It is amusing to reflect that the availability of a simulator would help here!” [4].

As has been emphasized before, we also identified the ease of use and portability of the toolchain to be a key requirement: “From the student perspective, our solution had to: a) Run on a range of host systems; b) Be easy to install; c) Be easy to use and maintain; d) Minimize side-effects on the host system; e) Provide a stable experience throughout the semester.” [26]

The concepts and materials we provided as help to the students, which have been either used already in the experiment or suggested by the interviewed participants (i.e. video tutorials are also similar to those other studies [11]).

7.2. Impact

Since previous studies also report similar observations in related contexts, it can be concluded that the obtained results not only indicate that the proposed approach helps to improve the teaching of OOP programming for undergraduate students with non-computer science majors, but may be also transferred to similar courses in other universities.

One important finding is the fact that for technology-agnostic students of non-computer science majors, the use of physical devices like the robots alone is not sufficient for improving motivation and understanding for OOP. Further boundary conditions have to be met to create an emotional relation of the students with the device and thus the subject taught. However, our findings strongly encourage to continue and extend the described style of teaching for this target audience.

7.3. Limitations

Nevertheless, the described concepts require further empirical validation. The most important limitation is the current restriction of the empirical study to only one semester of two Bachelor degree programs at one university. This limits the number of participants in the study and therefore the validity of the results obtained. Therefore, the study should be repeated in subsequent semesters within the same study programs as well as within other related study programs at other universities. In addition, the impact of e.g. gender, cultural and educational background or age of the students on the perception and effectiveness of the approach needs to be analyzed in more detail.

7.4. Future Work

Therefore, it is planned to continue the usage of the presented approach in the IMA and IMCC study programs, improving it according to the findings made and further evaluate it in the described way in future semesters. In addition, the approach should be transferred and evaluated to other universities and study programs in the near future and be evaluated there in a similar way to be able to compare the results.

8. Acknowledgements

The present work as part of the EVELIN project was funded by the German Federal Ministry of Education and Research under grant number 01PL12022E. The authors are responsible for the content of this publication.

9. References

[1] R. Goldman, A. Eguchi, and E. Sklar, “Using educational robotics to engage inner-city students with technology,” in Proceedings of the 6th International Conference on Learning Sciences, Santa Monica, California, 2004, pp. 214–221.
[2] J. Ruiz-del-Solar and R. Aviles, “Robotics courses for children as a motivation tool: the Chilean experience,” IEEE Transactions on Education, vol. 47, no. 4, pp. 474–480, 2004.
[3] S.H. Kim and J.W. Jeon, “Introduction for Freshmen to Embedded Systems Using LEGO Mindstorms,” (da), IEEE Transactions on Education, vol. 52, no. 1, pp. 99–108, 2009.
[4] D.J. Barnes, “Teaching introductory Java through LEGO MINDSTORMS models,” SIGCSE Bull, vol. 34, no. 1, pp. 147–151, 2002.
[5] N. Correll, R. Wing, and D. Coleman, “A one-year introductory robotics curriculum for computer science upperclassmen,” IEEE Transactions on Education, vol. 56, no. 1, pp. 54–60, 2013.

[6] R.U. Pedersen, J. Norbjerg, and M. P. Scholz, "Embedded programming education with Lego Mindstorms NXT using Java (leJOS), Eclipse (XPairtise), and Python (PyMite)," in Proceedings of the 2009 Workshop on Embedded Systems Education, Grenoble, France: ACM, 2009, pp. 50–55.

[7] D. Ewert, D. Schilberg, and S. Jeschke, "Problem based learning of object-oriented Programming with LEGO Mindstorms and leJOS," in Communication and Cybernetics in Science and Engineering Automation 2011/2012, S. Jeschke, I. Isenhardt, F. Hees, and K. Henning, Eds.: Springer Berlin Heidelberg, 2013, pp. 315–323.

[8] P. B. Lawhead, M. E. Duncan, C. G. Bland, M. Goldweber, M. Schep, D. J. Barnes, and R. G. Hollingsworth, "A road map for teaching introductory programming using LEGO© mindstorms robots," in Working group reports from ITiCSE, 2002, pp. 191–201.

[9] A. Heuer, K. Lauenroth, V. Stricker, and K. Phol, "Entwicklung eingebetteter Software in einem Softwarepraktikum mit Lego Mindstroms," in Software Engineering im Unterricht der Hochschulen, 2009, pp. 115–129. (in German)

[10] D. Cliburn, "Experiences with the LEGO Mindstorms throughout the Undergraduate Computer Science Curriculum," in Proceedings. Frontiers in Education. 36th Annual Conference, pp. 1–6.

[11] A. B. Williams, "The qualitative impact of using LEGO MINDSTORMS robots to teach computer engineering," IEEE Transactions on Education, vol. 46, no. 1, p. 206, 2003.

[12] B. Bagnall, A. Shaw et al, leJOS / EV3 Wiki / Home. Available: http://sourceforge.net/p/lejos/wiki/Home/ (2015, Jun. 15).

[13] D. Shell and L.-K. Soh, "Profiles of Motivated Self-Regulation in College Computer Science Courses: Differences in Major versus Required Non-Major Courses," Journal of Science Education and Technology, vol. 22, no. 6, pp. 899–913, 2013.

[14] A. Gaspar, S. Langevin, W.D. Armitage, and M. Rideout, "March of the (virtual) machines: past, present, and future milestones in the adoption of virtualization in computing education," Journal of Computing Sciences in Colleges, vol. 23, no. 5, pp. 123–132, 2008.

[15] C. van Elten and J. Graef, Tangible Programs — Interaktionsgestaltung Portfolio: Arbeit im Fach Invention Design 1 bei Prof. Jörg Beck. Available: http://ig.hfg-gmuend.de/Members/christian/meine-projekte/begreifbare-programmierung (2015, Jun. 08). (in German)

[16] Tufts University Human Computer Interaction Lab, Tern - Tangible Programming. Available: http://hci.cs.tufts.edu/tern/ (2015, Jun. 08).

[17] A. Jedlitschka and D. Pfahl, "Reporting guidelines for controlled experiments in software engineering," in 2005 International Symposium on Empirical Software Engineering, 2005, pp. 92–101.

[18] M. L. Brake and A. Tessmer, "Robots and girls," IEEE Antennas and Propagation Magazine, vol. 46, no. 1, pp. 142–143, 2004.

[19] O. Timcenko, and A. Friesel, "Competition-motivated teamwork and narratives to motivate girls for engineering," in 2011 Proceedings of the 22nd EAEEIE Annual Conference, Piscataway, NJ: IEEE, 2011, pp. 1–6.

[20] H. Schelhowe and H. Schecker, "Wissenschaftliche Begleitung des Projekts ROBERTA - Mädchen erobern Roboter." 2005. (in German)

[21] M. Müllerburg, J. Börding, G. Theidig, and U. Petersen, "Informatikausbildung, Roboter und Mädchen," pp. 143–147. (in German)

[22] D.C. Cliburn, "A CS0 course for the liberal arts," SIGCSE Bull, vol. 38, no. 1, p. 77, 2006.

[23] H.R. Arabnia, V.A. Clincy, A. Bahrami, and A.M.G. Solo, Eds, Proceedings of the 2010 International Conference on Frontiers in Education: Computer Science & Computer Engineering, Las Vegas, Nevada, USA: CSREA Press, 2010.

[24] A. Sayler, D. Grunwald, J. Black, E. White, and M. Monaco, "Supporting CS education via virtualization and packages," in the 45th ACM technical symposium, pp. 313–318.

[25] A. Delman, A. Ishak, L. Goetz, M. Kunin, Y. Langsam, and T. Raphan, "Development of a system for teaching CS1 in C/C++ with Lego NXT robots" in Proceedings of the 2010 International Conference on Frontiers in Education: Computer Science & Computer Engineering, Las Vegas, Nevada, USA: CSREA Press, 2010, pp. 396–400.

[26] S. Jeschke, L. Knipping, M. Liebhardt, F. Muller, U. Vollmer, M. Wilke, and X. Yan, Eds, Integrating Medical Robotics in the Robinson Program., 2008. ICALT '08. Eighth IEEE International Conference on Advanced Learning Technologies, 2008.

[27] A. Colombo, "Virtual Machine for Lego Mindstorm," 2013.

[28] D. Xu, D. Blank, and D. Kumar, "Games, robots, and robot games," in Proceedings of the 3rd International Conference on Game Development in Computer Science Education, pp. 66–70.

[29] A. Witzel, "Das problemzentrierte Interview," in Qualitative Forschung in der Psychologie: Grundfragen, Verfahrenweise, Anwendungsfelder, G. Jüttemann, Ed, Weinheim: Beltz, 1985, pp. 227–256. (in German)

[30] P. Schermerhorn, M. Scheutz, and C. R. Crowell, "Robot social presence and gender," in Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction, 2008, p. 263-270.