published as: Martin Malchow, Jan Renz, Matthias Bauer, Christoph Meinel: Improved E-learning Experience with Embedded LED System, In Proc. of the 2016 Annual IEEE Systems Conference (SysCon '16), 18-21 April, 2016, Orlando (USA)

Improved E-learning Experience with Embedded LED System

Martin Malchow, Jan Renz, Matthias Bauer, Christoph Meinel
Hasso Plattner Institute (HPI)
University of Potsdam
Potsdam, Germany
Email: {martin.malchow, jan.renz, matthias.bauer, christoph.meinel}@hpi.de

***Abstract*—During the last years, e-learning has become more and more important. There are several approaches like teleteaching or MOOCs to delivers knowledge information to the students on different topics. But, a major problem most learning platforms have is, students often get demotivated fast. This is caused e.g. by solving similar tasks again and again, and learning alone on the personal computer. To avoid this situation in coding-based courses one possible way could be the use of embedded devices. This approach increases the practical programming part and should push motivation to the students. This paper presents a possibility to the use of embedded systems with an LED panel to motivate students to use programming languages and solve the course successfully. To analyze the successfulness of this approach, it was tested within a MOOC called "Java for beginners" with 11,712 participants. The result was evaluated by personal feedback of the students and user data was analyzed to measure the acceptance and motivation of students by solving the embedded system tasks. The result shows that the approach is well accepted by the students and they are more motivated by tasks with real hardware support.**

I. Introduction

MOOCs enjoy great popularity with typically several thousand attendees in a course. Nevertheless, a major problem of motivation exists. Most of the courses have dropout rates of around 30% to 40%. Whereas, an active user is defined as a user completing at least 50% of the assignments and exams. Furthermore, in average only around 18% of the students are able to finish the courses with a certificate. Students are granted a certificate when they solve more than 50% of the assignments and exams correctly [1][2]. This paper focuses on an embedded approach to motivate users to increase the activity and certificate rates. This is caused by the motivation problem which occurs especially in the e-learning context besides learning in school, at university, etc. [3].

The idea of this paper is to motivate students by interactive programming. Especially, young students which are focused to learn a programming language or network programming. In our approach we describe students' influence of a real device to change its status. This can lead to additional motivation instead of just programming on a standard PC or a virtual machine with calculating something. Due to the use of the real device in combination with a camera the users can see results of their program instantly in a live captured image.

This paper deals with the question: Can we improve the learning experience in MOOCs with assessments based on real hardware support?

II. Related Work

The Cornell University has built a MOOC course for Microcontroller programming [4]. This course is focused on developing software for embedded systems using the Arduino platform. In the described MOOC there are 4 different labs which get more and more difficult. At the end of each lab users can upload their results and source code. The grading process is handled by a peer review of other students. This system of students reviewing students is clever to avoid overloading the teaching assistants. Nevertheless, this peer review idea is not done automatically which takes some time to evaluate the result by the student and when students report an evaluation the teaching assistants have to put additional effort in handling the reported evaluation and regrading the students' result.

Another approach is the introduction of gamification into MOOCs by openHPI to keep students motivated over a long term [5]. Therefore openHPI introduces badges, points and leader boards. Badges can be used as self-test rewards, points for up votes in questions and answers in the user forum. Finally, the leader board should be established. It should be relevant and related. Only close competitors who can possibly be reached and friends should be shown. So, users are motivated to compete with other students and have a solvable challenge. High scores are not the right solution in this approach due to the reason that students would think they cannot reach the high scores because they started the course later. So, there is no real motivation for them to go on due to the high score.

The Tele-Lab idea is to create virtual machines where users have an environment to test security features [6]. Therefore, students can access the virtual machines via VNC. In this secure area security scenarios like Trojan behavior can be tested by the students without any risks.

Further research in this area focuses on managing access of several test systems and a load balancing of student requests [7]. To handle several students on limited hardware with an estimated execution time it is possible to define the necessary number of resources. Furthermore, scheduling and test of hardware usage have to be considered to avoid code executions of different students at the same time on the same device. Otherwise the result could be completely random and non-deterministic for every student sharing hardware.

III. Approach

This section describes how an embedded system can be integrated in MOOCs with programming tasks and how students can reach points. It starts with the conceptual overview and goes on with the implementation of the embedded system and the infrastructure to handle programming the board and LED display in MOOCs.

A. Concept

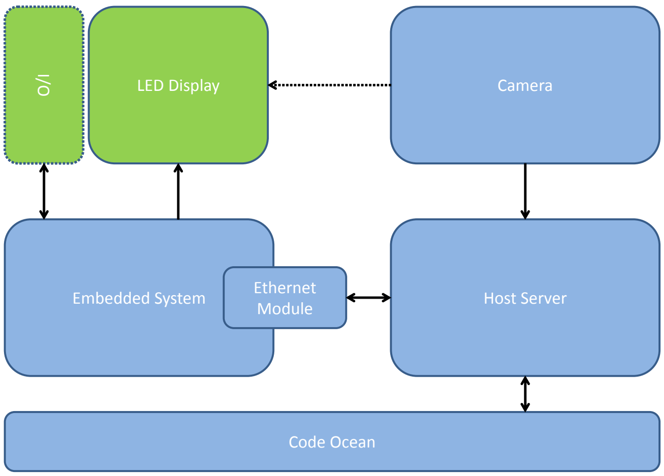

Fig. 1. Concept Overview

The concept overview is shown in Figure 1. The heart of the system is the embedded system. Which is in our case an STM32F4DISCOVERY board manufactured by STMicroelectronics[1]. This board uses a STM32F407VGT6 microcontroller and have several integrated I/O (input/output) peripheries like buttons, LEDs and a 3-axis accelerometer and plenty of I/O pins. These I/O pins can be used to connect additional I/O devices. In our case the I/O pins are used to connect the Ethernet module with an ENC28J60 chip and the LED display SLM1608MD2. The Ethernet module is used for communication with the host system. The basic idea is that the host sends an array message for the LED display matrix. After the board has received the array and checked it for the validity successfully the array content will be displayed on the LED matrix using the I/O pins of the board and LED matrix for the connection. Finally, the host system receives a message of success as answer. If the validity fails, the answer will indicate an error in the received array and the LED display will not change its content. After the host system has received a message of success, it will enable the camera which then will take an image of the LED display with the new submitted content.

Still, this system has no MOOC integration where students shall have the opportunity to write their code. Therefore, the web application Code Ocean is used in this approach which allows programming directly in the web browser. Code Ocean offers an LTI (Learning Tools Interoperability) interface to integrate learning tools and MOOCs like Coursera[2] and edX[3] to enable evaluation of implementation tasks. Students' implementation can be done in Code Ocean in several programming languages which are supported by Linux. Students can test their implementation directly in Code Ocean and grade their code automatically. The automated grading is done with Unit tests. In Java a possible unit test can be realized with JUnit.

[1]http://www.st.com

[2]https://www.coursera.org/

[3]https://www.edx.org/

To connect Code Ocean with our embedded system Code Ocean communicates with the host server of the embedded system and sends the array over a TCP socket to the host server. After this, the host server communicates with the embedded system like it is described in the beginning of this section. As answer the host server forwards the message of the embedded system and if the array check was successful the answer of the host server to Code Ocean additionally contains the picture of the LED matrix for the sent student's array.

B. Implementation

The system set up is shown on the following Figure 2. On the left the STM32F4DISCOVERY board is shown next to the small Ethernet module using an ENC28J60 chip which is next to the 16x16 LED display on the right.

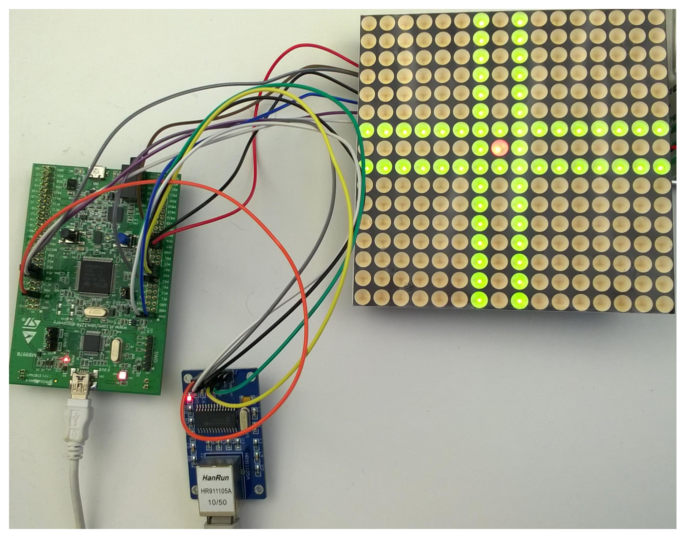

Fig. 2. Overview

1) Board: First the STM32F4DISCOVERY microcontroller has to be implemented after the LED display and Ethernet module have been connected by several cables. The implementation language is C which is typically used for embedded systems. As basis the "STM32F4DISCOVERY peripheral firmware" sources are used to simplify basic programming tasks on the embedded STM32F4 microcontroller like controlling LEDs and accessing I/O pins.

To control the LED matrix a driver had to be written which uses several pins to connect pins of the display like the clock, red LED, green LED, bright and reset. The board calls an interrupt every 10ms to set all 256 LEDs of the display. Therefore, the reset pin will be triggered and reset. Now every clock cycle the corresponding LEDs are activated one by one from left to right with the corresponding red and green LED pins. If both pins are set at the same time the LED will be orange due to red and green enabled LEDs. After every line the "bright" pin is set to enable the LEDs like they were set before. Due to the behavior the LEDs have to be turned on by the "bright" pin. For every line this has to be triggered by the interrupt every 10ms. Otherwise, the display would be turned on only shortly if new data were added and turned off again nearly instantly. The interrupt is also reasonable due to the fact that the Ethernet module could block the update of the display because of the large amount of data. By using an interrupt it can be guaranteed that the display task is run in time.

To handle the Ethernet module an additional driver has to be written to enable the TCP connection. Therefore, the "ENC28J60 EtherShield Library for Arduino"[4] is used and adapted to work with the STM32F4DISCOVERY board. This library handles incoming TCP connections. On the board a data stream validation is processed. A valid data stream starts with "MATRIX " followed by 256 digits which are between 0 and 3 describing every LED color line by line. The meaning of the numbers is 0 equals off, 1 equals green, 2 equals red and 3 equals orange. After these 256 digits there is an "X" marking the end of the data stream. If this stream validation is passed, the LED Matrix will be displayed and a success message will be sent by the microcontroller. Otherwise, the microcontroller would send an error message as answer to the TCP request.

2) Server: The server handles students' requests, takes care of the camera snapshots and synchronizes access of multiple students at the same time. This server runs on an Ubuntu 14.04 LTS with PHP environment[5] and a small super-server daemon xinetd[6]. Furthermore, fswebcam[7] is used to capture the image of the LED matrix with the camera. Every request is handled by the PHP script which uses the PHP function "flock()" to lock a file and realize exclusive access to a part of the PHP script. This enables exclusive access to the embedded system. As long as the file is locked and a student is accessing the embedded system and taking a camera shot of the LED display all other student requests will cycle as long as the request is done and the file is released. This could lead to an additional execution time but so wrongly captured images for students are avoided.

3) Code Ocean: Code Ocean is a web-based platform for practical programming exercises that was developed by the openHPI team to be used within the context of MOOCs. The code execution is handled within language specific Docker images, so it can handle multiple languages. For this course a Java-specific environment has been used. After the code is submitted to the server it is executed in the isolated container environment. Furthermore, tests can be executed, usually to grade the users' code submissions.

Due to the security reasons normally all network traffic from the execution container would be prohibited. Due to the fact that the lab server is addressed via TCP/IP an execution environment with network access was implemented for this experiment. To maintain the system security specific firewall rules were introduced, which only allow network traffic to the lab server's IP address while all other traffic is declined. As the code execution may take longer than the lab execution time a pool of execution environments is booted and then waits for code execution.

IV. Experiment

As experiment we integrated the embedded system into a MOOC course called "Java for Beginners" with 11,712 enrolled users. Until week 3 of the course conditions, loops and arrays were discussed. Therefore, in week 4 the embedded system tasks were released to consolidate this knowledge with the following matrix tasks. The idea of using the embedded system was to motivate students to implement the application and have fun doing so. This experiment focuses on evaluating if the embedded system can enhance learning motivation for implementing an application with students' personal feedback and analyzed feedback.

A. Tasks

There were two tasks implementing applications on with the embedded system students have to solve during the course. They need to send the correct array to display it on the LED matrix. They can solve it by setting every position in the matrix manually or by using loops and conditions to reduce the necessary source code.

To solve the first task correctly students had to draw a red line on the LED Matrix which goes from the left top to the right bottom (see Figure 3 on the left).

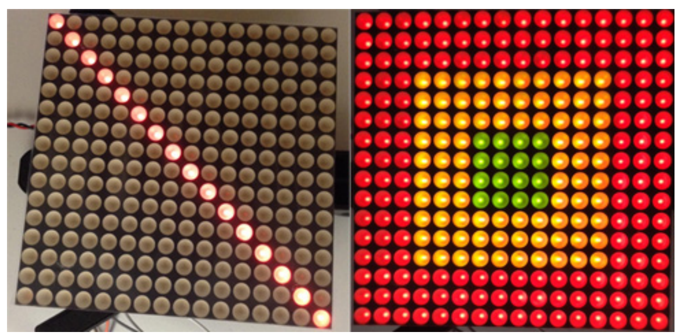

Fig. 3. LED Matrix red line and squares task

For the second task the students had to create three squares to display it on the LED matrix like it is shown in Figure 3 on the right.

Like described in Section III-B the display can be controlled by the students within Code Ocean. During the course "Java for Beginners" the programming language Java was used. Students can use the already existing Java method in a hidden file which creates the necessary bit stream out of an array and send it to the IP address of the embedded system. As advantage of the hidden file is that the IP of the embedded system is not available to the public and it avoids "Denial of Service" attacks. To use this Java method a proper 16x16 array has to be created and filled with the correct values for the corresponding task. Possible values for every array field were described in Section III-B1. The given source code in Code Ocean to solve these tasks is shown in Listing 1. The students have to initialize the array for the given variable *ledMatrix* in the size of the LED matrix and fill this array with the corresponding integer

[4]https://github.com/thiseldo/EtherShield
[5]http://php.net/
[6]https://github.com/xinetd-org/xinetd
[7]http://www.sanslogic.co.uk/fswebcam/

values. Finally, the *sentToBoard* method will send an array to the board and show the given corresponding output on the LED display.

Listing 1. Java source code as starting point to solve the LED matrix tasks

```
public class DisplayBoard {
    public int[][] ledMatrix;
    public DisplayBoard()
    {
        //Your code starts here

        TCPClient.sentToBoard(ledMatrix);
    }
    public static void main(String[] args) {
        new DisplayBoard();
    }
}
```

This architectural design of the embedded system with the corresponding server is enabling students to produce fast results with already learned knowledge about arrays, loops and conditions. Furthermore, the camera enables students to get fast feedback of the result of the LED display they controlled by themselves.

B. Feedback

During the course we got a lot of feedback from the users. Here we discuss some reactions of the LED matrix task. Since this course was held in German the feedback was translated into English.

"The last matrix task is so much fun. I believe that I tried all possibilities until I got the right result. I'm more the "Trial and Error" guy when it comes to tasks like this. Every new right lamp (or also complete row or column) was celebrated with cheers.... :-) :-) :-) Anyway, great challenge with a lot of fun!!! MORE of such experiments, PLEASE!!!"

"The LED matrix task which was more practical in comparison to the other tasks encourage me to experiment more than needed for the current task requirements. As always during programming applications I am coming from the butcher over the baker to the candlestick-maker. That is fun and before you know it you are engaged with entirely new stuff! Hence, thank you from my side!"

"Hello, I would like to thank you. Especially, for the LED matrix task. An awesome experiment. I particularly like that there is an different output with a real visible "result" of the implementing work in contrast to the normal text output of the other programming tasks. This task also helped me personally. I was very unsure about my current advanced course decision in school and my future plans. This course shows me how much fun I have with programming and also returned my hidden motivation and commitment. Therefore, I want to thank you so much. Best wishes"

"Awesome task! I get a kick out of it and I had several laughs. This task tempts certainly to try out something new. @HPI-Team awesome idea."

"That is the way programming makes way more fun. Please more interactive tasks in the following programming course."

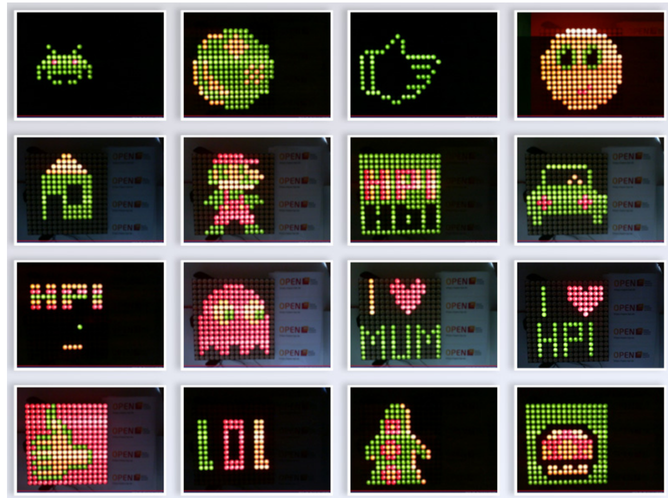

Fig. 4. Creative additional and voluntary matrix solutions of the students

"The idea with the real LED matrix motivates me. It would be nice if in further practical courses, such real devices would be used."

As these comments highlight that several students will not stop implementing Java after solving the LED matrix tasks. They went on programming different symbols and images on the matrix. A collection of creative results is shown in Figure 4.

Since not all students post comments in the discussion board we also provided a survey with several questions regarding the acceptance of the LED matrix tasks. The questions and answers of about 1500 students are visualized in Figure 5.

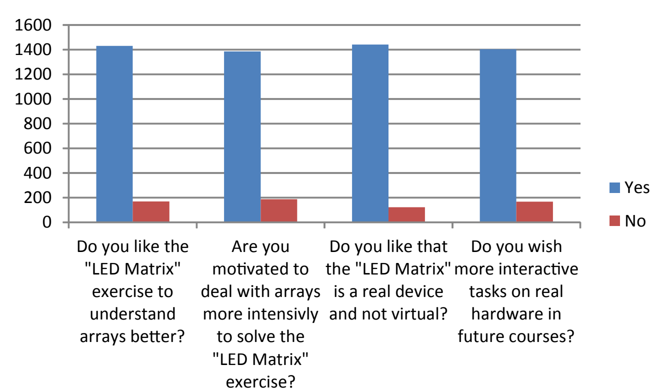

Fig. 5. Survey LED matrix acceptance

As an obvious result the majority of students enjoy the usage of the LED matrix during this MOOC and would like to have it in other further programming courses. It motivates them additionally to solve these tasks and also to implement voluntary further applications making it a personal goal to implement something new. This motivation leads to a better understanding of the programming environment.

Finally, one user also offered a Java application to improve the creation of LED images by clicking into a virtual matrix, setting the points and printing the matrix to add it directly in

Code Ocean and display the result. This additional tool uses the Java GUI (Graphical User Interface) elements "AWT" and "Swing" which were not discussed in the course. This student was so motivated to implement applications for the LED matrix that he learned programming Java GUIs voluntarily.

C. Analysis

In total 2,129 users at least once submitted code to the server. 26,653 submissions have been submitted for the first task, 27,951 submissions for the seconds. An average amount of 25.64 submissions has been made per user. This includes submissions for testing the code as well as those just running the assessment tests used for grading (on these test runs the array was not sent to the lab server, so no images have been created here).

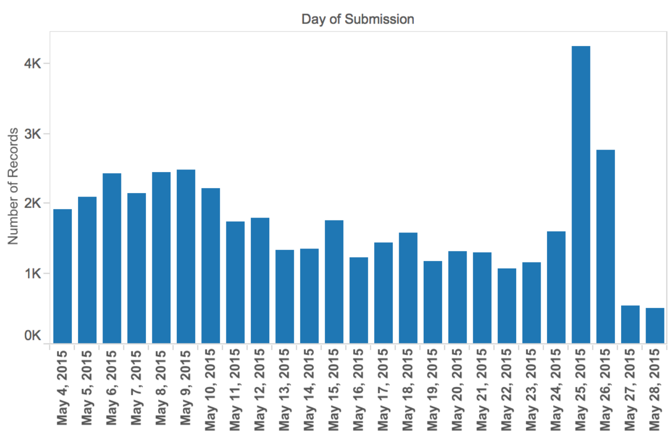

Fig. 6. LED Matrix submissions per day

Submissions are distributed throughout all days, peaking before the submission deadline. After that deadline the users were still able to run the code, but they could not receive points within the MOOC course anymore. On the 25th of May, 1 day before the deadline 4,242 submissions per day have been handled by the setup. This leads to the hypothesis, that while many users liked the type of the assessment and many used it to play around, still the main motivation was to earn some bonus points in the course context. This can be backed by the small amount of submissions after the deadline was passed. Therefore, it seems helpful to embed the lab based tasks like this in a course context, not only to provide knowledge needed to solve the lab assessment, but also to generate additional motivation.

To identify possible effects of the assessment towards the quality of the course completion (measured by the total amount of points per user achieved within the course) a possible correlation between the lab assessment and the course completion as shown in Figure 7 was investigated. This shows that nearly all users who had multiple submissions passed the course. It also shows that many users who passed the course with very good results (so they might have a very good understanding of the Java language) still submitted a large amount of submissions. In combination with the user feedback this can be explained by the fact that many good users used the provided assessment environment in an explorative manner, playing around even after solving the original assessment task. Furthermore a strong ($p < 0.0001$) correlation between the number of submissions and the course result could be found.

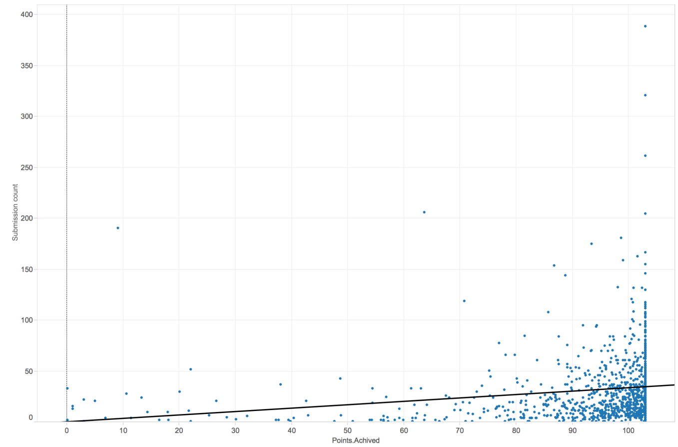

Fig. 7. Correlation between submissions and received points

D. Scalability

As this experiment took place in a MOOC environment the scalability of the lab setup was an important precondition. Every execution with in the lab took 2 seconds. Most of the time is used to initiate the camera and to take the picture. So on the busiest day the lab server was busy for approximately 8492 seconds or ca. 10 percent of the day. A simple queuing approach is used to handle the concurrent requests. If the request could not be processed within 10 seconds, an error message is presented to the user. Also the limited size of execution environments helped to avoid this situation. While the user would still be able to edit the code, after submitting the code another error message ("All executions environments are busy") was displayed.

While the availability of the system had no the bottleneck in this setup, the code execution through the Code Ocean, including uploading the code, compiling and running tests before sending the code to the lab led to a certain unresponsiveness, especially for the students who tend to code in an trial and erroneous manner. One course participant implemented a virtual matrix lab. He provided the Java code as a download within the course forum. The virtual lab allowed users to run their code in a local Eclipse IDE, a graphical output was then generated and displayed to the user.

Assuming this lab should be used in larger courses multiple actions could be used to improve the capacity of the lab. The simplest idea is to connect multiple units (each containing camera and board) to the lab server and let the lab server do some simple load balancing. Due to the low cost of the hardware this is an easy solution. As most of the time is used to initiate the camera, this time could be significantly decreased by keeping the camera always on and just capturing a picture signal from the camera's video stream once the matrix board has processed the input data.

Furthermore, caching could be introduced to generate a picture output without using the hardware setup. This caching could be very effective for the correct solution or typical coding errors. However, as the camera setup was built so that it is obvious that this is a real setup and no simulation (providing some stickers on the desk and an illumination that was heavily influenced by the daylight) this caching idea should respect the current daytime.

V. Results and Future Work

In the introduction of the paper we asked the question: Can we improve the learning experience in MOOCs with assessments based on real hardware?

The answer is yes! Based on the students' feedback and evaluation in Section IV-B it is obvious that students are satisfied with this approach. Furthermore, it allows explorative learning which is intensely used by the students and was proved by the additional submissions shown in Figure 4. Finally, also the analysis in Section IV-C shows how motivating this task is. An interesting fact is that students solving this interactive LED Matrix tasks are more successful in the complete course which is visualized in Figure 7. As a result, using real hardware in MOOCs teaching a programming language is reasonable to increase students' motivation and the learning experience. With a certificate claim rate of 28% this course is also clearly over the average which was discussed in the introduction in Section I.

Even though the system is already running successfully there is some future work to do. To increase the stability of the system the connection of the microcontroller with the LAN module and LED matrix should be handled with a printed circuit board (PCB) instead of the cable connections. This would look more professional and is more fault tolerant if something or someone hits the system. Additionally, the system should be connected with sensors and actuators which can be read or controlled by the students. As an example a temperature sensor can be used and evaluated. Furthermore, when enabling additional embedded systems a router has to be used to scale and allocate the embedded systems for the students. This will decrease an overload of one embedded system and decrease waiting time for students if simultaneous hardware access occurs.

References

[1] F. Grunewald, E. Mazandarani, C. Meinel, R. Teusner, M. Totschnig, and C. Willems, "openhpi - a case-study on the emergence of two learning communities," in *Global Engineering Education Conference (EDUCON), 2013 IEEE*, March 2013, pp. 1323–1331.

[2] C. Willems, N. Fricke, S. Meier, R. Meissner, K. Rollmann, S. Voelcker, S. Woinar, and C. Meinel, "Motivating the masses - gamified massive open online courses on openhpi," in *EDULEARN14 Proceedings*, ser. 6th International Conference on Education and New Learning Technologies. IATED, 7-9 July, 2014 2014, pp. 4042–4052.

[3] J. Keller and K. Suzuki, "Learner motivation and e-learning design: A multinationally validated process," *Journal of Educational Media*, vol. 29, no. 3, pp. 229–239, 2004. [Online]. Available: http://dx.doi.org/10.1080/1358165042000283084

[4] M. Dai, "Mooc for microcontroller design course," Ph.D. dissertation, Cornell University, 2015.

[5] J. R. C. M. Thomas Staubitz, Sebastian Woinar, "Towards social gamification - implementing a social graph in an xmooc platform," in *Proc. 7th International Conference of Education, Research and Innovation*, Sevilla, Spain, 0 2014.

[6] C. Willems and C. Meinel, "Awareness creation mit tele-lab it-security: Praktisches sicherheitstraining im virtuellen labor am beispiel trojanischer pferde." in *Sicherheit*, 2008, pp. 513–532.

[7] A. Rasche, P. Troger, M. Dirska, and A. Polze, "Foucault's pendulum in the distributed control lab," in *Object-Oriented Real-Time Dependable Systems, 2003. WORDS 2003 Fall. The Ninth IEEE International Workshop on*, Oct 2003, pp. 299–299.