95 A ROBOT EXERCISE FOR LEARNING PROGRAMMING CONCEPTS

Johannes Holvitie[1], Riku Haavisto[1], Teemu Rajala[1], Erkki Kaila[1], Mikko-Jussi Laakso[1] & Tapio Salakoski[1]

1 University of Turku, Turku, Finland, {jjholv, rahaav, temira, ertaka,milaak,sala}@utu.fi

ABSTRACT

Different methods of visualizing code execution and its relation to various state changes have been perceived useful in programming education. However, due to the nature of program visualizations, they are generally hard to successfully and efficiently integrate into different programming teaching curricula. ViLLE is an educational tool, capable of being used in teaching basic programming concepts via utilizing visualizations coupled with different forms of collaboration, immediate feedback and automatic assessment. Previous studies conducted with the tool have shown that it can be effectively and successfully used in its domain area. This article introduces a new exercise type as an extension to the ViLLE platform. Dubbed as the robot exercise, the main focus of the exercise is in providing visualizations for different types of repetitive tasks, commonly found throughout program implementations. Further, by providing the user with an easily comprehensible visual counterpart, the emphasis is in optimizing the crane or robot movement and visually demonstrating the outcome of possible programming hiccups as crane malfunctions.

Keywords: Program visualization, Repetition, Best-practices, Optimization,

1 INTRODUCTION

Visualizations promote programming learning. An exciting visual representation can help in motivating and engaging the students. Furthermore, the visualization can make the control flow of a program easier to follow. ViLLE is an online education platform featuring several different exercise types, some of which use visualization. However, the visualizations are currently limited to debugger-like information of the state in which the program is. While that information is vital for thorough understanding of how certain program works, it might be too verbose to optimally engage a novice programmer.

In this paper we present a new exercise type, in which the program code written by the students control a robot. The robot is animated and moves by the commands given in the student code. The students get immediate visual feedback of their code quality from the movements of the robot. As the animated robot is more visually thrilling than mere state representation of various variables affecting a program state, it is expected to be more approachable for a novice programmer. In the first version of the robot exercise, means to concretize repetition structures are provided. The main teaching goal is to help students to understand how different repetition structures work, and how their incorrect usage might lead to very long-running or non-terminating programs.

2 RELATED WORK

As stated in [1], automatic assessment, combined with immediate feedback, can substantially help students to understand the essential programming concepts. We are able to pursue this by following ViLLE format for exercise building and submission, discussed in greater detail in section 4.2. The importance of engaging the student while the program executes has been discussed in [2]. Achieving high student engagement is of top priority, and for that reason the student is given access to the execution flow controls. Access to multiple views during execution, code marking and dynamic rewriting are discussed in [3]. These features might be implemented in later versions, if studies conducted later with the current version encourage their integration. Further, [4] discusses how the execution controls must be comprehensive enough, and how the user interface must be clear and easily approachable [5] in order to avoid user frustration. Finally, the ability to provide language-independency is discussed in [6]. The program responsible for robot visualizations is capable of building animations based on strings representing command listings, and is therefore controllable by any language capable of producing them.

3 VILLE

ViLLE is a collaborative education platform, developed at the University of Turku. As mentioned previously, ViLLE is used as the implementation and distribution - as well as data gathering – tool in development of the robot exercise. The following shortly describes the most significant features of ViLLE in relation to the new exercise type:

Language-independency: the possibility to view exercises, as well as answer them in different programming languages, is an essential feature in ViLLE. Being able to switch effortlessly between different languages further enhances the learning experience and provides a clear linkage between commonalities of programming languages.

Flexible visualization controls: to further increase the learning process, the user may control the execution speed and direction of the visualizations in ViLLE. This feature was also adapted into the animation controls of the robot exercise.

Thin client distribution: ViLLE supports a method of building the exercise so that it can be distributed with minimal amount of prerequisites towards the user. The only requirement for using ViLLE is that the client has a JavaScript capable browser, which makes it easier to deploy ViLLE to new environments. Additionally, copies of the system can be run in local area networks – for example in exam situations where the access to internet needs to be restricted.

The student is presented with a clear user interface, providing instant access to all needed elements of the undergoing exercise. Questionnaires are simple and highlight the portion from which the question originates. After answering a question, the student receives immediate feedback (providing that the exercise creator has added feedback to the question). Further, upon completing an entire exercise the student gets to see a final draft of his or her performance before submitting.

The teacher side provides an easy-to-use editor for making the exercises. Teachers are also presented with tools to organize courses and rounds, and a comprehensive view with data gathered from a multitude of tracking points. The data gathered is available for review in both raw- and graph-forms. This is perceived especially useful in tracking the success of exercise presentation, as well as student performance and knowledge in different areas.

3.1 Previous studies

We have previously studied ViLLE in various occasions: the effectiveness of ViLLE was studied in [7] and in [8]. We found out, that ViLLE is especially useful for novice programmers, when the usage occurred in higher levels of engagement. This is supported by Laakso [1], who states, that automatic assessment and immediate feedback can substantially support the process of learning to program. We have also studied the collaborative use of ViLLE [9], and found out, that co-operation can be a beneficial form of learning when using program visualization tools. Further, the student opinions collected [10] support the results mentioned earlier: students seem to think that ViLLE is both effective and motivational tool to use. The complete list of publications can be found at ViLLE's web site at http://ville.cs.utu.fi.

4 THE ROBOT EXERCISE TYPE

With the new exercise type we wanted to provide students with a clear and inspiring visual presentation of repetition in programs. The most important learning goal of the exercise type is to understand the control flow and traversed value range in conditional repetition statements. To encourage an efficient use of source code space – and optimization in general – an optional penalty system was implemented: dynamic feedback can be provided based on the length of the program code and the cost of robot movement.

The idea of the exercise is to program a robot to move a number of indexed boxes from predefined starting locations into the required goal locations, while minimizing the number of steps needed. The robot itself starts from a predefined position and length. The three "arms" have different costs for their movement, relational to their position in the crane. A single command results in a single movement of an arm, which in its turn results in this cost being added to the total. The robot automatically skips commands that would make the crane hit edges of the area or other objects in the field.

4.1 Editor View

The editor is used for making new instances of the exercise. In the case of the robot exercise, this requires the creator to define the starting positions for the crane, as well as the start and goal positions for the boxes. Additional configurations include adjusting the number of boxes used, limiting the movement options for the crane, and tuning the aforementioned penalty system.

The basic editor view is represented in Figure 1: the three-part crane is on the left side of the screen, where two of the arms appear as orange honeycomb steel pylons and the last arm as a wire with a magnet attached to it. All of the arms have similar functionality: they may move up or down, or left or right in relation to their parent arm. Additionally, all arms may extend or

withdraw. If an arm is a parent of another arm, a command issued to this arm may resolve into several arms moving at once, which again increases the cost of the step.

One of the most important things to notice when creating a new exercise is the initial positioning of the crane regards to the initial and goal position of the boxes. This part of the crane movement corresponds to the series of commands usually issued outside of any repetitive structures, and is only done once.

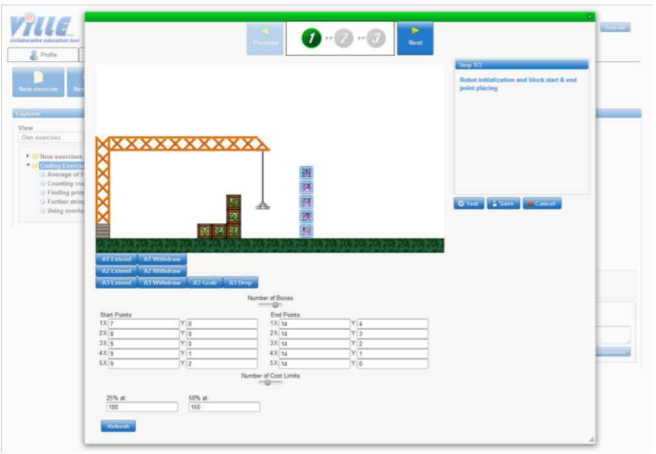

FIGURE 1. *The editor for a new robot exercise.*

The editor view also makes it possible to add cost limits. Exceeding these limits will reduce the student's score accordingly. These limits make it possible to reward control structures that do not take large amounts of space or require several costly commands to complete.

4.2. Student View

The student side of the robot exercise consists of two phases. The view in the first phase contains the crane in its starting position and the initial (brown) and final (blue) position of the boxes. To the right of this view is the programming area. Students use it to write a program that executes the required robot steps. Additional instructions are usually provided for the students, including the (possibly limited) set of commands usable for controlling the robot. The first phase view of the student side is presented in Figure 2.

When a student is finished writing the control code, the robot animation can be started by pressing the submit button. In phase two, a popup view opens, displaying the same initial state for the robot and the boxes. Instead of the coding area a set of execution flow controls is displayed. These may be used to control the speed and direction at the robot animation.

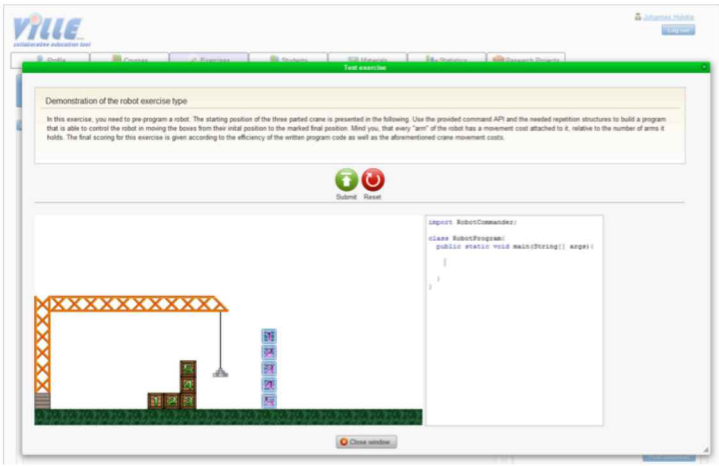

FIGURE 2. *the initial student view of the of the robot exercise.*

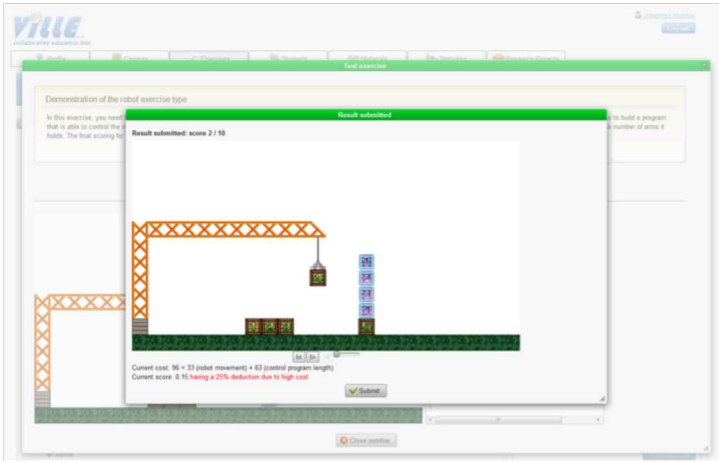

FIGURE 3. *the robot movement and exercise submission phase.*

Additionally, the final view shows the final score as well as keeps count of the live score relational to the robots movements so far. The point additions from placing boxes to correct goal positions and the possible penalties from exceeding cost limits are also displayed.

5 EXPECTED BENEFITS

By providing the students with an exciting representation of repetitive command execution in programs, we expect to promote the following learning goals:

1. Repetition control, especially value range definition and the consequences of unsuccessful definition.
2. Optimization, the importance of acknowledging command and algorithm costs as well as

cost multiplication due to repetition.
3. Importance of acknowledging a problem with a valid iterative and / or recursive solution, and the ability to implement it in a reasonable amount of time.
4. Efficient source code space usage and the ability to provide a solution that properly utilizes the expressiveness of the target programming language.
6. Conclusion

We have built a new exercise type for providing a clear and approachable method for understanding the basics of repetitive structures and command execution in programs. By integrating this newly built type into an already existing platform, we avoid the additional ground work, while bringing in several studied and tested features. We expect this new exercise type to provide the same level of ease of use and utilization as the other exercise types present in ViLLE do.

We acknowledge that to fully achieve the expected benefits discussed in section 5, a more evolved version of the new exercise type might be needed. The exercise will be studied and adjusted by the results of the studies. However, the exercise in its current state already contains the theoretically essential components for achieving the aforementioned expectations. As mentioned earlier, we are also planning on studying the robot exercise thoroughly. Later, required fine-tuning is to be made based on the results of the studies.

7 ACKNOWLEDGEMENTS

The authors would like to acknowledge the inspiration for the Robot exercise, Charles Thevathayan. His robot exercise has been used regularly in the teaching of Programming 1 in the School of CS & IT, RMIT, Melbourne, Australia.

REFERENCES

[1] M.-J. Laakso, Promoting Programming Learning: Engagement, Automatic Assessment with Immediate Feedback in Visualizations. TUCS Dissertations 131. Turku Centre for Computer Science, 2010.
[2] C. D. Hundhausen, S. A. Douglas and J. T. Stasko, "A Meta-Study of Algorithm Visualization Effectiveness", Journal of Visual Languages & Computing, Vol. 3, No. 1, pp. 259-290, 2002.
[3] T. Okamura, B. Shizuki, and J. Tanaka, "Execution Visualization and Debugging in Three-Dimensional Visual Programming". Proceedings of the 8th IV'04.
[4] R. E. Mayer, Multimedia Learning. Cambridge University Press, Cambridge, UK, 2001.
[5] M. Petre, "Why Looking Isn't Always Seeing: Readership Skills and Graphical Programming", Communication of the ACM, Vol. 38, No. 6, pp. 55-70, 1995.
[6] A. Moreno, N. Myller, E. Sutinen and M. Ben-Ari, "Visualizing Programs with Jeliot3", AVI'04, Gallipoli, Italy, May 25-28, 2004.
[7] T. Rajala, M.-J. Laakso, E. Kaila and T. Salakoski, "Effectiveness of Program Visualization: A Case Study with the ViLLE Tool", Journal of Information Technology Education: Innovations in Practice, Vol 7, pp. 15-32, 2008.
[8] E. Kaila, M.-J. Laakso, T. Rajala and T. Salakoski, "Evaluation of Learner Engagement in Program Visualization", 12th IASTED Internation Conference on Computers and Advanced Technology in Education (CATE 2009), St. Thomas, Virgin Islands, USA, November 22-24, 2009.
[9] T. Rajala, E. Kaila, M.-J. Laakso and T. Salakoski, "Effects of Collaboration in Program Visualization", Technology Enhanced Learning Conference (TELearn 2009), Taipei, Taiwan, 2009
[10] E. Kaila, T. Rajala, M.-J. Laakso & T. Salakoski. Effects, Experiences and Feedback from Studies of a Program Visualization Tool. Informatics in Education, Vol. 8, No. 1, pp. 17-34, 2009.