COMPARING PLAY-BASED LEARNING TO GRAPHICAL PROGRAMMING IN PROGRAMMING EDUCATION

Rolf Lindén¹,³, Ana García Díaz², Erkki Kaila¹, Erno Lokkila¹,³, Mikko-Jussi Laakso¹

¹ University of Turku, Department of Future Technologies (FINLAND)
² Universidad Complutense de Madrid, Department of Didactics (SPAIN)
³ UTUGS Graduate School (MATTI), University of Turku (FINLAND)

Abstract

Programming and computer science are nowadays taught to younger and younger children. Still, the methodology applied for programming education in elementary school is usually graphic programming, with environments such as Scratch and Alice being highly popular. We have previously conducted a study about teaching programming concepts to ten-year-olds in Finland. In the study, play-based learning with kinesthetic exercises was compared to graphic programming. According to these previous results, the students using play-based approach outperformed the graphic programming group statistically significantly in the post-test.

In this study, we repeated the study with the same approach in a Spanish elementary school. A setup was kept as similar as possible to that in Finland, except for translating the materials into Spanish. Surprisingly, the results are quite different to the results in Finland, as in Spain, neither of the groups showed significant differences between the pretest and the posttest results. We discuss about the possible causes of the differences between the Finnish and Spanish experiments in detail.

Keywords: Programming education, play-based learning, graphical programming, elementary schools, introductory programming.

1 INTRODUCTION

Nowadays, introducing technology into classrooms has become a worldwide trend. Most of the countries in Europe offer subjects such as basic computing science or technology in secondary schools. However, the expanding tendency is to introduce this kind of knowledge in elementary schools as well.

In Spain, both of the subjects named above have been offered to teenagers for more than 26 years [6] but programming has never been part of the official curriculum. The provided knowledge about technology needs to be constantly renewed, because the technological environment evolves so rapidly, and its influence in peoples’ lives is constantly growing. Therefore, some Spanish schools are starting to offer basic programming lessons to Primary School children. Regarding the direction society is taking towards the inclusion of programming learning in elementary schools, seeking for the best method to teach it becomes a need.

The experiment that is presented in this paper reproduces a previous research done in a Finnish Primary School with a similar setting in Spain. Two classrooms from a Spanish Primary School participated to the study. In one of them a play-based teaching method was used and in the other one, a graphical programming environment was used. The aim of the study was to gather enough data to be able to generalize the results obtained in Finland, where the play-based teaching turned out to be the best method when explaining programming concepts to children.

The paper is structured as follows: first, a theoretical framework and the related work are presented, followed by an explanation of the methodology. Next, the results of the study are presented with a discussion about them. Finally, conclusions, possibilities and need for the future research are included.

2 RELATED WORK

While countries like Finland have introduced programming in their curriculum [1], Spain keeps teaching basic computing knowledge, focused on teaching teenagers how to use the Office package
and different search engines [5]. However, some private schools are introducing programming as part of the computing subject or as an after-school activity. It is common to use a graphical programming environment to teach the basic concepts, normally the one called Scratch [8]. Nevertheless, no research has been conducted to search for the best methodological choice to teach programming in elementary schools in Spain.

To understand this study, it is necessary to quickly explain the general teaching methods used in Spain. Most of the schools in Spain are content-based. Therefore, the main teaching method used is rote learning [7], since the current law of education [5] encourages this methodology. Hence, the pupils are not used to active learning lessons. Yet, there is an expanding tendency to incorporate active lessons in the rigid curriculum, looking to develop self-regulated learners, active and critical citizens [7]. Having access to technology in classrooms has helped teachers to change their teaching methods and to develop self-regulated children. Having programming as a part of the curriculum can help students to understand where the society is heading.

Graphical programming has been studied quite extensively during the recent years. Scratch [8], probably the most popular one of the graphical programming environments, has been found useful for learning in some cases (see [9], [10], [11] and [12] for example). Still, as pointed out in [13], the environment might not always be useful for programming education, as the students might manipulate the media without understanding the connection to programming. Other popular graphical environments used to teach programming are for example Alice [14] and Greenfoot [15].

Kinesthetic learning exercises [16] are activities in which the students act scenes or play games requiring physical activities. The utilization of active learning is supposed to help students to learn new topics and concepts easily. One implementation is the CS Unplugged [17], in which easy games targeted for children are used to teach computer science related topics. Again, the results of utilizing the methodology are mixed (see for example [18], [19] and [20]). Hence, the definitive method for introducing programming and computer science to younger students is still to be found.

2.1 Earlier Study

Our research group has conducted a preliminary study about the comparison of play-based learning and the graphical programming in Finland [1]. In the study, two groups of students, aged 10, were taught introductory programming skills with two experimental approaches: one of the groups (GPC, N=14) utilized graphical programming environment, based on Google’s Blockly [3], while the other group (PBC, N=17) utilized play-based, kinesthetic exercises to learn the same concepts. Both groups answered to a pre-test before the experiment, and a post-test with similar types of questions afterwards. In addition to programming related problems, both tests contained survey questions about knowledge and attitudes about programming. The post-test also surveyed the students about the experiment. The total time reserved for the experiment was 90 minutes for both groups, respectively.

There were practically no differences between the groups in the pre-test. However, in the post-test, the play-based group outperformed the graphical group in all three questions. Still, the difference was statistically significant only in the second question, which was about repetition. The total difference in post-test between the groups was also statistically significantly better in favour of the PBC group. According to students’ perceptions, both groups found programming more interesting and less challenging after the experiment, regardless of the methodology applied. Moreover, both groups found the session entertaining: the average of answers to question “how fun did you have in this lecture” got averages 4.0 (GPC) and 4.6 (PBC) in a Likert scale of 1 to 5. Hence, both methods seemed to be beneficial in improving students’ attitudes towards programming.

3 METHODOLOGY

The main goal of the study was to ascertain which of the two teaching methods used was preferable in explaining basic programming concepts to Primary School pupils. The following chapters describe the research setup in detail.

3.1 Participants

Two fourth grade classes participated in the study. There were 23 and 25 children (aged nine) in the classes, respectively, so that overall 48 children took part in the study. None of the attendees had any previous programming training. Classes were randomly assigned the treatment. The classroom called
4A used the graphical programming environment (from now on GPC, N=25) while 4B was taught with a play-based programming lesson (PBC, N=23). The same person supervised and conducted both lessons.

3.2 Procedure

The full experiment lasted for 90 minutes for both groups. In the beginning of the session, students were given about 10 minutes to complete the pre-test questionnaire to measure their initial programming knowledge. After the pre-test, the same introduction about programming was given to both groups. However, due to the differences in the methodology, the final part of the introduction was a quick preface about the treatment used. In the GPC, the digital environment was presented while the PBC was given a short introduction about the different activities.

After the introduction, students spent 65 minutes practicing programming with the different methodologies. The session was finalized with students answering the post-test. The post-test was similar to pre-test with one additional, more challenging programming task.

3.3 Programming environments

The graphical programming environment used in the study was built for the earlier study at late 2014 and early 2015 [1, 2] and was based on the free Blockly library [3] offered by Google. In the environment, the solution to the problem is built by using program code blocks that are dragged and dropped into desired positions. The constructed program can be executed at any time. Hence, the environment resembles for example Scratch (Resnick et al. 2009), quite popular graphical programming environment, but was designed to be as simple and easy to adapt as possible. The language used in the user interface was originally Finnish, but English and Spanish translations of the system were created for the study. The localized environment is displayed in Fig. 1.
The play-based lesson was structured in three different activities. The first one was aimed to familiarize students with the concept of algorithm while building an origami frog. The second one was called “The Labyrinth”. In the activity, the students learned the concept of commands, introducing additionally concepts such as loop clauses. Finally, students played a card game to understand what conditional clauses mean.

3.4 Materials

The pre-test contained two programming questions, offered in graphical form. In the questions, a grid was displayed with a circle displayed in one of the squares. In addition, some pseudo code was attached next to the grid. The students were asked to determine and draw the final position of the ball after the code was executed. In the post test, similar questions were used, accompanied with one more challenging task. An example of the tasks, translated into English, is displayed in Fig. 2.

For the analysis, the answers to the programming tasks were evaluated in the scale of 0 to 5 points, based on their correctness. In addition to the programming tasks, the students were asked to answer a survey about their previous knowledge of, and attitudes towards, programming. These questions are displayed in Table 1.

4 RESULTS

The students’ responses from both the graphical programming class and the play-based class were analysed as a post-hoc analysis. First, the similarity of students’ attitudes and skills were evaluated by comparing the students’ responses to the pre-test survey (see Table 1 for details). This revealed what kind of differences there were between the two groups’ initial attitudes and skills. The students’ skills in programming were then compared between the two groups (PBC and GPC), by comparing the respective changes in the students’ skills between the pre-tests and post-tests. The similarities in the pre-test survey responses and programming skills led to the analysis of the post-test survey responses, which were consistent with the other results. The following sections go through these findings in more detail.

4.1 On the Similarity of Students’ Skills and Attitudes Before the Test

In order to verify the similarity of the two selected test groups, the score distributions of the pre-test results for the two programming exercises were compared between the groups using Kolmogorov-Smirnov test. For both programming exercises, the GPC and PBC distributions were deemed statistically similar with each other (Programming Exercise 1 p-value = 1, Programming Exercise 2 p-value = 0.7919).

The similarity of the students’ responses to multiple choice survey response items were evaluated by comparing the response counts for the GPC and PBC (see Fig. 3). The responses show great similarity between the two groups, and this visually apparent similarity was verified by using Chi-Square test for each survey item pair between the GPC and PBC. As none of the p-values were below 0.05, there was no reason to apply corrections to further control the Type I errors. The analysis showed that there are no statistically significant deviations between the two groups’ responses before the experiment.
Table 1. The questions in the pre-test and post-test explained. In the Type column, "cat." stands for categorical. If a hyphen is presented in the Pre-test or Post-test columns, the question was not included in that test. The Levels column indicates the number of choices in the question, if there were any.

<table>
<thead>
<tr>
<th>Pre</th>
<th>Post</th>
<th>Type</th>
<th>Levels</th>
<th>Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>interval</td>
<td>-</td>
<td>Student ID</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>interval</td>
<td>-</td>
<td>Internal paging identifier</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>text</td>
<td>-</td>
<td>Name</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>interval</td>
<td>-</td>
<td>Age</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>cat.</td>
<td>2</td>
<td>Gender</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>cat.</td>
<td>2</td>
<td>Have you programmed previously?</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>cat. + text</td>
<td>5</td>
<td>What is programming?</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>cat. + text</td>
<td>6</td>
<td>Who can be a programmer?</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>cat.</td>
<td>4</td>
<td>Which of the following requires programming?</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>cat. + text</td>
<td>5</td>
<td>What do you think about programming?</td>
</tr>
<tr>
<td>13</td>
<td>12</td>
<td>cat.</td>
<td>3</td>
<td>Do you find learning programming similar to learning mathematics?</td>
</tr>
<tr>
<td>14</td>
<td>13</td>
<td>cat.</td>
<td>3</td>
<td>Do you think that knowledge about programming could be useful to you in some other subject, or for example with your hobbies?</td>
</tr>
<tr>
<td>15</td>
<td>14</td>
<td>ordinal</td>
<td>5</td>
<td>How difficult do you think programming is? (if you have never programmed, tell how difficult you imagine it to be)</td>
</tr>
<tr>
<td>16</td>
<td>15</td>
<td>ordinal</td>
<td>5</td>
<td>Are you interested in learning programming?</td>
</tr>
<tr>
<td>-</td>
<td>16</td>
<td>ordinal</td>
<td>5</td>
<td>Did this lesson increase your interest towards programming?</td>
</tr>
<tr>
<td>-</td>
<td>17</td>
<td>ordinal</td>
<td>5</td>
<td>Did you find the things that were taught during the lesson easy or difficult?</td>
</tr>
<tr>
<td>-</td>
<td>18</td>
<td>ordinal</td>
<td>5</td>
<td>How fun was it during the lesson? (frowny face = didn't have fun at all and the smiley one = had a very good time)</td>
</tr>
<tr>
<td>17</td>
<td>19</td>
<td>grid</td>
<td>-</td>
<td>In which square will the ball reside once the program has finished? (first)</td>
</tr>
<tr>
<td>18</td>
<td>20</td>
<td>grid</td>
<td>-</td>
<td>In which square will the ball reside once the program has finished? (second)</td>
</tr>
<tr>
<td>-</td>
<td>21</td>
<td>grid</td>
<td>-</td>
<td>In which square will the ball reside once the program has finished? (third)</td>
</tr>
</tbody>
</table>

Figure 3. Students' responses to the multiple-choice questions in the pre-test survey for both graphical programming class (GPC) and play-based class (PBC). Each vertical column represents a single response item (question) in the survey. For both GPC (red) and PBC (blue), size of the circle represents the proportion of the students in the respective group that responded with the specific answer. Purple areas signify overlap between the two groups. As can be seen from the figure, the students in the two groups had very similar responses to the survey. This similarity was further verified by statistical analysis, as there are no statistically significant differences between the response ratios of the two groups.
4.2 On the Performance of GPC and PBC in the Experiment

The performances of the two tested methods were evaluated using three different programming exercises. Two of these programming exercises were present in both pre- and post-surveys, while the third exercise was included only in the post survey.

The first programming exercise consisted of five consecutive commands that moved an arrow in a grid. The exercise was very easy for the students, as 28 out of all 48 students got the full score on both pre- and post-tests. Due to this, the changes in the scores during the experiment followed a similar distribution for both groups (Kolmogorov-Smirnov test p-value = 0.9953). As the students were very good already before the experiment (combined pre-test mean = 4.4 out of 5), most students whose results declined received full score from the pre-test (see Fig. 4).

Programming Exercise 2 was more difficult (combined pre-test mean = 2.4) than Programming Exercise 1, and it provided more variance between the students’ scores. In order to evaluate the performance of the two teaching methods, the students in GPC and PBC groups were classified into three different groups: decline, neutral and improvement. The students were divided into these groups based on the change in their scores in pre-tests and post-tests. The number of students in each of the groups is listed in Table 2. The two groups were then tested for differences using the Chi-Squared Test, which showed that there is no statistically significant difference between the two methods (p-value > 0.77).

The last programming exercise was tested only in the post-test. The results of this exercise were in line with Programming Exercise 2, so that students in GPC group had a slightly higher mean than PBC group (3.44 and 2.96, respectively), but the difference between the two groups was not statistically significant (p-value > 0.066). As Programming Exercise 3 was not present in the pre-test, it is not possible to determine whether the differences in the scores were caused by differences in the group populations or differences in the tested methods.

4.3 On the Similarity of Students’ Skills and Attitudes After the Test

The students’ responses to the post-test survey were analysed to identify the possible differences in the responses between play-based and graphical programming classes. An overview of the students’ responses can be seen in Figure 5. In addition to the visual examination, similarity of each survey item was evaluated by conducting a Chi-Square test between the responses from GPC and PBC groups. As all of the tested p-values were greater than 0.05, it was verified that there were no statistically significant differences between the two groups. Due to the nature of the acquired p-values, there was no reason to apply any corrections to control the Type I errors.
Table 2. A summary on the number of positive, negative and neutral cases in exercise 2 between the pre-tests and post-tests for graphical programming class (GPC) and play-based class (PBC). While both methods show improvement, GPC has a slightly more positive outcome.

<table>
<thead>
<tr>
<th>Method</th>
<th>Decline</th>
<th>Neutral</th>
<th>Improvement</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPC</td>
<td>4</td>
<td>11</td>
<td>10</td>
<td>25</td>
</tr>
<tr>
<td>PBC</td>
<td>5</td>
<td>10</td>
<td>7</td>
<td>23</td>
</tr>
</tbody>
</table>

5 DISCUSSION

As this was a repetition study, the authors tried to avoid alterations that might cause unexplained changes in the test procedure. Due to this, some of the shortcomings of the original test process were transferred to the applied test procedure. For example, by changing the pre-surveys and post-surveys it would have been possible to provide a more thorough set of exercises, and provide a more complete picture on the students’ skills before and after the experiment. These results would likely have given a more fine-grained view on the differences between the two experimental groups. Unfortunately, alterations in the exercise set would make comparisons between the results of the two studies ambiguous. Also, the pre-test and post-test surveys could be extended to gain further knowledge on the background variables that are related to the learning outcomes. This will be done in a larger setting elsewhere.

The time-frame for the conducted experiment was relatively short, only ninety minutes, raising concerns about its effects on the measured results. As this is a repetition study in a new setting, and the previously conducted research produced consistent results, it remains inconclusive whether the deviations in the results are caused by the short time-frame or the change in the research setting.

Figure 5. Students’ responses to the multiple-choice questions in the post-test survey for both play-based class (PBC) and graphical programming class (GPC). Similarly to the pre-test survey, the responses for the two student groups are very similar with each other. There were no statistically significant differences between items in the two groups.

For all translated content, there is always a possibility that the original and translated content do not match each other well enough. As this is problem was well-known and anticipated, the translation process was done with care, and the content was double-checked where possible. Therefore, there should be only minimal differences that were caused due to the translation process itself, and the remaining issues should be superseded by other environmental and cultural differences. Still, identifying these differences and their effects is quite difficult, and will at least require a whole new study.

6 CONCLUSIONS AND FUTURE WORK

This research re-created the experimental setting, reported by [1], in a Spanish elementary school with a larger sample size. Post-hoc research and the larger N were used to show that the experimental setups for graphical programming and play-based programming do not cause significant transferable
differences in students’ attitudes or programming skills in the given short time-frame. If the ratios between the students’ programming performances for GPC and PBC were to stay the same as reported in this paper, the differences would remain insignificant even if the N was ten-fold compared to what it was for this study. Based on the results, it is reasonable to assume that the null-hypothesis presented in [1] was likely revoked largely due to the short time-span of the experiment, and in a lesser extent due to unanticipated environmental and cultural differences.

Based on the analysis and the received set of results, it would be beneficial to further extend the testing framework so that the framework includes repeated, topic-specific programming exercises that measure students’ skills in the topics that are taught to the play-based class and the graphical programming class. The time-span of the experiment should be extended to a longer time frame to ensure that the students are accustomed with the graphical programming environment and the play-based teaching methods.

REFERENCES


