[ai] Explore!: An Educational Game for Developing Programming Skills and Algorithmic Thinking

Vedrana Mikulić Crnković[1], Martina Holenko Dlab[2], Bojan Crnković [1] and Ivona Traunkar [1]

[1] University of Rijeka, Faculty of Mathematics, Rijeka, Croatia

[2] University of Rijeka, Faculty of Informatics and Digital Technologies, Rijeka, Croatia

vmikulic@math.uniri.hr
mholenko@inf.uniri.hr
bojan.crnkovic@math.uniri.hr
inovak@math.uniri.hr

Abstract: The game [ai] explore! was developed to introduce the algorithm behind the popular science exhibition. The idea behind the [ai] explore! exhibition was to show the application of mathematics and computer science in solving real engineering problems by creating the exhibit using the mathematical algorithm Heat Equation Driven Area Coverage (HEDAC). The algorithm is based on scientific research and has many applications: exploration and search in unknown space, painting, planning agricultural spraying, scanning 3D objects, etc. The exhibits show the process by which the algorithm paints objects with a virtual brush and explores their shape. By surveying the high school students, it was shown that new mathematical and engineering results can be explained and turned into a playable game. After playing the game, students respond positively to the game presented and state that they have a good understanding of the HEDAC algorithm presented and how it works. This motivated us to investigate [ai] explore! as an educational game for teaching mathematics and computer science. The aim of the research described in this paper is to investigate the potential of the developed game as an educational game. The game-based learning (GBL) approach is increasingly used to increase student motivation and engagement in STEM lessons. The main research question is: How do primary school teachers and students perceive the usefulness of the game [ai] explore! in supporting teaching and learning of computer science concepts? We have developed and implemented teaching scenarios on how the game [ai] explore! can be used as an educational game in computer science lessons. We show that the game has great potential to teach primary school students programming. In addition, the game promotes algorithmic thinking and computational thinking skills in general, which are valuable not only in computer science but also in other fields such as mathematics, where a systematic and analytical approach to problem-solving is required.

Keywords: Educational Game, Game-Based Learning, Algorithmic Thinking, Programming Skills, Mathematics

1. Introduction

Computational and analytical thinking should be encouraged in STEM education as they are crucial in various fields such as math, engineering and science. The use of algorithms and mathematical principles underlies many scientific concepts, and for this very reason it is important to foster an interdisciplinary approach to education and raise awareness of the connections between disciplines.

Today's education faces many challenges. The traditional approach to education does not sufficiently promote computer literacy, so new learning and teaching methods need to be developed.

The game-based learning (GBL) approach is increasingly being used to increase student motivation and engagement in STEM lessons.

Games have been part of the educational process since ancient times (Vankúš, 2005).

From an early age, children learn through play. This form of teaching can be integrated into formal education. Game-based learning and teaching uses the motivation and competition that naturally arise during play to achieve the goals and outcomes of the lesson. There are numerous studies that show many positive effects of this type of learning (Ku et al., 2014), (Vankúš, 2021), and although some of the articles report mixed results and the challenges teachers face with this type of teaching, no study reports negative effects.

The most used tools for game-based learning are computer games and board games. There are examples of the implementation of game-based learning in mathematics (Deeb and Hickey 2019), (Mikulić Crnković, Traunkar and Crnković, 2022), (Seebauer, Jahn and Mottok, 2020), (GAMMA, 2020) and in computer science (Holenko Dlab et al., 2020), (Tonbuloglu, 2019), all of which report very positive results. The effects of serious games in mathematics education shows a significant improvement in mathematical skills in different grade levels and class groups (Hieftje et al., 2017), (Fraga-Varela, Vila-Couñago and Martínez-Piñeiro, 2021).

Introducing programming concepts and improving programming skills is a challenge at all levels. As early programming lessons are seen as crucial to stimulate students' interest in these areas (Wong et al., 2016), GBL

is widely used as an innovative method (Topalli and Cagiltay, 2018), supported by tools with graphical elements to create programs such as Scratch 3.0 (2019) to increase students' motivation and engagement (Lau, 2018), (Holenko Dlab and Hoić-Božić, 2021). In addition to activities supported with information and communication technology, unplugged GBL activities can also be used to support the development of algorithmic thinking and programming skills in students of different ages (Jagušt at al., 2018).

This paper presents the game [ai] explore! and the results of a study conducted to investigate the potential of the game for the development of programming skills and algorithmic thinking in primary school. The research presented contributes to the field of GBL by designing and implementing a teaching scenario for computer science lessons that involves the use of [ai] explore! as an educational game and examining its perceived usefulness for the development of programming skills by primary school teachers and students. In addition to developing programming skills, the game also promotes the development of algorithmic thinking and computational thinking skills as well, which are valuable not only for computer science but also for other fields such as mathematics, where a systematic and analytical approach to problem solving is required. Therefore, the paper also discusses the connection between mathematics and computer science in the context of developing programming skills and algorithmic thinking.

2. Background

2.1 Computational Thinking and Programming

The development of computational thinking begins in preschool by engaging kids in different activities that strengthen pre-mathematical skills (Botički, Pivalica, and Seow, 2018): sorting, comparing and grouping objects, following multi-step instructions, working with patterns, orienting in space and evaluating. Similar essential 21st-century skills should continue to be developed set as the child enters school (Sneider et al., 2014.; Yadav et al, 2016; Ogegbo, A. A., Ramnarain, U. 2021).

Computational and analytical thinking involves breaking down complex problems into smaller, manageable pieces and developing algorithms to solve them (ISTE & CSTA. (2011)). More precisely, computational thinking is a set of skills and processes that enable students to tackle complex problems. It is a cognitive process that involves the following: abstraction, problem decomposition, algorithmic thinking, evaluation, and generalization (Shute, Sun, and Asbell-Clarke, 2017).

A part of computational thinking is algorithmic thinking. Algorithmic thinking is a cognitive process in which we arrive at a solution to a problem step by step, so the process can be repeated by humans or computers and is necessary for successful problem solving in computing. Algorithmic thinking is necessary for successful programming, *i.e.* the management of digital tools by students with the goal of having the digital tool solve a problem (Wing J. M., 2006).

Programming is an excellent way to learn, understand and apply mathematical principles while developing computational thinking. It is well known that there are many positive effects of cooperative learning where students learn together, learn from each other, etc. However, learning mathematics through programming can be interpreted in a similar way (Yadav et al.,2018). Successful programming requires communication between the student and the computer in a language that the computer understands, with the student giving sufficiently clear instructions to enable the computer to perform a mathematical procedure independently.

By applying interdisciplinary approach in mathematics and computer sciences education, both fields can prosper. Many mathematical applications involve the use of technology for analysis, simulation, and modelling. By incorporating technology and linking it to computer science topics in math class, students gain experience with tools and techniques that are relevant in the real world (Basu et al.,2013). This bridges the gap between theory and application and helps students understand the importance of math to the world around them.

Mathematical thinking complements computer science thinking, and many mathematical topics interact with computer science topics (Baldwin, Walker, and Henderson, 2013). That is way, solving mathematical problems fits naturally into the computer science classroom and helps develop programming skills.

2.2 Game [ai] Explore!

Exhibition [ai] explore! is an interdisciplinary activity aimed at explaining the results of scientific research and their application to the interested public and students.

The popular science exhibition [ai] explore! presents visualisations of engineering problems from the field of civil engineering as well as some computationally demanding scientific research examples. Each visualisation shows the process of the algorithm that creates the given image, with different default settings (brush settings, etc.). Figure 1 shows an exhibit, i.e. a visualisation. Each exhibit has a QR code behind which you can read more about the engineering problem that the exhibit shows, walk through the exhibit and read more about the mathematical algorithm that painted the exhibits.

The visualisation was created using the mathematical algorithm HEDAC (Ivić, Crnković and Mezić, 2017) and (Ivić et al., 2019). The HEDAC algorithm was introduced to control and coordinate the actions of a multi-agent space exploration system, but it has been used in several very different applications, such as spraying plants, painting pictures, controlling a robotic arm with 6 degrees of freedom, 3D scanning, maze exploration, and more. (Löw T., Maceiras J., Calinon S., (2022); Ivić et al., 2023., Bilaloglu C., Löw T., Calinon S., 2023; Crnković B., S. Ivić, M. Zovko. , 2024).

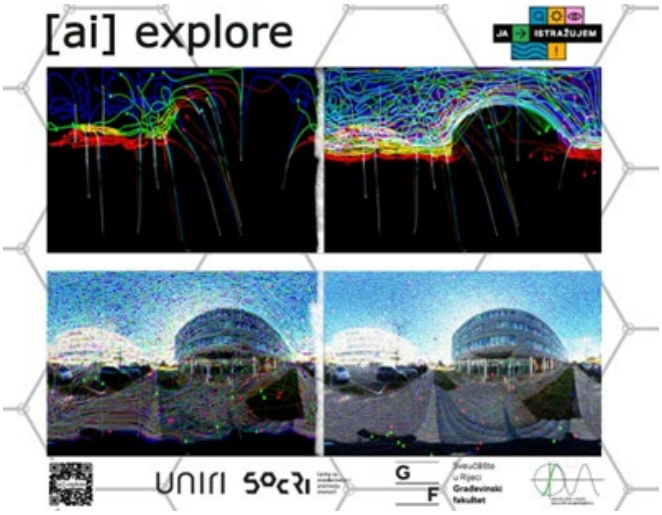

Figure 1: Exhibition [ai] explore!

More details about the exhibition can be found in Crnković, Mikulić Crnković and Traunkar (2024).

To complement the exhibition, two educational [ai] explore! game that illustrate how the mathematical algorithm works were developed: a game using GeoBoard and a game using micro:bits.

The aim of the [ai] explore! games is to illustrate the algorithm behind the exhibition and to put the students in the decision-making situation in order to motivate them to develop the strategy, i.e. the algorithm for playing the game. In order to play successfully, the players must follow the steps of the developed algorithm. The process of playing the games involves decomposing a problem, dealing with uncertainty, organising, planning, developing a heuristic algorithm, simulating and evaluating the results, all of which are fundamental to the development of computational thinking.

2.2.1 [ai] Explore Game Using Micro:bits

Micro:bit is a pocket-sized computer with a 5x5 grid LED light display, two buttons, sensors and many input and output functions that you can programme and interact with. This version of the game shows the application of the HEDAC algorithm for catching Pokémon. The playing field is a 5x5 board with micro:bits placed on some nodes of the board. Each micro:bit indicates the probability of the Pokémon being on that node. This is indicated by the level of heat, which is shown by the LED lights in a level from 1 to 5. The micro:bits are programmed so that the heat increases by one level every 4 seconds and cools down again by pressing the button on the micro:bit.

The rules of the game are simple: every 4 seconds, indicated by the sound of the main micro:bit, the student must take a step to the left, right, up or down to the next node towards the hottest group of micro:bits on the board. If there is a micro:bit on the node you stepped on, you must check whether there is a Pokémon there by pressing the button (thus cooling the micro:bit). The game ends after 20 moves (80 seconds) and the number of Pokémon collected is displayed on the main micro:bit, which is located outside the playing field. The game can

be played by several players and the students must follow the rules described and develop the best algorithm for co-operation and individual tactics when choosing the next moves in order to collect the most Pokémons.

2.2.2 How to Play and Generate the Game Using Geoboard

A GeoBoard is a board used primarily in primary school mathematics education to explore concepts in geometry. It is a square board with a certain number of pegs arranged in an orthogonal grid pattern and comes with rubber bands that are stretched around the pegs to create different shapes (Figure 2).

In the original version of the [ai] explore! game, the GeoBoard was modified by assigning different colours to the pegs. Each colour has assigned a numerical value that depends on the warmth of the colour, which corresponds to the wavelengths of light assigned to the colour. This ranking of colours is intuitive for most people, as they can compare this order to the colours of the rainbow. The rules of the game are very simple: connect the starting peg in the top left corner to the bottom right peg by a continuous shortest path with the maximum sum of peg values along the path. This means that you can only move down and to the right from any position.

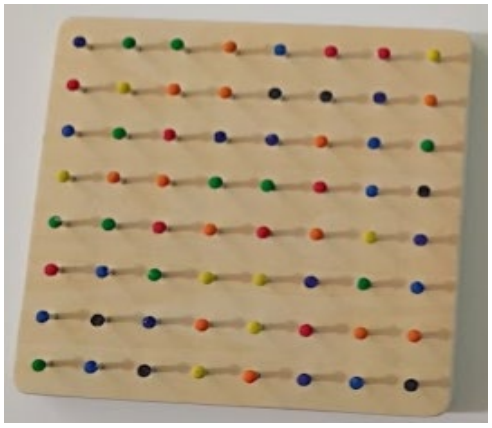

Figure 2: GeoBoard with 8 pegs in each row and column

The total number of possible paths depends on the number pegs in each row (or column) and is easy to calculate. This number is equal to the number of (n-1) subsets of a set of size 2(n-1), where n is the number of pegs in each row or column. The number of possible paths grows very quickly with n, but the number of optimal paths (shortest path with maximum sum) depends on the choice of board and the number of different colours. To find the optimal path, student needs to develop the tactic to collect the highest number of points, and following the developed tactic they choose the next step of the path.

The difficulty of the game depends on the size of the board. In the original implementation, the GeoBoards with 8 and 11 pegs in each row and column, coloured with 7 colours, were used (Figure 2). For example, there are exactly 3432 different shortest paths connecting opposite corners of the 8-peg GeoBoard. The colours of the board were randomly generated and often there is more than one optimal solution and several near-optimal solutions (1 point less than the maximum weight path). The Python programme with a recursive algorithm was used to generate all possible paths and rank them according to their total weight. Figure 3 shows the number of different paths and the distribution of near-optimal paths for two randomly generated boards. Although only a tiny fraction of all possible paths are optimal paths, students can find them in a reasonable amount of time. Students who find near-optimal paths can see from these graphs how close their solution is to the optimal solution and what percentile it is in.

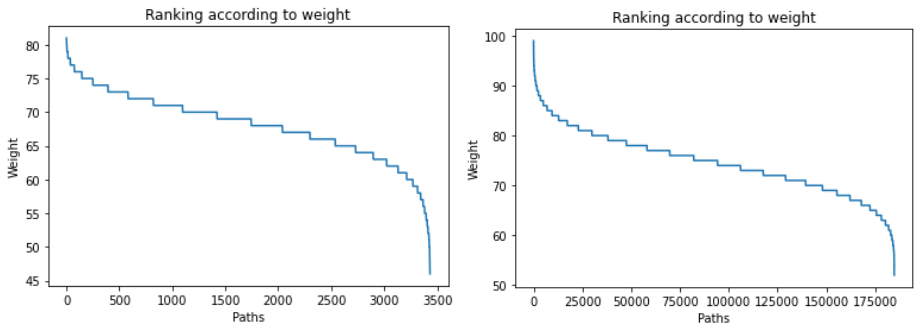

Figure 3: Distribution of all paths by weight for 8 and 11 peg GeoBoard with 7 colours

2.2.3 [ai] explore as Educational Game

After the presentation of the exhibition and the games to the students, a short survey was conducted among the students about their experiences and attitudes. The survey contained four statements (I found the game interesting; I would like to play the game again; I would recommend the game to friends; I understand how the algorithm presented works) and responses were measured using a 5-point Likert scale. A total of 66 high school students took part in the survey.

The results of the surveys were compared, for 3 different implementations of the [ai] explore! game: for a group of students playing [ai] explore! game with micro:bits, for a group of students playing [ai] explore! game on GeoBoard, and for a group of students who played both games. More details about the survey can be found in Crnković, Mikulić Crnković and Traunkar (2024).

From the survey results, the general impression is that the students react positively to the games presented. Most of the students themselves state that they have a good understanding of the algorithm presented and how it works. The results are slightly better when only one game is presented, and the inclusion of micro:bits and teamwork in the game makes it slightly more interesting for the students. However, implementing a game with digital tools in the classroom can be challenging for teachers in many ways. The authors reported that the reactions and interest were really positive during the European Researchers' Night 2023 in Rijeka, where up to 1000 kids and adults tried out the game for themselves. From this, the authors concluded that the developed educational game on GeoBoard has good playing characteristics, is very practical and can be easily used in different environments and situations, including classrooms.

3. Methodology

The game [ai] explore! was developed with the aim of teaching mathematics and explaining the HEDAC algorithm. Motivated by the results of the survey described in the previous section, the aim of the study presented in this paper was to investigate the potential of the game [ai] explore as an educational game for the development of programming skills and algorithmic thinking. The main research question was: *How do primary school teachers and students perceive the usefulness of the game [ai] explore! in supporting teaching and learning computer science concepts?*

To illustrate how the game [ai] explore! can serve as an educational game, a teaching scenario for one lesson in programming was developed and implemented with students from two primary school classes as part of regular lessons in Computer Science/Informatics subject. During the execution of the scenario, the researchers were present in the classroom together with the class teacher.

3.1 Context and Participants

Learning objectives related to the development of computational thinking and programming skills are implemented in Croatian schools in the subject Computer Science/Informatics according to the "National Curriculum for the Subject of Computer Science/Informatics for primary schools and secondary schools – gymnasiums". One of the four domains of the National Curriculum in which the learning outcomes and course content are organized is Computational Thinking and Programming. Computer Science/Informatics is a compulsory subject for students in 5th and 6th grade and an elective subject for students in 7th and 8th grade.

Teachers are autonomous in choosing the programming language for teaching programming. The most used languages are Scratch, Logo and Python.

The participants of the study were students (NS=35) from two primary schools in Rijeka, Croatia, together with their teachers (NT=2). The students involved in the study were from two mixed-gender classes aged 11-12 years (5th grade in Croatian schools). Both teachers involved in the study have extensive experience in teaching computer science and serve as mentors for pre-service teachers (i.e. computer science students) from the University of Rijeka during their practical lessons in schools.

3.2 Teaching Scenario for Using [ai] Explore! Game in Combination with Scratch

Developed teaching scenario demonstrate how the game [ai] explore! can be used as an educational game for teaching conditional statements (if-then-else) in Scratch.

After completing the lesson, students are expected to:

- correctly implement conditional statement in Scratch
- recognize situations in which is necessary to use conditional statement

The duration of the lesson is 45 minutes. Students are expected to be in the computer lab. During the lesson, the teacher should guide the students through the activities included in the scenario.

For this lesson, GeoBoards have been prepared with 5 squares in each row and column, coloured in 5 colours: red, yellow, green, blue, and purple with assigned values 5, 4, 3, 2 and 1, respectively. The prepared game has two optimal solutions out of seventy possibilities (all paths connecting the upper right point with the point in the lower left corner). The two optimal paths, i.e. the shortest paths that connect the upper right point with the point in the lower left corner so that the path has the maximum possible sum, are shown in Figure 4.

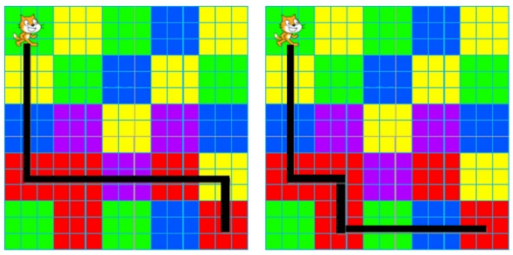

Figure 4: Two optimal solutions on the GeoBoard

The first part of the lesson is playing [ai] explore! game. Each student should have a GeoBoard, rubber bands, and instructions for playing. After all students have found the optimal solution on the board (any of them), the teacher moves on to the second part.

In this part, the students have to implement the algorithm they developed by playing the game [ai] explore! on the GeoBoard in Scratch to move the game character along the optimal path in Scratch. With this goal in mind, we prepared a stage in Scratch that looks the same as the GeoBoard (Figure 5).

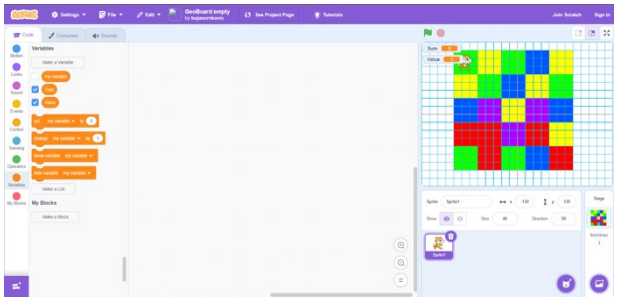

Figure 5: Prepared Scratch file

Creating and following the algorithm while playing the [ai] explore! game on GeoBoard is a preparation for implementing the algorithm in Scratch. The task in Scratch is linked to the game, i.e. the student is asked to

implement the path they have chosen on the GeoBoard, but takes into account the programming skills the students are developing in this lesson (namely conditional statements and variables).

In the second part, the students are given the following task:

Create a programme in Scratch in which your character follows the path you specified on your GeoBoard, step by step (each rubber band is one step). As it moves, the character counts points depending on the colour of the square it is on. In addition, your character thinks about how warm or cold it is on each field (depending on which filed it is on, cold colours are purple, blue, green, and warm colours are yellow and red).

Teacher should instruct the students to the use conditional statements and variables to add up points. Example of one possible realisation of such code can be seen in Figure 6.

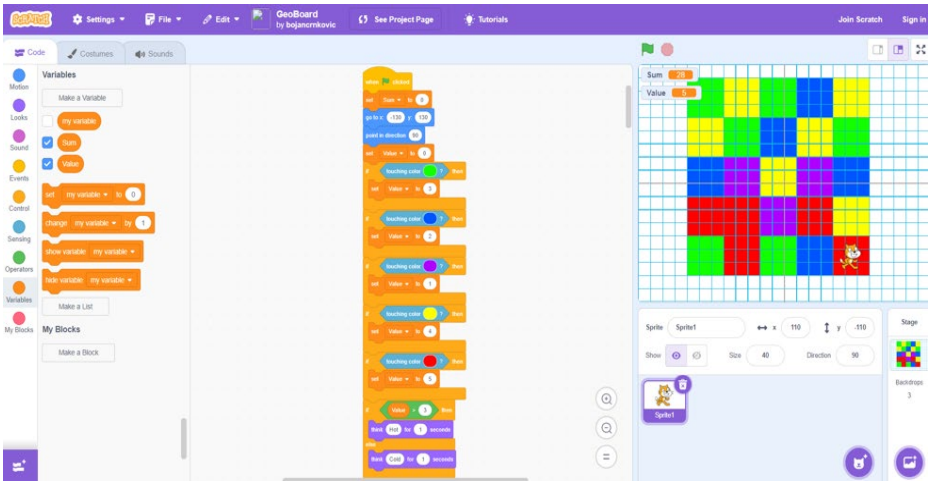

Figure 6: Part of the solution in Scratch

It should be noted that this lesson is intended as part of a series of lessons. It is anticipated that students will recognise the need for more advanced concepts (i.e. loops and procedures) during this lesson, so teachers can also use this example to introduce advanced programming concepts.

In addition, students can create a more advanced version of the game in the next lessons, e.g. by adding control buttons to move the game character, etc.

3.3 Instrument

A combination of quantitative and qualitative research methods was used to answer the research question. A questionnaire with a list of statements on a 5-point Likert scale (1 – strongly disagree, 5 – strongly agree) was developed for the study (see Tables 1 and 2) to measure students' attitudes. After the activity, students were asked to complete a questionnaire and indicate their attitudes towards their experience. The first set contained statements about the game [ai] explore! and its potential to support the learning of programming concepts. The second set of questions contained statements about their more general attitudes towards participating in the activity which involved using the [ai] explore! game in combination with Scratch.

In addition to the questionnaire, a short interview was conducted with the computer science teachers teaching students in two classes that participated in the survey. The teachers were asked about their observations and opinions on the game [ai] explore! in combination with Scratch and its potential to support the teaching and learning of programming concepts using the following questions:

- How would you describe the general atmosphere during the activity and students' engagement?
- In your opinion, how did the game-based learning activity and the GeoBoard help the students to solve the task in Scratch?
- Do you think the game could help students solve more complex problems in Scratch?
- How do you think such activities motivate students compared to more traditional teaching methods?

4. Results and Discussion

4.1 Results of Questionnaire for Students

Table 1 presents the results of responses gathered from students (N=35) regarding their attitudes towards the [ai] explore! game. Each statement was rated on a Likert scale ranging from 1 - Strongly Disagree to 5 - Strongly Agree.

Table 1: Attitudes towards the game [ai] explore!

Statements	N	AVG	SD	MIN	MAX
I found the game interesting.	35	4.57	0.77	2	5
The rules of the game were clearly explained to me.	35	4.57	0.77	2	5
The game made it easier for me to find a solution to the task in Scratch.	35	4.20	1.24	1	5
The geoboard helped me find the path that the character in Scratch should take.	35	4.49	1.05	1	5
I would like to solve more complex problems in Scratch using the game [ai] explore! on the geoboard.	35	4.26	1.08	2	5
I would like to create the game [ai] explore! in Scratch.	35	4.03	1.36	1	5
I would prefer to learn programming using games rather than normal tasks.	35	4.49	1.11	1	5

The results indicate a positive attitude towards the game [ai] explore! and its potential to enhance the learning experience. Students found the game interesting and expressed satisfaction with the clarity of the game rules (the average rating for both statements are 4.57). This result is important since providing concise instructions for educational games can contribute significantly to a positive student's experience.

The majority of students found the game to be a helpful tool for solving the task in Scratch and agreed that the GeoBoard was an effective tool in helping them find the path that the character should take in Scratch (AVG = 4.49). This shows that the integration of tools such as the GeoBoard can improve the application of programming concepts in a visual and interactive way.

Students also expressed an interest in further exploring the game [ai] explore! In combination with Scratch and creating more complex projects (AVG = 4.26), as well as a preference for learning programming through games rather than traditional tasks (AVG = 4.49), highlighting the potential of GBL approaches to increase student engagement and motivation.

In addition to exploring the attitudes towards game [ai] explore!, general attitudes towards participation in the activity using the game [ai] explore! in combination with Scratch were examined as well. The results are presented in Table 2.

Table 2: Attitudes towards participation in the activity using the game [ai] explore! in combination with Scratch

Statements	N	AVG	SD	MIN	MAX
I think I can learn something new through activities like this.	35	4.20	1.30	1	5
There are not enough activities like this in class.	35	4.49	0.87	2	5
Activities like this motivate me to study mathematics and computer science.	35	4.31	1.14	1	5
It was fun for me.	35	4.51	0.80	2	5
I would like to participate in similar activities in the future.	35	4.31	1.14	1	5
I would recommend other students to participate in similar activities.	35	4.63	0.64	3	5

Students expressed the opinion that they can learn something new through activities like this (AVG=4.20), indicating a potential of the activity for acquiring new knowledge and skills through hands-on, interactive learning experience. The perception that there are not enough activities like [ai] explore! Game in class was prevalent among participants (AVG=4.49). This indicates a desire for more engaging and interactive learning opportunities in the class. With an average rating of 4.31, participants indicated that activities like this serve as

a source of motivation to study Mathematics and Computer Science. This suggests that integrating GBL experiences into these subjects can positively influence students' attitudes and interest in these subjects.

Students found the activity fun (AVG=4.51) and expressed a desire to participate in similar activities in the future (AVG=4.31). They would also recommend such activities to other students (AVG=4.63). This suggests that teaching scenario that included the [ai] explore! game in combination with Scratch was well received by students and holds promise for promoting learning, motivation, and fun in math and computer science education.

To summarize, student responses indicate a very positive attitude towards this activity, suggesting that it has the potential to enhance learning experiences and foster motivation among students. These results indicate that the [ai] explore! game has the potential to support students in learning computer science concepts.

4.2 Teachers' Observations

According to the teachers' observations, the atmosphere was pleasant and the use of the game in the learning process has triggered excitement and curiosity among the students. The students were engaged throughout the activity, and all completed the task successfully. These observations were confirmed by researchers.

Teachers commented that the GeoBoard proved to be a tool that helped students visualize and navigate the path their character should take in Scratch. After the students physically manipulated the rubber bands on the GeoBoard, they mapped movements in their Scratch projects using conditional statement.

Both teachers expressed the attitude that this game, which includes hands-on activities, can be used to tackle more complex challenges in Scratch. The teachers mentioned that they are interested in exploring the activities that can be built on top of this teaching scenario. They also said that creating the game [ai] explore! in Scratch would be an enriching experience for the students as it would allow them to demonstrate their understanding of programming concepts in a fun and interactive way.

Teachers' observations also support the conclusion that the game [ai] explore! has the potential to support students in learning computer science concepts.

5. Conclusions and Plans for Future Work

The aim of the research described in this paper is to investigate the potential of the developed game as an educational game.

Teaching scenarios on how the game [ai] explore! can be used as an educational game in Scratch lessons were developed and implemented. The results of the student survey clearly show that the students find the game fun, interesting and captivating, that they would like to learn programming with this game and would recommend it to other students. The teachers' observations confirm that the games have generated excitement and curiosity among the students and that they should be used to master more complex programming challenges. From the responses of the teachers and students, it can be concluded that the game has great potential for teaching primary school students programming. In addition, the game promotes algorithmic thinking and computational thinking skills in general, which are valuable not only in computer science but also in other fields such as mathematics, where a systematic and analytical approach to problem solving is required.

The results of the survey suggest that this approach and the [ai] explore! game have the potential to be used in game-based learning in other areas of computer science and mathematics with topics of varying levels of difficulty. This indicates one direction for future work motivated by this research.

The development a series of learning scenarios for teaching programming in Scratch, from simple to more complex commands and tasks, using the [ai] explore! game is planned. Since computer science teachers in Croatia can choose programming language they want to use in the classroom to develop students' programming skills it would be useful, in addition to scenarios and Scratch lessons, to develop teaching materials and programming lessons based on the [ai] explore! game in Logo and Python.

In order to evaluate the effectiveness of the proposed teaching scenario, future work will include a comparative study comparing the learning success of an experimental group after programming lessons based on the described game-based approach and the game [ai] explore! with the results of the control group that will be taught in the traditional way.

Acknowledgements

The research has been funded by the Erasmus + Programme of the European Union, KA220-SCH - Cooperation partnerships in school education, under the project "Science&Math educational games from preschool to university" (023- 1-HR01-KA220-SCH-000165485) and by the University of Rijeka (Croatia) under the project uniri-iskusni-prirod-23-223.

References

Baldwin, D., Walker, H., and Henderson, P. (2013) "The Roles of Mathematics in Computer Science". ACM Inroads, 4, 74–80.

Basu, S., Dickes, A., Kinnebrew, J. S., Sengupta, P., & Biswas, G. (2013, May). CTSiM: A computational thinking environment for learning science through simulation and modeling. CSEDU, 369–378.

Bilaloglu C., Löw T., Calinon S., (2023) Whole-body exploration with a manipulator using heat equation,arXiv preprint arXiv:2306.16898

Botički I., Pivalica D., and Seow P. (2018) "The Use of Computational Thinking Concepts in Early Primary School," in Proceedings of the International Conference on Computational Thinking Education

Crnković, B., Mikulić Crnković, V., Traunkar, I. (2024). "The Exhibition, Games and Virtual Reality - Technologies in Math Education", In: Volarić, T., Crnokić, B., Vasić, D. (eds) Digital Transformation in Education and Artificial Intelligence Application. MoStart 2024. *Communications in Computer and Information Science*, vol 2124. Springer, Cham.

Crnković B., S. Ivić, M. Zovko. (2024), Fast algorithm for centralized multi-agent maze exploration, https://arxiv.org/abs/2310.02121

Deeb, F. A. and Hickey, T. J. (2019) "Teaching Introductory Cryptography using a 3D Escape-the-Room Game" in *2019 IEEE Frontiers in Education Conference (FIE)*, pp. 1-6

Fraga-Varela, F., Vila-Couñago, E. and Martínez-Piñeiro, E. (2021) "The impact of serious games in mathematics fluency: A study in Primary Education", in *Comunicar* 29.69: 125-135.

GAMMA Handbook for Teachers (2020), http://www.project-gamma.eu

Hieftje, K., Pendergrass, T., Kyriakides, T.C., Gilliam, W., and Fiellin, L. (2017) "An evaluation of an educational video game on mathematics achievement in first grade students", *Technologies*, 5(2), 30.

Holenko Dlab, M. *et all* (2020) "Supporting Croatian Primary School Teachers in Designing Game Based Learning Activities: A Case Study", in *European Conference on Game Based Learning (ECGBL)*, Brighton, UK.

Holenko Dlab, M. and Hoic-Bozic, N. (2021). Effectiveness of game development-based learning for acquiring programming skills in lower secondary education in Croatia. Education and Information Technologies, 26(4), 4433-4456.

ISTE & CSTA. (2011). Operational definition of computational thinking for K–12 Education. http://www.iste.org/docs/ctdocuments/computational-thinking-operational-definition-flyer.pdf.

Ivić, S., Crnković, B., and Mezić, I. (2017) "Ergodicity-Based Cooperative Multiagent Area Coverage via a Potential Field", *IEEE Transactions on Cybernetics*, 47(8), pp. 1983-1993

Ivić S., Andrejčuk A., Družeta S., (2019) Autonomous control for multi-agent non-uniform spraying, Applied Soft Computing, Volume 80, pp. 742-760,

Ivić S., Crnković B., Grbčić L., Matleković L. (2023), Multi-UAV trajectory planning for 3D visual inspection of complex structures, Automation in Construction, Volume 147, 104709, ISSN 0926-5805

Jagušt, T., Krzic, A. S., Gledec, G., Grgić, M., & Bojic, I. (2018). Exploring different unplugged game-like activities for teaching computational thinking. In 2018 IEEE Frontiers in Education Conference (FIE) (pp. 1-5). IEEE.

Ku, O., Chen, S.-Y., Wu, D.-H., Lao, A.-C-C., and Chan, T.-W. (2014) "The Effects of Game-Based Learning on Mathematical Confidence and Performance: High Ability vs. Low Ability", *Educational Technology and Society*, 17. 65-78.

Lau, W. W. (2018). Learning With Games and Digital Stories in Visual Programming. In Encyclopedia of Information Science and Technology (pp. 3309–3316). IGI Global

Löw T., Maceiras J., Calinon S., (2022) "drozBot: Using Ergodic Control to Draw Portraits," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11728-11734

Mikulić Crnković, V., Traunkar, I., and Crnković, B. (2022) "Treasure Hunt as a Method of Learning Mathematics", *Proceedings of the 16th European Conference on Games Based Learning*. UK: Academic Conferences International Limited, 349-357

Scratch 3.0 (2019), The Scratch Foundation, in collaboration with the Lifelong Kindergarten Group at the MIT Media Lab. https://scratch.mit.edu

Ogegbo, A. A., Ramnarain, U. (2021). A systematic review of computational thinking in science classrooms. *Studies in Science Education*, *58*(2), 203–230.

Seebauer, S., Jahn, S. and Mottok, J. (2020) "Learning from Escape Rooms? A Study Design Concept Measuring the Effect of a Cryptography Educational Escape Room," *2020 IEEE Global Engineering Education Conference (EDUCON)*, pp. 1684-1685

Shute, V. J., Sun, C., and Asbell-Clarke, J. (2017) "Demystifying computational thinking," Educ. Res. Rev., vol. 22, pp. 142–158

Sneider, C., Stephenson, C., Schafer, B., Flick, L. (2014). Computational thinking in high school science classrooms. *The Science Teacher*, 81 (5), 53.

Tonbuloglu B., Tonbuloglu I. (2019) The Effect of Unplugged Coding Activities on Computational Thinking Skills of Middle School Students, Informatics in Education 18, no. 2, 403-426

Topalli, D., and Cagiltay, N. E. (2018). Improving programming skills in engineering education through problem-based game projects with Scratch. Computers and Education, 120.

Vankúš, P. (2005) "History and present of didactical games as a method of mathematics' teaching", *Acta Didactica Universitatis Comenianae – Mathematics*, 5, 53 – 68

Vankúš, P. (2021) "Influence of Game-Based Learning in Mathematics Education on Students' Affective Domain: A Systematic Review", *Mathematics 9*, no. 9

Yadav, A., Hong, H., Stephenson, C. (2016). Computational thinking for all: Pedagogical approaches to embedding 21st-century problem-solving in K-12 classrooms. *TechTrends*, 60 (6), 565–568.

Yadav, A., Krist, C., Good, J., Caeli, E. N. (2018). Computational thinking in elementary classrooms: Measuring teacher understanding of computational ideas for teaching science. *Computer Science Education*, 28(4), 371–400.

Wing J. M. (2006), "Computational thinking," Communed. ACM, vol. 49, no. 3, pp. 33–35

Wong, G. K. W., Cheung, H. Y., Ching, E. C. C., & Huen, J. M. H. (2016). School perceptions of coding education in K-12: A large scale quantitative study to inform innovative practices. In Proceedings of 2015 IEEE International Conference on Teaching, Assessment and Learning for Engineering, TALE 2015 (pp. 5–10). Institute of Electrical and Electronics Engineers Inc.