Integrating STEM Teaching into Digital Game Aids to Explore Non-Technical Students' C# Programming Learning Outcomes and Perceptions

Yu-Sheng Su[1], Chia-Jung Chang[2], Jia-Hong Li[2], Shih-Yeh Chen[3]*

[1] Department of Computer Science and Information Engineering, National Chung Cheng University, Taiwan
[2] Department of Computer Science and Engineering, National Taiwan Ocean University, Taiwan
[3] Department of Engineering Science, National Cheng Kung University, Taiwan
ccucssu@gmail.com, s95026540@gmail.com, tom862828@gmail.com, sychen-ncku@gs.ncku.edu.tw

Abstract

With programming development skills becoming one of the most important skills for many jobs in the workplace, there is a growing demand for STEM talent. Beginners usually learn C# programming using traditional teaching methods, resulting in poor learning outcomes. This study endeavors to probe the efficacy of integrating diverse learning aids, specifically digital game programming aids and conventional teaching aids, within the ambit of STEM education to bolster C# programming proficiency. Non-science and technology students from a university in Taiwan were the experimental subjects. The students were divided into an experimental group and a control group. Both groups learned C# programming using the STEM teaching method. The experimental group focused on digital game programming aids, while the control group focused on traditional programming aids. The results found that implementing the STEM teaching approach with digital game programming aids for learning C# programming improved student learning outcomes and perceptions. Learning C# programming through game creation enables students to visualize code execution, deepen their understanding of programming concepts, and enhance their problem-solving skills.

Keywords: STEM education, C# Programming, Learning performance, Learning perceptions, Digital games

1 Introduction

1.1 Programming Education

In the 2018 Higher Education Sprout Project, the Ministry of Education in Taiwan mandated that at least half of all college students must take courses related to logical operations before graduation. The goal is to develop problem-solving skills through practical programming education [1]. Most colleges and universities in Taiwan offer courses related to basic logic operations in response to the policy to meet the needs of students. However, beginners who are new to programming often face difficulties. This leads to poor learning performance [2-3]. Tan et al. researched the factors that make it difficult for college students to learn programming in programming courses and the appropriate learning methods of programming. Their study highlights the importance of using appropriate teaching methods to help students overcome these challenges. They found that most students lacked a programming foundation and considered basic fundamental concepts of programming to be difficult to learn [2]. Tek et al. researched and analyzed why non-science and technology students struggle to learn programming. They found that most students believe it is irrelevant to their major, which affects their motivation to learn and, in turn, their learning achievement [3]. Bubica and Boljat researched the teaching methods for university programming courses and found that most universities still use traditional methods of instruction [4]. Therefore, to help non-science and technology students learn programming, universities should consider using specially designed courses or additional teaching aids.

Previous studies have used digital games to increase students' learning motivations [5]. For example, Végh and Takáč used Unity to teach and learn programming in their courses, and challenged students to create 2D games, thereby stimulating students' interest in game development and programming [5]. We decided to apply digital games to programming courses, hoping to reduce the difficulties encountered by non-science and technology students in programming courses. It is necessary to determine the game engine required for the development of digital games. Sharif et al. compared a variety of game engines and found that Unity had good ratings in terms of visual fidelity, functionality, etc., so we decided to use Unity as the game engine in our study. Because of the need to work with the game engine, the programming language learned in the programming course could be the C# programming language or Python programming language. Although a few people use Python in Unity, and Python is often used in basic programming courses, Unity cannot directly call Python scripts. C#, on the other hand, is the main programming language used in Unity [6]. Moreover, comparing the advantages and disadvantages of programming languages, Bahar et al.

*Corresponding Author: Shih-Yeh Chen; Email: sychen-ncku@gs.ncku.edu.tw
DOI: https://doi.org/10.70003/160792642025072604002

mentioned that Python programming language is prone to errors in programming, and it is difficult to clarify and track where the errors occur in the code [7]. They pointed out that the C# programming language has the advantages of easy integration of visual suites, easy learning of C# programming language for beginners, more teaching resources, and easy understanding of the code structure. After all the above considerations, we decided to use C# in the programming course.

1.2 Applying STEM Teaching to Digital Game Aids in Programming Education

In recent years, STEM teaching methods have become increasingly important and have attracted attention in various countries which have actively promoted STEM education to cultivate more relevant talents and improve STEM literacy [8]. STEM education incorporates ideas proposed by Dewey [6]. The core idea is to develop students' reasoning and problem-solving skills, and to enhance their motivation to learn and their learning performance [9-11]. Christensen et al. pointed out that STEM education has five characteristics, namely real-world application of learning, learning from real problems, applying learned skills to solve new problems, promoting teamwork for problem solving, and integrating knowledge across disciplines. These characteristics are different from conventional teaching methods [9]. Hu et al. found that integrating problem-solving teaching into an electronic music pencil-making course for non-science and technology students can enhance understanding of STEM concepts and improve learning outcomes and motivation [12]. Yuliati et al. integrated inquiry-based learning with STEM concepts into a physics mechanics course. Students were required to understand and observe Newtonian mechanics (science concept), learn to apply mechanical formulae to solve problems (technology concepts), assemble and create mechanical concept prototypes (engineering concepts), and perform calculations and verify physics experiments (mathematics concepts). Their results showed that their teaching approach effectively deepened students' understanding of physics concepts and scientific literacy [13].

However, Freeman et al. found that the application of the STEM teaching approach to programming did not effectively engage learners and did not significantly improve their programming learning outcomes [14]. They pointed out that programming courses should be engaging to increase students' motivation to learn. Calvo et al. developed a C language script using the LEGO Mindstorms add-on kit with NXC (C-like programming language for the NXT brick) to increase the fun factor of the course [15]. The activity required the application of STEM concepts, such as understanding electrical science (science concepts), using the NXC compiler to program components for desired effects (technology concepts), assembling the extension kit (engineering concepts), and performing arithmetic operations in the script (mathematics concepts). The results showed an improvement in student learning in programming and in course completion rates. Based on the above, the application of STEM teaching approaches in programming course still needs to increase the fun factor to enhance learning motivation, improve learning achievement and mitigate the negative impact of traditional teaching methods.

Li and Watson suggested that learning digital game programming differs from learning traditional programming [16]. Integrating digital games into programming education allows students to visually observe the results of code execution within the game, which helps to avoid difficulties in understanding abstract concepts. This approach enhances students' understanding of programming concepts, thereby improving their learning performance and motivation. Hacıoğlu et al. integrated STEM teaching strategies into the Scratch ecological aquarium game programming course for primary school students [17]. During the STEM activities, students have to learn and apply ecological concepts (the science concept) and image programming language to build algorithms (the technology concept), game object assembly, software engineering concepts (the engineering concept), and addition and subtraction (the mathematics concept) conditional judgement to solve course problems. The results showed that the students were actively involved in the course activities and that they could stimulate their interest in STEM and the concepts of programming.

To promote willingness to learn and reduce poor motivation, we integrated STEM teaching into digital game programming aids. This was achieved by using digital game design as STEM teaching content. Yuliati et al. [13] presented science concepts related to the production of the modular programming project for war games, including the physical phenomenon of the rebound of the reaction force when a piece hits a pawn or a blood replenishment element. The Unity engine is used for programming, along with a compiler, to achieve specified effects for game components and other scripting applications. This includes the technology concepts mentioned in Calvo et al. [15]. Hu et al. [12] discussed the engineering concept of assembling game components. Hacıoğlu et al. [17] discussed programming scripts for additive and subtractive operations in mathematical concepts.

Therefore, we applied STEM teaching to digital game aids for learning C# programming, using digital game design as STEM teaching content, and making and presenting the modular programming project from war chess games.

1.3 STEM Learning Perceptions

Previous scholars believed that learning perception is one of the factors that affects learning motivation and learning achievement [18]. Zheng and Li explored the influence of learning perception on learning outcome, and the results indicated that learning perception can significantly affect self-regulated learning strategies, thereby improving learning performance [18]. Wild and Schiefele developed the LIST (Learning Strategies in Studying) questionnaire to evaluate students' learning behavior and learning perceptions in a course. The questionnaire was divided into three strategies, namely cognitive strategy, metacognitive strategy, and resource management strategy [19]. Griese et al. increased the number of questions in the LIST questionnaire with the original questionnaire structure, and implemented the questionnaire in STEM courses. The results showed that the improved LIST questionnaire can well reflect the students' learning perceptions in the course [20]. Mathew et al. integrated problem-solving teaching into undergraduate programming courses. Through the analysis of learning questionnaires, it was found that most of the students' feedback on the teaching was positive, and the students could really understand the structure, concepts and problem-solving skills of the programming [21]. Chang and Tao integrated STEM into inquiry-based teaching, enabling students with no game programming experience to learn Web game programming. According to the learning perception questionnaire, most of the students gave positive feedback about their experience with STEM game programming courses [22].

Therefore, we modified the questionnaire designed by Wild and Schiefele [19], and Griese et al. [20], and applied its design concept to the C# programming STEM learning perception questionnaire, so as to analyze the impact of applying the STEM teaching approach in the digital game project of C# programming on non-science and technology students' learning perceptions.

1.4 Research Objective

In this study, we focused on STEM teaching and the digital game aids in the C# programming course, and used the digital game project of the war chess game modular programming framework to carry out course activities to improve the programming learning achievement and learning perceptions of non-science and technology students. The following are the two research questions we aimed to answer:

1. What is the impact of C# programming applying STEM teaching and digital game aids on the learning outcomes of non-STEM students?
2. What is the impact of C# programming applying STEM teaching and digital game aids on the learning perceptions of non-science and technology students?

2 Methodology

2.1 Participants

This study was an experiment that was carried out at a college in the east of Taiwan. The subjects of the experiment were first year non-science and engineering students studying programming courses, most of whom had no background in programming. One class was the experimental group, using the modular programming project for war chess games for digital game programming aids, using C# to design Unity game program scripts, with a total of 31 students. The other class served as the control group, who learned with the traditional programming aids, and comprised a total of 31 students. These classes had the same instructors and took place in a classroom.

2.2 Learning Materials

Both groups of students in this study used the C# programming language for course instruction. The teaching materials were selected from the "New Concept Visual C# Programming Example Textbook" [23], a resource designed to provide comprehensive coverage of C# programming concepts with a focus on practical application. The integration of this textbook aimed to provide a solid foundation in programming fundamentals for all participants. In addition, the programming courses were strategically designed to align with the principles of various STEM (Science, Technology, Engineering, and Mathematics) subjects.

Table 1. Learning materials for the experimental group

Item	STEM concepts	Description
Science (S)	Physical phenomena.	Component scripts are used to achieve the rebound function of the reaction force when the chess piece component hits.
Technology (T)	Application of compiler, debugger and scenario builder.	Application of Unity engine, Visual Studio compiler operation, debugging tools and scene building tools.
Engineering (E)	Analyze program requirements, assemble components, and implement appropriate program syntax.	Assemble game scenes and components, analyze program fragments, and implement conditional narrative grammar to enable conditional judgment effects in game components.
Mathematics (M)	Addition and subtraction	A program script that produces functions for addition and subtraction operations on blood volume, an application algorithm module for calculating the shortest distance between points, and a module for calculating the shortest distance between points and points.

This interdisciplinary approach sought to emphasize the interconnectedness of programming with other

STEM disciplines and to provide students with a holistic understanding of the role of programming in real-world applications. The infusion of STEM concepts aimed to increase the relevance and applicability of programming skills in different contexts. In order to ensure consistency in teaching content, both the experimental and control groups followed the same curriculum structure and used the same set of learning materials. The content covered in the programming courses included basic programming concepts and practical examples, facilitating a systematic and progressive learning experience for all participants. For a detailed comparison of the learning materials used by the experimental and control groups, see Table 1 and Table 2 below, which provide a detailed breakdown of the specific topics and modules covered during the course. This alignment of the learning materials helps to establish a standardized basis for evaluating the impact of the different learning materials used in the experimental and control groups on students' learning outcomes and perceptions.

Table 2. Learning materials for the control group

Item	STEM concepts	Description
Science (S)	Physical phenomena.	Develop a basic program for temperature conversion that utilizes the physical formula for converting Fahrenheit to Celsius.
Technology (T)	Application of compiler, debugger and scenario builder.	Application of Visual Studio compiler operation and debugging tools.
Engineering (E)	Analyze program requirements, assemble components, and implement appropriate program syntax.	To achieve the specified result, analyze the program fragments and import them into the program code. For instance, you can import conditional judgment into the temperature conversion program. If the input value exceeds the set temperature, a warning message will be displayed.
Mathematics (M)	Addition and subtraction	Operators are used to create computer programs for performing basic arithmetic operations such as addition, subtraction, multiplication, and division.

2.3 Procedure

This study employed the quasi-experimental research method. Two groups participated in experimental activities lasting 8 weeks, with one 3-hour session per week, in accordance with university course hours. The flow chart of the experimental procedure is shown in Figure 1.

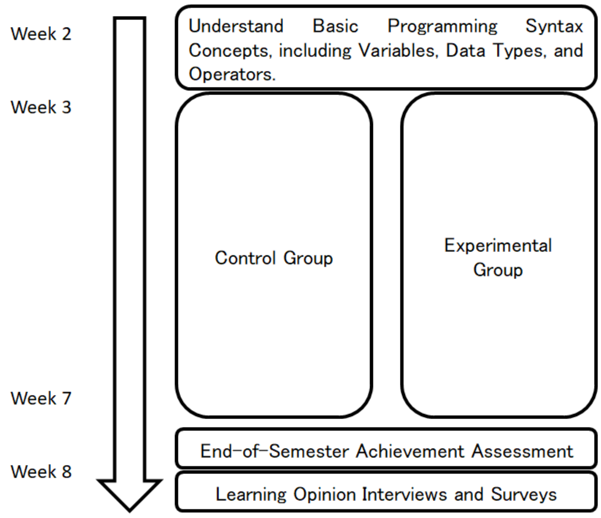

Figure 1. Experimental procedure

During the first week of the study, the curriculum focused on teaching basic programming concepts and pre-tests were conducted to establish a baseline understanding of the participants.

Subsequently, from the second to the seventh week, the experimental group (E.G.) and the control group (C.G.) engaged with different programming teaching aids, as outlined in Table 3, to navigate through the course material.

Post-tests were administered in the final week to measure the participants' learning. In addition to the quantitative assessments, a STEM learning perception questionnaire was administered to both groups to capture their subjective experiences and perceptions of the STEM learning environment. This questionnaire aimed to gain insights into how students in both the experimental and control groups perceived the integration of digital game-based programming tools or traditional teaching aids in their learning process.

Table 3. Curriculum content of the experimental group and control group

Time	Group	Description
Second week	E.G.	Understand basic programming syntax concepts, including variables, data types, and operators.
	C.G.	Understand basic programming syntax concepts, including variables, data types, and operators.

Third week	E.G.	1. Understand basic programming concepts, including conditional statements, loops, string arrays, function calls, etc. 2. Unity engine, compiler operation and debugging tool application.
	C.G.	1. Understand fundamental programming concepts, including conditional statements, loops, string arrays, function calls, etc. 2. Application of Visual Studio compiler operation and debugging tools.
Fourth week	E.G.	Use the component scripts in the teaching resources to create the physical effect of collision and bounce.
	C.G.	Refer to the program examples in the learning resources to create a simple temperature conversion program using the physical formula for converting Fahrenheit to Celsius.
Fifth and sixth weeks	E.G.	1. Use the components in the resources to create game scenes and assemble game components. 2. Refer to programming examples to analyze and insert code script fragments to make game components achieve specified effects.
	C.G.	Refer to the program examples in the programming aids to analyze the program fragments and insert the code to make the program achieve the specified results.
Seventh week	E.G.	Follow the programming examples in the teaching aids to create addition and subtraction functions, use algorithm modules to calculate and display the shortest distance between points.
	C.G.	Use the programming examples in the learning resources, use operators to create computer programs for addition, subtraction, multiplication and division.

2.4 Instruments

2.4.1 Pretest/Posttest

The test papers of this research mainly refer to the basic programming concepts practice test questions adapted from the New Concept Visual C# Programming Sample Textbook [23]. There was a total of 20 single-choice questions worth 100 points. The content of the pretest and posttest questions was the same. The aim was to understand the variance of the subjects' knowledge before programming and their learning after programming. According to Ebel and Frisbie's research [24], the higher the difficulty level, the easier the test questions are, and the lower the difficulty level, the more challenging the test questions are. It is more appropriate for the difficulty (0.58) of the test questions to be around 0.50. A higher discrimination indicates the ability to effectively discriminate between high and low scoring students. If the discrimination of the test questions (0.42) is greater than 0.40, this is considered to be excellent discrimination.

2.4.2 STEM Learning Perceptions Questionnaire

The STEM learning perceptions questionnaire used in this study was mainly based on and modified from the questionnaire developed by Wild and Schiefele [19] and Griese et al. [20]. The questionnaire used a 5-point Likert scale where respondents could indicate their level of agreement from "strongly agree" to "strongly disagree." On this scale, 5 points indicated strong agreement and 1 point indicated strong disagreement. For reliability analysis, Cronbach's alpha was used as a check between variables. According to Bland and Altman [25], the reliability coefficient of the questionnaire higher than 0.7 is an acceptable standard, and SPSS Statistics was used to analyze the reliability of the STEM learning perceptions questionnaire.

In this study, 59 students were selected to participate in the pretest before the experimental activities. The reliability coefficients of these six dimensions (0.72, 0.88, 0.83, 0.86, 0.93, 0.89, 0.92) and the whole questionnaire were all higher than 0.7, so the questionnaire has acceptable reliability.

2.5 Data Collection and Analysis

A total of 59 valid pretest and posttest papers and 59 valid STEM learning perception questionnaires were obtained, and the independent samples *t* test was performed using SPSS statistical software.

3 Results

3.1 Pretest and Posttest

The pretest results showed a slightly higher average score for the experimental group (E.G.) compared to the control group (C.G.). However, neither group's average score met the passing threshold, suggesting that most students lacked prior programming knowledge. The posttest results showed a significantly higher average score in the experimental group compared to the control group. Both groups showed improved average scores that met the passing threshold; at the same time, the experimental group showed relatively constant standard deviations, whereas the control group showed a slight increase in standard deviation. This indicates a tendency towards a greater dispersion of posttest scores in the control group. The results are shown in Table 4.

Table 4. Descriptive statistics of the pretest and posttest

Test	Group	N	Min	Max	Mean	S.D.
Pretest	E.G.	30	30	70	46.000	11.477
	C.G.	29	25	65	43.450	8.671
Posttest	E.G.	30	40	95	74.000	13.416
	C.G.	29	35	90	66.030	13.187

To further compare the average score of the posttest of the two groups, the independent samples *t* test was conducted on the pretest and posttest of both groups. The results showed an F value of 0.11 and a p value of 0.74, indicating non-significance. This suggests homogeneity of the regression coefficient within this dataset, which meets the necessary conditions for covariate analysis. After analysis of the independent samples t test, the results showed a significant difference between the experimental and control groups in the posttest, indicated by an F value of 5.180 and a corresponding p value of 0.027 ($p = 0.027 < 0.05$). This indicates statistical significance in the comparison. The results are presented in Table 5.

Table 5. Independent samples t-test analysis of the experimental and control groups

Source	Type III sum of squares	df	Average sum of squares	F	p
Two groups * pretest	20.160	1	20.160	0.110	.741
Two groups	933.022	1	933.022	5.180	.027*

$^*p <= 0.05$

3.2 STEM Learning Perceptions

The independent samples t test was performed on the STEM learning perceptions questionnaire. The results are presented in Table 6. For the statistics of the questionnaire, the experimental group (E.G.) had a higher average score compared to the control group (C.G.), and there was a significant difference ($p = 0.011 < 0.05$); this result means that the students in the experimental group had better learning perceptions compared to those in the control group. It also indicates significant differences in the learning perceptions of the two groups.

Table 6. The results of the independent sample t-test analysis for the STEM learning perception questionnaire in the two groups

Dimension	Group	Mean	S.D.	t	p
Organization	E.G.	3.500	.435	-.812	.420
	C.G.	3.606	.569		
Elaboration	E.G.	3.479	.499	.382	.704
	C.G.	3.418	.713		
Repetition	E.G.	4.366	.676	2.540	.014*
	C.G.	3.965	.523		
Monitor	E.G.	3.020	.650	-.973	.335
	C.G.	3.200	.767		
Attention	E.G.	3.683	.499	7.534	.000**
	C.G.	2.459	.730		
Peer Learning	E.G.	4.200	.815	2.108	.039*
	C.G.	3.793	.654		
All	E.G.	3.708	.371	2.635	.011*
	C.G.	3.407	.498		

$^{**}p <= 0.01$, $^*p <= 0.05$

Furthermore, the data obtained from each dimension of the questionnaire were analyzed. The average value of the experimental group was higher than that of the control group for the elaboration, repetition, attention, and peer learning dimensions. Additionally, significant differences were observed within the repetition, attention, and peer learning dimensions. This study demonstrates that the use of the digital game programming project by students in the experimental group facilitated the learning of basic programming concepts. The results indicated that the experimental group had a significantly better perception of learning compared to the control group, highlighting a clear distinction between the two groups.

According to the questionnaire results, it can be inferred that most students in the control group displayed inattention during the course. Most students in the experimental group believed in reviewing their learning or found that similar topics aided their memory. They had good concentration and spent more time discussing the course content with their classmates. When encountering problems in the course, they discussed solutions with their classmates.

4 Conclusion

In this study, the students from the experimental group applied the STEM teaching approach in the digital game project in the C# programming course, while the students from the control group used the STEM teaching approach to integrate the traditional programming aids in the C# programming course. This study aimed to explore the impact of integrating STEM teaching into different learning aids (the digital game programming aids and the traditional programming aids) on the learning of C# programming. The analytical results of the learning outcomes indicate that the experimental group of students had better learning outcomes than the control group. Yulia and Adipranata [26] found that using the digital game programming aids in programming courses can enhance students' learning performance. Feedback from the course showed that students in the experimental group generally felt that the digital game programming aids increased their motivation to learn programming. Similarly, Calvo et al. [15] reported that the use of learning materials and fun aids motivated students to learn, and improved their learning outcomes in programming. The findings suggest that integrating digital game programming into STEM teaching strategies can aid non-technical students in understanding fundamental programming concepts.

The analytical results of the STEM learning experience questionnaire indicated that the overall mean of the experimental group was higher than that of the control group, with a significant difference. This suggests that the use of digital game programming aids can lead to a better learning experience in basic programming concepts compared to traditional programming aids. Through the course feedback, it was also found that most students in the experimental group thought that using the digital game programming aids as learning materials could arouse their own learning interest and they would invest more time in learning, and they could maintain a focused state in the course. They also thought that when they encountered difficulties in game programming, they would actively seek help from their peers. This finding is similar to that of Chang and Tao [22], who found that integrating STEM digital game programming tools into the curriculum could make students feel good about learning and also improve their learning outcomes.

Acknowledgements

This study was supported by the Taiwan Comprehensive University System (TCUS) and the National Science and Technology Council, Taiwan, under grants NSTC 111-2410-H-019-006-MY3, NSTC 111-2423-H-153-001-MY3 and NSTC 111-2410-H-019 -029 -MY2.

References

[1] Ministry of Education, *Higher education deep cultivation plan performance indicators, measurement standards and target values*, 2018. https://www.edu.tw/News_Content.aspx?n=0217161130F0B192&sms=DD4E27A7858227FF&s=48B633D53D143A26

[2] P. H. Tan, C. Y. Ting, S. W. Ling, Learning difficulties in programming courses: undergraduates' perspective and perception, *2009 International Conference on Computer Technology and Development*, Kota Kinabalu, Malaysia, 2009, pp. 42-46.

[3] F. B. Tek, K. S. Benli, E. Deveci, Implicit theories and self-efficacy in an introductory programming course, *IEEE Transactions on Education*, Vol. 61, No. 3, pp. 218-225, August, 2018.

[4] N. Bubica, I. Boljat, Teaching of novice programmers: strategies, programming languages and predictors, *International Conference of Information Technology and Development of Education 2014 (ITRO 2014)*, Zrenjanin, Republic of Serbia, 2014, pp. 180-185.

[5] L. Végh, O. Takáč, Teaching and learning computer programming by creating 2D games in Unity, *14th annual International Conference of Education, Research and Innovation, ICERI2021 Proceedings*, Seville, Spain, 2021, pp. 5696-5700.

[6] J. Dewey, Experience and education, *The educational forum*, Vol. 50, No. 3, pp. 241-252, 1986.

[7] A. Y. Bahar, S. M. Shorman, M. A. Khder, A. M. Quadir, S. A. Almosawi, Survey on Features and Comparisons of Programming Languages (PYTHON, JAVA, AND C#), *2022 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS)*, Manama, Bahrain, 2022, pp. 154-163.

[8] A. Zollman, Learning for STEM literacy: STEM literacy for learning, *School Science and Mathematics*, Vol. 112, No. 1, pp. 12-19, January, 2012.

[9] R. Christensen, G. Knezek, T. Tyler-Wood, Alignment of hands-on STEM engagement activities with positive STEM dispositions in secondary school students, *Journal of Science Education and Technology*, Vol. 24, No. 6, pp. 898-909, December, 2015.

[10] Y. Kalaani, R. J. Haddad, B. Guha, A hybrid simulation and hands-on experimentation for electric machines laboratory, *2015 ASEE Southeast Section Conference*, Gainesville, FL, USA, 2015, pp. 1-13.

[11] H. T. Wu, P. C. Hsu, C. Y. Lee, H. J. Wang, C. K. Sun, The impact of supplementary hands-on practice on learning in introductory computer science course for freshmen. *Computers & Education*, Vol. 70, pp. 1-8, January, 2014.

[12] C. C. Hu, H. C. Yeh, N. S. Chen, Enhancing STEM competence by making electronic musical pencil for non-engineering students, *Computers & Education*, Vol. 150, Article No. 103840, June, 2020.

[13] L. Yuliati, F. Yogismawati, E. Purwaningsih, Y. Affriyenni, Concept acquisition and scientific literacy of physics within inquiry-based learning for STEM Education, *Journal of Physics: Conference Series*, Vol. 1835, No. 1, Article No. 012012, March, 2021.

[14] J. Freeman, B. Magerko, T. McKlin, M. Reilly, J Permar, C. Summers, E. Fruchter, Engaging underrepresented groups in high school introductory computing through computational remixing with EarSketch, *Proceedings of the 45th ACM technical symposium on Computer science education*, Atlanta, Georgia, USA, 2014, pp. 85-90.

[15] I. Calvo, I. Cabanes, J. Quesada, O. Barambones, A multidisciplinary PBL approach for teaching industrial informatics and robotics in engineering, *IEEE Transactions on Education*, Vol. 61, No. 1, pp. 21-28, February, 2018.

[16] F. W. Li, C. Watson, Game-based concept visualization for learning programming, *Proceedings of the third international ACM workshop on Multimedia technologies for distance learning*, Scottsdale, Arizona, USA, 2011, pp. 37-42.

[17] Y. Hacıoğlu, N. D. Usta, Digital game design-based STEM activity: Biodiversity example, *Science Activities*, Vol. 57, No. 1, pp. 1-15, 2020.

[18] L. Zheng, X. Li, The Effects of Motivation, Academic Emotions, and Self-Regulated Learning Strategies on Academic Achievements in Technology Enhanced Learning Environment, *2016 IEEE 16th International Conference on Advanced Learning Technologies*, Austin, TX, USA, 2016, pp. 376-380.

[19] K. P. Wild, U. Schiefele, Lernstrategien im studium: Ergebnisse zur faktorenstruktur und reliabilität eines neuen fragebogens, *Zeitschrift für Differentielle und Diagnostische Psychologie*, Vol. 15, No. 4, pp. 185-200, 1994.

[20] B. Griese, M. Lehmann, B. Roesken-Winter, Refining questionnaire-based assessment of STEM students' learning strategies, *International Journal of STEM Education*, Vol. 2, No. 1, pp. 1-12, 2015.

[21] R. Mathew, S. I. Malik, R. M. Tawafak, Teaching Problem Solving Skills using an Educational Game in a Computer Programming Course, *Informatics in education*, Vol. 18, No. 2, pp. 359-373, 2019.

[22] C. J. Chang, S. Y. Tao, The Impact of Inquiry-based Integrated STEM on Student's Perception of Learning Science and Computer Programming, *International Conference on Computers in Education*, Nanchang, China, 2021, pp. 357-360.

[23] H. Chen, *New concept visual C programming example textbook*, China Machine Press, 2019.

[24] R. L. Ebel, D. A. *Frisbie, Essentials of educational measurement*, Prentice-Hall, 1972.

[25] J. M. Bland, D. G. Altman, Statistical methods for assessing agreement between two methods of clinical measurement, *International journal of nursing studies*, Vol. 47, No. 8, pp. 931-936, August, 2010.

[26] Yulia, R. Adipranata, Teaching object oriented programming course using cooperative learning method based on game design and visual object oriented environment, *2010 2nd International Conference on Education Technology and Computer*, Shanghai, China, 2010, pp. V2-355-V2-359.

Biographies

Yu-Sheng Su is currently an associate professor of Department of Computer Science and Information Engineering at National Chung Cheng University, Taiwan. His research interests include cloud computing, big data analytics, intelligent system, and metaverse.

Chia-Jung Chang is a master's degree student in the Department of Computer Science and Engineering at National Taiwan Ocean University, Keelung, Taiwan. Her research interests include Artificial Intelligence, Metaverse, Augmented Reality, and Virtual Reality.

Jia-Hong Li received the M.S. degree from the Department of Computer Science and Engineering, National Taiwan Ocean University, Keelung, Taiwan, in 2023. His research interests include digital games based learning, computer architecture and algorithmic applications.

Shih-Yeh Chen received the Ph.D. degree from the Department of Engineering Science, National Cheng Kung University, Tainan, Taiwan, in 2016. Since 2024, he has been an Associate Professor with the Department of Engineering Science, National Cheng Kung University, Tainan. His main research interests include digital twin, human interactions, and learning support system.