International Journal of Evaluation and Research in Education (IJERE)
Vol. 9, No. 4, December 2020, pp. 939~946
ISSN: 2252-8822, DOI: 10.11591/ijere.v9i4.20653

Strengthening programming skills among engineering students through experiential learning based robotics project

Mohd Faisal Ibrahim, Aqilah Baseri Huddin, Fazida Hanim Hashim, Mardina Abdullah, Ashrani Aizzuddin Abd Rahni, Seri Mastura Mustaza, Aini Hussain, Mohd Hairi Mohd Zaman
Department of Electrical, Electronic and Systems Engineering, Universiti Kebangsaan Malaysia, Malaysia

Article Info

Article history:

Received Apr 9, 2020
Revised Sep 23, 2020
Accepted Oct 20, 2020

Keywords:

C programming
Experiential learning
Kolb model
Robotics project

ABSTRACT

This study examined the educational effects in strengthening programming skills among university's undergraduate engineering students via integration of a robotics project and an experiential learning approach. In this study, a robotics project was conducted to close the gap of students' difficulty in relating the theoretical concepts of programming and real-world problems. Hence, an experiential learning approach using the Kolb model was proposed to investigate the problem. In this project, students were split into groups whereby they were asked to develop codes for controlling the navigation of a wheeled mobile robot. They were responsible for managing their group's activities, conducting laboratory tests, producing technical reports and preparing a video presentation. The statistical analysis performed on the students' summative assessments of a programming course revealed a remarkable improvement in their problem-solving skills and ability to provide programming solutions to a real-world problem.

Corresponding Author:

Mohd Faisal Ibrahim,
Department of Electrical, Electronic and Systems Engineering,
Faculty of Engineering and Built Environment,
Universiti Kebangsaan Malaysia,
43600, Bangi, Selangor, Malaysia.
Email: faisal.ibrahim@ukm.edu.my

1. INTRODUCTION

Programming is an important computing skill that must be grasped by engineering students considering that future graduates with impeccable programming skills are increasingly demanded in the era of industrial revolution 4.0 (IR4.0). Good programming skills can assist students to solve a wide range of real-world problems. Programming skills can also help them to build effective logical and critical thinking skills that are essential in science and technology fields [1-4]. Besides, programming is a basic requirement for students to learn and understand advanced computer systems.

A conventional approach to deliver computer programming courses in universities is through classroom lectures and guided laboratories that focus on the basic understanding and concepts of a programming language. Such an approach is effective in inculcating the understanding of students on programming. However, it has a few drawbacks. First, the approach provides minimum experience for students to connect theoretical knowledge with real-world applications. Second, students cannot appreciate the usage of programming in the real world with such a monotonous approach. Therefore, existing learning methods and teaching styles should be improved to complement students' skills in adopting the theory that they have learned in the form of real-world applications [5, 6].

Various active learning methods have been proposed by past research works to increase students' motivation to learn. Some frameworks were suggested such as How People Learn (HPL) [7] to examine the motivation of students to learn programming language highlighting the contextualisation of programming applications to students' specialisation subjects. For example, work by previous researcher [8] proposed the use of programming in a bridge design project for undergraduate students of civil engineering. The results found that contextualisation can effectively improve students' motivation and appreciation of the importance of programming skills.

The mastery of programming language can also be enhanced using interactive and visual-looking programming software as shown in [9-15]. In a work by Erol and Kurt [9], students were divided into two groups, where, one group used Scratch's visual software and another group adopted the flowchart method, to learn the basic concepts of programming. The achievement of both groups was measured by the learning process and evaluation during the C# programming language session. The statistical observations found that the group of students who started learning programming with Scratch's visual software have higher achievement and motivation than the other group. The results of the qualitative study conducted by [10] are also in line with the above observations. The work proposed the usage of an integrated method of programming learning using web applications combining formal teaching in the classroom and practicing using Scratch software. Another work by [11] analysed the effectiveness of interactive programming software that enables real-time feedback given to students when writing programmes. Other interactive tools like Pencil Code collaborative software [12], Resource Flow Diagram (RFD) [13], Alice microworld [14], and virtual reality [15] were also used to promote intuitive learning. The use of the interactive feedback system demonstrates an improvement in overall student achievement.

The use of robotics systems in programming courses has also shown that it can enhance the coding skill of students and inculcate computational thinking among them [16-18]. A robot platform was used in [19] to increase the ability of electronic engineering undergraduate students to build algorithms. The students were asked to build a low-cost robot and use C programming for coding the robot behaviour. Assessment of students' motivation measured using Keller's attention, relevance, confidence, and satisfaction (ARSC) model found that the levels of attention, confidence and satisfaction of students' override that of a control group who did not use robots. Meanwhile, an online laboratory experiment that incorporates Lego Mindstorms programming for robot learning was developed in [20]. Five challenges were given to the students related to the control of robot movements. The study found that 71% of the students agreed that the usage of robot modules eases programming. Another closer approach to robotics applications in programming is called gamification [21, 22].

Another promising method of learning is via an experiential learning approach. The Kolb learning model is a prominent experiential learning approach. Many works have shown that the Kolb model is suitable to be implemented in engineering courses as an alternative approach to students' learning styles [23]. The Kolb model was implemented in [24] to design laboratory activities of mechanics of materials course for undergraduate mechanical engineering students. The survey analysis found that the method encourages students to experience hands-on learning for complementing theoretical knowledge they learned in class. The Kolb model was adopted in [25] to facilitate social collaboration in mobile cloud-based learning among team members. The results showed that the approach can fill the gap of socio-technical mechanisms to enhance teamwork performance by considering learners' behaviours and preferred computational choices. Other experiential learning methods that were previously investigated by researchers are pair programming [26], engineering design [27], game realism [28], and quasi-experimental design [29].

This work presents an integrated approach that adopts a robotics project combined with the Kolb model of experiential learning approach in a programming course attended by 114 first-year undergraduate students of electrical and electronics engineering programme at the Faculty of Engineering and Built Environment, Universiti Kebangsaan Malaysia in the 2016/2017 academic session. By implementing the proposed approach, students were given the opportunity to test their programming skills by developing programming codes for a microcontroller system that is responsible for robot movements. The microcontroller system is an important application in today's automation industry. It is widely used not only in robotics applications but also in most modern smart electronic devices.

The primary contribution of this study is the method used to implement an experiential learning approach based on the Kolb model in a robotics programming project. The second contribution is the evaluation to measure the effectiveness of the proposed method to strengthen programming skills among participated students. The proposed approach is designed with the goal of facilitating students to transfer the theoretical knowledge learned in class to the task of building computer programmes for robot navigation problems. In this project, students effectively learn and experience external factors that influence the effectiveness of building programmes. Students are required to make observations via a series of experiments and improve their programmes continuously throughout the project.

2. RESEARCH METHOD

The experiential learning approach adopted in this work is based on the Kolb model [30]. The model states that experiential learning has four stages in one learning cycle that are 1) Experiment, 2) Experience, 3) Reflect, and 4) Conceptualise. In accordance with the Kolb model, students can begin their learning experience at any stage but should follow the four stages in sequence. In the first stage, called active experimentation, students develop or choose their experiments on the basis of their pre-existing experience or theoretical knowledge. The second stage, referred to as concrete experience, students acquire new experience from their experiments conducted in the preceding stage. On the basis of this experience, in the third stage (reflective observation), students observe to find relationships between their new experience and theoretical knowledge or concepts. Finally, in the abstract conceptualisation stage, the students build a new or updated concept on the basis of their reflections in the third stage. Using these four stages, students can repeat the cycle several times to achieve any given learning objective(s). This model is useful in achieving complex learning objectives by breaking down each objective into a series of easy sub-objectives.

A robotics programming project was conducted to test the effectiveness of the Kolb model approach. The project requires the students to build computer programmes for the navigation of a mobile robot. It was executed for three weeks with a 3-hours contact session per week. Figure 1 illustrates a summary of the conducted sessions. The project started with a briefing session on robotics navigation and microcontroller-based mobile robot. Students were divided into groups with the maximum group's size of five. Next, four contact sessions were conducted throughout the project. Each contact session took 1 or 2 hours of learning time. In this session, students developed robotics navigation programmes gradually from a simple programme to a complex and complete navigation programme. After finishing the programming task, all groups competed in a robot competition hosted by the Department of Electrical, Electronic and Systems Engineering (JKEES) UKM and its student club INVEBOT. The aim of the competition, which is named as Maze Runner Robot Competition, is to navigate a mobile robot autonomously throughout a given course on a maze from a starting location to a destination location. The competition has four rounds, where each round has a different course setup. The format of the competition is a knock-out system. In every stage, each group uploaded their programmes into the robot's microcontroller and ran the robot in the maze. The navigation time of the robot was recorded by the judge. The faster the robot goes without any collision, the higher the chances to win the game.

The implementation of the Kolb model in the robotics programming project is shown in Figure 2. Four Kolb model cycles were conducted during the completion of the robotics programming tasks. Each contact session in Figure 1 represents one cycle of Kolb model implementation. In each cycle, the four stages of active experimentation, concrete experience, reflective observation and abstract conceptualisation were gone through.

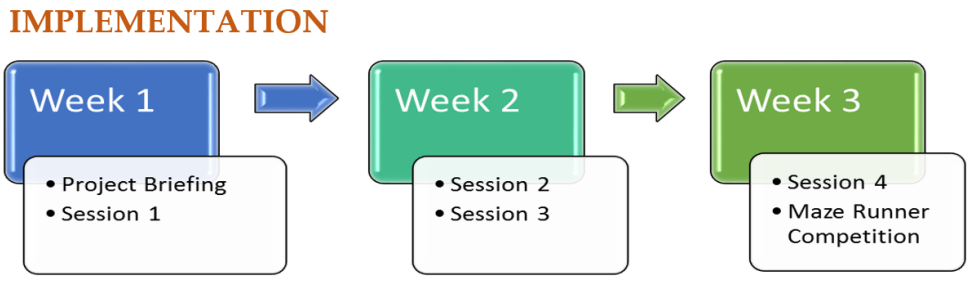

Figure 1. Contact sessions for the robotics programming project

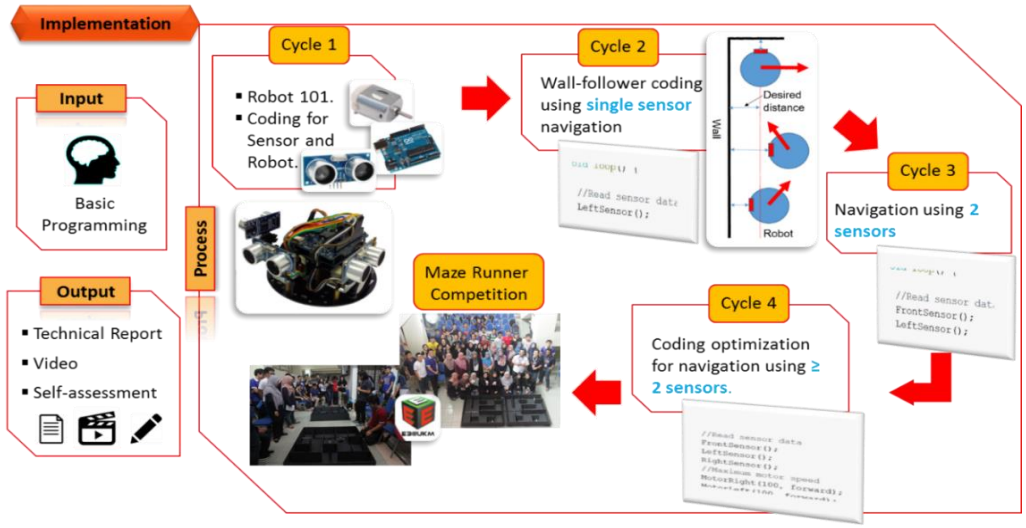

Figure 2. The complete process of the robotics programming project involving four Kolb model cycles

Before starting the project, students were taught on the concept of programming language via lecture and tutorial sessions spanning 11 weeks. With the theoretical concepts they have learned and a brief introduction to the robotics system, students started the first session (cycle 1) with the active experimentation stage of the Kolb model. At this stage, students executed an experiment that was designed for them. In this experiment, they attempted to programme a microcontroller called Arduino Uno for the first time. The experiment objective is to obtain the reading of ultrasonic distance sensors and to control the robot's wheels speed via the microcontroller. Once the programme was ready, students uploaded it into the microcontroller, turned on the robot hardware and observed the response from sensors and wheel motors. These actions were done at the concrete experience stage. Next, at the reflective observation stage, students compared the result between the expected response on the basis of their knowledge when building the programme and the actual response from the hardware. If both responses are not similar, then the student may systematically find the cause, correct the programme and test the hardware again until the intended response is achieved. Finally, before finishing the first session, students completed the abstract conceptualisation stage. Students analysed the response data and updated the preliminary concept by considering factors such as sensor noise and wheel alignment in real-world applications.

In the second session (cycle 2), students were given an opportunity to build their own experiments. In this open-ended experiment, students built a simple robotics navigation programme that controls the robot's wheels movement based on the feedback of sensor reading of one ultrasonic sensor located either on the left or the right side of the robot. The programme must be able to change the direction of robot movement such that the parallel distance between the robot and the sidewall is maintained at a certain value. The programme is also known as the wall-follower algorithm. To produce such a programme, students must use the new concepts they obtained in cycle 1 to plan an experiment. Thus, the students started this cycle with an active experimentation stage as in session 1. However, this time each group may have different experimental procedures depending on their experiment's plan. Again, all other stages in the Kolb model were followed in sequence. An example of common observation among students in this cycle is the limitation of sensor reading on a certain angle relative to the sidewall. Initially, students assume the sensor will give correct distance readings regardless of the angle of reflection between the sensor and the wall. However, during the experiment, students can observe that the distance reading from the sensor is incorrect when the robot angle relative to the wall increases. Therefore, on the basis of this observation, students redesigned their programme. In this manner, the wheels' motors act rapidly to ensure that the robot is always in a valid angle relative to the wall.

The Kolb model cycles continued in session 3 and 4. The task in the later session is more complex than that in the preceding session. Each session also required inputs from the previous session(s). Session 4 was the last cycle, where the objective is to build a complete robot navigation programme for various maze courses and multiple sensors. The programme from this final session was used by the students in a maze runner robot competition. Finally, all experiences and new concepts learned by the students were documented by them. The documents were used as evidence for assessing the ability of the students to perform the robotics programming project using Kolb's experiential learning approach. The documents consist of: 1) a

technical report on the description of the programme being built; 2) a creative video describing the robot and the experience of building the robot programmes; and 3) a self-assessment form allowing students to assess their skills and teamwork.

3. RESULTS AND DISCUSSION

This section discusses the findings on the designated research question "How does Kolb experiential learning approach implemented in the robotics programming project affect students' programming skills?" Quantitative data of the course were gathered to analyse the effectiveness of the project in strengthening students' programming skills. The primary data used for the analysis are the project marks, mid-semester examination marks and final examination marks. The project marks consist of a few assessments, namely, technical report (50%), video presentation (30%), students' commitment (10%), and competition result (10%). Mid-semester examination and final examination contain two major parts in principle. The first part is a set of questions assessing the basic knowledge of programming (50%), whereas the second part is another set of questions related to the application of programming in solving real-world problems (50%). The mid-semester examination had been conducted before the project execution. Meanwhile, the final examination was organised after the project had been completed. In this analysis, the mid-semester examination and final examination represent the pre-test and post-test marks, respectively.

To measure the degree of effectiveness of the proposed approach, two groups of students are compared. The first group is an experimental group, that is, the students' cohort who underwent the Kolb model-based robotics programming project. The second group is a control group, that is, the past students' cohort who did not experience the Kolb model-based project. The numbers of students for the experimental and control groups are 114 and 91, respectively. The control group performed a conventional problem-based learning project. Groups' members and project titles were pre-determined by the lecturer. The execution of the project is mainly depending on an unguided approach as long as the students can submit all required output including sample programmes, peer review and presentation slides.

3.1. Analysis of pre-test and post-test marks

Pre-test and post-test marks of both groups are used to observe any significant differences in terms of the academic performance of students in programming course. The result indicates the improvement of students' ability to implement programming skills in solving problems. Table 1 shows the mean of pre-test marks and post-test marks for both groups. The data revealed that the experimental group has a mean of 51.1% and 48.4% for pre-test marks and post-test marks, respectively. Thus, the mean gain from pre-test to post-test is −2.7%. The decrease in the mean score is expected, given that the cognitive level of questions for the final examination is typically higher than that for the mid-semester examination. For comparison, the mean of pre-test marks and post-test marks of the control group are 52.6% and 34.1%, respectively, giving the total mean gain of approximately −18.5%. The results show that the implementation of the Kolb model-based project improves students' programming skills measured via their formative assessment marks with 85.4% of the mean gain.

Table 1. Means of pretest (mid-semester exam marks) and post-test (final exam marks)

Group	Samples	Pre-test mean	Post-test mean	Mean gain
Experimental	114	51.1%	48.4%	−2.7%
Control	91	52.6%	34.1%	−18.5%

To verify the results, the following null hypothesis, H_o is tested:
H_o: No significant difference is found between the mean performance marks of the students who underwent the Kolb model-based project (experimental group) and the students who underwent programming project without Kolb model implementation (control group).

To test the hypothesis, the statistical data of mid-semester exam marks and final exam marks for both groups are analysed with unpaired student's T-test analysis. Table 2 and Table 3 tabulate the results of the analysis. Table 2 proves no statistically significant difference in mid-semester exam marks for the experimental group (M: 51.139, SD: 16.678) and the control group (M: 52.599, SD: 16.540) with p-value 0.5326. This result shows that both groups have programming skills capability nearly at the same strength before the project started. From Table 3, the statistical data indicate an extremely statistically significant difference in mid-semester exam marks for the experimental group (M: 48.443, SD: 18.848) and control group (M: 34.077, SD: 14.334) with p-value <0.0001. This finding suggests a statistically significant

improvement of final exam marks to the experimental group compared with the control group. Thus, the null hypothesis can be rejected. This finding is parallel to the result in [9] that utilised academic achievement to measure the students' ability to learn programming logic.

Table 2. Student's T-test results of the mid-semester exam for the experimental and control groups

Group	Experimental	Control
Mean (M)	51.139	52.599
Standard deviation (SD)	16.678	16.540
Standard error mean (SEM)	1.562	1.734
No. of samples (N)	114	91

Two-tailed p-value = 0.5326 (95% confidence interval)

Table 3. Student's T-test results of the final exam for the experimental and control groups

Group	Experimental	Control
Mean (M)	48.443	34.077
Standard deviation (SD)	18.848	14.334
Standard error mean (SEM)	1.765	1.503
No. of samples (N)	114	91

Two-tailed p-value < 0.0001 (95% confidence interval)

3.2. Analysis of final exam questions related to the application of programming to real-world problems

An objective of this project is to provide students with ample experience to implement programming skills for solving real-world applications. Problem solving skill is an important required skill to learn programming and remains as a great challenge for students [5]. On this basis, we measure how such experience benefits students of the experimental group in answering questions in the final exam related to the application of programming to real-world problems compared with the control group. For this type of questions in the final exam, students can supposedly 1) translate a given problem into a logical flow of process using a flowchart, 2) create C programmes based on the flowchart and 3) test the logic of the programme by inserting a specific instance of variables used in the programmes. Figure 3 shows the boxplots of the final exam marks of real-world problem questions, where the maximum accumulated marks are 50. The median marks for the control and experimental groups are 11.0 and 21.0, respectively, showing the improvement of final exam marks for about 10 marks. This positive finding is aligned with the study by [19] that shows students academic achievement increased by about 15% with the use of a robotic-based project.

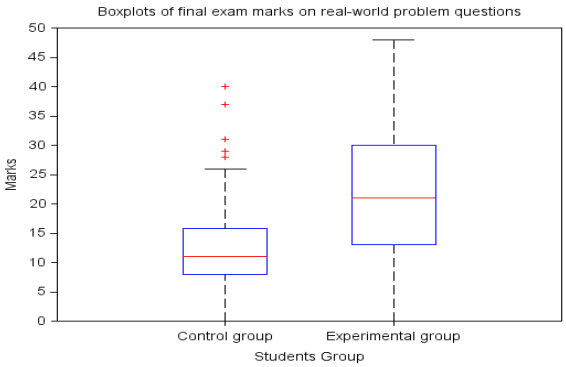

Figure 3. Boxplots of the final exam marks on real-world problem questions for the control and experimental groups

Table 4 shows the statistical analysis of student's T-test on the data. A statistically significant difference is found in the final exam marks on real-world problem questions for the experimental group (M: 21.864, SD: 11.122) and control group (M: 12.275, SD: 7.706) with p-value <0.0001. Thus, the Kolb model-based project can help students in designing solutions for real-world problems with programming. The finding is also in line with the study by [7] in the context of a knowledge-centre approach that students will be more motivated when the contextualised applications of programming are emphasized.

Table 4. Student's T-test results of final exam marks on real-world problem questions for experimental and control groups

Group	Experimental	Control
Mean (M)	21.864	12.275
Standard deviation (SD)	11.122	7.706
Standard Error Mean (SEM)	1.042	0.808
No. of Samples (N)	114	91

Two-tailed P value < 0.0001 (95% confidence interval).

3.3. Correlation between project marks and final marks

This test aims to observe whether students in the experimental group who did better in the project were also got good marks in the final examination. For verifying whether the case is true, Spearman's rank correlation test is calculated between the project marks and final examination marks. As a comparison, the statistical test was used by [13] to find correlation between video-based learning materials and ability of students to understand programming.

The scores of 114 students were ranked for the project and the final examination. The lower the rank, the lower the marks of a student compared with the others, and vice versa. Table 5 shows the results of Spearman's rank correlation analysis for the ranking of the students in the experimental group based on project marks and final exam marks. The table presents Spearman's correlation coefficient, its significance value (p-value) and the sample size N.

Spearman's correlation coefficient is 0.241. It is statistically significant with p-value 0.00991 (95% confidence interval). Thus, the coefficient indicates a correlation between the rankings of students in the project and final exam marks. However, this minimum correlation might be contributed by the mixture of evaluation of individual and group assessments in the project. Although it is not strongly monotonically related, the tendency of a student to perform better in the final exam when he did well in his project is promising.

Table 5. Spearmans's rank correlation analysis for ranking of students in experimental group

			Project	Final exam
Spearman's rho	Project	Correlation coefficient	1.000	0.241
		Two-tailed p-value		0.00991
	Final exam	Correlation coefficient	0.241	1.000
		Two-tailed p-value	0.00991	

All calculation is based on sample size N = 114

4. CONCLUSION

This paper presents an experiential learning-based robotics project to strengthen programming skills among electrical and electronics engineering students. The approach was successfully implemented by using the Kolb learning model. The project has four cycles in which each cycle implements four stages of the Kolb model. The analysis shows that the implementation of the Kolb model-based robotics project improves the score of students' formative assessment of approximately 85.4% from the control group. This is supported by the post-test results reporting statistically significant improvement of final exam marks of the experimental group. There is also a correlation between the rankings of students in the project and final exam marks. In addition, the Kolb model-based robotics project improves engineering students' programming skills in the context of understanding the concept of programming, interest in programming languages and the ability to solve real- world problems with coding.

ACKNOWLEDGEMENTS

Authors especially thank JKEES UKM, INVEBOT club and Pn. Jamaliah Abu Bakar from FKAB, UKM for providing the technical and administrative support in this project. This article is financially supported by a grant with no. GUP-2018-103 from Universiti Kebangsaan Malaysia.

REFERENCES

[1] F. Kamaruzaman, R. Hamid, A.A. Mutalib and M.S. Rasul, "Comparison of Engineering Skills with IR 4.0 Skills," *International Journal of Online and Biomedical Engineering*, vol. 15, no. 10, pp. 15-28, 2019.

[2] A.M. Seoane-Pardo, "Computational Thinking between Philosophy and STEM - Programming Decision Making Applied to the Behavior of 'Moral Machines' in Ethical Values Classroom," *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 13, no. 1, pp. 20-29, 2018.

[3] A. Babatope, *et al.*, "Competence-driven engineering education: A case for T-shaped engineers and teachers," *International Journal of Evaluation and Research in Education*, vol. 9, no. 1, pp. 32-38, 2020.
[4] A.Z. Abidin, E. Istiyono, N. Fadilah, W.S.B. Dwandaru, "A computerized adaptive test for measuring the physics critical thinking skills," *Int. Journal of Evaluation and Research in Education*, vol. 8, no. 3, pp. 376-383, 2019.
[5] R.P. Medeiros, G.L. Ramalho and T.P. Falcao, "A Systematic Literature Review on Teaching and Learning Introductory Programming in Higher Education," *IEEE Transactions on Education*, vol. 62, no. 2, pp. 77-90, 2019.
[6] N.D.S. Chetty, *et al.*, "Learning styles and teaching styles determine students' academic performances," *International Journal of Evaluation and Research in Education*, vol. 8, no. 3, pp. 610-615, 2019.
[7] N.A. Azmi, K. Mohd-Yusof, F.A. Phang and S.A.H. Syed Hassan, "Motivating Engineering Students to Engage in Learning Computer Programming" In: Auer M., Kim KS. (Eds), *Engineering Education for a Smart Society. WEEF 2016, GEDC 2016. Advances in Intelligent Systems and Computing*, vol. 627, Springer, Cham, 2018.
[8] J.D. Bowen, "Motivating Civil Engineering Students to Learn Computer Programming with a Structural Design Project," *2004 Proc. American Society for Engineering Education Annual Conference and Exposition*, pp. 9.931.1-9.931.11 2004.
[9] O. Erol and A. Kurt, "The Effects of Teaching Programming with Scratch on Pre-Service Information Technology Teachers' Motivation and Achievement," *Computers in Human Behavior*, vol. 77, no. 1, pp. 11-18, 2017.
[10] J. Cárdenas-Cobo, *et al.*, "Recommending Exercises in Scratch: An Integrated Approach for Enhancing the Learning of Computer Programming," In: Auer M., Kim KS. (Eds), *Engineering Education for a Smart Society. WEEF 2016, GEDC 2016.* Vol. 627, pp. 255-271, Springer, Cham, 2018.
[11] J. Coenen, S. Gross and N. Pinkwart, "Comparison of Feedback Strategies for Supporting Programming Learning in Integrated Development Environments (IDEs)," In: Le NT., van Do T., Nguyen N., Thi H. (eds), *Advanced Computational Methods for Knowledge Engineering. ICCSAMA 2017.* vol 629, pp. 72-83, Springer, Cham, 2018.
[12] W.B. Deng, *et al.*, "Pencil Code improves learners' computational thinking and computer learning attitude," *Computer Applications in Engineering Education*, vol. 28, no. 1, pp. 90-104, 2019.
[13] S. Tsukuta, T. Yamaguchi, T. Kano, Y. Kishimoto and E. Nunohiro, "Practical problem-based programming learning using video," *Artificial Life and Robotics*, vol. 24, no. 3, pp. 422-429, 2019.
[14] J.M. Costa, "Microworlds with Different Pedagogical Approaches in Introductory Programming Learning: Effects in Programming Knowledge and Logical Reasoning," *Informatica*, vol. 43, no. 1, pp. 145-147, 2019.
[15] M.D.G. Rios and M.P. Velasco, "Augmented Reality as a Methodology to Development of Learning in Programming," *International Conference on Technology Trends*, vol. 895, pp. 327-340, 2019.
[16] Y. Ohnishi, *et al.*, "Evaluation for Task Achievement of Robotics Programming Based on Image Information," *Journal of Robotics & Mechatronics*, vol. 31, no. 3, pp. 427-433, 2019.
[17] J.T. Aparicio, S. Pereira, M. Aparicio and C.J. Costa, "Learning Programming Using Educational Robotics," *14th Iberian Conference on Information Systems and Technologies*, vol. 560, pp. 215-222, 2019.
[18] B.L.S. Lima, *et al.*, "Robotics Application Project-Oriented Programming: Learning and Competition," *International Conference on Applied Physics, Power and Material Science*, vol. 1172, pp. 1-10, 2019.
[19] O.O. Ortiz, *et al.*, "Innovative Mobile Robot Method: Improving the Learning of Programming Languages in Engineering Degrees," *IEEE Trans. on Education*, vol. 60, no. 2, pp. 143-148, 2017.
[20] T.O. Almeida, J.F.M. Netto and M.L. Rios, "Remote robotics laboratory as support to teaching programming," In *Proc. IEEE Frontiers in Education Conference*, pp. 1-6, 2017.
[21] A. Rojas-Lopez, *et al.*, "Engagement in the course of programming in higher education through the use of gamification," *Universal Access In The Information Society*, vol. 18, no. 3, pp. 583-597, 2019.
[22] R.M. Flores and M.M.T Rodrigo, "Wheel-Spinning Models in a Novice Programming Context," *Journal of Educational Computing Research*, vol. 58, no. 6, 2020. (Early access).
[23] L.S.J. Arturo and P.N. Zapata, "Learning styles, a correlational study in engineering students," In *2010 Proc. IEEE ANDESCON*, pp. 1-9, 2010.
[24] M. Muscat and P. Mollicone, "Using Kolb's Learning Cycle to Enhance the Teaching and Learning of Mechanics of Materials," *International Journal of Mechanical Engineering Education*, vol. 40, no. 1, pp. 66-78, 2012.
[25] G. Sun and J. Shen, "Facilitating Social Collaboration in Mobile Cloud-Based Learning: A Teamwork as a Service (TaaS) Approach," *IEEE Transactions on Learning Technologies* vol. 7, no. 3, pp. 2017-220, 2014.
[26] T.A.N. de Oliveira and A.D. Reboucas, "The use of Pair programming to support introductory programming teaching: A qualitative study," In *Proc. XIII Latin American Conf. on Learning Technologies*, pp. 65-68, 2018.
[27] W.F. Lu, H.W. Lim and K.H. Goh, "Engineering Design and Education: A Case Study on Designing a Competition Fuel Efficient Vehicle through Experiential Learning," In *Proc. The ASME International Design Engineering Technical Conferences and Computers and Information In Engineering Conference*, pp. 741-750, 2012.
[28] A.C. Urquidi-Martin, C. Tamarit-Aznar and J. Sanchez-Garcia, "Determinants of the Effectiveness of Using Renewable Resource Management-Based Simulations in the Development of Critical Thinking: An Application of the Experiential Learning Theory," *Sustainability*, vol. 11, no. 19, pp. 1-15, 2019.
[29] Mutmainah, Rukayah and M. Indriayu, "Effectiveness of experiential learning-based teaching material in Mathematics," *International Journal of Evaluation and Research in Education*, vol. 8, no. 1, pp. 57-63, 2019.
[30] D.A. Kolb, *Experiential learning: Experience as the source of learning and development* (2nd Ed.). New Jersey: Pearson Education, 2015.