Ogata, H. et al. (Eds.) (2015). Proceedings of the 23rd International Conference on Computers in Education. China: Asia-Pacific Society for Computers in Education

Open Designettes, Flowcharts, and Pseudocodes in Python Programming with the Aid of Finch

Cheah Huei YOONG[a*], Hyowon LEE[a], Ngai-Man CHEUNG[a], Geok See NG[a], Yue ZHANG[a] & Subhajit DATTA[a]
[a]*Information System and technology Design (ISTD), Singapore University of Technology and Design, Singapore*
* andrew_yoong@sutd.edu.sg

Abstract: Many universities around the world have introduced design in their engineering and computer-related curriculum. Design has played a role in student learning. Flowcharts and pseudocodes have been used in teaching computer programming and computational thinking. However, the combination of open designettes, flowcharts, and pseudocodes used to aid in teaching programming language are not investigated yet. In this paper, we have explored the teaching of Python programming language using open designettes, flowcharts, and pseudocodes in a series of weekly mini-projects with the help of a small, simple and educational programmable robot called Finch. Our studies revealed that these techniques have contributed positively to student learning and students' participation through observation, student participation rate and survey result. We believe the sharing of this experience in the paper will help academics and educators in the similar mission of enhancing students' learning in courses where students are from various countries and with different learning abilities.

Keywords: open designettes, flowcharts, pseudocodes, Finch, Python programming language

1. Introduction

Universities around the world are integrating design into their curriculum including engineering, architectures, sciences and arts. Design and technology have shown as frontrunner to many new discoveries. Design based research is part of the methodology from theory to practice (Dede, 2004; The Design-based Research Collective, 2003). Design plays a crucial role to enhance learning (Veurink, 2008), create useful knowledge, advance theories and principles of learning and teaching in educational environment, notably in complex settings.

The concept of *designettes* is coined by Singapore University of Technology and Design (SUTD) (Wood, 2012). Broadly, designettes in a sense includes the glimpses, snapshots, small-scale, short turnaround and well-scoped design problems, or mini-projects, providing a significant design experience, while requiring a minimal amount of time and resources. Designettes can be scoped according to time and resource allocation. They may be apportioned to particular technological or project domains, they may focus on certain phases of design process (Wood, 2012). Designettes have helped to increase students' awareness of applications and learning content (Telenko, 2014). The effectiveness of designette lies in the fundamentals of well-known learning approaches such as Kolb's cycle, Bloom's revised taxonomy, scaffolding, Myers-Briggs Type Indicator (MBTI), five factor model and Novak's model of learning. These approaches guide the formation, evaluation and advancement of designiettes within the active learning pedagogy. However, not all of these learning approaches may be used in forming designettes for student learning. Designettes can be grouped into two categories: open designettes and constrained designettes. Open designettes cover open-ended design problems which may be scoped according to time and resource allocation. This method promotes creativity and enhances problem-solving skills. The end design results will have multiple solutions to the problem. The second category, constrained designettes, contains design problems with limitations like using a maximum of three features of the given robot and/or some stated requirements, for instance navigating the robot to each side of a rectangular wall (Yoong, 2014). For the latter, the robot may be designed to visit each adjacent side inside the rectangle. There are some other possibilities to this problem too. Constrained designettes may be tailored to time limit and resource requirements at the sametime. This

method also encourages creativity and improves problem solving skills but to a limited extent. Thus, the design solutions to the problem will likely be fewer than open designettes. Nevertheless, students may find the design solutions to constrained designettes to be easier as the scope is narrower and less challenging compared to open designettes. Well-structured small-scale design project that has only one design solution is not considered as a basic characteristic of designettes (Wood, 2012). As there is only one answer, there is little room for students to think holistically and creatively, and they can easily verify their solution.

The use of program design before coding produces positive results (Faux, 2006). Algorithms are part of program design as program design is made up of steps a software developer should follow before starting writing codes in a computer language such as Python and Java. An algorithm is a series of steps to be performed to solve a problem. *Flowcharts* and *pseudocodes* are two of the techniques used to formulate and express algorithms. Flowchart was first introduced in 1920s (http://en.wikipedia.org/wiki/Flowchart). Pseudocodes were used many years before the first computer was invented. There are still no standard for pseudocodes syntax because pseudocodes are not executable programs. Flowchart has a higher abstraction than pseudocodes. Both flowcharts and pseudocodes are still used today as part of program design (Giordano, 2015; Ni, 2006). Flowcharts (Carlisle, 2004; Jesus, 2011) and pseudocodes (Lekkos, 1978) are used in teaching programming. Designettes have been used to teach design in various educational settings (Telenko, 2014; Wood, 2012). However, the combination of open designettes, flowcharts and pseudocodes used together to teach programming language have not be reported in the literature. In this paper, we have investigated the contribution of open designettes, flowcharts and pseudocodes in mini-projects to aid in teaching Python programming using a small, off-the-shelf, programmable robot called Finch (http://www.finchrobot.com). As will be discussed later in the paper, these methods have positively contributed to student learning and strong student engagement, evidenced from rough sketches of stories to demonstrations (Figures 1 to 14).

Section two briefly described the learning theories used in our investigation and how the theories were utilized in running the mini-projects. Section three highlights the background and context of the pedagogical stance and delivery approach at SUTD. Section four discusses the implementation of the series of mini-projects particularly evidences of learning process and demonstration of mini-project four. Section five reports the findings. Section six concludes the paper.

2. Learning Theories and Methods

Three learning theories are briefly discussed because they are used in our investigation. They are Bloom's revised taxonomy, active learning, and scaffolding. Bloom's revised taxonomy (Andersin, 2001) is a multi-tiered model of six cognitive complexity levels. The thinking skills from the lowest level to the highest level are remembering, understanding, applying, analysing, evaluating, and creating. The lowest level corresponds to lowest level thinking skills (remembering) and the increasing levels correspond to a more advanced learning process all the way up to the highest level thinking skill (creative thinking). The open designettes used in our research encourages creative thinking (highest level in Bloom's model) because it helps to generate/create ideas as students are free to use any features of the robots, other equipment and materials to tell their stories. Active learning improves overall student learning (Aglan, 1996) and is used effectively in Science, Technology, Engineering and Mathematics curriculum today. Active learning is a model of instruction that engages students in pedagogical activities to promote learning (Calabro, 2008; Celebi, 2011; Hake, 1998; Linsey, 2009). In active learning, the instructor's role is a facilitator rather than a teacher. Active learning is embedded in the four mini-projects, notably the fourth mini-project used in our research where open designettes engaged students to be creative in problem solving and design the environment to tell a story. Scaffolding is an instructional technique to move students progressively towards greater understanding of the materials in learning process. Scaffolding is used to bridge learning gaps of what the students know and what they are expected to know and be able to do at a certain point in their education. Scaffolding was employed in our course as the sequence of the four mini-projects incrementally challenged our students discussed in section four.

Flowcharts and pseudocodes are chosen in our experiment because both are structured methods. This prevented many ambiguities common in statements of natural languages. Moreover, they are easy to understand and use especially for beginners in program design and computer programming. Both methods are effective if the program logic is not complex. In fact, the emphasis to use both methods in our investigation is to stress the importance of design first before implementation. Many students tend to write program first before design thinking. Hence, we wanted to make students aware the proper steps of software engineering that is design before coding in a specific computer language.

3. Pedagogical Context and Background

The Finch robot was chosen because it had good value for money compared with other higher-end robots such as LEGO NXT, Scribble-2 and CSbots. The Finch robot is inexpensive and easy to program and is operated by a universal serial bus cable directly connected to a computer running the program. The cheap cost of Finch, about US$90 each, has allowed us to pair two students for each robot. This gave better opportunity for student to engage and learn. Although there were minor problems with one of the sensors, the robot helped us achieve our objectives at this level of education.

At SUTD, cohort-based active learning is mandated from the university level. For this term, there are more than 280 Freshmore (Freshman plus sophomore first half) students who take the Digital World (DW) module. The students' nationalities are diverse. The students have a wide range of learning abilities. There are seven cohorts and each cohort consists of more than 40 students. Each cohort has two instructors and one or two teaching assistants (TAs). Students may ask questions to any instructors and TAs to clarify their doubts during or after class. In DW module, we assume that students have no prior program design and programming experience. The pedagogy activities integrate in engaging students through active learning are the understanding of open designettes problem, hands on experience using flowcharts, pseudocodes, writing Python programs and Finch. Team-based-learning and problem-based learning are used to promote active learning. This setting allowed us to conduct the study both qualitatively (observation within the class) and quantitatively (survey and participation percentages).

4. Mini-projects Implementation

Scaffolding technique was employed in mini-projects to motivate students progressively towards greater understanding of program design and computer programming through week two to week six. There were a total of four mini-projects using the Finch. In week two, the mini-project was the simplest – explore the Finch features and basic programming to make the Finch move forward, backward, and rotate. In week three, students were to solve very easy problems using functions and parameter passing in programming the robot. In week four, the mini-project was to utilize the if-else-statement, loops and lists in controlling the robot. During the grading of each mini-project, students were required to show the flowcharts and pseudocodes. Each mini-project required at least some knowledge gained from the previous mini-project(s).

The difficulties of mini-projects increased from week two to week six. The last mini-project provided opportunities for students to be more independent and creative in the learning process. This was an open designettes cum flow chart and pseudocodes exercise. This mini-project encompassed the knowledge learnt from the previous three mini-projects and any other aspects learned throughout the course thus far. The first mini-project (week two) was to get students know the basic Finch features such as the three sensor types (light, temperature and obstacle) and buzzer. The students were required to apply fundamental Python commands to control Finch movements including rotating clockwise/ anticlockwise, moving forward/backward and triggering some sensor readings. The students needed to design flowcharts and pseudocodes for very simple Python programs. The demonstration was carried out during class in front of each of the two instructors or any teaching assistants. We had viewed that the students were interested to explore the Finch as it was designed to look cute. At the cognitive point of view, students were able comprehend the features of Finch and the use of them in a short period of

time. During testing, some students were excited about the robot's movement and the music it had produced.

The second mini-project (week three) developed on the previous one to write short Python programs to solve slightly more sophisticated problems. The main objective of was to use functions and parameter passing in the programs. Design of flowcharts and pseudocodes for these programs were stated in the objectives. In the third mini-project (week four), longer and more difficult problems were required to solve by students using two-dimensional list, selection and loops. Flowcharts and pseudocodes were part of the objectives. The programs written by students may vary but the ultimate results were the same. In the final mini-project (weeks five and six), the open designettes problem was part of the small scale mini-project to be completed within 2 weeks using Finch. The objectives of this project were clearly stated. The deliverables were creativity in telling a story around what the robot does, flowcharts, pseudocodes, and demonstration of any application program using Finch as part of the story. Both flowcharts and pseudocodes were part of program design as all previous mini-projects. Students were to create an intriguing story, from which they design a plan for implementation to fit the unfolding of the story. Finally, they shared the story using the developed application in Python programming language for the Finch.

In order to get some sense of how the mini-project was delivered and what the level of student engagement was like, we illustrate the progress and student engagement using the Figures 1 to 14. In the learning process, draft snippets of story (Figures 1-3), flowcharts (Figures 4-5) and pseudocodes (Figure 6) were observed. Moreover, active student discussion was noticed to derive a story (Figures 7-10) and sell the story (Figure 11). During demonstration, students had shown creativity by using boxes (Figures 12-14), books, and other objects as part of the design to tell the story. Many short stories like hot coffee machine and strong Finch were told during the demonstration.

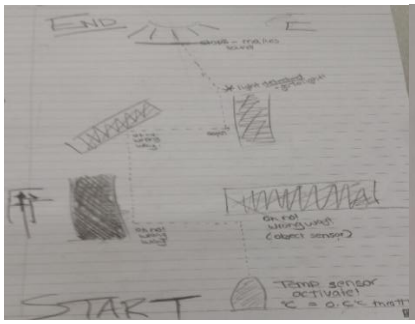

Figure 1. Sketch a story

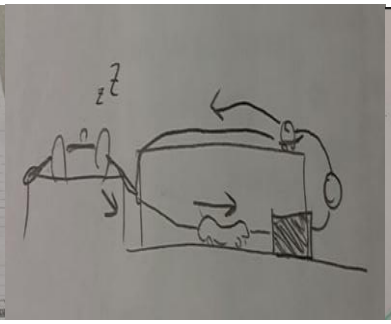

Figure 2. Rough sketch of another story

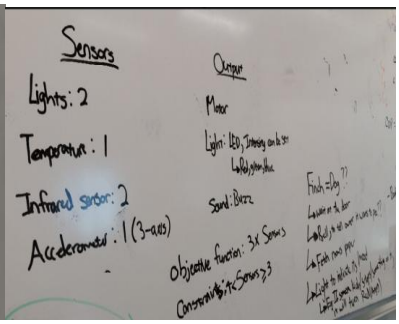

Figure 3. Forming idea of a story

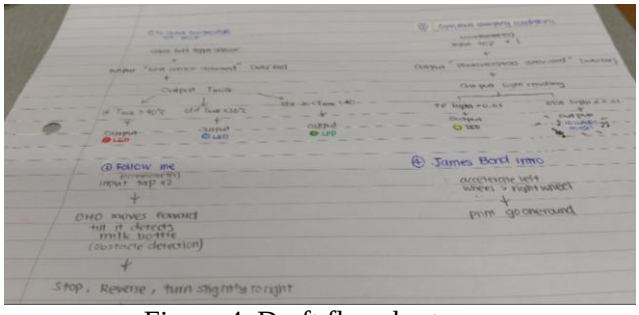

Figure 4. Draft flowchart

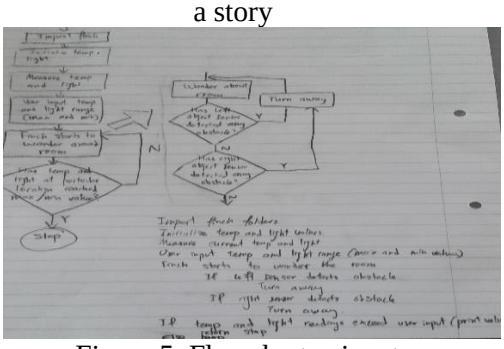

Figure 5. Flowchart snippet

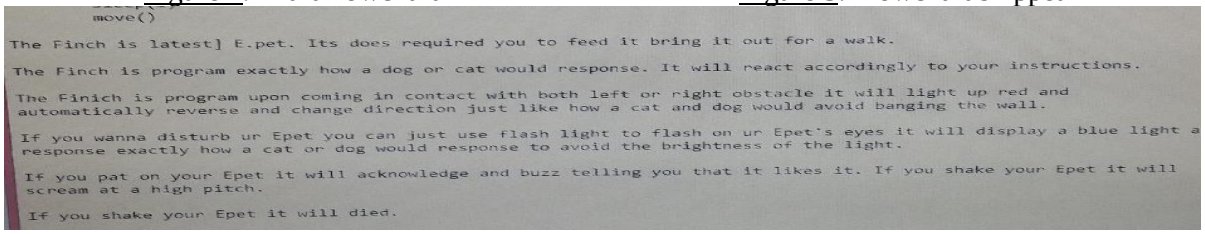

Figure 6. Draft pseudocodes

Figure 7. Student engagement

Figure 8. Student working together

Figure 9. Student discussion

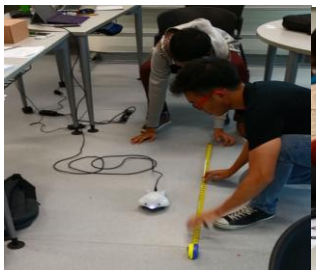

Figure 10. Teamwork

Figure 11. Practise to sell the story

Figure 12. Box to tell story

Figure 13. Box and cartoons to tell story

Figure 14. Boxes to tell story

5. Results and Discussion

Average percentage for students who had completed mini-projects one to four was about 99%. This had showed active participation from students, specifically mini-projects one and four. Figures 7 to 14 had supported the findings of mini-project four. Students' engagement enhanced their learning.

The survey exercise attracted 57 respondents. Q1. The Finch mini-projects 1 to 3 helped me to better understand loops, if-else statements and lists in Python programming language (87.7%). Q2. The Finch mini-project 4 had enhanced my creativity, research and problem solving skills (86.0%). Q3. The Finch projects 1 to 4 gave me practice on flowcharts and pseudocodes (94.7%). Q4. On the whole, the Finch project 4 had helped to improve my Python programming skills (93.0%). Q5. Overall the Finch projects motivated me to learn Python programming language (80.8%). Q6. In general, the Finch projects were important additions to the existing teaching methods for Python programming language in Digital World module (89.5%). The results of survey questions one, two and four indicated that the mini-projects had helped students learn Python programming language and improved their creativity, research and problem-solving ability. Open designettes, flowcharts, and pseudocodes were part of the mini-projects deliverables, particularly mini-project four (Figures 1 to 6). Hence, they had contributed positively to student learning. Many students at least agreed that the mini-projects gave them practice on flowcharts and pseudocodes (Q4) and motivated them to learn Python (Q5). On the whole, many students felt that the mini-projects were positive additions to existing teaching methods of DW module. For the price of a Finch, the robot was good enough to serve our purposes and achieve desired results. Our observations in class, survey results, and participation rates showed that there was a certain level of student enthusiasm during the mini-projects period.

6. Conclusions

We have determined that the use of flowcharts, pseudocodes, and open designettes in mini-projects have contributed to the positive results of student learning as they are part of mini-projects deliverables, especially mini-project four (Figures 1 to 6). Furthermore, these techniques have added to active student engagement in mini-projects as shown from Figures 7 to 14 in section four and in section five. By being engaged, students learning have been enhanced. Demonstration of mini-projects during cohort classrooms, surveillance of students working with the robot, participation rates and survey are methods used to measure the effectiveness of our research. The three learning theories that are utilized to various extend in our research and course are active learning, revised Bloom taxonomy, and scaffolding. Flowcharts and pseudocodes are used in mini-projects to introduce design first before coding as both methods are part of program design. Last but not least, we expect that the experiences and materials shared in this paper will be beneficial to readers interested to follow an analogous approach in courses where students come from different countries and with a wide range of learning abilities. In future, we hope to expand our pedagogical activities with more fine-tuned series of mini-projects and add other related technologies to the mini-projects.

References

http://en.wikipedia.org/wiki/Flowchart

http://www.finchrobot.com

Andersin, L. W. and et al. (2001). A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives, Addison Wesley Longman.

Calabro, K. M., Kiger, K. T., Lawson, and W, Zhang. G. (2008). New directions in freshman Engineering Design at the University of Maryland. ASEE/IEEE Frontiers in Education Conference, pp. T2D: 6-11.

Carlisle, M. C., Wilson, T. A., Humphries, J. W., and Hadfield, S. M. (2004). RAPTOR: introducing programming to non-majors with flowcharts. Journal of Computing Sciences in Colleges, 19(4), pp. 52-60.

Celebi, H. and Qarage, K. A. (2011). Engagement of Undergraduate Students in Advanced Research: A Case Study. IEEE Global Engineering Education Conference (EDUCON), pp. 529-532.

Dede, C. and et al. (2004). Design-based research strategies for studying situated learning in a multi-user virtual environment. 6th International Conference on Learning Sciences, pp. 158-165.

Faux, R. (2006). Impact of preprogramming course curriculum on learning in the first programming course. IEEE Transactions on Education , 49(1), pp. 11-15.

Giordano, D. and Maiorana, F. (2015). Teaching algorithms: Visual language vs flowchart vs textual language. IEEE Global Engineering Education Conference (EDUCON), Tallinn, Estonia, pp. 499-504.

Hake, R.R. (1998). Interative-engagement versus traditional methods: A six-thousand-student survey of mechanics test data for introductory physics courses. Journal of Physics, 66(1), pp. 546-552.

Jesus, E. (2011). Teaching computer programming with structured programming language and flowcharts. Workshop on Open Source and Design of Communication, pp. 45-48.

Lekkos, A. A. (1978). How to develop module logic using pseudo-code and stepwise refinement. 15th Design Automation Conference, pp. 366-370.

Linsey, J., Talley, A., White, C. K., Jensen, D., and Wood, K. L. (2009). From Tootsie Rolls to Broken Bones: An Innovative Approach for Active Learning in Mechanics of Materials. ASEE Journal of Advances in Engineering Education (AEE), 2009, 1(3), pp. 1-23.

Ni, X. and et al. (2006). The design and implementation of HAVC system. International Conference on Power System Technology, pp. 1-6.

Telenko, C., Camburn, B., Katja, H-O., Wood, K. and Otto, K. (2014). Designettes: New approaches to multidisciplinary engineering design education. International Design Engineering Technical Conference & Computers and Information in Engineering Conference, Buffalo, New York, USA.

The Design-Based Research Collective (2003). Design-based research: An emerging paradigm for educational inquiry. Educational Researcher, 32(1), pp. 5-8.

Veurink, N. and Hertel, J. (2008). Integrating solid modelling and computer programming for freshman design experience. 38th ASEE/IEEE Frontiers of Education, Saratoga Springs, New York, USA (pp. SID-3 to SID-7).

Wood, K. L. and et al. (2012). A symphony of designiettes exploring the boundaries of design thinking in engineering education. 119th ASEE Annual Conference, San Antonio, Texas, USA.

Yoong, C. H. (2014). Benefits and introduction to python programming for freshmore students using inexpensive robots. IEEE Teaching, Assessments & Learning, Wellington, New Zealand, pp. 12-17.