Design of a VR Application Based on Cognitive Load Human Movement Effect to Aid Basic Programming

Carlos A. Arévalo Mercado[1], Jorge A. Muro Rangel [1] and Estela Lizbeth Muñoz Andrade [1]

[1,2,3] *Autonomous University of Aguascalientes (UAA), Av. Universidad #940, Ciudad Universitaria, Aguascalientes, México*

Abstract

The increasing need to train capable software developers from universities to the IT industry, and the inherent complexity of learning programming, drives the exploration of new learning methods to aid novice students. Many variations of VR applications have been reported in this context, using multiple designed principles such as metaphors, puzzle solving and visualization. The confluence of cognitive psychology, brain structure biology and Virtual Reality is a promising area of research. In this study, a VR application to help students solve pseudocode exercises was designed, developed, and refined using human movement effect and completion problem effect as instructional guidelines from cognitive load theory, which have shown through empirical research to be an effective learning strategy. A user experience questionnaire (UEQ) was applied to first year computing students to refine a first prototype. A second version was developed using the feedback of UEQ which indicated a need for better dependability, better efficiency, and a better novelty factor.

Keywords

Programming, Virtual Reality, Cognitive Load Theory, Human Movement Effect, Completion Effect.

1. Introduction.

The increasing use of information technologies in personal, commercial, and educational spheres in modern societies has led to an increase in the demand for software developers, so the ability to write code in programming languages has become key. for the economic and social development of regions and countries. However, the software industry is in constant shortage of qualified programmers to develop these technologies [1].

In university programs, developing a programming logic is one of the first skills that new students in careers related to computer science must acquire. But these tend to be difficult for new students, due to the barrier represented by the process of acquiring problem-solving strategies, the creation of relevant mental models, programming language syntax, the development of algorithmic thinking and even emotional barriers derived from previous experiences [2]–[4] among other factors reported in literature. This inherent difficulty has led to global passing percentages of introductory programming courses to be slightly higher than 60% [5], [6], which suggests the need to increase motivation and reduce the perception of difficulty of subjects related to programming [6].

In this study, we propose a Virtual Reality application to aid programming instructors with pseudocode exercises that can be solved using human movement principles, using motion controllers. The design used instructional design guidelines taken from cognitive load theory, specifically, the completion problem effect and the human movement effect. The application was tested for user experience and later refined and is yet to be tested for learning effectiveness.

CISETC 2023: International Congress on Education and Technology in Sciences 2023, December 04–06, 2023, Zacatecas, Mexico
carlos.arevalo@edu.uaa.mx (C. Arevalo); al289298@ed.uaa.mx (J. Muro); lizbeth.munoz@edu.uaa.mx
0000-0002-8349-7985 (C. Arevalo); 0000-0003-4182-5044 (E. Muñoz); (J. Muro) 009-0006-3315-3179

2. State of the Art.

2.1. Virtual Reality

The concept of Virtual Reality (VR) can be defined as a 3-dimensional computer-generated simulation of images or environments where you can interact visually and physically using specialized [7]electronic equipment.
VR technology has gone through several commercial waves, where it has not been until the last decade with advances in sensors, processors, and cameras, that commercial VR equipment became accessible to the home and educational consumer. and in which devices with HMD (Head Mounted Displays) already provide an experience with a high degree of [8] immersion.

2.2. Virtual reality as an aid for teaching programming

Code visualization and visual metaphors [9] have been used as a starting point for the development of VR-based support tools for learning programming, translating abstract concepts into immersive and interactive environments.

For example, [10] proposed a programming environment, called 'Cubely', based on the popular game MineCraft where the participant must solve programming problems by assembling the answer using cubes, which have the instructions of the program. and where the interaction occurs in the assembly of puzzles.

It is reported that interaction with virtual reality helps release the cognitive load by making the [11] student understand programming concepts faster, through code analysis in two simultaneous spaces: a computer panel and a space for action. The user navigates the code from start to finish in the panel while observing changes in the environment and performing actions on variable objects in the action space. The code generation option displays a joint window for the user to relate their actions to the resulting code.

A virtual reality application is reported for the visualization, navigation, and transmission of information of code structures in an immersive and interactive way to support the cognitive, exploratory, analytical and descriptive processes of code, through a VR based prototype [12] called FlyThruCode (VR-FTC). The objective of this prototype is to help developers have a better visualization of code structures and encourage compression processes.

2.3. Cognitive load theory.

Cognitive Load Theory (CLT) takes as reference the human cognitive architecture [13], which has two types of memories: working memory and long-term memory. The first is limited in terms of the number of discrete elements that it can process and store simultaneously and the second is unlimited in terms of storage capacity. It is postulated that learning takes place when the so-called 'schemas' are created, organized, and stored in long-term memory, which are cognitive constructs that allow multiple elements to be organized as a single block and that automate the processing of large amounts of information. information without using additional short-term memory resources. The concept of schema was described in the seminal works of Frederic Bartlett and Jean Piaget [14], [15].

The central part of CLT describes that during learning, short-term memory is subject to three types of "cognitive load": intrinsic cognitive load, related to the inherent complexity of the topic being studied, extrinsic cognitive load, related to the teaching material and used instructional procedures, and the germane cognitive load, related to the mental effort required to build connections between new information and existing knowledge in long-term memory. It is the latter that is directly related to learning and facilitates the acquisition of new knowledge and skills, through constant and conscious practice that encourages the automation of schemes.

CLT describes a series of 'effects' [16] that allow learning to be optimized according to the characteristics of the topic to be studied and that are guidelines and guides to reduce the student's cognitive load. In the context of teaching programming, the most reported effects with positive empirical evidence are the "worked example" and the "completion problem" effects [17]. To date, 17 effects of the theory have been identified.

2.4. Human Movement Effect.

One of the recent and least explored effects of CLT is the one called "human movement effect" [18] which refers to the way the human brain processes physical gestures and movement for learning. It takes elements from [19] where it is stated that the human brain possesses primary biological knowledge which can be learned, but not taught because it is genetically included, such as face and pattern recognition, and "mirror neuron" reflexes. Oppositely, secondary biological knowledge can be learned and taught, as is the case of knowledge acquired in schools and cultural surroundings.

The human movement effect involves the use of bodily movements, gestures, or physical actions to reduce cognitive load, based on the principle that incorporating physical movements can reduce cognitive load in working memory [20]. Even animations or videos that explicitly include human movements can make use of this effect.

The human movement effect in combination with VR has been studied in other educational settings, such as surgical training [21], [22] with positive learning outcomes. Studies in the context of teaching programming that use movement, but without reference to the cognitive load theory, [23]report positive results in the motivation of the participants.

As such, CLT in particular [16], [24], and the human movement effect, can provide guidelines for the design of immersive prototypes for complex learning domains such as programming, mathematics, statistics, and engineering.

On the other hand, it has also been reported that a good part of the studies associated with VR and education are conducted in laboratories and few include formal tests to improve their usability and learning effectiveness, leaving many of them in prototypes or first versions [25].

Thus, the objective of this study is to design, test and refine a VR-based application to support the teaching of basic programming through problem-solving strategies, which has elements of the effect of human movement and the completion problem effect. It is worth mentioning that the version described in this article is a second iteration, with cognitive load and learning measurement tests pending.

3. Development.

A pilot version (see **Error! Reference source not found.**) was developed as a proof of concept in which multiple choice problems were presented to the user in an immersive environment in a ' Canvas ' type object and where interaction with the problems was carried out by through a point-and-select interface.

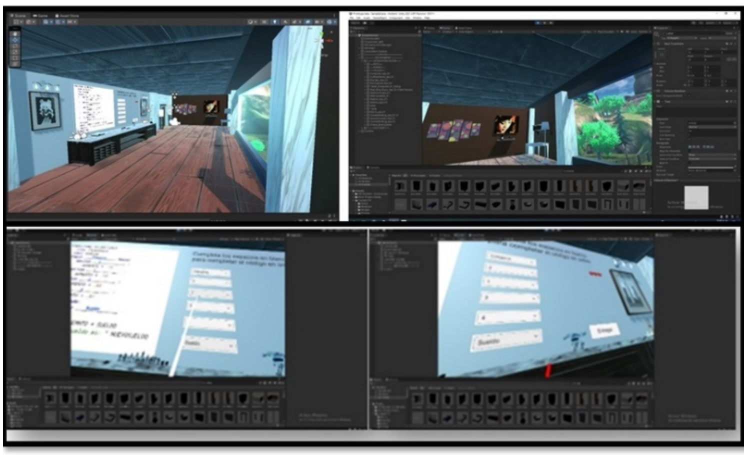

Figure 1. Classroom and problem solving on a blackboard

The VR environment was developed in Unity using an XR plugin (see **Error! Reference source not found.**) that works regardless of the device the user uses and is a package of libraries and scripts. The objects used from this library and in Unity are called 'Sockets', ' Prefabs ' and 'Scripts'. The 'sockets' determine the behavior of objects in the environment and with other objects. The ' Prefabs ' are predefined 3D models to create the environment and the 'Scripts' are the part that can be programmed by the developers.

The camera refers to how it would behave within the environment as an object with respect to the player's point of view. The 'Canvas' and the environment set (Environment, Operation Room) contained the elements that the user can view and interact with.

The 'Canvas' object that displays the basic programming exercises is based on the 'Completion Problem Effect' [16], [26], [27] of Cognitive Load Theory. This effect happens when the instructional designer replaces conventional tasks with 'completion tasks' that provide learners with a partial solution they must complete. The use of the effect has consistently reported positive empirical results in skill transfer and reduction of cognitive load in students. In this case, the partial solution is provided in Spanish pseudocode format that needed to be completed in selected parts to implement the completion effect. For example (see

Figure 2), in the prototype, a fragment of pseudocode shown was:

```
Segun __ hacer
Caso 1:
       Si ( __  y ___ ) >= 8.8) entonces
              Escribir __ , " ", __ , ___, "Aceptado";
Caso __:
       Si sem > 6 y __ ) entonces
              Escribir ____
Caso __ : Caso __:
       Si __ y __ entonces
              Escribir __ , " ", __ , ___, "Rechazado";
De otro modo:
       Escribir "Opción Incorrecta";
```

Figure 2*. Example pseudocode in Spanish pseudocode completion format*

Previous training in the classroom -via traditional lectures- provides the basic concepts and syntax of control structures, and the VR application is expected to provide subsequent training and complementary practice. The examples in completion form were kept in the refined version although with better interactivity and user experience (see section 4.1).

4. Results

The user experience of the first version of the application was evaluated with the UEQ questionnaire [28]. This instrument contains six rating scales: attraction, perspicuity, efficiency, dependability, stimulation and novelty, measured on a 1-7 Likert scale of 26 items. For the analysis of the results, the order of the positive and negative terms of an item are randomized. By dimension, half of the items begin with the positive term and the other half with the negative. For analysis, the results are transformed from the 7-point Likert scale to a range of -3 to +3. +3 represents the most positive value and -3 the most negative.

To evaluate the user experience, 12 students from first year of Intelligent Computing Engineering program participated, with previous experience with video games, but without experience in VR applications, with the following results (see **Table 1** and **Figure 3**). They were asked to solve three basic pseudocode tests in the canvas, that corresponded to three levels of difficulty. At the end of the test, the participants filled the UEQ instrument to measure their perception of the experience.

***Table 1**. User experience measurement results with UEQ*

UEQ Scales (Mean and Variance)		
Attraction	*2,426*	*0.20*
Perspicuity	*1,583*	*0.39*
Efficiency	*1,639*	*0.30*
Dependability	*1,583*	*0.81*
Stimulation	*1,389*	*1.61*
Novelty	*2,056*	*0.75*

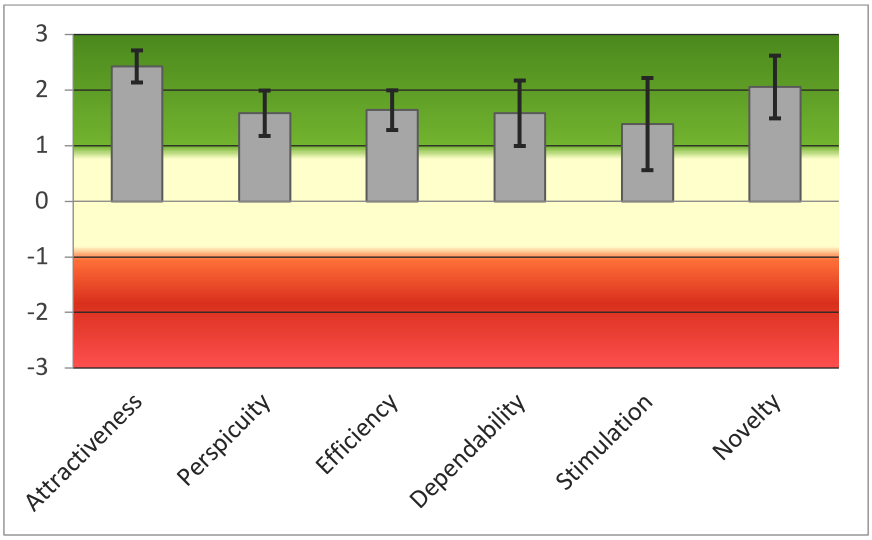

***Figure 3**. Means and variances graph of the UEQ test.*

The lowest rated category by the participants was '*stimulation*', which measures whether using the product is interesting, exciting, and motivating. The ratings were also lower in the categories of *dependability* (the product is easy to understand, clear, simple, and easy to learn) and *efficiency* (tasks can be carried out with the product can be carried out quickly and easily, the user interface looks organized). The scale with the highest rating was *attraction* which describes whether 'The product looks attractive, pleasant and friendly'. The 'novelty' category (which indicates whether the product is innovative, inventive and has a creative design) also had a high rating. It was observed that none of the 6 scales had negative results.

The UEQ instrument allows additional analysis by grouping the 6 categories into 3 general categories called pragmatic quality (Controllability, Efficiency, Reliability) and hedonic quality (Stimulation, Originality). Pragmatic quality describes the quality aspects related to the task and hedonic quality the quality aspects not related to the task (see **Table 2**). To do this, the average of the aspects of pragmatic and hedonic quality is calculated, contrasted with the most highly evaluated category (attraction).

***Table 2**. Pragmatic and Hedonic quality results*

Pragmatic and Hedonic Quality	
Attraction	2.43

Pragmatic Quality	1.60
Hedonic Quality	1.72

In this way, the pragmatic aspect which was the one with the lowest average (although within the positive range) would be improved by incorporating feedback to the participants about the correct or incorrect options within the exercise, as seen in the next section.

4.1. Refined protype with gestures.

A second version was designed also using the Unity platform with the 'XR Plugin', 'Sockets' and 'Scripts' components, but with emphasis on addressing the areas of improvement given by the results of the user experience test and the improvement in the aspects of gestures and human movement to align with the theory (see **Figure 4**).

Thus, gestures were incorporated to manipulate the options to fill in the blank spaces of the exercises to be completed, imitating grabbing and holding cubes that are inserted into spaces on a virtual whiteboard. This functionality was achieved through 'Prefab' models assigned to ‘RightController’ and ‘LeftController’ objects, in turn associated with the Oculus Quest controls to detect movement and gestures. Hand movement when pressing buttons and other interactions is an included as a default animation.

A ‘verify’ option provided feedback on whether the cubes were placed correctly and to address the pragmatic aspect identified in the results.

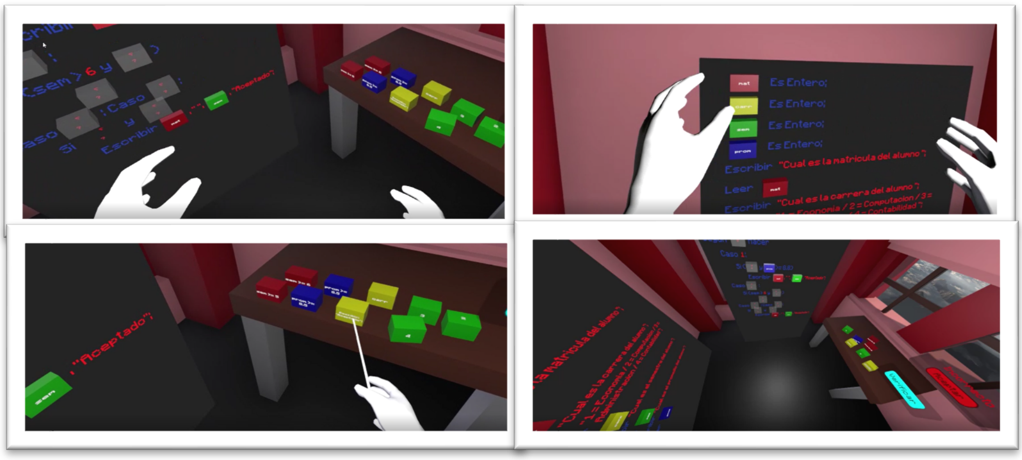

Figure 4*. Improved VR application incorporating gestures*

5. Conclusions

The results of the first prototype indicated that users found it attractive and novel (highest average and lowest variance) but and in contrast, they also rated it as not very ‘stimulating’ and ‘controllable’ (lowest average and greater variance). These results were interpreted as the perception of novelty quickly dissipating once the user became accustomed to the virtual classroom environment.

The low ‘dependability’ may correspond to the use of wireless controls in VR that in this version required the user to ‘point and shoot’ the multiple options of the exercise, which suggests improvements in the interaction aspects with objects of the virtual room. The results also suggested improvements in the ‘stimulating’ category, which was achieved by including greater variability in the objects of the VR environment adding more elements to interact with the exercise.

Finally, it must be stated that design guidelines of CLT to incorporate the human movement effect are of heuristic nature. It can be inferred that using it in combination with other cognitive theories such as multimedia learning theory and dual coding theory [29], [30] could provide for a more specific framework. Currently there is work in this direction by several researchers of the cognitive load community [18].

6. Limitations and future work

The presented learning application is a work in progress. In this context, we report that an extensive learning curve and development life cycle of VR software is a limitation to researchers interested in exploring the possible benefits of VR technology, where knowledge about development environments, imply the use of objects and complex calibrations with the selected proprietary VR devices. These extensive UI development and refinement cycles are typically not ideal for learning testing. In addition, it is necessary to have computers with sufficient computational power and video resources for processing. In addition, the short obsolescence cycle of commercial VR devices is also another limitation in terms of cost.

The current refined version of the software will be used to verify the cognitive load of participants using the NASA-TLX instrument adapted for programming [31] with a pre-post controlled experimental design. Also, the measurement of the effect on learning to solve basic programming problems is expected to be carried out through standardized tests and experimental randomized designs.

References

[1] S. Torrisi, "Software Industry," in *The Palgrave Encyclopedia of Strategic Management*, Palgrave Macmillan UK, 2018, pp. 1589–1592. doi: 10.1057/978-1-137-00772-8_731.

[2] G. Bain and I. Barnes, "Why Is programming so hard to learn?," in *ITICSE 2014 - Proceedings of the 2014 Innovation and Technology in Computer Science Education Conference*, New York, New York, USA: Association for Computing Machinery, 2014, p. 356. doi: 10.1145/2591708.2602675.

[3] T. Jenkins, "On the difficulty of learning to program," in *3rd Annual LTSN-ICS Conference, Loughborough University*, LTSN Centre of information and computer sciences, 2002.

[4] D. McCall and M. Kölling, "A new look at novice programmer errors," *ACM Transactions on Computing Education*, vol. 19, no. 4, pp. 1–30, Jul. 2019, doi: 10.1145/3335814.

[5] J. Bennedsen and M. E. Caspersen, "Failure rates in introductory programming," *AcM SIGcSE Bulletin*, vol. 39, no. 2, pp. 32–36, 2007.

[6] C. Watson and F. W. B. Li, "Failure rates in introductory programming revisited," in *ITICSE 2014 - Proceedings of the 2014 Innovation and Technology in Computer Science Education Conference*, New York, New York, USA: Association for Computing Machinery, 2014, pp. 39–44. doi: 10.1145/2591708.2591749.

[7] L. Freina and M. Ott, "A literature review on immersive virtual reality in education: state of the art and perspectives," in *The International Scientific Conference eLearning and Software for Education*, 2015, pp. 10–1007.

[8] J. Radianti, T. A. Majchrzak, J. Fromm, and I. Wohlgenannt, "A systematic review of immersive virtual reality applications for higher education: Design elements, lessons learned, and research agenda," *Comput Educ*, vol. 147, p. 103778, Apr. 2020, doi: 10.1016/j.compedu.2019.103778.

[9] S. Romano, N. Capece, U. Erra, G. Scanniello, and M. Lanza, “On the use of virtual reality in software visualization: The case of the city metaphor,” *Inf Softw Technol*, vol. 114, pp. 92–106, Oct. 2019, doi: 10.1016/j.infsof.2019.06.007.

[10] J. Vincur, M. Konopka, J. Tvarozek, M. Hoang, and P. Navrat, “Cubely: Virtual reality block-based programming environment,” *Proceedings of the ACM Symposium on Virtual Reality Software and Technology, VRST*, vol. Part F1319, no. 2, 2017, doi: 10.1145/3139131.3141785.

[11] G. Singh, “Using virtual reality for scaffolding computer programming learning,” *Proceedings of the ACM Symposium on Virtual Reality Software and Technology, VRST*, vol. Part F1319, 2017, doi: 10.1145/3139131.3141225.

[12] R. Oberhauser and C. Lecon, “Virtual reality flythrough of program code structures,” in *ACM International Conference Proceeding Series*, New York, New York, USA: Association for Computing Machinery, Mar. 2017, pp. 1–4. doi: 10.1145/3110292.3110303.

[13] W. G. Chase and H. A. Simon, “Perception in chess.,” *Cogn Psychol*, vol. 4, no. 1, pp. 55–81, 1973, doi: 10.1016/0010-0285(73)90004-2.

[14] C. C. Carbon and S. Albrecht, “Bartlett’s schema theory: The unreplicated ‘portrait d’homme’ series from 1932,” *Quarterly Journal of Experimental Psychology*, vol. 65, no. 11, pp. 2258–2270, Nov. 2012, doi: 10.1080/17470218.2012.696121.

[15] S. Carey, D. Zaitchik, and I. Bascandziev, “Theories of development: In dialog with Jean Piaget,” *Developmental Review*, vol. 38, pp. 36–54, Dec. 2015, doi: 10.1016/J.DR.2015.07.003.

[16] J. Sweller, J. J. G. van Merriënboer, and F. Paas, “Cognitive Architecture and Instructional Design: 20 Years Later,” *Educational Psychology Review*, vol. 31, no. 2. Springer New York LLC, pp. 261–292, Jun. 15, 2019. doi: 10.1007/s10648-019-09465-5.

[17] J. H. Berssanette and A. C. de Francisco, “Cognitive Load Theory in the Context of Teaching and Learning Computer Programming: A Systematic Literature Review,” *IEEE Transactions on Education*, vol. 65, no. 3, pp. 440–449, 2021, doi: 10.1109/TE.2021.3127215.

[18] S. Sepp, S. J. Howard, S. Tindall-Ford, S. Agostinho, and F. Paas, “Cognitive Load Theory and Human Movement: Towards an Integrated Model of Working Memory,” *Educational Psychology Review*, vol. 31, no. 2. Springer New York LLC, pp. 293–317, Jun. 15, 2019. doi: 10.1007/s10648-019-09461-9.

[19] D. C. Geary, “Evolutionary educational psychology.,” in *APA educational psychology handbook, Vol 1: Theories, constructs, and critical issues.*, American Psychological Association, 2011, pp. 597–621. doi: 10.1037/13273-020.

[20] F. Paas and J. Sweller, “An Evolutionary Upgrade of Cognitive Load Theory: Using the Human Motor System and Collaboration to Support the Learning of Complex Cognitive Tasks,” *Educ Psychol Rev*, vol. 24, no. 1, pp. 27–45, Mar. 2012, doi: 10.1007/s10648-011-9179-2.

[21] J. G. Frederiksen *et al.*, “Cognitive load and performance in immersive virtual reality versus conventional virtual reality simulation training of laparoscopic surgery: a randomized trial,” *Surg Endosc*, vol. 34, no. 3, pp. 1244–1252, Mar. 2020, doi: 10.1007/s00464-019-06887-8.

[22] S. A. W. Andersen, P. T. Mikkelsen, L. Konge, P. Cayé-Thomasen, and M. S. Sørensen, “The effect of implementing cognitive load theory-based design principles in virtual reality simulation training of surgical skills: a randomized controlled trial,” *Advances in Simulation*, vol. 1, no. 1, pp. 1–8, Jan. 2016, doi: 10.1186/s41077-016-0022-1.

[23] D. Parmar *et al.*, “Programming moves: Design and evaluation of applying embodied interaction in virtual environments to enhance computational thinking in middle school students,” *Proceedings - IEEE Virtual Reality*, vol. 2016-July, pp. 131–140, 2016, doi: 10.1109/VR.2016.7504696.

[24] J. Sweller, P. Ayres, and S. Kalyuga, *Cognitive Load Theory: Explorations in the Learning Sciences, Instructional Systems and Performance Technologies*. London: Springer, 2011. [Online]. Available: http://www.springer.com/series/8640

[25] A. Dey, M. Billinghurst, R. W. Lindeman, and J. E. Swan, “A systematic review of 10 Years of Augmented Reality usability studies: 2005 to 2014,” *Frontiers Robotics AI*, vol. 5, no. APR, 2018, doi: 10.3389/frobt.2018.00037.
[26] M. Bannert, “Managing cognitive load-recent trends in cognitive load theory,” 2002. [Online]. Available: www.elsevier.com/locate/learninstruc
[27] J. J. G Van Merrienboer and H. P. M Krammer, “Instructional strategies and tactics for the design of introductory computer programming courses in high school,” 1987.
[28] B. Laugwitz, T. Held, and M. Schrepp, “Construction and Evaluation of a User Experience Questionnaire,” in *HCI and Usability for Education and Work*, A. Holzinger, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 63–76.
[29] M. Rudolph, “Cognitive Theory of Multimedia Learning,” 2017.
[30] J. L. Alty, “Dual Coding Theory and Computer Education: Some Media Experiments To Examine the Effects of Different Media on Learning.,” in *ED-MEDIA 2002*, Association for the Advancement of Computing in Education (AACE), Feb. 2002. [Online]. Available: http://www.eric.ed.gov/ERICWebPortal/custom/portlets/recordDetails/detailmini.jsp?_nfpb=true&_&ERICExtSearch_SearchValue_0=ED476964&ERICExtSearch_SearchType_0=no&accno=ED476964
[31] B. B. Morrison, B. Dorn, and M. Guzdial, “Measuring cognitive load in introductory CS: Adaptation of an instrument,” in *ICER 2014 - Proceedings of the 10th Annual International Conference on International Computing Education Research*, New York, New York, USA: Association for Computing Machinery, 2014, pp. 131–138. doi: 10.1145/2632320.2632348.