Seeking the Treasures of Theoretical Computer Science Education: Towards Educational Virtual Reality for the Visualization of Finite State Machines

Andreas Dengel
Mathematics and Informatics Education Unit
University of Passau
Passau, Germany
andreas.dengel@uni-passau.de

***Abstract*—As metaphorical representations showed beneficial effects on several parts of computer science, this paper focuses on the question: How can metaphorical representations in VR enhance the understanding of theoretical computer science concepts? A concept for an immersive educational virtual environment for learning the theoretical computer science concept of finite state machines is presented. A first draft for the virtual reality aims to immerse students in a world of islands (states) with treasure islands (final states). Each symbol from the input alphabet is illustrated through another boat on every island. Students navigate through the islands by looking at one of the boats on the current island using a head-mounted-display. They then take the shipping route of the selected boat (the state-transition function) by pressing a button on a controller. The goal of the game can either be to reach Treasure Island and open the treasure chest or to complete the map.**

Keywords—immersive learning, finite state machine, computer science unplugged, computational thinking, theoretical computer science

I. Introduction

Recent developments in the research field of Virtual and Augmented Reality have led to an increasing interest in the practical use of immersive virtual environments in education [1-3]. Virtual Reality (VR) is a digital environment which is completely synthetic with the purpose to immerse the participant-observer with enabled interaction [4]. Especially the MINT subjects (mathematics, information technology, natural sciences and technology) can benefit from the possibilities of accurate, schematized, substantiated and metaphorical representations in VRs [5]. The power of analogies and metaphors is well documented in the cognitive sciences [6].

CS education already uses metaphorical representations of abstract concepts, for example in programming [7][8]. But metaphorical representations may be useful in other parts of CS education as well: When we take a look at the theory of computation, the covered topics are not only hard for students to understand, but lead to considerable dropout rates and poor performance within exams [9]–[11].

Due to the abstract nature of this field, metaphorical representations may provide good assistance for students. With regard to this matter, this paper focuses on the question of how metaphorical representation in VR can enhance the understanding of theoretical CS using the example of finite state machines. To address this question, we first give a brief overlook of regular languages and their importance for computer science and in everyday life. Further, we present existing approaches to visualize finite state machines. The idea of a hands-on-experience to learn about finite state machines is then pursued into an immersive educational virtual environment. Its methodical and didactical considerations are explained and an example of a level design is given. Afterwards, possibilities to evaluate the effectiveness of the educational virtual environment in terms of the students' learning outcomes as well as the challenges of metaphorical representations in VR are discussed.

II. Finite State Machines as a Fundamental Idea of Computer Science

In fact, the finite state machine itself is actually a metaphorical representation of the entirety of regular language of the type-3 grammar in the Chomsky hierarchy [12].

To describe a given regular language L with a finite state machine M, we use $M = (Q, \Sigma, \delta, s0, F)$, with

$Q = \{q0, q1, q2, \ldots, qn\}$ as the finite set of states,

Σ as the alphabet containing the symbols of the language,

δ as the set of the language's transition functions,

$\delta = Q \times \Sigma \rightarrow Q$,

$s0 \in Q$ as the start state, and

$F \subseteq Q$ as the amount of accept states.

The automaton processes a string w, starting in its initial state q0. M reads the symbols from the string one by one from left to right. After each symbol, M moves from one state to another according to the corresponding transition function. After reading the last symbol, M produces the output (accept or reject), depending on whether the machine is in an accept state or not. The word w is accepted by M if the current state after reading the last symbol is an accept state, otherwise, it is rejected [13].

Finite state machines find usage in many cases of everyday life like automatic doors and various electromechanical devic-

978-1-5386-6522-0/18/$31.00 ©2018 IEEE 4-7 December 2018, Wollongong, NSW, Australia
2018 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)

es. Furthermore, the finite state machine can be used as an introduction to computational models in order to understand the functionality of a real computer.

Language, in general, has been identified as a fundamental idea of computer science based on four criteria [14]. Based on the work by Bruner [15], Schwill states that a fundamental idea "is a schema for thinking, acting, describing or explaining" [14] that:

- has to be applicable or observable in many areas of a domain;
- may be taught at every intellectual level;
- is observable in the domain's historical development; and
- is related to the everyday life.

Finite state machines belong to the section syntax and the subsection accepting in the category language from Schwills taxonomy of fundamental ideas of computer science [14].

The concept of regular languages which are represented by finite state machines finds many uses in computer science [16] as well as in everyday life [17] and language [18]. Furthermore, Wing de-scribes finite state machines as a class of computational abstraction which helps developing Computational Thinking as a thought process to formulate a problem and its solution using multiple ways of abstraction and decomposition so that a computer could effectively carry out [19].

III. Visualizations of Finite State Machines

As a part of the curriculum of CS undergraduate studies and of some school curricula, several kinds of software for the transformation of regular expressions to drawn state machines are already established [20]–[22]. To increase visualization and interaction in automata theory courses, Hung and Rodger present the learning and teaching tools JFLAP (Java Formal Languages and Automata Package) and Pâté. Both JFLAP and Pâté visualize the transformation process between different representations of languages. JFLAP, for example, contains the preparation of regular expressions and their conversion from and to nondeterministic finite automata. This helps the learner to explore and understand the connection between different representations of the same language [23].

This visualization process can be helpful during the learning process of finite state machines and regular languages but requires an adequate previous knowledge in the topic, so it may not be adequate as an introduction. Because of the importance of regular languages for understanding everyday life concepts and enhancing Computational Thinking skills, an early adaption of the concept of finite state machines which fits children's cognitive abilities and interests is needed.

Research has shown that using games for teaching basic concepts of theoretical computer science like finite state machines and Turing Machines can be a useful approach. Korte et al. developed a didactical sequence to teach these concepts using a purpose-built game engine. In this learning by game-building approach, under-graduate students were asked to copy an existing game or to write their own game using the game engine. In case of the finite state machine topic, students took existing finite state machines and set them into a narrative context. In a student's game, Becoming Leader of the Conservative Party, the player has to complete several steps (which represent the transition functions) in order to become the new leader of the conservatives. The requirements for designing a game corresponding to the finite state machine topic where 1) It is always possible to reach the goal, 2) Your game contains at least 1 'long' cycle (i.e. at least 5 states before a repeat), 3) Within at least one game-level it is always possible to reach the initial state and 4) Your game has at least one interleave. By giving the students the possibility to create their own game, they could choose a context which was relevant to them personally. Doing so, the students did not only understand the concepts of finite state machines but created a connection to their everyday life and thinking all by their own [24].

With regard to everyday life problems and thinking patterns, Computer Science Unplugged introduced a fun activity based on concepts of finite state machines which "is based around a fictitious pirate story which leads to the unlikely topic of reasoning about patterns in sequences of characters" [25]. The goal of the game Treasure Hunt is to find a way to the Treasure Island (the final state) on a map which consists of several islands (the states of the finite state machine which are represented by other students) and a fixed set of routes (the state-transition functions). The two departing ships on every island, A and B, are the two possible inputs from the alphabet and lead to another (or the same) island. In the activity, 7 children are chosen to represent the islands, holding cards to identify their island. The cards have secret instructions (namely the shipping routes) on their back. On every island, a student gets to choose between the two boats, not knowing their destination. After making a decision, the student representing the island tells the player his or her next destination. Every student gets an empty map where only the islands are drawn without the routes of the ships. During the game, the students complete their map with the discovered shipping routes what results in an abstract representation of the underlying finite state machine [17]. The treasure hunt game of CS Unplugged is widely accepted and has been adopted in analogue [26] and digital [27][28] ways.

The concept of the developed VR application tries to refine the ideas of CS Unplugged and puts the user into an immersive educational virtual environment for an introduction into Theory of Computation. When using VR as a medium to visualize an alternate reality suited for learning CS concepts, it seems appropriate to speak of a 'plugged' unplugged activity for CS education.

IV. Design Of the Immersive Educational Virtual Reality "Treasure Hunt VR"

The VR program has been developed and implemented with Unity. It converts any given regular language in a specific text format with a maximum of six states and four different input characters into an island structure similar to the CS Unplugged activity. When starting the game, the learner enters a world of islands which represent the states of the finite state machine by the use of a head-mounted-display (HMD). The HMD as a head-tracking device is used to enhance the feeling of presence for the user. In the VR, one or more islands are

considered as treasure island(s), which represent the final state(s) and the acceptance of a given input (the route). The goal for every student can either be to reach an island with a treasure chest (the easy part) or to complete the map by finding all routes between the islands (the hard part), starting from Palm Island (the initial state). The symbols from the input alphabet are the different ships which can be taken from every island. In order to enhance the gaming experience, the islands are unique and distinguishable due to their appearance (palms, a shipwreck, seagulls, etc.). For orientation purposes, a sign with the island's (state's) name is placed on every island. Each input character is illustrated by another boat on the current island. By looking at one of the boats, the students can navigate through the islands with the use of a HMD and activate the state-transition function by pressing a button on a controller. When looking down, the user sees him- or herself holding a map of the previous discovered islands and routes. Every time the student discovers a new route or island, the discovery is added to the map so that a complete map shows a complete finite state machine.

When the student reaches one of the treasure islands, he or she can open the treasure chest and end the game or get to the next level. A game can consist of several levels which may increase in difficulty (the complexity of the underlying finite state machine), size and transitions. As the difficulty rises, an island without return (an error state where all routes lead back to the same island) can be added as well. While the process of finding a working route to the Treasure Island can be an easy try-and-error process on simple finite state machines, larger and more complex finite state machines require an understanding on how state-transition functions work, more precisely on the structure of the language which is accepted by the automaton. Even though the map can be helpful on finding the correct route to Treasure Island, a full understanding of the underlying regular language is needed in order to reach the next level by opening the treasure chest. Another game objective could be to complete the map with all possible routes. The game objective can be set by a teacher who also selects the finite state machines underlying the island structure.

One big challenge in the design of the VR was to enhance presence as the feeling of being there [29][30] and simultaneously not to distract the learner from the underlying learning content. We decided to let the player discover the island by himself by giving him or her the possibility to look around and to spot the key element of the island (for example a group of palms on Palm Island). Besides looking around and discovering the current position, the player has no possibility to move on the island, in order not to distract the learner from completing the task of finding a treasure island or the task of completing the map. The player is not able to see any other island from a specific island. This tries to eliminate misleading conclusions about shipping routes.

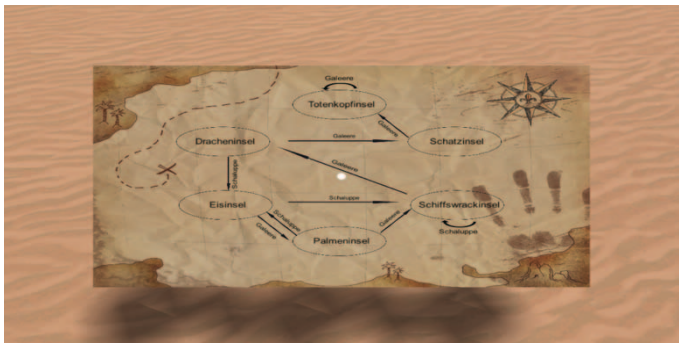

Fig. 1. The ingame map with islands and ship routes.

Within the first concept, we did not provide a map to help the learner understanding the automatons structure. The player would have had the possibility to exit the game and take notes by him- or herself like in the CS Unplugged activity. This may had resulted in a more constructivist learning approach because the learner would need to find an appropriate representation for the islands structure by him- or herself but would have destroyed any feeling of presence when exiting the VR every minute to take notes in reality. In order to keep presence as a reinforcing factor for learning outcomes [31][32], we decided in favor for the use of an in-game map (Fig. 1) where all discoveries (islands and shipping routes) are noted down.

According to this concept, a typical gameplay of a simple level would look like this: The player lands on Palm Island and has two or more possibilities to take a ship. The player decides to take a specific ship and lands on an island which is populated by seagulls. By that, the Seagull Island is drawn on his or her map next to Palm Island where he or she started from. The two islands are then connected with an arrow from Palm Island to Seagull Island with the labeling of the ship. The player then takes the headset back on and returns to the VR where the next decision is made. This process is repeated until a treasure island is reached where the player opens the treasure chest.

The current version provides six different islands and four different boat positions, together with their related island drawings and possible routes on the map (36 in total). Thus, the converter is currently limited to a maximum of six states and to a maximum of four input symbols. Each of the islands has a hidden treasure chest which becomes visible if the island is a final state of the finite state machine.

V. Example of the Level Design

To connect the concept of finite state machines to known conditions from the everyday life, we use the example of a coffee vending machine which takes 1.50€ for a coffee. The automaton only accepts 1€ and 50ct coins. If the total of the inserted coins exceeds 1.50€, the last inserted coin is returned (so there is no change and the amount has to be exact). There is also a Cancel button which resets the machine to the initial state and returns all the inserted coins. When the 1.50€ are inserted correctly, the automaton does not longer react to users inputs and the coffee is brewed. This simple concept of a coffee vending machine can be translated into a language over the alphabet $\Sigma = \{50ct, 1€, Cancel\}$, which contains all accepted

inputs on the vending machine. In this case, the word w is the total of all inputs from the user. The language which contains all accepted inputs that result in the coffee vending machine brewing a cup of coffee is L_{coffee} = {(50ct, 1€), (1€, 50ct), (50ct, 50ct, 50ct), (Cancel, 50ct, 1€), (Cancel, 1€, 50ct), (1€, 1€, 50ct), …}.

To describe the corresponding finite state machine A_{coffee}, we use A_{coffee} = (K, Σ, δ, s0, F), with:

K = {q0, q1, q2, q3},

Σ = {50ct, 1€, Cancel},

δ = { δ (q0, 50ct) → q1,

δ (q0, 1€) → q2,

δ (q0, Cancel) → q0,

δ (q1, 50ct) → q2,

δ (q1, 1€) → q3,

δ (q1, Cancel) → q0,

δ (q2, 50ct) → q3,

δ (q2, 1€) → q2,

δ (q2, Cancel) → q0,

δ (q3, 50ct) → q3,

δ (q3, 1€) → q3,

δ (q3, Cancel) → q3}

s0 = q0, and

F = {q3}.

Figure 2 shows the visualization of this finite state machine using a classical drawing method as it is taught in schools and universities. In the immersive educational virtual reality "Treasure Hunt VR", this representation is used as a basis for the design of the level map, which builds up during the learner's discoveries in the island world, resulting in Figure 3. However, the level setting is only the underlying structure and the final result of completing the ingame map and not shown in the visual structure on the HMD as every 3D-representation of an island is a distinct place without any game related environment. In order to avoid the mental image of a geographic structure, no other island is visible from a specific island as each island is implemented as a separate scene.

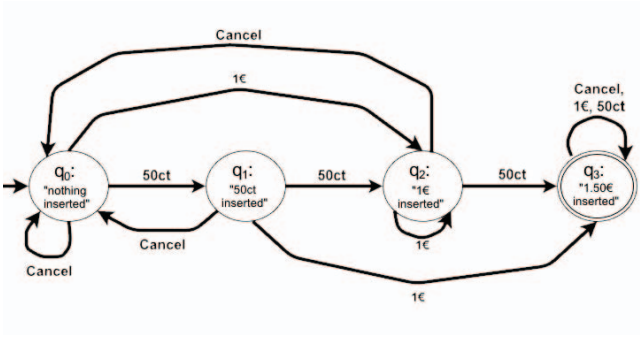

Fig. 2. Representation of a finite state machine $\mathbf{A_{coffee}}$.

The states q0, q1, and q2 are represented by Palm Island (initial state z0), Seagull Island and Shipwreck Island. The final state q3 is represented by Treasure Island, where the treasure chest can be opened. Figure 3 shows a possible map with typical island names. Of course, it would be possible to name the islands after the initial regular language (e.g. q0, q1, …) or to give them a semantical meaning (e.g. "50ctInserted", "1€inserted", …).

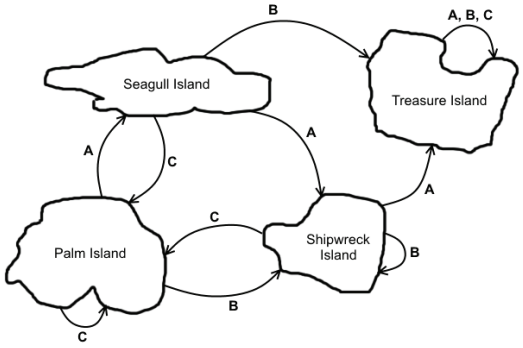

Fig. 3. Island map in "Treasure Hunt VR", based **on** $\mathbf{A_{coffee}}$.

A possible 3D-visualization of Palm Island in the educational virtual reality "Treasure Hunt VR" is shown in Figure 4. Through exploring the virtual reality on their own, learners get a subjective learning experience, similar to an anchored instruction [33].

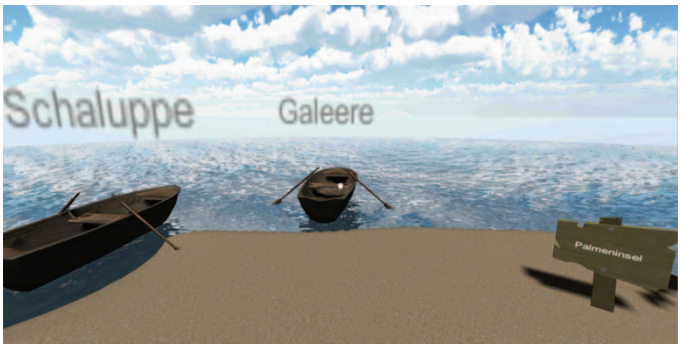

Fig. 4. 3D visualization of the initial state q0 as "Palm Island" with two possible boats.

VI. Discussion And Conclusion

This paper described the immersive VR visualization of a metaphor for finite state machines. Pursuing existing approaches of the visualization of regular languages, the CS Unplugged approach Treasure Hunt has been identified as an appropriate and fun game-based learning activity. The metaphor used in the Treasure Hunt game transfers states to islands, final states to treasure islands, symbols of the input alphabet to ships and transfer functions between the states to ship routes. This metaphor was used in order to design an immersive educational virtual environment for a first introduction into the topic finite state machines. Challenges of the current implementation and an example of the level design were described.

Using metaphorical representations of abstract concepts is a double-edged sword in terms of computer science. On one hand, Carroll and Thomas note that learning through metaphors can be questionable for students without data-processing training; they may never get past an initial metaphorical representation of computing systems [6]. This can be a challenge when implementing the educational VR in the classroom: Getting past the representations in the VR game of islands as states and shipping routes as transition functions is crucial for the transfer of the concept in terms as to enhance a broad transfer in Computational Thinking. Understanding the concept of abstraction can be a main skill taught through the whole process of working with theoretical CS.

On the other hand, Korte et al state in their evaluation of the game-building project for teaching finite state machines that weaker students may gain a larger profit from game-based learning concepts and work more enthusiastically. Stronger students, however, may be less enthusiastic about game-based approaches, what may indicate a preference for assignments in a more traditional way [24]. Taking this into account, the visualization tool will surely be appropriate for motivating children to dive into the topic in general, but should not be used as the only representation. Therefore, a combination of traditional teaching for basic concepts of theoretical computer science and the use of new media like VR in the classroom may be the right way to address all learners' needs. This is particularly the case when a continued work beyond a pure introduction to the topic of finite state machines is desired, for example in higher K-12 grades.

It is obvious that the topic of finite state machines covers only a small area of theoretical computer science education and that the educational VR does not assume or replace the role of the teacher in the learning process. But still, giving children an understanding of the concept of finite state machines can contribute to the development of their Computational Thinking skills However, the conception of an educational VR for the visualization of finite state machines may provide a good opportunity for an introduction of the complex topic of regular languages. Especially for young children, where intrinsic motivation can be considered as the key to learning efficiency in game-based learning environments [34], the use of an immersive medium in terms of gamification may promote high motivation.

With an appropriate didactical sequence, CS teachers may benefit from a mixed variety of methods and media. As a motivation (depending on the age of the target group), the CS Unplugged activity or the "Treasure Hunt VR" application could be used. For further training and a game-based learning setting, the VR may be useful as well, using more difficult levels, explaining the meaning of treasure islands or of islands without a returning possibility in combination with traditional lessons teaching the components of the finite state machine. Together with a teacher, the students can design their own island structures as well, formulate them as text files and convert them into an educational virtual environment using the developed software. As students' skills get better and the topic covers different concepts of representing a regular language such as regular expressions or non-deterministic automata, teachers and students may benefit from the use of several visualization conversion tools such as JFLAP or Pâté. In addition to that, designing own games based on finite state machines or visualizing real life processes using the underlying concept can be a motivating project for students, applying their skills and knowledge to a complex problem.

In a next step, it would be possible to examine on how effective this gaming setting actually is using a pre- and a posttest and what factors determine the learning results. Especially the impact of the subjective feeling of presence, which was one of the focuses when working on the concept of the VR, on the motivation and the learning outcomes of a specific target group would be interesting to examine. Furthermore, the effect of immersion (for example using different HMD devices) on both learning outcome and motivation of the children would be interesting as well. With regard to the results of Korte et al., it would be interesting to examine the students' motivation on working with the VR as a learning tool, especially with the connection to their overall performance in school generally and particularly in CS.

Additionally, with the implementation of the software which transfers the 5-tupel of a given finite state machine into a set of islands, ships, and shipping routes, an appropriate didactical sequence of island levels with underlying finite state machines has to be considered as well. Therefore, a target group has to be selected which could be, thanks to the didactical reduction of the topic in the game, every K-12 age group. The cognitive abilities of the target group have to be taken into account as well as the objectives in terms of

developing Computational Thinking skills and pursuing contents of the particular CS curriculum.

With regard to the immersive medium VR, there are still many research desiderata in terms of educational use of the technology. As CS uses many abstract, non-tangible concepts and ideas, new ways of explaining and learning are exactly what is needed to reinforce K-12 CS education. Differentiation between existing knowledge, skills and interests may be the key for getting more children interested in CS in the future. Next, to the visualization of finite state machines, further possibilities to help visualizing and understanding Computational Thinking concepts in immersive media such as VR are yet to be discovered.

REFERENCES

[1] J. Hedberg and S. Alexander, "Virtual reality in education: Defining researchable issues," *Educ. Media Int.*, vol. 31, no. 4, pp. 214–220, Jan. 1994.

[2] J. Psotka, "Immersive training systems: Virtual reality and education and training," *Instructional Sci.*, vol. 23, no. 5, pp. 405–431, 1995.

[3] R. Schroeder, "Learning from virtual reality applications in education," *Virtual Reality*, vol. 1, no. 1, pp. 33–39, 1995.

[4] P. Milgram, H. Takemura, A. Utsumi, and F. Kishino, "Augmented reality: A class of displays on the reality-virtuality continuum," *SPIE,* vol. 2351, Telemanipulator and Telepresence Technologies, pp. 282–292, Jan. 1994.

[5] S. Schwan and J. Buder, *Virtuelle Realität und E-Learning*, Tübingen, Germany, 2006.

[6] J. M. Carroll, and J. C. Thomas, "Metaphor and the cognitive representation of computing systems," *IEEE Trans. Syst. Man Cybern.,* vol. 12, no. 2, pp. 107–116, 1982.

[7] D. Schweitzer and W. Brown, "Interactive visualization for the active learning classroom," in *Proc. 38th SIGCSE Tech. Symp. Comput. Sci. Educ.,* Covington, KY, 2007, pp. 208–212.

[8] L. J. Waguespack, "Visual metaphors for teaching programming concepts," *Common LISP object system specification. Assoc. for Computing Machinery / Special Interest Group on Programming Languages: Sigplan notices 23.: Special issue*. ACM Press, New York, N.Y., pp. 141–145, Jul.1998.

[9] F. Kiehn, C. Frede, and M. Knobelsdorf, „Was macht theoretische Informatik so schwierig? Ergebnisse einer qualitativen Einzelfallstudie," *INFORMATIK 2017*, M. Eibl and M. Gaedke, Eds., Gesellschaft für Informatik, pp. 267–278, 2017.

[10] M. Knobelsdorf, and C. Frede, "Analyzing student practices in theory of computation in light of distributed cognition theory," in *ICER'16. Proc. 2016 ACM Int. Conf. Computing Educ. Research*, Melbourne, Australia, 2016, pp. 73–81.

[11] M. Knobelsdorf, C. Frede, S. Böhne, and C. Kreitz, "Theorem provers as a learning tool in theory of computation," in *Proc. 2017 ACM Int. Conf. Computing Educ. Research - ICER '17*. Tacoma, WA, USA, 2017, pp. 83–92.

[12] N. Chomsky, "On certain formal properties of grammars," *Inform. & Control* , vol. 2, no. 2, pp.137–167, June 1959.

[13] M. Sipser, *Introduction to the Theory of Computation*. Thomson Course Technology, Boston, 2009.

[14] A. Schwill, "Computer science education based on fundamental ideas", *Information Technology - Supporting Change Through Teacher Educ.*, D. Passey and B. Samways, Eds., Chapman Hall, 1997, pp. 285–291.

[15] J. S. Bruner, *The Process of Education*. Harvard Univ. Press, Cambridge, MA, 1960.

[16] M. Shaw, Ed. *The Carnegie-Mellon Curriculum for Undergraduate Comput. Sci.* Springer, New York, NY, 1985.

[17] Computer Science Unplugged (2018, Oct. 21). *Treasure Hunt - Finite-State Automata* [Online], Availabe: https://classic.csunplugged.org/wp-content/uploads/2014/12/unplugged-11-finite_state_automata.pdf

[18] M. Mohri, "On some applications of finite-state automata theory to natural language processing," *Nat. Lang. Eng.*, vol. 2, no. 1, pp. 61–80, Mar. 1996.

[19] J. M. Wing, "Computational thinking," *Commun. ACM*, vol. 49, no. 3, pp. 33–35, Mar., 2006.

[20] S. Diehl, *Software Visualization: Visualizing the Structure, Behaviour, and Evolution of Software*. Springer, Berlin, 2007.

[21] M. Hulden, "Foma. A finite-state compiler and library," in *Proc. 12th Conf. European Chapter of the Assoc. for Computational Linguistics*, Athens, Greece, 2009, pp. 29–32.

[22] L. van Zijl, J.-P. Harper, and F. Olivier, "The MERLin environment applied to ⋆-NFAs," in *Implementation and Application of Automata. 5th Int. Conf., CIAA 2000 London, Ontario, Canada, July 24-25, 2000, Revised Papers*, S. Yu and A. Păun, Eds. Lecture Notes in Computer Science 2088. Springer, Berlin, 2001, pp. 318–326.

[23] T. Hung and S. H. Rodger, "Increasing visualization and interaction in the automata theory course," in *Proc. 31st SIGCSE Tech. Symp. Comput. Sci. Educ. (SIGCSE '00),* Austin, TX, 2000, pp. 6–10.

[24] L. Korte, S. Anderson, H. Pain, and J. Good, "Learning by game-building," in *Proc. 12th Annu. SIGCSE Conf. Innovation and Technology in Comput. Sci. Educ.*, Dundee, Scotland, 2007, pp. 53–57.

[25] Computer Science Unplugged (2018, Oct. 21). *Finite State Automata* [Online] Available: https://classic.csunplugged.org/finite-state-automata/

[26] C. Heeren, T. Magliery, and L. Pitt, *Lesson 5: Finite State Machines*. 1998.

[27] K. Anderson, J. J. López, C. Ho, and J. Martens, *Komputer Kingdom. An Online Gaming Environment to Build Foundational Comput. Sci. Skills*, 2014.

[28] swisseduc.ch (2018, 2018, Oct. 21) *Learn Programming with Kara. Kara - Programming with State Machines* [Online] Available: https://www.swisseduc.ch/compscience/karatojava/kara/

[29] C. Heeter, "Being there. The subjective experience of presence" *Presence*, vol. 1, no. 2, pp. 262–271, Jan. 1992.

[30] M. J. Singer, and B. G. Witmer, *Presence Measures for Virtual Environments: Background and Development.* United States Army Research Institute for the Behavioral and Social Sciences, 1996.

[31] E. A.-L. Lee, K. W. Wong, and C. C. Fung, "How does desktop virtual reality enhance learning outcomes? A structural equation modeling approach," *Comput. & Educ.*, vol. 55, no. 4, pp. 1424–1442, Dec. 2010.

[32] T. A. Mikropoulos, "Presence. A unique characteristic in educational virtual environments," *Virtual Reality,* vol. 10, no. 3–4, pp. 197–206, Sep. 2006.

[33] J. D. Bransford, R. D. Sherwood, T. S. Hasselbring, C. K. Kinzer, and S. M. Williams, "Anchored instruction: Why we need it and how technology can help," in *Cognition, Educ., and Multimedia: Exploring Ideas in High Technology*, D. Nix, Ed. Erlbaum, Hillsdale, NJ, 1990, pp. 115–141.

[34] M. P. J. Habgood and S. E. Ainsworth, "Motivating children to learn effectively: Exploring the value of intrinsic integration in educational games," *J. Learning Sci.*, vol. 20, no. 2, pp. 169–206, Apr. 2011.