Computational Thinking in the Danish High School: Learning Coding, Modeling, and Content Knowledge with NetLogo

Line Have Musaeus[†]
Center for Computational Thinking and Design
Aarhus University
Denmark
lh@cc.au.dk

Peter Musaeus
Centre for Health Sciences Education
Aarhus University
Denmark
peter@cesu.au.dk

ABSTRACT

Computational thinking (CT) is emerging as an important theme in computer science and high school education. However, research is needed to inform high-school teachers how to foster students' development of CT in computer science and other subjects. Evidence suggests that agent-based modeling is a valuable way for students to learn CT in different subjects. This paper reports a teaching experiment where researchers, developers, and high school teachers collaborated to develop six NetLogo models. The models were used in nine Danish High Schools in the following four subjects: Biotechnology, chemistry, biology, and social science. Teachers and students had no or very limited experience with programming. Students build CT and content knowledge by using, modifying, and creating code in the models. This paper provides details for others to adopt the models and the underlying CMC framework, which integrates: Coding, Modeling, and Content. The paper evaluates the results from an open-ended questionnaire with all participating students (n=210) and semi-structured interviews with all teachers (n=15). Thematic analysis was applied to categorize the qualitative data. Results showed that students were able to use, modify, and create code in NetLogo that enabled them to develop CT and content knowledge. The CMC framework represents a fruitful way for teachers to design and teach and for students to tinker with learning CT.

CCS CONCEPTS

• Social and professional topics → Computing education; K-12 education; Computational thinking;

KEYWORDS

Computational thinking; K-12 education; Educational intervention; Computer models, Teacher professional development.

ACM Reference format:

Line Have Musaeus and Peter Musaeus. 2019.Computational Thinking in the Danish High School: Learning Coding, Modeling and Content Knowledge with NetLogo. In SIGCSE '19: SIGCSE '19: The 50th ACM Technical Symposium on Computer Science Education, *Feb. 27–March 2, 2019, Minneapolis, MN, USA*. ACM, NY, NY, USA, 6 pages. https://doi.org/10.1145/3287324.3287452

[†] Corresponding author

1 INTRODUCTION

Seymour Papert [1] coined the notion of Computational Thinking (CT) in 1980 to describe the inquiry necessary to solve problems by means of computational ideas or computers. He applied the notion to make formal geometry more accessible to children by means of computational models. In 2006, Wing [2] referred to CT as the competence to think like a computer-scientist. Wing argued that CT represented a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use [2 p. p. 33]. This definition referred back to Papert's [1] original insight that CT was an inquiry process in any subject, not only say computer science. Papert emphasized that the essence of the computer was its universality that gave power to the computational thinker with the computer to simulate real-world phenomena [1 p.VIII].

CT can be a way of teaching high school students modelling and coding [22] but to what extent does such activities lead high-school students to build content knowledge in other subjects than computer science such as biology, chemistry, English etc.? According to Papert [1] and Wing [2], CT stimulates students' understanding or construction of content knowledge. Although CT is being introduced into the subject of high school computer science [4, 11, 12, 13] and other subjects [10], it is still unclear how CT should be taught in high-school, whether learning CT helps learning another subject and finally whether CT should be taught in most or any subject.

CT can be said to have two meanings in the high school curricula. First, CT can be a learning goal in computer science (or Informatics as high school computer science is mostly called in Europe). In this case, CT is a means towards learning skills or knowledge relating to computer science such as learning programming. Second, CT can be a means towards learning content knowledge (in math, social science, biology etc.) by using CT to close the students' gaps in understanding between representing a phenomenon and the phenomenon itself. It follows from this second approach that CT is a competence that should be taught in high-school in order for the students to learn to grasp content in a subject (not just computer science) by means of computational ideas or the computer.

Researchers have demonstrated that agent-based modeling can be an effective way of teaching CT in relation to content knowledge [6, 7, 8]. We therefore choose to use NetLogo, which is an agent-based modeling tool that has been used in both biology teaching [16, 18] social sciences [17] chemistry [10] and mathematics [22].

In this paper, we report a design-based research study focusing on high-school students (n=210) and teachers (n=15) using NetLogo to develop CT and content knowledge. We used a sequential transformative design with quantitative data (from students) and qualitative data (from students and teachers). The research was conducted in four Danish high school subjects: Biotechnology, chemistry, biology and social/political science in nine different schools. The research aim was to develop teachers and students' integration of Coding, Modeling and Content construction (CMC see figure 1):

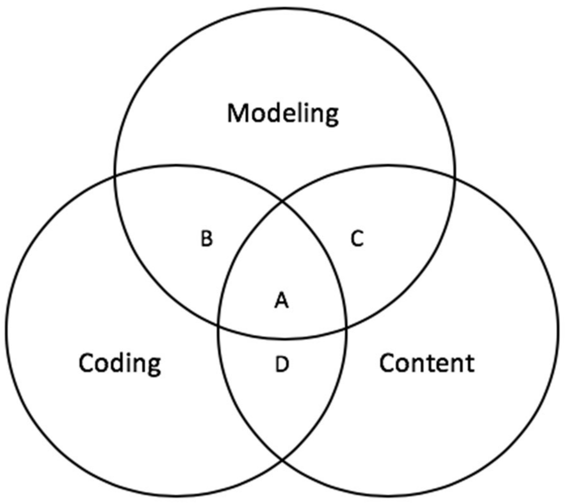

Figure 1: The CMC approach. A: Taking both modeling, coding as well as content into consideration when designing the models. B: Altering sections of code to modify the model. C: Comparing and modifying the model to fit the content matter. D: Concepts from content matter are illustrated in the code to facilitate students work with coding.

1.1 Research question

The objective of this teaching development project was to provide models and code with interface elements that students could tinker with and hereby develop CT. In particular we wanted to evaluate whether the CMC (coding, modelling, and content) approach can aid researchers, developers, and teachers to produce learning environments for students' acquisition of CT and content knowledge?

2 METHOD

Participants: Fifteen high school teachers (nine males and six females) in fifteen high school classes (in nine different high schools) employed the models in their classes. Altogether, 210 students participated.

NetLogo: NetLogo is an agent-based modeling environment. A NetLogo model includes the Interface tab, the Info tab, and the Code tab. The model is displayed in the Interface and students can manipulate its accompanying plots and parameters. The Info tab describes both the model as well as its structure, properties, and rules (i.e. rules assigned to the entities in the model). It also gives suggestions as to what to explore and extend in the model. The Code tab includes the code for the model with comments. Any changes made in the Code tab will result in changes to the model and the interface [15].

Step 1: Developing NetLogo models. Three researchers and two developers constructed six NetLogo models with five high school teachers as collaborators.

The aim was that students could tinker with the models and examine how patterns at a macro level, e.g. of ion transport across a cell membrane, arise from simple rules and interactions at the micro level, such as atoms and molecules, in a system (see figure 2).

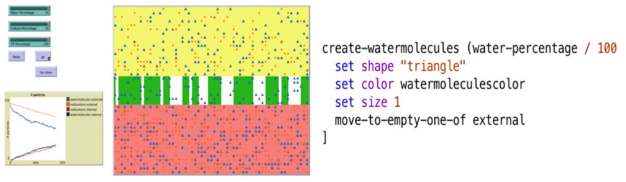
```
create-watermolecules (water-percentage / 100
  set shape "triangle"
  set color watermoleculescolor
  set size 1
  move-to-empty-one-of external
]
```

Figure 2: NetLogo model of cell membrane, showing both the 'Interface' and 'Code'.

On running the model, students could explore how these simple rules resulted in complex and unpredictable patterns.

Step 2: Designing learning activities. In designing the models, we used procedures and functions in the code to ensure that students worked at the appropriate level of abstraction (see figure 2).
Abstraction refers here to understanding a representation, identifying, and analyzing the elements that makes up the model of the represented phenomenon. All students were given the same one-minute video introduction to NetLogo.

The CMC approach (figure 1) served both as a framework in the design phase/step of the models and the pedagogy around using these models in teaching. Thus researchers, developers, and high school teachers used the approach throughout the study, to simultaneously design the computational models and plan the learning activities. The team designed the models primarily to serve as initial starting points or idea generators for students to tinker with, use and modify. Students could interact with models built as half-baked microworlds. A microworld is defined as a less developed representation of a phenomenon [14, 15].

Subsequently, we used structured learning activities to take the students through tasks such as creating a new set of agents in NetLogo, e.g. potassium ions, with specific structures and rules, in the model of a cell membrane that already contained sodium ions (figure 2). The students worked in pairs to support peer-to-peer collaboration, using an online questionnaire describing the tasks and related questions they needed to answer. The learning activities lasted for a total of 90 minutes.

Teachers were able to gain support from researchers and developers online or in short prerecorded guiding videos before and during classroom sessions.

Step 3: Students working with the models. We used a constructionist approach to learning [20, 21] that involved students tinkering with the model interface and code. This involved students playfully manipulating the code of a computational model to generate and pursue questions in relation to the model, much as described in Wagh et al. [15]. This is especially important for students who are novices in computing and programming [23].

We encouraged students to use, modify, and create the model and code by giving them specific tasks to perform. Thus, the model was what Vygotsky [21] conceived of as a semiotic sign meaning a representation for someone that could become a mediated activity for the student. Hereby the model, when appropriated by the student, resulted in a regulatory change in the students' behavior, understanding etc. In the interest of time and because both teachers and students had limited or no experience with computer programming, we chose to let students work with pre-designed computer models rather than have them develop models from scratch. But students were encouraged to modify models and ponder to what extent they furthered the construction of their understanding of content knowledge or CT.

The underlying code in NetLogo consisted of variables and rules such as for molecules in a system. Our aim was that the code should represent a phenomenon as well as the structure and rules of the agents and the dynamic relations between the agents that we wanted to investigate [1]. Hence, the interface should illustrate these relations with accuracy. Furthermore, the code and the interface were designed in order to facilitate the intended learning activities we wanted to achieve. In summary, our aim was to provide students with models and code with interface elements that were easy to tinker with.

Step 4: Collection of data. We conducted design-based research with a sequential transformative design, in which we collected quantitative and qualitative data from students and qualitative data from teachers. Students filled out online questionnaires, while working with the model. In order to analyze data, we employed a description of CT competences and learning goals from the literature [19, 3] as shown below in table 1.

CT category	CT sub-category
CT baseline	CT baseline Perspective development
Subject progress	Subject baseline Subject progress
Understanding the model	Using the model Analyzing the model Understanding the model Decomposing the model Levels of thinking Scaling the phenomenon Pattern recognition Sorting relevant from irrelevant information
Model/subject	Representation of the model Representation of the subject Transferring subject knowledge to the model and back
Interface/code	Analyzing macroscopic syntax Formulating macroscopic syntax with elements of microscopic syntax
Working with code	Creating solutions Defining problems in the model Generalize solutions by algorithmic thinking Capture essential properties common to a set of agents Motivation for working with computational models

Table 1: Categories used for classification in data analysis

The online questionnaire covered the 22 sub-categories of CT (see table 1). The questionnaires all contained the same pre-post question regarding the students' perspective on using a computer model in the subject. Approximately 10% of the questions were multiple-choice questions, the rest required the students to formulate their own answers. Students answers were scored by two independent researchers and pooled into six categories (see table 1) and the percentage of students able to accommodate the learning goals described were calculated (figure 3).

Three independent researchers coded qualitative and quantitative data separately.

Each of the teachers participated in a semi-structured group interview. The interviews were conducted between one and three weeks after the learning activities had taken place. Each interview consisted of three parts: Firstly, teachers were asked if there were any challenges when using the model. Secondly, teachers were asked to elaborate on the advantages for both students and teachers on using the CMC approach. Finally, in the third and last part of the interviews, teachers were asked to consider if they would use the CMC approach when working with other phenomena or even with other subjects. The interviews were audio-recorded and analyzed by two independent researchers who gathered, analyzed, and discussed relevant quotes.

3 RESULTS

3.1 Quantitative data: Students

We examined how students were able to use and understand the model they worked with, analyze and evaluate the model in regard to the subject, if the students were able to draw connections between the code and the interface, and finally if the students were able to successfully modify the code in order to improve the model.

As shown in figure 3, almost all of the students were able to perform the tasks related to these learning goals, with successfully performing students ranging from 74 to 90%. However, 79% of the students were able to perform tasks such as modifying the properties of an agent or adding a new type of agent (figure 3, last category "working with code"). This is suggestive of a highly positive effect in light of the fact that 93% of all students reported that they had no or very little experience with programming.

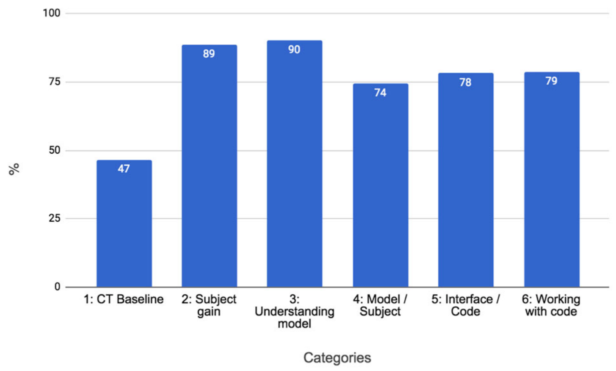

Figure 3: Percentage of students that successfully solved tasks in category

With a pre- and post-test in the form of a questionnaire we probed what students perceived they could learn from models in the subject (e.g. chemistry) as shown in figure 4.

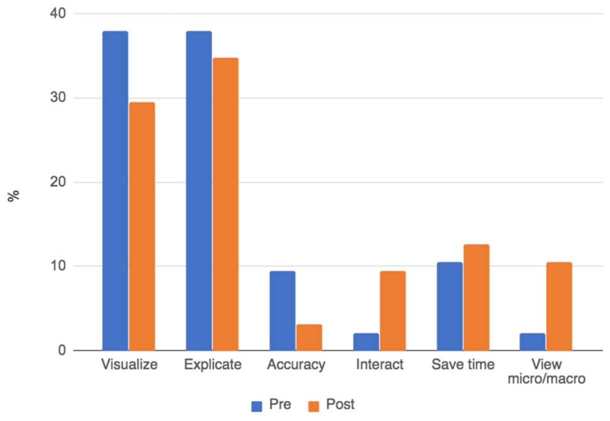

Figure 4: Students' perception of learning objectives acquired after working with computer models.

Interestingly, before having worked with a model, more students reported that computer models helped them visualize and explicate a phenomenon better as opposed to after. One explanation for this finding could be that a pictorial understanding of something is not a deeper conceptual understanding. Thus, students might realize that a visualization (e.g. a picture) does not have adequate information or does not adequately represent the more complex representations that a computer model is capable of. This might be the case in dynamic phenomena (e.g. transport across cell membrane osmosis) where a simple visualization carries less dynamic information that a NetLogo model.

After having worked with a model there was an increase in students who described being able to interact with the model, to save time as oppose to conducting an experiment in real life, and to see micro- and macro-levels of the phenomenon as important purposes of computer models.

3.2 Qualitative data: Teachers

In the semi-structured interviews teachers were asked to elaborate on their experiences with the CMC approach and how NetLogo worked as a learning environment in their class. Of the 15 teachers, only two had prior programming experience in NetLogo. Nevertheless, teachers found that using NetLogo, as a programming environment, made them confident enough to conduct the learning activities without any prior experience in programming or in NetLogo.

"When they [the students] see that the computational model does not match with their own perception of the phenomenon, they can change the code until it fit with what's in their head. That is what the program [NetLogo] can really give me. I can simply check, or see visually, what the students thinks. I have not been able to do that in any other way."

Figure 5: Quotation from a biotechnology teacher

Four teachers mentioned that when students use and modify the code in a model and interact with the teacher while doing so, the students reveal their perception of the phenomenon to the teacher, expressed by a male science teacher in figure 5. Some teachers mentioned that the activities activated other groups of students (such as quiet girls or poor students) than would normally be the most active in class. However, some teachers expressed concern over the fact that a small group of students persistently over the project period lacked motivation for working with code (see quotation in figure 6). More than half of the teachers expressed a wish for more models of other phenomena relevant to their subject and with more complex content knowledge embedded in the models.

"All students started tinkering with the model and most of the students worked very concentrated with the tasks given in the questionnaire. However, a few students didn't seem to be motivated by the tasks, and I found it difficult to engage them."

Figure 6: Quotation from a social science teacher

4 DISCUSSION

4.1 Quantitative data: Students

The finding (figure 3) that a majority of students were able to use, understanding, analyze, and evaluate the models and modify the code behind the models to improve them needs explanation. We take it to mean two things. First, the CMC approach can be used to design meaningful learning activities in which students obtain CT learning goals. Second, NetLogo is an appropriate programming environment for high school students in any of the four subjects tested.

Pre- and post-questions suggested that students change their perspective on what computer models can be used for in subjects when working with a model, see figure 4. Especially interesting is the finding, that students become increasingly aware of levels in phenomena after working with a model. Students emphasized the importance of levels thinking by pointing out how the interface of a NetLogo model can visualize the macro-level described by the micro-level in the code behind. This touch on a point that other researchers have made about affordances of interacting with code to engage in inquiry that help students think through and learn about mechanism [27].

4.2 Qualitative data: Teachers

An interesting quotation is from a male teacher (see figure 5). He elaborates how the application of the CMC approach enabled him as a teacher to elicit the students' perception of the phenomenon. Hence, the quotation demonstrates how powerful a tool computer models can be to teachers, as it helps them to scaffold and evaluate the student' progress in regards to both CT and the content knowledge. However, we must also be aware of the concern illustrated in figure 6 that not all students become motivated by the CMC approach.

Teachers asked for the project team of developers to develop more models in all subjects. Given the teachers enthusiasm about the models, this in no way suggests laziness on their behalf, but rather that teachers wanted to experiment with more models as learning activities. However, we did not adequately provide more models due to lack of resources.

The tinkering approach produced both eureka moments for students (when a representation was especially informative and thus an aha-experience) as well as moments of frustration when students were uncertain about how the model worked and what it represented. Other researchers have concluded when working with NetLogo models that a minimally structured activity followed by a highly structured activity (e.g. the use of worksheets) might lead to longer-term learning gains [24, 25]. The second design principle is to apply three steps: Use, modify, and create as a progression in learning tasks leading the students down a path of working with and learning code and acquiring aspects of CT [5, 26]. By applying these principles, we believe that the students become more motivated, engaged and less intimidated by working with computer models and with code. Thereby rendering the measurement of students' modeling skills and CT [9].

Keeping in mind that this design-based research study focused on developing CT and modeling in high school subjects, our study suggests that it is possible to plan short interventions of CT sequences and obtain student learning in CT in other subjects than computer science. However, future research is needed to examine how students with a limited computer coding proficiency learn to work with increasingly more complex code. As students refine the models and build the code to represent more and more complex phenomena it would be relevant to investigate student experiences: What do students perceive as complex code and why? This has ramifications for how the model should be designed in order to be adequately complex and accommodate relevant learning goals in CT. In particular, we would, based on this study, suggest that future design-based research use agent-based modeling in order to develop models of teaching CT in different high-school subjects.

5 CONCLUSION

In this work, we have focused on developing teaching activities around computer models that represent phenomena in both biotechnology, chemistry, biology, and social sciences. The modeling approach helped students tinker with both models and content knowledge. Although this teaching experiment found that students gained knowledge in both CT and content knowledge through working with computer models of phenomena in four different subjects more work should be conducted on extending the models into new subjects and domains. Four issues should be of concern here. First, to make sure we are not just describing a Pygmalion effect where anything new (e.g. a new NetLogo model) will have a short-term positive effect but not a long-term positive effect on performance. Second, we need to conceive of abstraction in terms of not only student understanding of content in a given high school subject but abstraction as a concrete manifestation or a computational solution that can be used in subjects for problem solving. Third, students problem solving should be measured using a continuous variable and not a Boolean variable. For instance, a limitation of this study leading to measurement bias was the use of a methodology where we rated poor performance as zero and successful performance as one. CT is hardly a dichotomous variable that students either have or do not have, but a set pf complex problem solving skills. Thus CT should be tested with dynamic (cognitive and computational) models in terms of whether students are successful in using, modifying and creating code over time. Fourth, more comparative research should be conducted in terms of how well different subjects might facilitate students' development of CT. We found biotechnology, social science and chemistry equally amenable to model phenomena whereby students could learn CT. All participating teachers were able to collaborate with researchers and developers in this project to produce models that generated CT in students regardless of their subjects.

CT helps raise fundamental ontological questions in high-school teaching about what is biology, social science etc.? This was originally pointed out by Papert [1 p.140] who asked: What is the potential influence of computation on students' understanding of physics? Will CT bring student nearer to grasping what a subject is or merely confuse them about phenomena, representations, codes and models? Our teaching experiment showed that by letting students tinker with models they were able to integrate both coding, modelling and content knowledge.

ACKNOWLEDGMENTS

We are grateful to PhD Palle Nowack for engaging in fruitful discussions leading to the CMC approach. We thank professors: Michael E. Caspersen, Deborah Tatar, Keld Nielsen, Arthur Hjort, and research assistant Nick Nielsen for discussions about CT. We thank the anonymous reviewers for constructive and relevant feedback. We are grateful for financial support from Region Midt and It-vest. Last, we extend our thanks to the 15 teachers and 210 students participating in the study.

REFERENCES

[1] Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas.* Basic Books, Inc.

[2] Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33-35.

[3] Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada* (Vol. 1, p. 25).

[4] Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., & Punie, Y. (2016, July). Exploring the field of computational thinking as a 21st century skill. In *Proceedings of the International Conference on Education and New Learning TechnologiesJuly 2016Barcelona, Spain Page* (pp. 4725-4733).

[5] Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., ... & Werner, L. (2011). Computational thinking for youth in practice. *Acm Inroads*, *2*(1), 32-37.

[6] Centola, D., Wilensky, U., & McKenzie, E. (2013, April). A hands-on modeling approach to evolution: Learning about the evolution of cooperation and altruism through multi-agent modeling-The EACH Project. In *International Conference of the Learning Sciences: Facing the Challenges of Complex Real-world Settings* (p. 166). Psychology Press.

[7] Machluf, Y., & Yarden, A. (2013). Integrating bioinformatics into senior high school: design principles and implications. *Briefings in bioinformatics*, *14*(5), 648-660.

[8] Ioannidou, A., Repenning, A., Lewis, C., Cherry, G., & Rader, C. (2003). Making constructionism work in the classroom. *International Journal of Computers for Mathematical Learning*, *8*(1), 63-108.

[9] Weintrop, D., Beheshti, E., Horn, M. S., Orton, K., Trouille, L., Jona, K., & Wilensky, U. (2014, July). Interactive assessment tools for computational thinking in High School STEM classrooms. In *International Conference on Intelligent Technologies for Interactive Entertainment* (pp. 22-25). Springer, Cham.

[10] Tatar, D., Harrison, S., Stewart, M., Frisina, C., & Musaeus, P. (2017). Proto-computational thinking: The uncomfortable underpinnings. In *Emerging research, practice, and policy on computational thinking* (pp. 63-81). Springer, Cham.

[11] Lockwood, J., & Mooney, A. (2017). Computational Thinking in Education: Where does it fit? A systematic literary review. *arXiv preprint arXiv:1703.07659*.

[12] Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher*, *42*(1), 38-43.

[13] Caspersen, M. E., Gal-Ezer, J., Nardelli, E., Vahrenhold, J., & Westermeier, M. (2018, February). The CECE Report: Creating a Map of Informatics in European Schools. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (pp. 916-917). ACM.

[14] Kynigos, C. (2007). Using half-baked microworlds to challenge teacher educators' knowing. *International journal of computers for mathematical learning*, *12*(2), 87-111.

[15] Wagh, A., Cook-Whitt, K., & Wilensky, U. (2017). Bridging inquiry-based science and constructionism: Exploring the alignment between students tinkering with code of computational models and goals of inquiry. *Journal of Research in Science Teaching*, *54*(5), 615-641.

[16] Wilensky, U. (1999). NetLogo (and NetLogo User Manual), Center for Connected Learning and Computer-Based Modeling, Northwestern University. http://ccl.northwestern.edu/netlogo/.

[17] Hjorth, A., & Wilensky, U. (2014). Re-grow Your City: A NetLogo Curriculum Unit on Regional Development. Boulder, CO: International Society of the Learning Sciences.

[18] Wilensky, U., & Reisman, K. (2006). Thinking like a wolf, a sheep, or a firefly: Learning biology through constructing and testing computational theories—an embodied modeling approach. *Cognition and instruction*, *24*(2), 171-209.

[19] Wing, J. 2011. Research notebook: Computational thinking "What and why? ". In *TheLink Magazine, Spring. Carnegie Mellon University, Pittsburgh.*

[20] Papert, S., & Harel, I. (1991). Situating constructionism. *Constructionism*, 36(2), 1-11.

[21] Vygotsky, L. S. (1980). *Mind in society: The development of higher psychological processes.* Harvard university press.

[22] Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, *25*(1), 127-147.

[23] Nowack, P., & Caspersen, M. E. (2014, November). Model-based thinking and practice: A top-down approach to computational thinking. In *Proceedings of the 14th Koli Calling International Conference on Computing Education Research* (pp. 147-151). ACM.

[24] Berland, M., Martin, T., Benton, T., Petrick Smith, C., & Davis, D. (2013). Using learning analytics to understand the learning pathways of novice programmers. *Journal of the Learning Sciences*, *22*(4), 564-599.

[25] Jang, H., Reeve, J., & Deci, E. L. (2010). Engaging students in learning activities: It is not autonomy support or structure but autonomy support and structure. *Journal of educational psychology*, *102*(3), 588.

[26] Sentance, S., & Waite, J. (2017, November). PRIMM: Exploring pedagogical approaches for teaching text-based programming in school. In *Proceedings of the 12th Workshop on Primary and Secondary Computing Education* (pp. 113-114). ACM.

[27] Sherin, B. L. (2001). A comparison of programming languages and algebraic notation as expressive languages for physics. *International Journal of Computers for Mathematical Learning*, 6, 1-61.