A Hands-On Cybersecurity Curriculum Using a Robotics Platform

Bernard Yett
Vanderbilt University
Nashville, TN, USA
bernard.h.yett@vanderbilt.edu

Nicole Hutchins
Vanderbilt University
Nashville, TN, USA

Gordon Stein
Vanderbilt University
Nashville, TN, USA

Hamid Zare
Vanderbilt University
Nashville, TN, USA

Caitlin Snyder
Vanderbilt University
Nashville, TN, USA

Gautam Biswas
Vanderbilt University
Nashville, TN, USA

Mary Metelko
Vanderbilt University
Nashville, TN, USA

Ákos Lédeczi
Vanderbilt University
Nashville, TN, USA

ABSTRACT

This paper presents a study where high school students were taught computing and cybersecurity concepts using a robotics platform. 38 students attended a week-long summer camp, starting with projects such as a simulation-only game and a simple autonomous driving program for the robots to learn and apply computational thinking (CT) and networking skills. They were then assigned a series of challenges that required developing progressively more advanced cybersecurity measures to protect their robots. This culminated in a final challenge that required implementing defensive measures such as encryption, secure key exchange, and sequence numbers to prevent cyber attacks during robot operations. We used an evidence-centered design framework to construct rubrics for grading student work. The pre- and post-test results show that the interventions helped students learn cybersecurity and CT concepts, but they had difficulties with networking concepts. These results correlate with scores from the game and the final challenge. Overall, surveys show that the competition-based robotics learning framework was engaging to students, and it supported their learning. However, our intervention needs to be modified to help students learn networking concepts.

CCS CONCEPTS

• **Applied computing → Interactive learning environments**; • **Security and privacy**; • **Social and professional topics →** *Computational thinking*;

KEYWORDS

block-based programming, robotics, computer science education, computational thinking, cybersecurity, networking

SIGCSE '20, March 11–14, 2020, Portland, OR, USA
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-6793-6/20/03…$15.00
https://doi.org/10.1145/3328778.3366878

ACM Reference Format:
Bernard Yett, Nicole Hutchins, Gordon Stein, Hamid Zare, Caitlin Snyder, Gautam Biswas, Mary Metelko, and Ákos Lédeczi. 2020. A Hands-On Cybersecurity Curriculum Using a Robotics Platform. In *The 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20), March 11–14, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3328778.3366878

1 INTRODUCTION

There has been a lot of emphasis on introducing computer science (CS) and computational thinking (CT) concepts and practices into K-12 curricula [3]. Educators, researchers, and industry stakeholders have recognized the importance of this integration, not only as a means for better preparing students for the 21st century workforce [22] but also for helping them to develop the abilities to create the next wave of computing innovations [17]. In particular, cybersecurity has become an important topic that is featured widely in multiple studies [6, 10]. In this work, we develop a set of curricular tasks that focus on learning cybersecurity concepts. Our hypothesis is that students will be more engaged in learning cybersecurity and other CS topics through hands-on activities and competitive projects. To facilitate such learning, we have adopted a robotics platform along with a block-based programming language (BBPL) [11] to develop a curriculum with a sequence of tasks that start by teaching CT concepts and practices, and then expand to cover advanced topics in networking and cybersecurity.

In the summer of 2019, we ran an exploratory research study with 38 high school students to evaluate their ability to learn about cybersecurity, networking, and CT topics during a week-long camp. As discussed, our intervention starts with networking and CT concepts to enable students with minimal previous experience to establish baseline knowledge similar to their peers. Then students transition to the cybersecurity problem-solving modules on relatively equal footing and progressively learn to implement new attack and defense strategies on the robotics platform. The platform allows for creating BBPL code, running experiments, and observing results. If a student's program does not work correctly, the BBPL allows for easy debugging by the students and their teachers who may not be well-versed in programming. Our overall study with the intervention included concept-based pre-post-tests, a survey of students' past programming experiences, and a self-assessment of the students' task values and motivation. We also modified the curriculum and assessments from a past study to accommodate new evidence-centered design (ECD)-based rubrics that allowed for the tracking of student understanding of relevant concepts over time.

In this paper, we analyze data collected during the study to answer the following research questions:

(1) **[Student learning]** Did the intervention help students learn the targeted CT, networking, and cybersecurity concepts as determined by their pre-post-test learning gains?
(2) **[Effectiveness of intervention]** Did performance on components of our intervention, such as game development, autonomous driving, encryption methods, and combining security methods correlate with students' pre-post-test learning gains?
(3) **[Student interest and self-efficacy]** Did the answers to the survey and self-assessment questions indicate that students showed engagement and interest in the intervention activities and the topics of study?

The rest of this paper discusses our intervention and the results of our data analyses. Section 2 reviews background work in computing and CT education, and previous work on use of robots to teach CS topics. Section 3 outlines our intervention, providing a day-by-day breakdown of the activities that students completed during the cybersecurity camp. This is supplemented by descriptions of the rubrics used to evaluate several of the projects that the students undertook during the intervention. Section 4 presents our analyses of the student data and discussion of the results. Finally, section 5 presents the conclusions and directions for future research.

2 BACKGROUND

The advent of CSforAll and the accompanying increase in school districts in the USA implementing required CS standards have resulted in a surge in efforts to engage K-12 students in CS and CT education. Curricula such as *Exploring Computer Science* [8] and the *Advanced Placement Computer Science Principles course* [2] have targeted broadening participation in CS by introducing students to constructs - such as networking and the internet, computing, and data analysis - while also promoting important practices such as creativity, abstraction, decomposition, and debugging. Courses such as *The Beauty and Joy of Computing* [1] offer innovative and engaging ways to implement CS and CT curricula in K-12 classrooms. Moreover, the development of BBPEs, such as Scratch, Snap!, and App Inventor, have been integral in increasing dissemination. Their ease-of-use has lessened difficulties students face in introductory programming (e.g., syntax [9]), promoting engagement, confidence, and creativity in working on authentic projects [4, 21].

The growth of CS education research has also led to the development of innovative tools for introducing computer science topics, such as encryption and cybersecurity in K-12 classrooms. PNW GenCyber [10] and SecurityEmpire [16] utilize game based learning approaches, and have produced promising learning gains in K-12 participants. Similar gains were experienced by approaches that integrate robotics and cybersecurity (e.g., Junior Cyber Discovery [20] and Roboscape [11]). However, in order to integrate these tools and approaches into K-12 classrooms, it is important to include methods for assessing students against key standards and teacher-defined learning objectives.

Recent efforts have targeted approaches to assessing student learning during programming tasks, such as those utilizing BBPEs to support programming projects that are creative and personally meaningful [4]. Catete, et al [5] used ideas from auto-graders to create refined rubrics and aid high school teachers, some of whom had no CS background, in offering courses such as *The Beauty and Joy of Computing*. Basu [4] decomposed Scratch projects down to their component parts to create multi-dimensional evaluation rubrics. Grover [9] focused on the language used by participants in her study, looking for them to better express complex ideas using relevant jargon and knowledge. This allows the learning and assessment to go beyond a pure programming focus. For this paper, we extend this approach by mapping student work to rubrics defined by concepts and practices that are assessed on our pre-post-tests. To do this in a systematic manner, we adopt an ECD approach [14] so that we can track students' learning gains temporally as they perform their assigned tasks.

3 METHOD

Thirty-eight high school students (50% female) participated in our week-long, cybersecurity and robotics summer camp. Students worked about 6 hours per day on their assigned tasks. A pre-survey question asked about prior CS knowledge, and only 11 students said that they had completed one or more CS courses in the past. However, 22 students indicated they had worked with BBPEs. Other experience listed included prior web-programming experience (17 students) and experience with programming languages, such as Python (12 students) and Java (8 students). Three students claimed that they had no programming experience.

The curriculum of the 5 day camp was refined based on our design-based research approach using feedback and observations from a previous pilot implementation [11]. The curriculum targeted key standards from the K-12 CS Framework [7] and NGSS [19]. Students worked individually on introductory CS units targeting key programming and CT skills as well as system tools needed to be successful in implementing the cybersecurity tasks on Day 1. The goal for this unit was to minimize the impact of differences in students' prior knowledge (as seen with this cohort) and helping students gain confidence in the environment and programming the robots. For the remainder of the camp students worked in dyads or triads. On days 2 and 3 students performed tasks to become familiar with Roboscape commands. The tasks were designed to support skill development and to use CT constructs learned on Day 1 in programming the robots. By the end of Day 3, students were implementing rudimentary security measures, such as observing a count of both commands sent to and commands received by their robot to detect possible attacks.

On Day 4, students incorporated stronger defensive measures using encryption. They first utilized a simple Caesar's cipher, then defended against brute force and overheard key attacks with secure key exchange and stronger encryption methods, such as the

Vigenère cipher. The tasks were designed for students to not only plan and implement methods for protecting their robots, but also to test their algorithms with their robots. On Day 5, students worked on a comprehensive project to defend their robot against replay attacks, which allowed other student groups to capture and reuse an encrypted command being sent from a student to the robot. To do so, students were introduced to sequence numbering, and tasked with adding this measure together with their existing defensive strategy to create a much stronger defense against attackers.

Our exploratory study (no control group) primarily focused on evaluating students learning gains in CT, networking, and cybersecurity using a summative, pre-post-test and formative evaluations of student projects based on predefined rubrics. We implemented an ECD approach to assessment development to map key CT, networking, and cybersecurity constructs addressed in the curriculum to the evaluation measures.

Our pre-post-tests consisted of 8 questions. The two networking questions were free response, while the four cybersecurity questions and the two CT pre-test questions were multiple-choice. For the post-test, we changed the CT format to one multiple choice and one fill-in-the-blank question. In the CT section, the question about loops tasked students with choosing the correct output for two sections of pseudo-code containing repeat loops; partial credit was awarded for correctly choosing the correct output for one of the two cases. The CT question on conditionals consisted of an "if-then, if-then" structure on the pre-test but an "if-then-else" structure on the post-test. This question will be standardized from pre-post in the future to remove any possibility of inconsistent results. The first networking question presented a message passing scenario with identifier tags, and asked students to explain the consequences of removing those identifiers (it would cause a network overload). The second networking question dealt with cooperating armies attempting to send attack and confirmation messages back and forth through enemy territory. The key idea was that it was impossible to guarantee messages were successfully received from both sides no matter how many messages were sent. The cybersecurity questions were:

(1) What makes cryptographic algorithms secure?
(2) (After describing a denial of service attack) Which of the following is the name for this kind of attack?
(3) Which of the following would allow you to decrypt a long message containing English text encrypted by Caesar's cipher?
(4) If the key is known, is encryption or decryption more difficult?

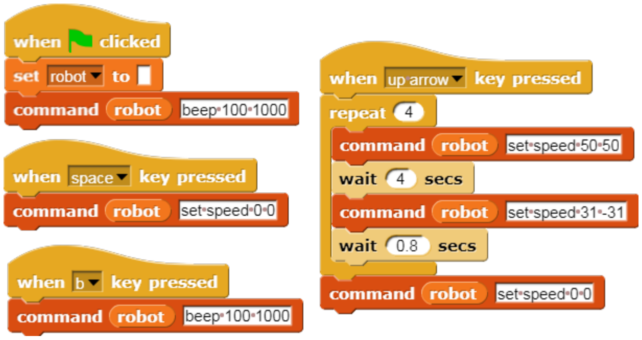
We found the normalized change [12] between each student's pre-post-test scores with the following set of equations:

$$c = \begin{cases} \dfrac{\text{post} - \text{pre}}{1 - \text{pre}} & \text{post} > \text{pre} \\ \text{drop} & \text{post} = \text{pre} = 1 \\ 0 & \text{post} = \text{pre} \neq 1 \\ \dfrac{\text{post} - \text{pre}}{\text{pre}} & \text{post} < \text{pre} \end{cases}$$

This scoring system shows a student's improvement relative to room for improvement. Students who achieve perfect scores on both pre- and post-tests are dropped because of the ceiling effect. The normalized change is applied when establishing connections between project and test results. To study learning gains, we used the averages and standard deviations.

The analysis of student work during the intervention presented in this paper involved the scoring of three projects: (1) the "cat-and-mouse" game (Day 1); (2) the autonomous robot driving problem (Days 2-3); and (3) the final project incorporating replay attack defense on top of other security measures (Day 5). All of the rubrics went through an iterative development process to ensure grading consistency. The "cat-and-mouse" game targeted CT constructs assessed on the pre-post-test (e.g., loops and conditional logic) through the implementation of a game in which a player needed to avoid an adversary sprite.

Students were asked to implement code to move all sprites, handle sprite collision events, implement a game end sequence, and increment a score variable. A maximum score of nine was possible and was achieved by two students. For the autonomous robot project, students were tasked with creating a program that would allow the robot to drive in a square autonomously (Figure 1). CT constructs including loops were required to successfully implement this task - mapping it to the "cat-and-mouse" task from Day 1. A sample rubric is provided in Table 1.



```
when ⚑ clicked
set robot ▾ to □
command robot beep·100·1000

when space ▾ key pressed
command robot set·speed·0·0

when b ▾ key pressed
command robot beep·100·1000

when up arrow ▾ key pressed
repeat 4
    command robot set·speed·50·50
    wait 4 secs
    command robot set·speed·31·-31
    wait 0.8 secs
command robot set·speed·0·0
```

Figure 1: Example autonomous driving program

Description	Score
No project submission	0
Submission with inconsistent timing, trying to complete the task with a circle around the obstacle, or incorrect implementation	1
Submission with correct implementation but lacking loops or other conditional usage	2
Submission with correct implementation including loops	3
Submission with correct implementation including loops plus additional features such as well used custom blocks or side lengths controlled by a variable	4

Table 1: Square Rubric

The final project was based around replay attack defense – described previously – and required students to build a comprehensive defense strategy against various attacks. Four categories made up the grade for the final project. Coding best practices deals with an initialization process, proper variable usage and naming, and the (hopeful) absence of unused blocks and duplicated scripts. The CT score is linked to the proper use of conditional statements, and a "repeat until" loop to guarantee the robot receives a changed encryption key. A custom block category required the proper use of the indicated blocks, and correctly updating them to use the advanced encryption and sequence numbering implementations. The final and most important category, cybersecurity, is displayed as an example rubric in Table 2. The majority of concepts were graded for their presence (meaning some attempt was made to use the relevant concept), completeness (meaning the concept was applied everywhere that it should have been applied, though not necessarily correctly), and correctness (meaning the concept was used correctly everywhere that it was applied, though not necessarily applied everywhere that was required).

The maximum score achievable for this project was 39 points. The highest score achieved by a group was 27.5 points. It is possible that students had the knowledge to obtain perfect scores, but the time constraints imposed by the camp did not provide them with sufficient time on the tests. We hypothesized that students doing well on the cybersecurity section of this project would also show larger pre-post-test gains for the other sections of the tests. Considering the all-encompassing nature of the project, we also compared the overall project scores with the pre-post-test learning gains.

Finally, self-assessment surveys administered to students asked them to indicate their confidence in acquiring particular skills. They were also asked how highly they valued the different topics covered in the intervention. The questions were designed carefully so that they would not bias the answers [13]. Student responded on a scale from 1 to 10, with 1 indicating "Strongly Disagree" and 10 indicating "Strongly Agree." The same self-assessments were administered on the first and last days of the camp for comparison purposes.

4 RESULTS AND DISCUSSION

Research Question 1: Pre-post-test analysis for Learning Gains. A two-tailed *t-test* was conducted to test for significance in learning gains from pre- to post-test. Table 3 reports the pre- and post-test scores along with the computed p-scores and the effect sizes (*Cohen's d measure*). Overall, students had significant learning gains ($p < 0.01$), but the effect size of the overall learning gains was small (0.26). The learning gains were significant ($p < 0.01$) for the CT and the cybersecurity questions with moderate effect sizes of 0.42 and 0.33, respectively.

Students did not seem to improve on the networking questions; on average, their scores from pre- to post-test did not change. It is possible that the lack of improvement in the networking scores could be attributed to a disparity between the concepts covered in the intervention in comparison to what was on the test. From our discussion with the students, it seems that they developed an overly simplistic view of the networking concepts from the materials presented to them and the tasks they performed during the intervention.

Table 4 presents a more detailed analysis of the individual pre- and post-test questions. The rows in blue text represent questions where students showed an increase in the pre- to post-test scores, and the rows in red text represent questions in which the students' scores decreased or showed no change. Improvement was significant ($p < 0.01$) with a large effect size (1.26) for the "Attack Type" question. Improvement was significant at the $p < 0.05$ level, with moderate effect sizes for the "Loops" (0.41), "Conditionals" (0.44), and "Encryption vs. Decryption" (0.38) questions. The "Cryptography Algorithms" question showed a significant ($p = 0.01$) decrease from pre- to post-test, indicating a problem with how we discussed that material with the students. These ideas were not specifically taught, and the terminology used in the question was not explicitly discussed during the intervention. On the other hand, the Caesar cipher was covered quite thoroughly through instruction during the camp. In this case, the lack of a significant learning gain may be attributed to a ceiling effect. For the "Attack Type" question, students understood that a described scenario with no relation to robotics matched up with the Denial of Service attacks they had been taught.

Research Question 2. Correlations between Intervention Task Scores and Pre-Post-test Learning Gains. We compare scores on the individual project tasks against students' pre-post-test learning gains to demonstrate the effectiveness of our intervention. Table 5 displays the average scores and standard deviation for each project. The final cybersecurity project grade is displayed separately. The relevant pre-post section or question scores are also listed as average normalized changes (see equations in Section 3). To study the relationships, we computed the Pearson product-moment correlation coefficient, r, between each project scores and the relevant pre-post gain. It should also be noted that we only included the pre-post results of students who had made an attempt at the project. This number is denoted by n.

We first analyze the "cat-and-mouse" game results. Out of the 38 students who completed the camp, we were able to find saved projects for this game at various stages of development for 31 of them. It is impossible to say how much of the project the remaining seven students completed. Therefore, we completely excluded them from the calculations. There was a direct correlation between CT concepts and the game task (high value of $r = 0.60$ between game score and CT pre-post learning gains). In fact, the high scoring students also had high pre-post learning gains overall (moderately high value of $r = 0.52$ for this pairwise correlation). These results can be seen graphically in Figure 2 after separating students into "Complete", "Semi-Complete", and "Incomplete" groups based on project performance, then comparing to pre-post-test results. This is a clear indication that the CT-focused starter units are very important for success in overall learning during the intervention. In the future, we will ensure students do not fall behind in these units, as they form the core for all subsequent work.

For the autonomous robot driving project, we compared scores with pre-post results on the question on loops for the $n = 33$ students who submitted a project. Similarly to the previous game project, five students had no saved version of this project and are excluded from calculations. It appears that there is only a very weak positive correlation ($r = 0.05$) between these two scores across all students who completed the project. Those students who learned and applied knowledge of loops successfully for this project did not seem to perform noticeably better on this pre-post question than their peers who used less efficient or incorrect approaches. Loops

Category	0 points	1 point	2 points	3 points
Encryption	Only using basic level encryption	Incorrect attempt at using improved encryption	Correctly using improved encryption in some locations, or incorrectly using improved encryption in all locations	Completely and correctly using improved encryption
Sequence numbering	No attempt at sequence numbering	Incorrect attempt at sequence numbering	Correctly implementing sequence numbering but not for all commands	Correctly implementing sequence numbering everywhere
Replay attack	No attempt at a replay attack	Incorrect attempt at a replay attack	N/A	Completely correct replay attack
Key changing	No attempt at changing keys	Incorrect attempt at changing keys	Slightly incorrect attempt used in appropriate locations	Completely correct implementation used appropriately
Attack vulnerabilities	Vulnerable to replay attacks, overheard keys, and brute force attacks	Vulnerable to two of those attacks	Vulnerable to one of those attacks	Not vulnerable to any of those attacks

Table 2: Final Project Rubric - Cybersecurity Section

Section	Pre average (SD)	Post average (SD)	P-Score	Effect Size
Overall	0.64 (0.17)	0.74 (0.14)	<0.01	0.26
CT	0.70 (0.32)	0.84 (0.20)	<0.01	0.42
Networking	0.60 (0.26)	0.59 (0.24)	0.89	-0.02
Cybersecurity	0.64 (0.22)	0.79 (0.18)	<0.01	0.33

Table 3: Pre-Post-Test Results

Question Topic	Section	Pre average (SD)	Post average (SD)	P-Score	Effect Size
Loops	CT	0.75 (0.28)	0.86 (0.23)	0.04	0.41
Conditionals	CT	0.66 (0.48)	0.83 (0.30)	0.03	0.44
Overloading the Network	NET	0.51 (0.27)	0.51 (0.25)	1	0
Receiving Confirmation	NET	0.68 (0.41)	0.67 (0.41)	0.86	-0.03
Cryptographic Algorithms	CYBER	0.87 (0.34)	0.71 (0.46)	0.01	-0.39
Attack Type	CYBER	0.34 (0.48)	0.87 (0.34)	<0.01	1.26
Caesar's Cipher	CYBER	0.87 (0.34)	0.89 (0.31)	0.71	0.08
Encryption vs. Decryption	CYBER	0.5 (0.51)	0.68 (0.47)	0.03	0.38

Table 4: Individual Question Results

Project	n	Topic	Project Score Average (SD)	Pre-Post Gains Average (SD)	Correlation
Game	31	CT	5.69 (2.11)	0.46 (0.48)	0.60*
Game	31	Overall	5.69 (2.11)	0.23 (0.29)	0.52*
Square	33	CT - Loops	2.33 (1.14)	0.36 (0.61)	0.05
Final	38	Cybersecurity	8.05 (2.32)	0.39 (0.45)	0.25
Final	38	Overall	23.07 (3.36)	0.25 (0.29)	0.08

* indicates result is statistically significant

Table 5: Selected Project Results

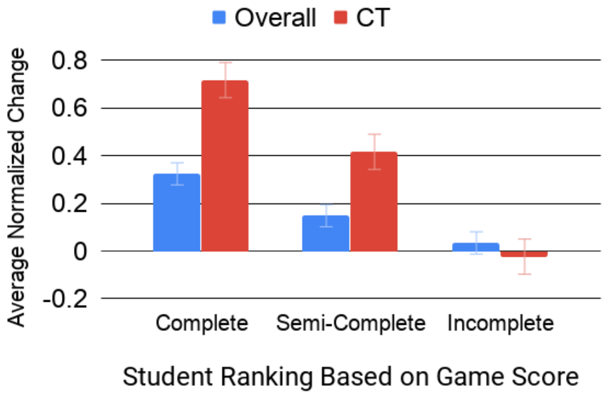


Figure 2: Students are sorted into categories depending on their game scores. The average normalized change in CT and overall results from the pre-post-test is calculated and displayed along the y-axis for each category.

are an important CT construct, so we will have to address this topic in a more systematic way in future studies.

Finally, we discuss the final project results in Table 5. One comparison involved just the cybersecurity section of the project as compared to the pre-post cybersecurity results. Those two data sets were weakly positively correlated ($r = 0.25$), indicating a small trend upward for students that were able to properly apply cybersecurity concepts. On the other hand, our second comparison between the overall project grades and the overall pre-post normalized learning gain showed a very weak ($r = 0.08$) correlation in the positive direction.

Research Question 3: Student Interest and Self-Efficacy. In the self-assessment surveys conducted before and after the intervention, we found students' responses to be generally positive. Their self-efficacy in the CT and networking topics improved as a result of the intervention. This provides a contrast between our test-based assessments and student self-assessment for networking. Two questions specifically related to networking - "I understand what computer networking is" and "I use networked applications on a daily basis" - saw significant positive gains in self-efficacy ($p < 0.001$). Another question that asked if students recognized that they use network messaging applications daily also saw significant gains ($p = 0.005$). This does not provide clarity between the disparities of what was taught and what was tested. However, it does point to a simplistic student understanding of networking concepts while further indicating a need for more applicable knowledge.

Other key areas of growth included recognition of computer scientists and the importance of their professional work, as well as distributed computing skills gained from working specifically on related projects. In contrast, we saw no growth among students for the following statements: "I am interested in computer science" and "I am interested in a career that involves computer programming". These issues will have to be probed in greater detail in future work.

Statement	Pre	Post	P-Score
I understand the types of projects that computer scientists work on and the skills they use in their careers	5.65	7.24	<0.001
Computer scientists play an important role in solving many of the global challenges we face today	8.46	8.92	0.030
I understand what computer networking is	5.54	7.65	<0.001
I use networked applications on a daily basis	7.54	9.03	<0.001
I regularly use applications that send or receive network messages on a daily basis	8.59	9.38	0.005
I know how to build a distributed application	3.11	6.49	<0.001

Table 6: Selected Survey Results

Limitations of current study. Many topics that were covered during the camp were not tested due to the limited time that was allocated for these tests. Lists, variables, custom blocks, events, message passing (in a more explicit way), different types of blocks (reporter, command, etc.), programming efficiency, Unicode, binary, and replay attacks are all areas that were not included in the tests this summer but could potentially be included in the future. In addition, we can include formative assessments with our units to facilitate self-assessment by the students, thereby providing mechanisms for them to overcome their difficulties with certain concepts and become better learners.

For networking, improvements could be made by introducing a visual representation of how the commands are passed from (for example) user to robot, along with a better theoretical framing of the subject as opposed to the more implicit current structure. Key networking concepts such as message latency, delivery failures, types of networks, and networking protocols could be introduced to students in an interesting way with the help of our robotics platform. We plan to create a specific networking curriculum to encompass these ideas, and we expect that this would reproduce the strong self-assessment results while helping students show more improvement in applying their knowledge of this domain.

Looking at some other responses from the post-camp survey, a few areas stand out. Almost half of respondents cited issues with communication between themselves and their robots or between each other during collaboration, with comments such as "Server not always reliable; blocks of script known to work in Scratch did not yield the same results (or any at all); problems with saving or editing" and "We had trouble collaborating using NetsBlox which made it challenging to code the robots. The program would often not let one person edit and would show different versions of the same project to different people."

Additionally, many students took issue with finding the blocks they wanted to use at any particular time. Comments such as "It (NetsBlox) was somewhat confusing to use, and I didn't always understand exactly how to set the blocks up in a way that would make my code work." support this, along with many students requesting features such as help sections for the blocks or a search bar to more easily locate them. These features are already present in the system, but they were not always mentioned to students.

5 CONCLUSIONS AND FUTURE WORK

Overall, our results indicate that a hands on robotics platform connected to a BBPE helps K-12 students learn basic CT and cybersecurity concepts while also keeping them engaged and retaining their interest in computing. There is also evidence to suggest that some of the initial projects, particularly the "cat-and-mouse" game, helped students develop their basic CT skills, which made it easier for them to go through the more difficult parts of the curriculum. It also helped students who had lower levels of prior knowledge catch up with the class.

Despite these promising results, many improvements can be made to the intervention, some of which we have already discussed as limitations in the previous section. In the future, we plan to spend more time in giving students an understanding of the theoretical concepts in networking and cybersecurity. More learning opportunities can be created by providing feedback that help students reflect on their projects at the end of each unit. Formative assessments should also provide students with more learning opportunities.

We are interested in implementing our system on different hardware platforms. Less costly robots, drone systems, or simulation-based options can be assimilated into our framework to increase deployment in K-12 school settings. Additionally, there is some space to branch out into collaborative problem solving with robots. Cheaper robots would make it an easier to provide multiple robots to a group or even an individual, and they could then develop programs that implement cooperative problem solving by the robots.

We also plan to work with teachers on implementing variations of our curriculum in classroom settings for middle and high school. Some of the teachers we worked with this summer expressed concern that their wireless networks - or even general computer access - may not be suitable for supporting our curricular tasks in their current form. While simulation-based work is one option, there can also be value in designing an offline executable environment for students to develop and test their algorithms before they implement them on physical robots. Finally, an automated grading or feedback generation system [15, 18] would be a useful supplement to our current selection of example projects and rubrics, both to reduce the load on teachers and to detect bad programming habits.

6 ACKNOWLEDGEMENTS

This material is based in part upon work supported by National Security Agency Science of Security Lablet H98230-18-D-0010 and National Science Foundation grants CNS-1644848, CNS-1521617, and DRL-1640199. Any opinions, findings, and conclusions expressed in this material are those of the author(s) and do not necessarily reflect the views of the US Government.

REFERENCES

[1] beauty and joy of computing: 2016-17 findings from an ap cs principles course.

[2] O. Astrachan, T. Barnes, D. D. Garcia, J. Paul, B. Simon, and L. Snyder. Cs principles: piloting a new course at national scale. In *Proceedings of the 42nd ACM technical symposium on Computer science education*, pages 397–398. ACM, 2011.

[3] V. Barr and C. Stephenson. Bringing computational thinking to k-12: What is involved and what is the role of the computer science education community? *Inroads*, 2(1):48–54, 2011.

[4] S. Basu. Using Rubrics Integrating Design and Coding to Assess Middle School Students' Open-ended Block-based Programming Projects. In *SIGCSE 2019 - Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pages 1211–1217, 2019.

[5] V. Cateté, E. Snider, and T. Barnes. Developing a rubric for a creative cs principles lab. In *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education*, pages 290–295. ACM, 2016.

[6] W. chang Feng, R. Liebman, L. Delcambre, M. Lupro, T. Sheard, S. Britell, and G. Recktenwald. Cyberpdx: A camp for broadening participation in cybersecurity. In *2017 USENIX Workshop on Advances in Security Education (ASE 17)*, Vancouver, BC, 2017. USENIX Association.

[7] C. S. F. S. Committee. K–12 Computer Science Framework. *(https://k12cs.org/)*, 2016.

[8] J. Goode, G. Chapman, and J. Margolis. Beyond curriculum: the exploring computer science program. *ACM Inroads*, 3(2):47–53, 2012.

[9] S. Grover and S. Basu. Measuring student learning in introductory block-based programming: Examining misconceptions of loops, variables, and boolean logic. In *Proceedings of the 2017 ACM SIGCSE technical symposium on computer science education*, pages 267–272. ACM, 2017.

[10] G. Jin, M. Tu, T.-H. Kim, J. Heffron, and J. White. Evaluation of Game-Based Learning in Cybersecurity Education for High School Students. *Journal of Education and Learning (EduLearn)*, 12(1):150, 2018.

[11] Á. Lédeczi, M. Metelko, X. Koutsoukos, G. Biswas, M. Maróti, H. Zare, B. Yett, N. Hutchins, B. Broll, P. Völgyesi, M. B. Smith, and T. Darrah. Teaching Cybersecurity with Networked Robots. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pages 885–891. ACM, 2019.

[12] J. D. Marx and K. Cummings. Normalized change. *American Journal of Physics*, 75(1):87–91, 2007.

[13] J. Mirkovic, M. Dark, W. Du, G. Vigna, and T. Denning. Evaluating Cybersecurity Education Interventions: Three Case Studies. *IEEE Security & Privacy*, 13(June):63–69, 2015.

[14] R. J. Mislevy and G. D. Haertel. Implications of evidence-centered design for educational testing. *Educational Measurement: Issues and Practice*, 4(25):6–20, 2006.

[15] J. Moreno and G. Robles. Automatic detection of bad programming habits in scratch: A preliminary study. *Proceedings - Frontiers in Education Conference, FIE*, 2015-Febru(February):1–4, 2015.

[16] M. Olano, A. Sherman, L. Oliva, R. Cox, D. Firestone, O. Kubik, M. Patil, J. Seymour, I. Sohn, and D. Thomas. SecurityEmpire: Development and Evaluation of a Digital Game to Promote Cybersecurity Education. In *2014 {USENIX} Summit on Gaming, Games, and Gamification in Security Education (3GSE 14)*, pages 1–10, 2014.

[17] R. B. Schnabel. Educating computing's next generation. *Communications of the ACM*, 4(54):5–5, 2011.

[18] R. Singh, S. Gulwani, and A. Solar-Lezama. Automated feedback generation for introductory programming assignments. *Acm Sigplan Notices*, 48(6):15–26, 2013.

[19] N. Standards. Next generation science standards: For states, by states (vol 1) washington. 2013.

[20] H. Tims, G. E. Turner III, G. Cazes, and J. M. Marshall. Junior cyber discovery: Creating a vertically integrated middle school cyber camp. In *American Society for Engineering Education*. American Society for Engineering Education, 2012.

[21] D. Weintrop and U. Wilensky. Comparing Block-Based and Text-Based Programming in High School Computer Science Classrooms. *ACM Transactions on Computing Education*, 18(1):1–25, 2017.

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