Analyzing Computational Thinking Gameplay: Identifying Struggles and the Role of Experience

Sotheara Veng
University of Delaware
Newark, DE, USA
sotheara@udel.edu

Yifan Zhang
Beijing Normal University
Beijing, China
zhangyifan@bnu.edu.cn

Ekaterina Bergwall
University of Delaware
Newark, DE, USA
ebergwal@udel.edu

Teomara Rutherford
University of Delaware
Newark, DE, USA
teomara@udel.edu

Abstract

Computational thinking (CT) skills are essential for navigating daily life, but how best to teach these skills is still being explored. Educational programming games can provide engaging environments to develop CT, yet uncovering student struggles during gameplay remains challenging. This study analyzed 15 undergraduates with varying programming experience playing the CT game, Fox and Field, utilizing gameplay videos, heart rate data, and surveys. Results revealed common struggle points in understanding the character's perspective, degree angles, and debugging, with notable differences between novice and experienced players. Findings suggest support targeting abstraction, pattern recognition, and debugging skills to improve future game design and scaffolding.

ACM Reference Format:

Sotheara Veng, Yifan Zhang, Ekaterina Bergwall, and Teomara Rutherford. 2025. Analyzing Computational Thinking Gameplay: Identifying Struggles and the Role of Experience. In *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 2 (SIGCSE TS 2025), February 26-March 1, 2025, Pittsburgh, PA, USA*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3641555.3705127

1 INTRODUCTION AND RELATED WORK

Computational thinking (CT) is a crucial 21st-century skill and supports success in a digital world [6, 14]. Educational programming games offer a promising approach to developing CT in an engaging way [9, 15]. Prior research often uses post-gameplay surveys and interviews to understand students' struggle points (e.g., [7]), which may miss key in-the-moment struggles. Few studies have used multimodal approaches that incorporate qualitative analysis of gameplay data, along with physiological data, such as heart rate, and students' explanation of their strategies, which can provide a deeper understanding of the gameplay. Our study examines students' struggle points in a block-based programming CT game, using these multiple data sources, to answer the following research questions:

- RQ1: What are students' struggle points and misconceptions in the CT game?
- RQ2: How do these struggles vary by programming experience?

SIGCSE TS 2025, February 26-March 1, 2025, Pittsburgh, PA, USA

ACM ISBN 979-8-4007-0532-8/25/02
https://doi.org/10.1145/3641555.3705127

2 METHODOLOGY

A block-based programming game, Fox and Field, was created to test whether certain scaffolds could make novice programmers behave more like experts. Fox and Field is similar to other block-based programming environments used in prior research (e.g., [4, 13]). The game had eight levels, each with a unique objective and condition. We selected a level (L412) that most players experienced and that allowed for enough variation in play, as players of L412 needed to handle angle application, path distractions, and code efficiency prompts (see Figure 1).

Figure 1: Game map for level 412

We recruited players for the study through open advertisement. They were categorized as Novice (66.7%) or Experienced (33.3%) based on a pre-game questionnaire. The sample was predominantly women (93.3%) and included participants identifying as White (46.7%), Asian (26.7%), Hispanic or Latinx (6.7%), Black or African American (6.7%), Hispanic or Latinx-Asian (6.7%), and Hispanic or Latinx-White (6.7%). During the one-hour gameplay session, heart rates (per 116 milliseconds), screen and face video, and gameplay data (time spent, code blocks, and attempts) were recorded, and players completed open-ended reflection surveys on their solving strategies.

Three gameplay videos coded based on students' gameplay performance groups (high, intermediate, and low) to generate coding schemes. Then, we used the directed content analysis approach for all videos to analyze the data using the coding schemes but without ignoring new emerging concepts or issues [8]. We obtained Cohen's kappa inter-rater reliability of $k = 0.86$ indicating a high level of agreement. Finally, both coders discussed and resolved any discrepancies in a consensus meeting. Due to corruption of some data, the heart rate and end survey data were only used to verify and confirm programmers' struggle points.

3 RESULTS AND DISCUSSION

Players primarily struggled with three aspects: (1) **Concept of Degree Angle**—33.3% of players (*n*=5) struggled with understanding degrees. Novices spent an average of 536.1 seconds (*SD* = 484.5) and 13.9 attempts (*SD* = 19.4) resolving errors compared to 291.8 seconds (*SD* = 117.7) and 3.2 attempts (*SD* = 3.8) for experienced players. Two novice players showed a significant increase in heart rate (10-20 bpm). Melina (all names pseudonyms) noted, *"It was more difficult to find the angle to get to the house, so I kept guessing until I could get to the right spot."*

(2) **Fox's Perspective**—53.3% of players (*n*=8) struggled with programming from the fox's viewpoint, often confusing its current direction. Although no experienced players made this mistake, 80% of novices (*n*=4) misjudged the direction more than once, leading to repeated errors and taking an average of four minutes (*SD* = 1.2) to resolve each instance. There was no statistically significant change in heart rate for these players, possibly because they noticed the errors quickly and stopped the code before completing the attempt.

(3) **Debugging**—33.3% of players (*n*=5) had difficulty identifying and correcting code errors, leading to inefficient strategies like restarting the code or deleting blocks one by one. Luna spent over 20 minutes (1,230 seconds) and 40 moves trying to identify the correct error. Her heart rate spiked from 70 bpm to over 100 bpm. She shared, *"I thought the problem was the angle … the actual problem was the mushroom."* Similarly, Maria struggled to pinpoint errors and made changes to unrelated code, resulting in increased heart rates of 15-20 bpm.

Overall, on average, experienced players completed the level faster (*M* = 4.8 minutes, *SD* = 1.9) and with fewer errors (M = 3.2 attempts, SD = 3.4) compared to novices (*M* = 8.9 minutes, *SD* = 8.1; *M* = 13.9 attempts, *SD* = 19.4), but they avoided using non-right angles, missing one of the level objectives. Although novices, having attempted more complex maneuvers, experienced more frequent heart rate spikes during debugging and conceptual struggles.

These identified difficulties are linked to the CT skill of abstraction, aligning with previous findings that novice programmers often struggle with this skill [5, 12]. The struggles with degree angles and perspective suggest the importance of spatial reasoning skills, such as mental rotation, for effective gameplay [2, 3]. Debugging difficulties, on the other hand, are consistent with known novice challenges in error identification [10].

Our comparison between novice and experienced players revealed that whereas experienced players completed the tasks faster with fewer errors, their solutions lacked the expected efficiency approach, as they avoided using non-right angles. This could indicate a potential link between programming experience and spatial reasoning skills as have been shown in some prior work (e.g., [1, 11], a topic requiring further research. Given these findings, scaffolding that targets spatial reasoning and debugging strategies may help novices build stronger CT skills.

4 CONCLUSION

Although limited by a small sample and the focus on a single game level, this study highlights the value of a multimodal approach to uncover real-time challenges during gameplay. In the future, we will expand this work by incorporating data from all levels and eye-tracking data, which will increase the sample size and enhance the robustness of our analysis of biometric data, allowing for more reliable insights into students' challenges and learning patterns throughout the game.

Acknowledgments

The authors would like to thank the members of the Rutherford Lab, including Andrew Rodrigues and the research assistants that assisted with data collection. We also acknowledge the contributions of Dr. Ray Patt to the creation of Fox and Field. This research was carried out with the support of the "University of Delaware General University Research" fund.

References

[1] Ana Paula Ambrosio, Leandro Da Silva Almeida, Joaquim Macedo, and Amanda Franco. 2014. Exploring Core Cognitive Skills of Computational Thinking.. In *PPIG*. 4.
[2] Sarah A Burnett and David M Lane. 1980. Effects of academic instruction on spatial visualization. *Intelligence* 4, 3 (1980), 233–242.
[3] Giuseppe Città, Manuel Gentile, Mario Allegra, Marco Arrigo, Daniela Conti, Simona Ottaviano, Francesco Reale, and Marinella Sciortino. 2019. The effects of mental rotation on computational thinking. *Computers & Education* 141 (2019), 103613.
[4] Adina Deiner and Gordon Fraser. 2024. NuzzleBug: Debugging Block-Based Programs in Scratch. In *Proceedings of the 2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE '24)* (New York, NY, USA). ACM, Lisbon, Portugal, 13 pages. https://doi.org/10.1145/3597503.3623331
[5] C. Gibbs and Y. Coady. 2010. Understanding Abstraction: A Means of Leveling the Playing Field in CS1?. In *Proceedings of the ACM International Conference Companion on Object Oriented Programming Systems Languages and Applications Companion.* 169–174.
[6] Sarah Gretter and Aman Yadav. 2016. Computational thinking and media & information literacy: An integrated approach to teaching twenty-first century skills. *TechTrends* 60 (2016), 510–516.
[7] Danial Hooshyar, Margus Pedaste, Yeongwook Yang, Liina Malva, Gwo-Jen Hwang, Minhong Wang, Heuiseok Lim, and Dejan Delev. 2021. From gaming to computational thinking: An adaptive educational computer game-based learning approach. *Journal of Educational Computing Research* 59, 3 (2021), 383–409.
[8] Hsiu-Fang Hsieh and Sarah E Shannon. 2005. Three approaches to qualitative content analysis. *Qualitative health research* 15, 9 (2005), 1277–1288.
[9] Tak Yeon Lee, Matthew Louis Mauriello, John Ingraham, Awalin Sopan, June Ahn, and Benjamin B Bederson. 2012. CTArcade: learning computational thinking while training virtual characters through game play. In *CHI'12 Extended Abstracts on Human Factors in Computing Systems.* 2309–2314.
[10] Laurie Murphy, Gary Lewandowski, Renée McCauley, Beth Simon, Lynda Thomas, and Carol Zander. 2008. Debugging: the good, the bad, and the quirky–a qualitative analysis of novices' strategies. *ACM SIGCSE Bulletin* 40, 1 (2008), 163–167.
[11] Marcos Román-González, Juan-Carlos Pérez-González, and Carmen Jiménez-Fernández. 2017. Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in human behavior* 72 (2017), 678–691.
[12] M. Sbaraglia, M. Lodi, and S. Martini. 2021. A Necessity-Driven Ride on the Abstraction Rollercoaster of CS1 Programming. *Informatics in Education* 20, 4 (2021), 641–682.
[13] B. T. Tabarsi, A. Limke, H. Reichert, R. Qualls, T. Price, C. Martens, and T. Barnes. 2022. How to Catch Novice Programmers' Struggles: Detecting Moments of Struggle in Open-Ended Block-Based Programming Projects using Trace Log Data. In *Proceedings of the 6th Educational Data Mining in Computer Science Education (CSEDM) Workshop*, B. Akram, T. Price, Y. Shi, P. Brusilovsky, and S. Hsiao (Eds.). Durham, United Kingdom, 57–65.
[14] Jeannette M Wing. 2006. Computational thinking. *Commun. ACM* 49, 3 (2006), 33–35.
[15] Weinan Zhao and Valerie J Shute. 2019. Can playing a video game foster computational thinking skills? *Computers & Education* 141 (2019), 103633.