Manipulatives for Teaching Computer Science Concepts

Jennifer Parham-Mocello
School of EECS
Oregon State University
Corvallis, USA
parhammj@oregonstate.edu

Garrett Berliner
School of EECS
Oregon State University
Corvallis, USA
berlineg@oregonstate.edu

Ayushi Gupta
School of EECS
Oregon State University
Corvallis, USA
guptaay@oregonstate.edu

2023 IEEE Frontiers in Education Conference (FIE) | 979-8-3503-3642-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/FIE58773.2023.10343500

***Abstract*—Despite many initiatives to increase participation in K-12 computer science (CS), only about half of the public high schools offer CS, and only about a third of the K-8 public schools offer CS. To make matters worse, even if schools offer CS, interests wane from late elementary through post-secondary education. The lack of participation has been attributed to feelings of not belonging, technology-rich programs creating divides among students, and negative belief systems that CS is socially isolating, lacks creativity or fun, and is for intelligent, white males, and we believe one contributing factor is the way CS is introduced, taught, and scaffolded.**

In this full paper, we present innovative pedagogical approaches to teach fundamental CS concepts, such as abstraction, representation, algorithms, and computation, to 6th grade students using manipulatives, which are physical objects that students interact with to teach or reinforce a concept. Teaching and learning using manipulatives has a long history in science and mathematics education, but the development of and research on manipulatives to teach CS concepts is less common. Through observational field notes from a 6th-grade classroom and interviews with the teacher, we discuss the affordances and drawbacks of the different approaches and manipulatives.

We found that using manipulatives led to increased student engagement and participation with the material and made teaching the material more exciting and engaging for the teacher. In addition, we found that the manipulatives provided a way for student misunderstandings and errors to be more apparent through tinkering with the physical objects, and the teacher was able to easily understand how to extend the activities and find ways to connect multiple CS concepts. However, we observed several drawbacks with different manipulatives and approaches for using the manipulatives that could be helpful for future changes and the development of new manipulatives. For example, the puzzle-like games we gave students to construct algorithms for tossing a coin to see who goes first in a game and playing Rock, Paper, Scissors were challenging for students to recreate a given algorithm that was not necessarily an algorithm they would have designed themselves.

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

We developed a curriculum for introducing computer science (CS) centered on identifying basic computing concepts in simple non-electronic games. We define CS as a discipline studying the foundation of computing and all related concepts, and we use non-programming computational thinking activities and examples to illustrate fundamental concepts in CS, such as *abstraction*, *representation*, *algorithm*, and *computation*.

One major goal of our approach is to debunk negative perceptions that CS is socially isolating, lacks creativity or fun, and is better suited for male students [1], [2]. We do this by demonstrating that learning basic concepts of CS is as fun, social, and gender-neutral as playing non-electronic games. Just as with games, computing is divided into the static algorithm (or instructions) and the dynamic computation (or game play). We believe choosing simple, physical games, such as tossing a coin to see who goes first or Tic-Tac-Toe, makes CS more widely accessible for students, teachers, and schools.

Our approach is similar to the approaches taken in CS For Fun (CS4FN), Teaching London Computing, and CTArcade [3]–[5], which also employ physical games to teach CS concepts, but it differs in a fundamental way. Instead of focusing on the strategy for winning games or playing against the computer, we use the instructions/rules for playing games without the use of a computer as a model to help students understand basic CS concepts before introducing them to programming.

In the 2022/2023 academic year, we began creating and piloting innovative practices using manipulatives, which are defined as physical objects that students interact with to teach or reinforce a concept [6], We were interested if students would engage more with the new practices using manipulatives instead of worksheets and slides to convey important CS concepts and rules. Specifically, we wanted to answer the following questions about our new innovative practices through classroom observations.

1) How do students engage differently with manipulatives versus worksheets/handwritten work?
2) What are the advantages and disadvantages to different manipulatives?

II. Motivation and Related Work

There is a long history of using physical manipulatives in K-12 mathematics [7]–[14]. As one article pointed out, many civilizations during ancient times understood and applied mathematics using manipulatives like counting boards and the abacus [8]. Within present-day society, manipulatives continue to prove to be very effective tools in early mathematics education, and for decades, the National Council of Teachers of Mathematics created efforts to increase the number of

manipulatives being implemented in K-12 schools nationwide [8]. Early education research showed that the use of manipulatives was important for developing students to have a broad range of materials they interacted with and manipulated in order to develop and construct mathematical knowledge [15]. One study showed that manipulatives allowed young students to discover essential skills through curiosity and the creative freedom to interact with each tactile piece, and the most valuable learning occurred when students actively constructed their own mathematical understanding through utilizing manipulatives [8].

While manipulatives are usually associated with early education, past research shows that the use of manipulatives in mathematics benefits students across any grade level and from any ability or culture [14], and research shows that the proper use of manipulatives leads to an increase in mathematical achievements, retention, problem-solving [8], [9], [14], specifically because manipulatives allow students to draw connections from concrete experiences to abstract reasoning. Research also shows that manipulatives greatly reduce students' anxiety toward math [10].

In computer science (CS), research on the use of manipulatives is much more sparse than in mathematics [16]–[22]. Even within the literature on manipulatives in CS, a couple publications are only abstracts [17], [18] and another couple are about the same manipulative created to simulate Microsoft's Kodu Game Lab, which is a virtual, block-based programming environment for young children [16], [19]. The idea of using physical objects to teach programming dates back to 1995 with AlgoBlock, which was a physical block-based programming environment that connected to a computer to execute [22], and these researchers and other CS researchers commonly refer to the use of physical objects to construct a program as tangible programming. Just as in mathematics, research shows that the use of manipulatives to teach computational concepts is beneficial to student learning [19], [21] and to students with visual impairments [20].

Another broader, more popular term for the use of non-electronics and physical objects to teach CS is "CS Unplugged", and the CS Unplugged approach [23] has been shown to broaden participation [24]. Several studies demonstrate that unplugged activities are a viable alternative to traditional programming activities for teaching introductory computational skills and algorithms [23], [25], [26], and supporting studies show the positive impacts unplugged activities have on students' perspectives of, engagement in, and motivation to study computer science [27]–[30]. However, many of the unplugged curricula introduce a new CS/CT concept using another concept that is also new or very technical, such as binary numbers, data structures for searching, or sorting, which may distract from learning the computing concept and be irrelevant or unmotivating to the students and teachers.

Games also have a long tradition as learning tools in education, especially in the form of gamification, which is the idea of representing a learning process as playing a game [31]. K-8 teachers use games to make learning certain concepts from different subjects more engaging and understandable, such as using Madlibs to teach students about nouns, adjectives, etc. in English, and studies show that playing physical board games teaches problem-solving skills and creativity [32]–[34] and improves math skills in elementary school students [35]. Additionally, playing games involve computational thinking activities [36]–[39], but research shows that simply playing games does not increase one's computational thinking skills, unless guided instruction about the skills is given [40].

Over the last 10 years, many new games to teach computational thinking and coding without digital technology have been invented, such as RaBit EscAPE (ages 6-10) [41], Cubetto (ages 3-6) [42], and Crabs and Turtles [43]. While a new game can be designed to capture specific computational concepts, using newly invented games presents many disadvantages. First, learning the rules of a new game can cause unnecessary extraneous cognitive load, which takes away cognitive resources from the learning of the computational concepts. Second, teachers and students may not identify with the game. Third, schools, students, and families might not have access to the new games. Last, the games usually do not have a curriculum with guided instruction for the teachers and students.

III. Curriculum Background

We developed a curriculum in collaboration with two middle school teachers that centers around the use of physical games to teach computing concepts. In the following section, we briefly summarize the research-practice partnership (RPP) and the curriculum for which we created the innovative practices.

A. A Research-Practice Partnership

Building on a well-established collaboration with a local dual-language immersion middle school, we developed an unplugged CS curriculum with two middle school mathematics teachers and the Assistant Principal. One teacher was a 6th-grade mathematics teacher with a BS in primary education, and the other teacher was an 8th-grade mathematics teacher with an MS in secondary education. Neither teacher had a background in CS or prior programming experience.

B. The Child's Play 6th Grade Curriculum

The goal of our curriculum was to introduce basic CS concepts, such as *representation*, *abstraction*, *algorithm*, and *computation* using physical games and the human as the computer. We wanted to eliminate the distraction of a machine and level the playing field between those who had and did not have prior programming experience.The resulting curriculum covered approximately 15 hours of instruction over 4.5 weeks in two units (see Table I).

Unit 1 contained four lessons motivating the concepts of representation (an entity that stands for something else) and abstraction (the process of omitting detail). It did this by correlating the categories/kinds of things (types) and the actual things (values) that were used in the game instructions to representations. Students played with using different representations in Tic-Tac-Toe to motivate the appropriate choice of a

TABLE I: Resulting Level 1 Curriculum.

Curriculum	CS Concepts	Games
Unit 1: Abstraction and Representation (1 week)	Abstraction Representation Kind of Thing/Type Thing/Value	*Tic-Tac-Toe*
Unit 2: Algorithm (3.5 weeks)	Algorithm Input/Output Placeholder/Variable Control Instruction Condition	*Coin Toss* *Rock-Paper-Scissors* *Nim* *Tic-Tac-Toe*

representation that was abstract enough to omit unnecessary detail but remain easily distinguishable, such Xs and Os versus pictures of team members or two different sides of a coin.

Unit 2 contained eight lessons which introduced the concept of algorithms and furthered the concept of representation. It did this by relating algorithms to game instructions and by addressing pros and cons of using different representations (or types of values) in a game, along with how the instructions/rules change with different representations. Unit 2 also introduced the idea of placeholders for values (variables) in algorithms and formal if-then-else constructs with conditions.

The curriculum used stories to describe the games of Tic-Tac-Toe, tossing a coin, and Nim to motivate the concepts of representation and abstraction. For example, the game of Tic-Tac-Toe was initially represented as an island map with eight treasure chests and a story about two teams trying to be the first to recover a treasure by three teams members in different sections of the island pulling on the same rope (see top of Fig. 1). We use the idea that a story describes how to play a game as motivation for an algorithm to describe how to play a game more precisely, since there are ambiguities and details that might be left out in a story.

Students were tasked with understanding the differences between the game and the story, as well as which aspects of the story were important for playing the game. Then, students were asked to think about changing the representations used in the game, such as using shapes instead of ropes and a line instead of a grid for the map, to motivate the need to talk about algorithms and how algorithms use representations (see bottom of Fig. 1). After writing and presenting algorithms for games, we advanced students' understanding of how to formally express algorithms using the idea of Parsons Problems [44] with pieces of an algorithm jumbled and an outline for sequencing the algorithm pieces (see Fig. 2).

Fig. 1: Story Representation of Tic-Tac-Toe

IV. Manipulatives to Teach CS

We believe that the most important attributes for a CS curriculum are to be *effective*, *widely engaging*, and *flexible*. Thus while employing teaching tools, any material should be easy to learn and support students in acquiring a basic understanding of fundamental CS concepts.

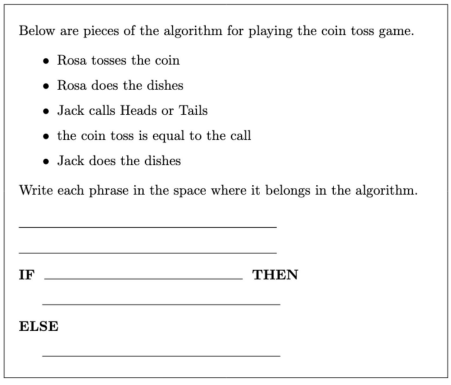
However, we also need these tools to engage everyone, not just a set of students who are already interested in CS, programming, or some other very specific application domain like robotics. Lastly, we want the game to be flexible and extensible to support a variety of difficulty levels and ways to deliver new concepts, in order to broaden participation and scaffold learning for new concepts.

Through multiple conversations with the 6th-grade teacher and classroom observations, we came to the conclusion that students were actively disengaging when provided a worksheet to enforce their learning, especially when the worksheet had a lot of writing. Therefore, developing ways to negate this disengagement was focal, leading us to create manipulatives as a teaching tool to engage students with the CS material and fundamental concepts in an alternative, more appealing way.

We created and deployed three innovative practices centered around utilizing manipulatives primarily within a 6th-grade classroom, but also in a first-year University studio section. All three developed practices provided a unique approach to bringing CS concepts into the learning experience for students. In the following subsections, we explain each innovative practice, provide our observations of the manipulative used in the classroom, and the teacher's perspective of using manipulatives.

A. Innovative Practice 1: Algorithmic Puzzle Pieces

From classroom observations and conversations prior to manipulative integration, we noticed students were reluctant to either fill out a worksheet, like in Fig. 2, or write out their own individual algorithms for tossing a coin or playing Rock-Paper-Scissors to see who goes first in a game, and we also noticed that many students had difficulty with coming up with


Below are pieces of the algorithm for playing the coin toss game.

- Rosa tosses the coin
- Rosa does the dishes
- Jack calls Heads or Tails
- the coin toss is equal to the call
- Jack does the dishes

Write each phrase in the space where it belongs in the algorithm.

IF _____ THEN

ELSE

Fig. 2: Worksheet with Algorithm as Parsons Problem

their own algorithm from scratch. Therefore, minimizing the reluctance to writing and providing an easier way to get started were two main ideas behind this innovative practice.

We expected the puzzle pieces to reduce the students from disengaging with the material, due to writing, and provide the scaffolding needed to get started. We wanted to provide the students with the algorithmic puzzle pieces to create different algorithm for games, such as Coin Toss and Rock-Paper-Scissors.

We created puzzle pieces (or blocks) with different colors and different shapes representing inputs, outputs, if-then-else statements, operators, and values (see Fig. 3). Utilizing the puzzle pieces while actively playing either game, students were tasked to create an algorithm representing the game and winning condition. This innovative practice emphasized the fundamental CS concepts of *algorithms* and *computation*.

1) Classroom Observations: During the first implementation of this practice within a 6th-grade classroom, we started by giving the students all the pieces without any scaffolding. We noticed that even though all the students were interacting with and interested in the puzzle pieces, they were very confused about how to start, specifically how to set up the algorithm. This prompted the teacher to provide a skeleton outline of the algorithm with the input first, the conditional being next and two different outputs depending if the condition is true or false (see Fig. 3). After the skeleton was provided, students seemed more engaged and willing to attempt solving the algorithm.

After that, all other algorithms using the puzzle pieces were started with input, conditionals, and outputs for an initial scaffolded state, such as the more complex puzzle for tossing a coin to see who goes first (see Fig. 4). With this scaffolded state, we noticed that students were able to recognize where the relational operators went in the conditions of the ifs, and they were able to more easily get started. However, even when students were very close to solving the puzzle accurately, they did not try to debug their solutions to see if they were right (see Fig. 5).

While we noticed that this helped the students not feel overwhelmed with expressing their algorithms, we also noticed that the use of the puzzle pieces seemed to limit their algorithmic creativity, since students were only able to build the algorithm in a way the puzzle pieces provided. Therefore, we provided students with many if/else and output pieces and allowed the students to write on the pieces to add more flexibility and creativity to using a die to see who goes first in a game (see Fig. 6). This did not seem to be as engaging due to writing.

2) Teacher Interview: We conducted an interview with the teacher of the 6th-grade class centered around manipulative integration into the classroom. During this interview, we asked "Across rotations, which activities for learning algorithms have seemed the most engaging for the students and why?", to which the teacher responded with the coin-toss version of the algorithmic puzzle pieces. The teacher stated students

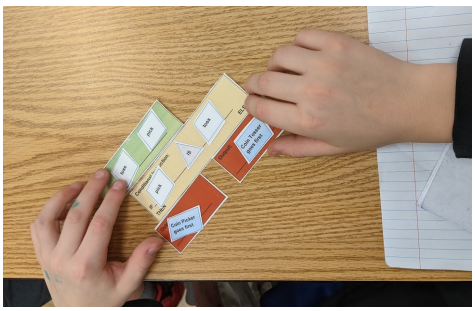


Fig. 3: Basic Puzzle Solution for Coin Toss

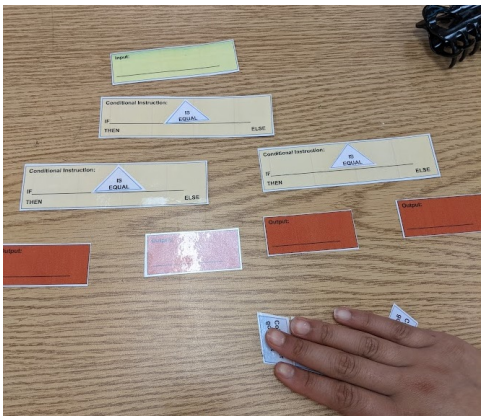


Fig. 4: More Complex Puzzle Outline for Coin Toss

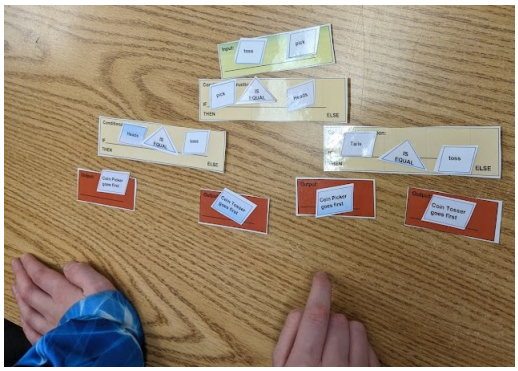


Fig. 5: Complex Puzzle Solution for Coin Toss

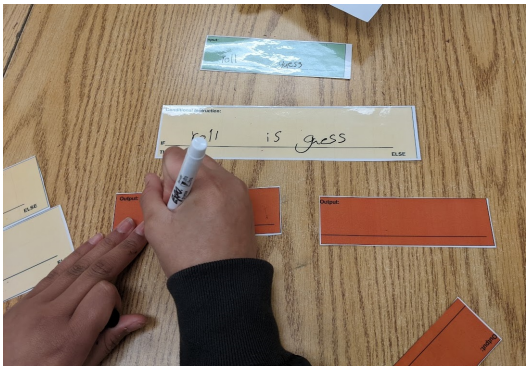


Fig. 6: Flexible Puzzle for Rolling a Die

enjoyed the challenge of solving an algorithm with scaffolding provided, as they felt like they were solving a real computer science problem. During this response, the teacher also stated they especially appreciated how well the puzzle pieces correlated coding-like instructions taught within the Level 2 curriculum.

B. Innovative Practice 2: The MoveIt! Game

We developed the MoveIt! game based on all the Level 1 concepts of *algorithms*, *computation*, *representation*, and *abstraction*. In the game, students are tasked to develop and execute a plan to move an arrangement of objects from a starting configuration to a target configuration through a set of verbal and written instructions (see Fig. 7). In groups of two, students are delegated as either the "mover", who moves the pieces on the board without seeing the target configuration, or the"instructor", who provides instructions to the mover without looking at the mover or board (see Fig. 8).

The instructor initializes the board to the starting state and is the only one who knows the target arrangement, and the mover moves the pebbles according to the move instructions provided by the instructor. In the initial version of the game, the instructions were spoken by the instructor, but we had to modify the game to use cards to verbalize each instruction in a crowded classroom while still motivating the necessity to write down instructions (see Fig. 9). Later, we expanded the game to include an activity where students created their own starting and target configurations that they could exchange with another group or try to have their own partner rearrange based on their instructions (see Fig. 10).

1) Classroom Observations: We observed the use of the MoveIt game with three different groups: in two 6th-grade classes (with 14 and 17 students, respectively) and in one first-year university class (with 105 students). Even though the game was designed for 6th-grade students, we wanted to see if the game was flexible enough to be used with first-year university students.

We observed that initiating the game with verbal instructions showed positive results for several reasons. Firstly, since many students showed reluctance and expressed their discontent towards writing, engaging them through verbalizing instructions significantly increased their participation. Secondly, verbalizing instructions proved beneficial for motivating the importance of documenting instructions. When providing verbal instructions, several students faced challenges in recalling the crane's position, leading them to place their finger on the visual representation of the transformation card. Throughout the game, the students remained actively engaged, with one group even requesting to move to a quieter area for better

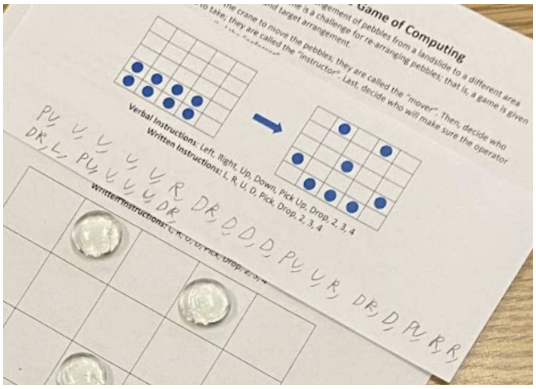


Fig. 7: Initial and Target Configurations for MoveIt!

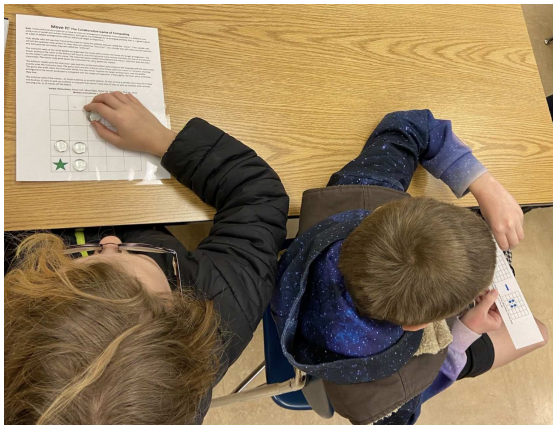


Fig. 8: Students playing MoveIt! game

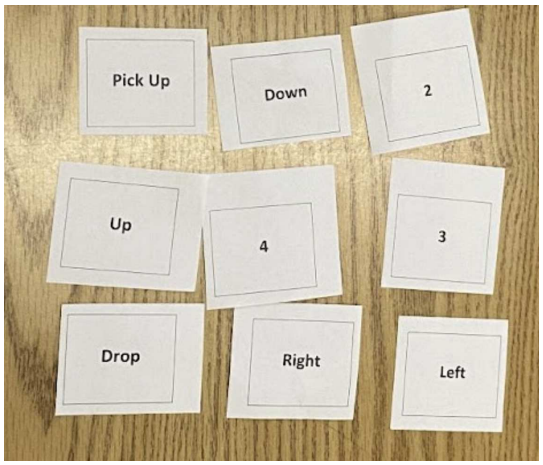


Fig. 9: Card-based instructions for MoveIt!

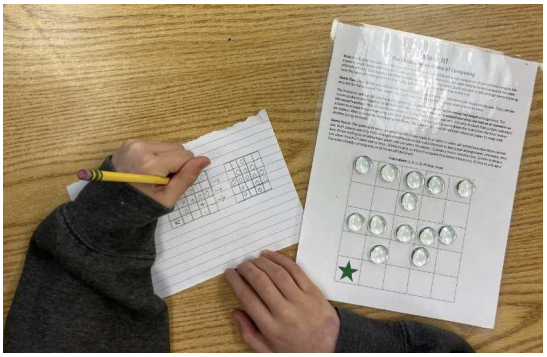


Fig. 10: Student making their own MoveIt! states

concentration.

It was interesting to note that many students became invested in the game, demonstrating a willingness to attempt multiple times until they achieved the final state. Afterwards, the students were asked to write down the instructions they used. Initially, they hesitated to write instructions until they were informed that they only needed to write a letter or two for each instruction, which sparked their enthusiasm for writing. Subsequently, the groups exchanged solutions and checked the solutions of other groups. It was observed that students enjoyed having their solutions validated for correctness and also took pleasure in reviewing and identifying errors in others' solutions. They also expressed interest in comparing their own solutions with alternative ones. Some students were excited and surprised to discover identical solutions, while others were astonished by the differences in the number of instructions between their solutions and others.

However, we did encounter certain aspects that were less successful. Firstly, verbalizing instructions proved challenging in a room with 16 students, leading to a chaotic environment that hindered some students' ability to focus. Secondly, some students immediately requested repetition and inquired if they could provide a numerical value with the instruction to perform it multiple times, such as moving right three times. Third, some students quickly grew bored if the game did not increase in difficulty. Additionally, there were students who remained disinterested as they either did not want to replay the game or simply did not enjoy it from the beginning.

Next, we presented an iteration where students had to use instruction cards instead of verbalizing the instructions. Prior to introducing this iteration of the game, we sought the students' opinion on how the game related to Computer Science. A majority of students responded by likening the instructor to a programmer and the mover to a computer. They recognized that the set of instructions constituted an algorithm and discussed how the computer follows instructions precisely, highlighting the importance of a correct algorithm to obtain accurate results. Interestingly, one student referred to the instructions as input and the final transformation state as output. Additionally, we presented them with a less intuitive configuration involving 8 pebbles, which immediately piqued their interest in the game. We observed that the use of cards for verbalizing instructions worked effectively. Both the students and the teacher appreciated the idea of holding up a card to indicate the instruction the mover was supposed to execute.

The students particularly enjoyed having numerical values for moving more than one square at a time. Initially, we anticipated that they would either hold the instruction cards above their heads or pass them to the mover, who would then return them. Contrary to our expectations, we noticed students utilizing the cards in two different ways: 1) some preferred to hold up a card pair displaying both the number and instruction simultaneously, especially when repeating instructions; 2) some students opted to point to the instruction on the table or slide it over to indicate which instruction should be executed, instead of holding it up for the mover to see (as shown in Fig. 11). Interestingly, when asked to write down the instructions for this particular iteration, many groups wrote them as separate movements without incorporating the direction and number language. For instance, they would write "R, R, R, L, D" instead of "R3, L, D" or "3R, L, D".

We presented one group of 6th graders with the option for students to create their own starting and ending states (see Fig. 10). The students showed great enthusiasm and eagerly agreed to the idea. Throughout the activity, all the students remained fully engaged. After a while of drawing their own states, we asked the students to test either their own or another group's transition states. We noticed that most students initially chose to test their own states and then proceeded to exchange and test the states of other teams. However, many students forgot to label or draw transition arrows between the start and end states. This led to some confusion when they exchanged states and the paper was turned upside down. Interestingly, throughout the activity, we observed that all-female groups were particularly engaged and actively involved. Overall, this iteration of the game effectively utilized the game's flexibility to enhance participation and interaction. It encouraged students to think about various possible modifications they could make to the states or the board.

2) Teacher Interview: During the interview with the 6th-grade teacher, we asked them "Have you noticed a difference in student *engagement* based on which activities are used to teach re-enforce learning (ex: teaching a topic through worksheets vs. manipulatives)? ", to which they responded the level of engagement has significantly increased at the start when using manipulatives. The teacher then stated the MoveIt! game had continually been a very engaging and fun activity for students across rotations, specifically the game did

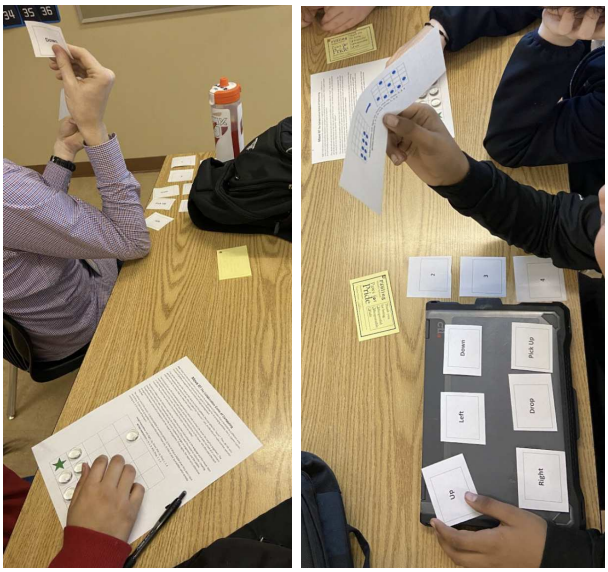


Fig. 11: (Left to Right) (a) Student holding up the card-based instructions (b) Student pointing toward the card-based instructions placed on the table

a great job at drawing students in at the beginning. They stated the objective of getting from the starting to the end state of MoveIt! wasn't too overwhelming for students to grasp, which made them very excited and engaged to do more examples.

C. Innovative Practice 3: Physical Floor Game Play

Physical floor games, like Twister and Charades, have been a popular and fun way to engage a larger crowd than a game that sits on a table, and more recently, life-sized versions of tabletop games have gained popularity in parks, restaurants, and now in the classroom. We believed that a physical floor representations of a games would engage a larger audience, such as the whole classroom, for conveying concepts and introducing a game and its rules, especially since the written instructions did not seem to engage students.

We hoped that floor games would provide the ability to engage students through actively moving throughout the classroom. We created floor-sized versions of the treasure hunt game presented in Fig. 1 and the MoveIt game to engage the classroom in game instructions and reinforce the fundamental CS concepts of *abstraction*, *representation*, *algorithms*, and *computation* without lecturing.

We created a large, floor-sized version of the Treasure Hunt game story representing Tic-Tac-Toe within a 6th-grade classroom. The initial board was set up having standard tic-tac-toe win conditions represented through ropes across the map (see Fig. 12). We divided the classroom into two teams, allowing them to play the first game state for two rounds and making one student represent a piece for a team. Then, we modified the win conditions to be a different layout of the ropes, emphasizing the concepts of *representation* to the students, as the win condition represents the treasure game and not Tic-Tac-Toe (see Fig. 13). Students played this version for another two rounds, and then, they followed this activity with a class discussion led by the teacher to further emphasize the concepts of *representation* and *abstraction*.

We introduced the MoveIt! game to students by incorporating a larger 3x3 version, where we requested volunteers to physically act out the game as a demonstration. Initially, as the game was being set up and introduced, several students expressed audible confusion, asking questions like "What's the point of this?" or making comments such as "This seems pointless." However, after the 6th-grade teacher selected two students to play a round, the class gathered around and grasped how the game worked and its underlying meaning. This floor version of MoveIt! effectively highlighted key CS concepts such as *algorithms*, *computation*, and *representation*. One student was assigned the role of computing and providing instructions (or an algorithm) to another student, guiding them from a given starting state to an end state (Fig. 14). Subsequently, the entire class participated in playing the MoveIt! game on the game board version, working in pairs.

1) Classroom Observations: We observed that the physical demonstration captured the attention of a larger number of students and sparked their enthusiasm to try the game on the game board version. Throughout the class period, all the students remained actively engaged and willingly sought different configuration states once they had completed their current one. The teacher expressed great satisfaction with the high level of class involvement and believed that the physical introduction to the game played a significant role in fostering such engagement, as well as facilitating a quick understanding of the concepts.

For both iterations of the physical board games, students were actively engaged and wanted to do as many examples in front of the class as possible. We noticed that providing the

Fig. 13: Floor Game of Tic-Tac-Toe

Fig. 12: Floor Game of Tic-Tac-Toe

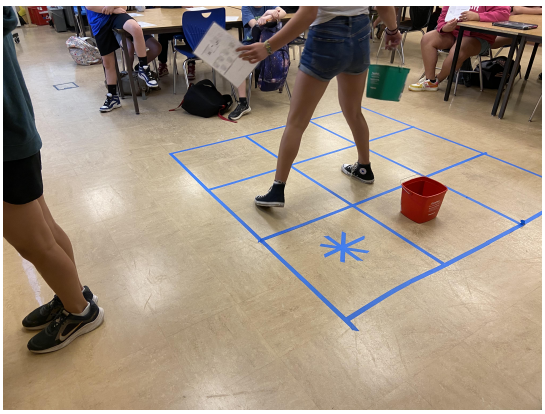


Fig. 14: MoveIt! Floor Version

ability for 6th-grade students to learn CS fundamentals while moving around in order to let some energy out was a key takeaway throughout the implementation of this practice.

Specifically for the large-size Treasure hunt game, all the students were actively engaged with each other and the material, as they were collaboratively discussing as a team which moves they wanted each other to execute in order to satisfy either win condition. This was especially apparent after the switch in the representation of win conditions occurred, as it caused all the students within a team to collaborate with each other and share ideas on which moves they thought were best based on which way of winning they were going for. The more the students were able to share their thought process, the further engaged they seemed to be with the material.

2) Teacher Interview: The final question we asked during our interview with the 6th-grade teacher was, "What other kinds of manipulatives or activities would you like to see more or less of and why?", to which they responded they wanted more manipulatives that would engage the students through making them the pieces, just as they did for the Treasure Hunt game. The teacher emphasized that their students were very actively engaged while playing the floor-size Treasure Hunt game, to which they followed up this point by stating in general 6th grader students will seize any opportunity to move around out of their chairs.

V. Evaluation of Manipulatives

A. Perspective of the Teacher

During an interview with the 6th-grade teacher, we first asked 1) "Which type of materials, either worksheets or manipulatives, do you prefer using to teach Computer Science concepts such as abstraction, representation, and algorithms? Why?" to which they responded they preferred using manipulatives for 6th-graders in particular. They then followed their response by stating manipulatives take the cognitive demand of grammar off the students by removing the difficulty of wording and sentences, along with also mentioning that manipulatives have provided students with a larger level of play, specifically noting more students are engaged when manipulatives are utilized to teach and enforce concepts.

We then asked the teacher 2) "What are students' reactions when assigned tasks through *worksheets*?", to which they responded they have noticed students typically feel as worksheets can be quite difficult due to the limitations 6th-graders have on being able to articulate their ideas through grammar. This question was followed up by asking 3) "What are students' reactions when assigned tasks through *manipulatives*?", to which the teacher stated there is an easier way for the students to bridge connections between ideas and manipulatives also provide a smaller cognitive load to initiate with.

B. Broaden Engagement

Almost all 6th-grade and first-year university students were enthusiastic about playing the game and exchanging their plans/algorithms with other groups. Compared to other unplugged activities the students had used previously, the activities centered around using and discussing the MoveIt game were significantly more engaging, in particular, when previous activities involved writing and analyzing algorithms, even if these only involved minute tasks.

Especially at the 6th-grade level, we observed that students engage more with activities that only require a limited amount of writing, yet allow for ownership of the solution. In fact, many students in both 6th-grade classrooms sighed in relief and blurted out " Yay!", when they were told that they would begin by using cards or verbalizing the instructions before writing them. Not only was this more engaging for the students to begin by verbalizing their instructions, but it motivated the need to write down the instructions in the algorithm for debugging purposes and sharing with others.

C. Flexibility

We wanted the game to be easily extendable vertically with increasing difficulty and horizontally to introduce new CS concepts incrementally, but we also found that the game offered a variety of ways of playing that can engage different groups of students. For example, we had students design and draw their own game challenge with a start and target configuration to exchange between groups, and we observed that this was very engaging for an overwhelming number of female students. We envision extending this creativity to designing new game boards for interacting with the game, such as using pictures on cells to limit which operations are allowed or disallowed (as discussed at the end of Section **??**). Additionally, the game was designed to be collaborative and non-competitive, but we found that many students got excited when asked to come up with the most efficient algorithm for a common challenge the teacher drew on the board.

We observed that the flexibility of the game activities allowed us to engage a broader group of students throughout a single class period, and the various levels of difficulty that could be incorporated into each activity provided another avenue for engaging those who bore easily, in addition to scaffolded learning. We also believe the broad engagement ranging from 6th-grade students to first-year university students shows the game's flexibility for introducing basic CS concepts across a wide range of age groups.

Acknowledgment

This work was supported by the National Science Foundation under the grant DRL-1923628.

References

[1] A. Gokhale and K. Machina, "Online learning communities to recruit and retain students in information technology programs," 2010.

[2] B. Cheryan, Drury and M. Vichayapai, "Enduring influence of stereotypical computer science role models on women's academic aspirations," 2013.

[3] CS For Fun: Queen Mary, University of London, "Welcome to cs4fn : the fun side of computer science," http://www.cs4fn.org/, 2011, accessed: 2021-01-07.

[4] ——, "Teaching london computing: A resource hub from cas london & cs4fn," https://teachinglondoncomputing.org/, 2015, accessed: 2021-01-07.

[5] T. Y. Lee, M. L. Mauriello, J. Ahn, and B. B. Bederson, "Ctarcade: Computational thinking with games in school age children," *Int. Journal of Child-Computer Interaction*, vol. 2, no. 1, pp. 26–33, 2014.

[6] Merriam-Webster, "Manipulatives," https://www.merriam-webster.com/dictionary/manipulatives, accessed: 2023-06-02.

[7] S. K. Stigberg, H. Stigberg, and M. Maugesten, "Making manipulatives for mathematics education," in *6th FabLearn Europe / MakeEd Conference 2022*, ser. FabLearn Europe / MakeEd 2022. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi-org.oregonstate.idm.oclc.org/10.1145/3535227.3535228

[8] M. Boggan, S. Harper, and A. Whitmire, "Using manipulatives to teach elementary mathematics." *Journal of Instructional Pedagogies*, vol. 3, 2010. [Online]. Available: https://files.eric.ed.gov/fulltext/EJ1096945.pdf

[9] C. A. Kelly, "Using manipulatives in mathematical problem solving: A performance-based analysis," *The mathematics enthusiast*, vol. 3, no. 2, pp. 184–193, 2006.

[10] M. Cain-Caston, "Manipulative queen," *Journal of instructional psychology*, vol. 23, no. 4, p. 270, 1996.

[11] C. Kamii, B. A. Lewis, and L. Kirkland, "Manipulatives: when are they useful?" *The Journal of Mathematical Behavior*, vol. 20, no. 1, pp. 21–31, 2001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0732312301000591

[12] P. S. Moyer, "Are we having fun yet? how teachers use manipulatives to teach mathematics," *Educational Studies in mathematics*, vol. 47, no. 2, pp. 175–197, 2001.

[13] K. J. Carbonneau, S. C. Marley, and J. P. Selig, "A meta-analysis of the efficacy of teaching mathematics with concrete manipulatives." *Journal of educational psychology*, vol. 105, no. 2, p. 380, 2013.

[14] D. H. Clements and S. McMillen, "Rethinking "concrete" manipulatives," *Teaching children mathematics*, vol. 2, no. 5, pp. 270–279, 1996.

[15] C. Seefeldt and B. A. Wasik, *Early education: three-, four-, and five-year-olds go to School*. Recording for Blind & Dyslexic, 2007.

[16] D. S. Touretzky, "Teaching kodu with physical manipulatives," *ACM Inroads*, vol. 5, no. 4, p. 44–51, dec 2014. [Online]. Available: https://doi-org.oregonstate.idm.oclc.org/10.1145/2684721.2684732

[17] M. Goadrich, "Cs2mulch: Physical manipulatives for teaching advanced data structures," in *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 1374. [Online]. Available: https://doi-org.oregonstate.idm.oclc.org/10.1145/3408877.3439533

[18] S. Ludi and S. Kurkovsky, "Using tangible manipulatives for hands-on activities in undergraduate computer science classes (abstract only)," in *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, ser. SIGCSE '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 727. [Online]. Available: https://doi-org.oregonstate.idm.oclc.org/10.1145/3017680.3022349

[19] A. Aggarwal, C. Gardner-McCune, and D. S. Touretzky, "Evaluating the effect of using physical manipulatives to foster computational thinking in elementary school," in *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, ser. SIGCSE '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 9–14. [Online]. Available: https://doi-org.oregonstate.idm.oclc.org/10.1145/3017680.3017791

[20] A. M. Stefik, C. Hundhausen, and D. Smith, "On the design of an educational infrastructure for the blind and visually impaired in computer science," in *Proceedings of the 42nd ACM technical symposium on Computer science education*, 2011, pp. 571–576.

[21] T. J. Ramabu, I. Sanders, and M. Schoeman, "Teaching and learning cs1 with an assist of manipulatives," in *2021 IST-Africa Conference (IST-Africa)*, 2021, pp. 1–8.

[22] H. Suzuki and H. Kato, "Interaction-level support for collaborative learning: Algoblock - an open programming language," 01 1995, pp. 349–355.

[23] T. Bell, I. H. Witten, and M. Fellows, *CS Unplugged. An Enrichment and Extension Programme for Primary-Aged Students*, 2015, https://www.freetechbooks.com/cs-unplugged-an-enrichment-and-extension-programme-for-primary-aged-students-t1323.html.

[24] T. J. Cortina, "Reaching a Broader Population of Students Through "Unplugged" Activities," *Communications of the ACM*, vol. 58, no. 3, pp. 25–27, 2015.

[25] J. Parham-Mocello, S. Ernst, M. Erwig, E. Dominguez, and L. Shellhammer, "Story Programming: Explaining Computer Science Before Coding," in *ACM SIGCSE Symp. on Computer Science Education*, 2019, pp. 379–385.

[26] R. Thies and J. Vahrenhold, "On Plugging "Unplugged" Into CS Classes," in *ACM SIGCSE Symp. on Computer Science Education*, 2013, pp. 365–370.

[27] T. Bell, P. Curzon, Q. I. Cutts, V. Dagiene, and B. Haberman, "Overcoming Obstacles to CS Education by Using Non-Programming Outreach Programmes," in *Int. Conf. on Informatics in Schools*, ser. LNCS 7013, 2011, pp. 71–81.

[28] Q. I. Cutts, M. I. Brown, L. Kemp, and C. Matheson, "Enthusing and informing potential computer science students and their teachers," in *ACM SIGCSE Symp. on Computer Science Education*, 2007, pp. 196–200.

[29] R. Taub, M. Ben-Ari, and M. Armoni, "The Effect of CS Unplugged on Middle-School Students' Views of CS," in *ACM SIGCSE Symp. on Computer Science Education*, 2009, pp. 99–103.

[30] C. Mano, V. Allan, and D. Cooley, "Effective In-Class Activities for Middle School Outreach Programs," in *Annual Conf. on Frontiers in Education*, 2010, pp. F2E–1–F2E–6.

[31] K. M. Kapp, *The Gamification of Learning and Instruction: Game-Based Methods and Strategies for Training and Education*. San Francisco, CA: Pfeiffer, 2012.

[32] L. A. Sharp, "Stealth Learning: Unexpected Learning Opportunities Through Games," *Journal of Instructional Research*, vol. 1, pp. 42–48, 2012.

[33] C. Harris, "Meet the New School Board: Board Games Are Back–And They're Exactly What Your Curriculum Needs," *School Library Journal*, vol. 55, no. 5, pp. 24–26, 2009.

[34] C. Ragatz and Z. Ragatz, "Tabletop Games in a Digital World," *Parenting for High Potential*, vol. 7, no. 7, pp. 16–19, 2018.

[35] S. Cavanagh, "Playing Games in Class Helps Students Grasp Math," *Education Digest: Essential Readings Condensed for Quick Review*, vol. 74, no. 3, pp. 43–46, 2008.

[36] M. Berland and S. Duncan, "Computational Thinking in the Wild: Uncovering Complex Collaborative Thinking through Gameplay," *Educational Technology*, vol. 56, no. 3, pp. 29–35, 2016.

[37] M. Berland and V. R. Lee, "Collaborative Strategic Board Games as a Site for Distributed Computational Thinking," *Int. Journal of Game-Based Learning*, vol. 1, no. 2, pp. 65–81, 2011.

[38] N. R. Holbert and U. Wilensky, "Racing games for exploring kinematics: a computational thinking approach," in *7th Int. Conf. on Games + Learning + Society*, 2011, pp. 109–118.

[39] C. Kazimoglu, M. Kiernan, L. Bacon, and L. MacKinnon, "Learning programming at the computational thinking level via digital game-play," *Procedia Computer Science*, vol. 9, pp. 522–531, 2012.

[40] T. Y. Lee, M. L. Mauriello, J. Ingraham, A. Sopan, J. Ahn, and B. B. Bederson, "CTArcade: Learning Computational Thinking Thile Training Virtual Characters Through Game Play," in *Human Factors in Computing Systems*, 2012, pp. 2309–2314.

[41] P. Apostolellis, M. Stewart, C. Frisina, and D. Kafura, "RaBit EscAPE: A Board Game for Computational Thinking." in *Conf. on Interaction Design and Children*, 2014, pp. 349–352.

[42] Primo, "Cubetto: Screenless coding toy for girls and boys aged 3-6," 2018, https://www.primotoys.com.

[43] K. Tsarava, K. Moeller, and M. Ninaus, "Training Computational Thinking Through Board Games: The case of Crabs Turtles," *Int. Journal of Serious Games*, vol. 5, no. 2, pp. 25–44, 2018.

[44] D. Parsons and P. Haden, "Parson's Programming Puzzles: A Fun and Effective Learning Tool for First Programming Courses." vol. 52, 2006, pp. 157–163.