Learning Programming from Tutorials and Code Puzzles: Children's Perceptions of Value

Kyle J. Harms, Evan Balzuweit, Jason Chen, Caitlin Kelleher
Department of Computer Science & Engineering
Washington University in St. Louis
St. Louis, Missouri, United States
{kyle.harms, ebalzuweit, chenjy, ckelleher}@wustl.edu

***Abstract*— Tutorials and code puzzles are commonly used in today's novice programming environments to introduce computer programming to children. While research has explored the effectiveness of each instructional format at teaching different kinds of information independently, little work has explored learners' perceptions of value in each or the strategic decisions users make around the instructional format when learning to program. We present a study in which learners selected from a set of tutorials and puzzles with an identical set of programming content. We explore the reasoning behind their choices and the potential implications for the learning support available in future programming environments.**

Keywords—novice programming; code puzzles; programming completion problems; Parsons problems; tutorials

I. INTRODUCTION

Children's programming environments predominantly use two instructional formats to support learning: tutorials that present step-by-step instructions for learners to follow [1], [2] and code puzzles that provide learners with a set of code elements and a specific challenge [3]–[5]. Prior research suggests that novices show more evidence of programming knowledge when learning via puzzles compared to tutorials [6].

However, the same study found a surprising mismatch between learning and motivation during the formative and summative phases of the study [6]. The researchers observed stronger motivation for puzzles during the formative, but found no statistically significant difference in motivation during the summative evaluation [6]. These findings suggest that children may perceive value in both tutorials and puzzles.

In this paper, we build upon the prior study's results by conducting a qualitative study with an explicit analysis of the motives and decisions middle school children make when choosing a particular instructional format while learning to program. Over the course of the study, we asked participants to select and complete six programs using their choice of tutorials and puzzles. Throughout the study, we interviewed participants about their experiences and intentions for choosing between the instructional formats. From this study, we hoped to learn about the 1) circumstances under which novices choose to use tutorials or puzzles, and 2) novices' perceptions of value for each of the instructional formats. The results suggest that users' own personal interests play a large role in their format selection and that the range of scaffolding provided by the formats enables learners to better follow and manage their own goals.

II. RELATED WORK

We consider two areas of related work: 1) adult students and their perceptions of learning, and 2) the effectiveness of learning programming independently, especially for children.

A. Learners' Perceptions

Learners' perceptions and choices influence the way they navigate learning materials. For example, independent learners often choose not to read directions and tend to avoid repetitive practice exercises when following instructional materials [7], [8]. Instead, learners prefer to begin working on their own tasks immediately and use instructional materials as a reference only when needed [8], [9]. Traditional-style instructional manuals can be especially frustrating for these task-oriented learners [9]. Aligning instructional materials more appropriately to learners' preferences improves their learning efficacy [9].

An alternative to aligning materials to learners' preferences is giving learners a choice in their learning and enabling them to make their own learning decisions. In the context of programming, students liked the ability to choose the instructional format (e.g. self-guided lab, tutorial) of supplemental classroom material [10], [11]. Students also felt that choosing their learning materials, like programming languages [12], projects [13], and online discussion format [14] helped them learn better. Similarly, students reported that compared to traditional classroom instruction, informal resources (e.g. online courses) enabled them to choose materials that were appropriate for their level, which they perceived as improving their ability to learn [15]. However, beyond students' perceptions, these researchers present little evidence that demonstrates that choice actually helps students learn better. Furthermore, we know little about how these results relate to independent learners, especially children, when trying to learn programming outside of the classroom experience.

B. Independent Learning Support

Novice programmers, and in particular children, who seek to learn programming independently can choose from several different instructional formats. Broadly, these formats fall into four categories: learning from online courses, examples, tutorials, and puzzle-like systems.

Massive open online courses (MOOCs) like Codecademy [16], Khan Academy [17], and edX [18], are an increasingly popular way for novices to learn programming. However, online courses typically have high attrition rates, which can be

978-1-5090-0252-8/16/$31.00 ©2016 IEEE

attributed to an inconsistency between learners' goals and the style of instructions [19], [20].

Instead, task-oriented learners can leverage example code and tutorials when seeking instruction [21]. Tutorons annotate online example code with explanation to help learners understand the code [22]. Online tutorials that incorporate example code with live program replay, like Online Python Tutor [23] and Codepourri [24] may also help learners understand code. Further, tutorials have been shown to help children learn new programming concepts [25]. Yet, it is unclear what value novices perceive in these tools.

An increasingly popular alternative is puzzle-like support for learning programming independently. Probably the most well-known puzzle-like system is the Hour of Code [3]. The Hour of Code is specifically targeted at enabling children to learn programming independently. Another popular type of code puzzle, the Parsons problem [26], have also been used in independent contexts [27], but they are more commonly used in classrooms. Puzzle-like systems generally set an explicit goal that users accomplish through programming. Frequently, this involves navigating an object through a grid [5], [28], [29]. Puzzle-like systems have been shown to be effective tools to enable children to learn programming independently [6], [28]. However, little is known about children's perceptions of instructional formats and how those perceptions factor into their decisions on when to use a particular format.

III. Evaluation

The goal of our exploratory study was to identify the factors that influenced participants' decisions about learning content and instructional format. We asked participants to complete six instructional tasks. For each task, participants selected both the program (i.e. animation) they wanted to learn to create and the instructional format (i.e. tutorial or puzzle). We interviewed participants throughout the study to gather information about the choices they made when choosing an instructional format.

A. Instructional Formats

For this study, we decided to use the novice programming environment, Looking Glass [30]. Looking Glass is a blocks-based programming environment designed to motivate middle school children to program by authoring 3D animations.

1) Tutorials

The tutorials presented users with a sequence of steps. For each step, the tutorial provides short textual instructions as well as a looping video demonstrating how to perform that step as shown in Fig. 1-E. Prior research comparing the effectiveness of tutorials and puzzles used this format [6], which was originally based on tutorials implemented within Scratch [1].

2) Puzzles

When completing puzzles, participants had a complete set of programming statements that comprise the solution (Fig. 1-B). To complete a puzzle, participants dragged the programming statements into the editor (Fig. 1-C). Participants could view the output of their current program, as well as the target output. This style of puzzle is inspired by the problem completion effect [31], in which high school students who learned programming by completing incomplete programs showed greater learning than those who constructed the same programs from scratch. We note that the puzzle mechanics are similar to Parsons problems [26], however these puzzles provide additional support intended to facilitate independent learning [6].

B. Participants

We recruited 30 participants between the ages of 10 and 15 years (14 female, 16 male; age: $M = 11.2$, $SD = 1.3$) from the Academy of Science of St. Louis mailing list. The Academy of Science is a not-for-profit organization in the St. Louis metropolitan area dedicated to science outreach. Participants self-reported their prior programming experience. Eighteen participants had less than three hours of prior programming experience, whereas twelve participants had at least three hours of prior programming experience. We compensated participants with a $10 gift card.

Fig. 1. Participants are given a list of 14 instructional tasks (A). Participants choose whether to complete an instructional task (A) as a tutorial or a puzzle. When completing a puzzle, participants drag unused statements (B) and reassemble them into the correct order (C). For the tutorial, participants work in the Looking Glass programming environment (D) and follow the video and textual instructions in the tutorial pane (E).

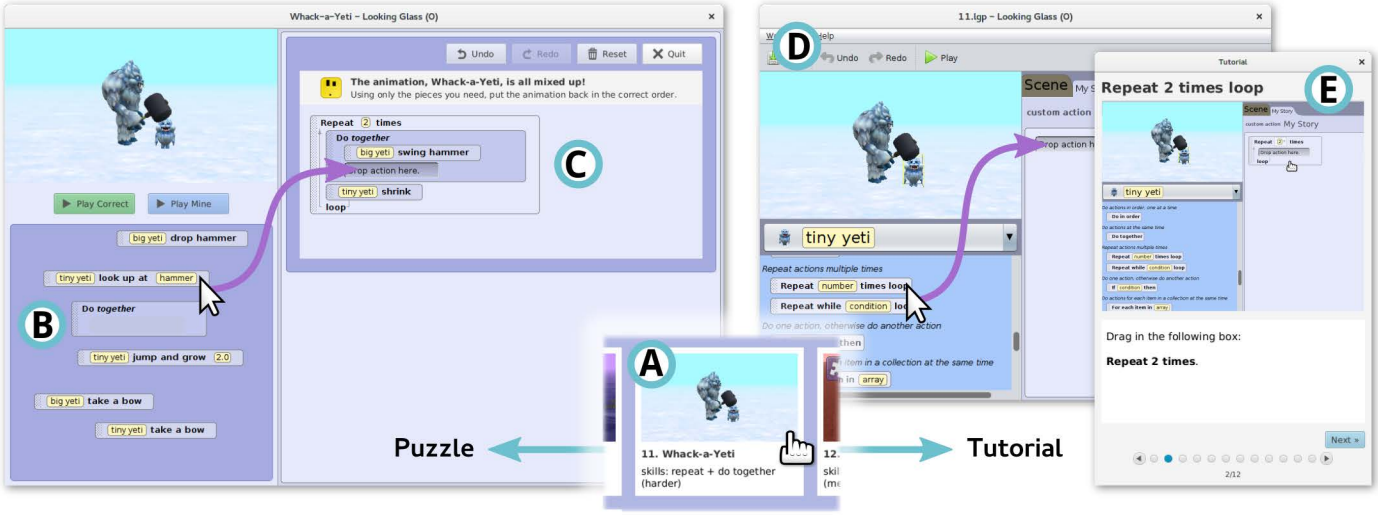
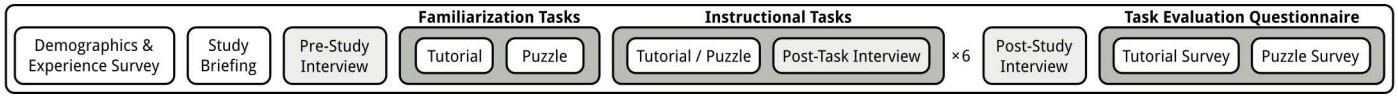


Fig. 2. From left to right, the order of the study's activities.



C. Methods

We conducted our study through 30 individual, two-hour sessions. See Fig. 2 for an overview of the study procedure. In the first part of the study, we familiarized participants with the study procedure. When participants arrived, we asked them to complete a demographic and programming experience survey. We then briefed participants on the study format. During the briefing we also primed participants by requesting that they focus on the goal of increasing their programming skills while working their way through the study. During our pilot testing we discovered that many participants made decisions based on whether they found an animation interesting. We have chosen to include this priming in our study in order to gather perceptions of value beyond compelling animations.

After the briefing, we conducted the pre-study interview and then asked participants to complete two familiarization tasks. The familiarization tasks were designed to introduce participants to the experience of using a tutorial and puzzle. We randomly assigned participants to complete a tutorial or puzzle first. During the familiarization tasks, we informed each participant that we could provide assistance and answer questions. We also informed participants that once they began the instructional task part of the study, we would be unable to provide any assistance.

For the remainder of the study, we asked participants to complete six instructional tasks. Participants selected from among fourteen animations. For each animation, participants decided whether they would use a tutorial or a puzzle to learn how to create that animation (see Fig. 1-A). There was no time limit for the instructional tasks. Upon completion of each task, a researcher interviewed the participant about that task. We allowed participants to stop working on a task at any time. If a participant choose to stop before completing a task, we interviewed them about their decision to quit that task.

After completing the instructional tasks, we interviewed participants about their entire study experience in the post-study interview. We then asked them to complete two motivation inventories, one for tutorials and one for puzzles. Participants decided which motivation inventory they completed first.

D. Materials

Our study included familiarization tasks, instructional tasks, semi-structured interviews, and surveys. We iteratively developed and refined these materials through a pilot study with 29 participants (9 female, 20 male; age: $M = 11.2$, $SD = 1.8$).

1) Familiarization Tasks

The familiarization tasks each consisted of a simple, sequential animation with four statements. Each animation could be completed as a tutorial or as a puzzle.

2) Instructional Tasks

We developed fourteen instructional tasks. An instructional task is a complete program (i.e. animation) that participants can choose to complete as a tutorial or as a puzzle. The animations spanned a range of difficulties and programming concepts. Programming constructs included sequential execution, repeat, do together, repeat + do together, and do together + do in order.

Additionally, we developed an interface that allows participants to select an instructional task (Fig. 1-A). The interface shows all available animations, ordered and numbered by difficulty (1 to 14). We established the difficulty ordering based on pilot testing. For each task, the interface contains a thumbnail of the animation, the animation's title, the ranked difficulty (1-14), a difficulty label (easier, medium, and harder), and the programming constructs demonstrated in the program. Participants could also click the thumbnail to view the full animation for each task.

3) Semi-Structured Interviews

We developed questions for four semi-structured interviews: a *pre-study interview*, a *post-task interview*, an *early termination interview*, and a *post-study interview*. Each interview contained questions that sought extended responses and one-word responses. We captured participants' responses in audio recordings. The interview questions were informed and refined by the themes that emerged through our pilot study.

The *pre-study interview* contained five questions about participants' prior programming experience. Additionally, we asked participants to rank their programming skill using a Likert scale from novice (1) to expert (5) [32]. For all Likert scales in the study, we presented both numerical and ordinal values.

In the *post-task interview*, we asked participants a series of 17 questions about their decision, the quality and value of their experience with the task, and what they planned to do next. Three questions asked participants to rank their expected difficulty for the task, the actual difficulty for the task, and how difficult the task would have been if they had used the other instructional format using a Likert scale ranging from extremely easy (1) to extremely difficult (9) [33].

The *early termination interview* consisted of a subset of the questions from the post-task interview. These questions focused on perceived difficulty and their plans for what to work on next. We included an additional question asking participants about their reasons for terminating the task early.

In the *post-study interview*, we asked participants twenty questions exploring their experiences during the study, their preferences regarding instructional task format, difficulties with tasks, and comparative questions about the value between the instructional formats. We also asked participants to re-rate their programming experience from novice (1) to expert (5).

4) Surveys

Our study included two surveys. The first survey, completed before starting the study, asked participants to self-report their age, gender, schooling and prior programming experience. For the second survey, completed at the end of the study, we used the Intrinsic Motivation Inventory's Task Evaluation Questionnaire (TEQ) [34]. The TEQ is a 22-item Likert survey with four subscales: interest/enjoyment, perceived competence,

perceived choice, and pressure/tension. Participants rated their agreement with individual statements from not true at all (1) to very true (7). We asked participants to complete two TEQ surveys, one for tutorials and one for puzzles.

IV. Analysis

We analyzed the reliability of the TEQ motivation subscales and performed a qualitative coding on the interview responses.

A. TEQ Surveys

We determined the reliability of the TEQ motivation subscales by combining the results from the tutorial and puzzle TEQ surveys and computing Cronbach's alpha for each subscale. Three of the subscales were reliable ($\alpha > .70$): interest/enjoyment ($\alpha = .92$), perceived choice ($\alpha = .77$), and pressure/tension ($\alpha = .73$). We have not reported the results for perceived competence subscale ($\alpha = .68$).

B. Interview Responses

The primary data analysis we performed was the qualitative coding of the interview data. Over the course of the study, we collected an average of 40.2 (SD = 8.8) minutes of recorded interview responses for each participant. We transcribed the audio recordings and separated the responses by question. This produced a total of 3,915 question responses. Because we used semi-structured interviews, the responses to some questions included follow-on questions and responses. We grouped any follow-on questions with the original interview question.

Using a grounded theory approach, we developed a set of high-level categories for the responses. We chose to categorize based on participants' responses rather than the questions for two reasons: 1) the interviews included related questions and 2) participants sometimes responded with information that did not perfectly align to each question. We developed the initial categories by manually sorting roughly 10% of the responses into similar groups. We identified five categories: decision rationales, expected task difficulty, sources of ease and difficulty, experience outcomes, and other. Reassuringly, the categories that emerged from this process align closely with the themes we tried to incorporate into the interviews. Once the categories emerged, two researchers labeled a new random sample of 20% of the responses. We reached very high agreement (Cohen's $\kappa > .8$) for the categories ($\kappa = .95$, $p < .001$). One researcher then categorized the remaining responses.

We subsequently repeated this process to develop sublabels for each of the high-level categories, with the exception of the *other* category. We omitted the *other* category, because many of the responses were unrelated. Because the sublabels were based on the original high-level categories, we report Bonferroni adjusted p values for Cohen's κ. We reached very high agreement for all sublabels as shown in Table I. We report the sublabels for each category in the results section.

TABLE I. Sublabel Interrater Agreement

High-Level Category	% of Responses	Cohen's κ	adj. $p <$
Decision Rationales	16%	.89	.001
Expected Task Difficulty	13%	.85	.001
Sources of Ease & Difficulty	15%	.88	.001
Experience Outcomes	38%	.86	.001
Other	18%	*n/a*	*n/a*

V. Results

We share our findings in this study by reporting each category separately. Each of the categories provides insight into the decisions that participants made, the factors that influenced those decisions, and participants' perceptions of value that they received from the task and instructional format. For each category, we share relevant survey results and summarize the general themes that emerged from participants' responses.

When summarizing the interview responses, we report each category's sublabels in two ways: 1) the overall percentage a sublabel was cited across all tasks in the study and 2) the percentage of participants who cited the sublabel at least once during the study. The percentage across all tasks gives insight into what types of decisions participants made, whereas the percentage across all participants helps demonstrate what factors participants identify as important. We report a summary of each high-level category's sublabels in Tables II-V. In these tables we summarize the interview responses in the same two ways: 1) *% of Tasks*, and 2) *% of Participants*, respectively. For each high-level category we had two additional sublabels that we do not discuss in our results: *no reason / neutral* and *other*. The *no reason / neutral* sublabel captured responses that provided no reason and were often neutral in sentiment, for example "it was good." The *other* sublabel captures responses that did not answer the question or that are off topic.

We begin by first discussing the decision rationales category to understand participants' decisions. We then follow-up with the remaining high-level categories shown in Table I.

A. Decision Rationales

The decision rationales category analyzes the explicit reasons participants gave for choosing an instructional task and format. In total, participants worked on 167 instructional tasks (62 tutorials, 105 puzzles). 21 participants (70%) completed all six instructional tasks; the remaining 9 participants (30%) completed between three and five tasks (Mdn = 4, M = 4, SD = .83). Most participants chose to work on both tutorials (Mdn = 2) and puzzles (Mdn = 3). Overall, 10% of participants decided to work on more tutorials than puzzles, 60% of participants worked on more puzzles, and 30% of participants decided to work on an equal number of both.

Participants indicated that they valued both the tutorial and puzzle formats. In the post-study interview, we asked participants which format they were more likely to use on their own after this study and 53% of participants responded that they would use both, while 30% said puzzles, and the remaining 17% said tutorials. This is further supported by the variety of decisions participants made during the study. Overall, we see that the majority of participants (73%) used both tutorials and puzzles. While a small minority show a strong preference (number of tasks minus one) for tutorials (3%) or puzzles (13%).

From the sublabels, we see that participants made decisions based on personal preference, improving their programming skills, and challenge (see Table II). We found that participants were more likely to choose a task based on their personal preference rather than choosing a task based on improving their programming skills. However, when they did make a decision to improve their programming skills, many participants chose

TABLE II. DECISION RATIONALE RESPONSE THEMES

	% of Tasks (*Tutorials*[a], *Puzzles*[a])	% of Participants (*Females*, *Males*)
Personal Preference	56% (52%, 58%)	87% (71%, 100%)
Enjoyed Animation	48% (47%, 49%)	83% (71%, 94%)
Liked Format	16% (8%, 20%)	53% (43%, 63%)
Improve Programming Skills	32% (52%, 21%)	73% (79%, 69%)
Improve Skill	12% (19%, 8%)	37% (36%, 38%)
Discover Level	21% (34%, 13%)	53% (64%, 44%)
Challenge	66% (65%, 68%)	100% (100%, 100%)
Avoid Challenge	22% (39%, 12%)	60% (43%, 75%)
Seek Challenge	48% (32%, 57%)	87% (93%, 81%)
Other	17% (13%, 19%)	53% (36%, 69%)
No Reason / Neutral	12% (13%, 11%)	40% (29%, 50%)
Other	5% (0%, 8%)	27% (21%, 31%)

[a]. Percent of sublabels for all tutorials/puzzles; not percent across all tasks.

the tutorials, the format that has been shown to be less effective [6]. Throughout the study participants also actively managed the difficulty of their tasks by choosing an instructional format that they believed would increase or decrease each task's challenge.

1) Personal Preference

Personal preference was a common reason participants cited for their decisions. Overall, our results suggest that participants enjoyed completing puzzles more than tutorials.

Participants cited enjoying the instructional format as an important factor in their decisions. In total 53% of participants made a decision during the study based on whether they liked the instructional format. 43% of participants based at least one decision on enjoying the puzzles, "I like the puzzles. They've all been fun." In contrast, only 17% of participants cited making a decision based on liking the tutorials, "I like the tutorials better." The greater enjoyment of puzzles is also reflected in the percentage of tasks: participants cited enjoyment of puzzles in 20% of tasks, while only citing enjoyment in 8% of tutorials.

In the post-study interview, we asked participants which instructional format they enjoy more. Again, we see a preference towards puzzles: 80% of participants stated that they enjoy the puzzles more, compared to 13% who enjoy the tutorials more, and 7% who enjoy both. This preference is also reflected in the TEQ motivation survey's interest/enjoyment scale. Using a multilevel model, we found that participants rated the puzzles as significantly more enjoyable (M = 5.8, SD = 1.0) than the tutorials (M = 4.3, SD = 1.5), $\chi^2(1) = 17.52$, $p < .001$.

In 48% of tasks, participants cited enjoying an animation as the reason for making a decision. In total 83% of participants made a decision that was based on whether or not they liked the animation. Participants cited the overall appeal of an animation: "Probably just because it looked like a fun animation to do." Some participants also described specific attributes: "I liked how the alien kicked the spaceship to start it up and when it flew away." The specific content presented via learning resources is an important attribute for children learning to program.

2) Improve Programming Skills

Participants made fewer decisions that prioritized improving their programming skills compared to their personal interests. When making decisions to improve their programming skills, they did so either directly, by seeking to acquire new programming skills or indirectly, by trying to gain a better understanding of their skill level or the instructional formats.

A relatively small number of participants (37%) cited a goal of improving their programming skills across only 12% of tasks. This is a surprisingly low result given that we primed participants to make decisions to improve their skills. However, when participants did cite improving programming skills as a goal, they were more than twice as likely to have chosen a tutorial (19% of tasks) than a puzzle (7% of tasks). This difference seems to reflect a belief from users that they learn more using tutorials. When asked for the reason behind a tutorial decision, one participant explained, "Because it was a new skill that I hadn't learned before, so if I did it as a puzzle I'd probably not quite understand exactly what it was trying to teach me." This tendency to select tutorials when prioritizing learning is interesting because it does not reflect the results of a study comparing the two formats which found that puzzles were both more efficient and effective for learning new concepts [6].

Choosing appropriate learning goals also requires both accurate knowledge of the available learning resources and knowledge of their own abilities. Participants described a variety of goals that contribute towards these types of self-knowledge. In 21% of tasks, participants described their desire to discover their skill level. Some attempted discovery by equally using both formats. One participant justified a decision as "Just so I can get a feel for both tutorials and puzzles." Other participants talked about comparing the difficulty between the two formats: "I wanted it to be a little harder than the tutorial because I wanted to see if it'd make a big difference or not."

Participants' responses indicate a strong preference towards enjoyment rather than improving skills. However, when attempting to improve their programming skills, their responses suggest a nuanced approach in which they explore what roles resources could play, assess their own skill level, and select resources that they believe could contribute towards that goal.

3) Challenge

Participants frequently cited *challenge*, pushing their abilities, as a factor in their decisions. Overall, 87% of participants sought challenge while 60% also avoided challenge. Plotting the expected difficulty of each instructional task for each participant, we observed fluctuation between easy and hard tasks for roughly 80% of participants. Of the remaining 20%, 7% of participants consistently avoided challenge, and 13% consistently sought challenge. This suggests that participants do not always want to work on the hardest task, they prefer a mix between challenging and non-challenging tasks.

When seeking challenge, participants tended to select puzzles. Participants described seeking a challenge for 57% of the puzzles and 32% of the tutorials. When avoiding challenge, participants favored tutorials. In 39% of tutorial decisions and 12% of puzzle decisions, participants expressed a desire to decrease the challenge level. This disparity suggests that participants perceived tutorials as being easier. The post-interview provided further support for the perception that tutorials are easier. In each post-task interview we asked participants to rank the difficulty of their chosen instructional format from 1 to 9. Participants ranked puzzles as significantly more difficult (M = 4.8, SD = 1.9) than tutorials (M = 3.8, SD =

1.6), $\chi^2(27) = 86.75$, $p < .001$. Further, participants also never ranked a tutorial above a six, suggesting a difficulty ceiling for tutorials. Participants also reported significantly less pressure/tension in the TEQ motivation survey when using the tutorials ($M = 1.9$, $SD = .9$) compared to puzzles ($M = 2.8$, $SD = 1.1$), $\chi^2(1) = 10.46$, $p < .01$.

Participants' desires to seek challenge were mostly straightforward: "I like having to challenge my mind more." We saw two classes of reasons for avoiding challenge. The first group of participants expressed a desire to complete an animation they expected to be too challenging for them. These participants perceived tutorials as a more gentle introduction to the content required for a challenging animation. "I picked it as a tutorial because it looked like it had lots more complexity than the other ones and I didn't want to just jump right in without knowing what I was doing." The second group of participants avoided challenge after extending themselves outside of their comfort range. "I think I'm going to do mostly tutorials from now on because this was a bit hard."

While there is some variety in what participants perceived as too difficult, participants who gave up on a task before successfully completing it arguably felt that they had taken on too much. We note that this was relatively rare; it happened for 8 of 167 total tasks. However, all eight of these tasks were puzzles. For these eight, participants uniformly went on to easier tasks afterwards. Six chose to complete a tutorial for their next task. The other two chose an easier puzzle. While the failed tasks are a more extreme situation, participants' decisions to reduce the difficulty were consistent with the overall pattern of pushing a little bit and then scaling back. The presence of both puzzles and tutorials empowered participants to adjust the experience to meet their own learning and confidence needs.

B. Expected Task Difficulty

In the decision rationales section we saw that challenge was a prominent reason given for making a decision. When trying to manage challenge, participants considered a variety of information that informed their expectations about task difficulty (see Table III). Our results suggest that participants used appropriate factors for predicting task difficulty. Participants typically weighed several practical factors: their prior programming experience, their perception of the instructional format's intrinsic difficulty, and the animation's labeled difficulty. However, there is one factor that may hamper their ability to do this accurately: animation complexity.

TABLE III. EXPECTED TASK DIFFICULTY RESPONSE THEMES

	% of Tasks (*Tutorials* [a], *Puzzles* [a])	% of Participants (*Females*, *Males*)
Programming Experience	23% (21%, 24%)	77% (71%, 81%)
Programming Experience	20% (19%, 21%)	70% (71%, 69%)
Code Structure	2% (2%, 3%)	10% (0%, 19%)
Instructional Format	14% (31%, 4%)	47% (50%, 44%)
Labeled Difficulty	31% (21%, 36%)	67% (64%, 69%)
Animation Complexity	23% (23%, 23%)	63% (71%, 56%)
Other	16% (13%, 18%)	43% (50%, 38%)
No Reason / Neutral	14% (10%, 17%)	40% (50%, 31%)
Other	2% (3%, 1%)	10% (7%, 13%)

[a.] Percent of sublabels for all tutorials/puzzles; not percent across all tasks.

1) Programming Experience

70% of participants considered their own programming experience when reasoning about difficulty. One participant expected a task to be easy "Because all of the skills in this one I had already learned." An additional 10% of participants reasoned about code structure. Another participant expected a selected task to be difficult: "Because it was a new skill that I hadn't learned and I'd have to figure out that you would have to put the *Do in Order* inside the *Do Together* with something outside of the *Do in Order*."

2) Instructional Format

47% of participants based their expectations of difficulty on the instructional format. For example, one participant expected that, "The tutorial experience would be pretty easy." Participants often stated that tutorials were easier "Because all you do is look at the tutorial, it tells you what to do, you do that, and then you turn the page and you do that again with the next instruction."

During the post-study interview, we asked participants to summarize their advice to potential new users about when they should use tutorials and puzzles. Their advice reiterates the theme of using the different instructional formats to manage challenge. 17% of participants stated that you should use a tutorial when you expect a task to be difficult. 13% of participants stated that you should use a puzzle when you want to challenge yourself. Lastly, 41% of participants thought that if something is new for you, you should do a tutorial first, and then practice with puzzles later.

3) Labeled Difficulty

Our task selection interface labeled all tasks with one of three difficulty labels: easier, medium, and harder. In 31% of tasks, participants based their expectations on these difficulty labels.

4) Animation Complexity

63% of participants stated that they expected a task to be easy or difficult based on watching the animation. One participant expected a task to be challenging "just because there was, I mean, it looked like there was lots of stuff happening." In total, participants used animation complexity to inform their difficulty assessment in 23% of tasks. While this may be a natural thing to do, it creates an interesting tension. Participants often considered the appeal of an animation in selecting a task. Participants' desire to complete a compelling animation creates an incentive to create more complex and visually interesting animations. However, the same push towards compelling animations may also lead users to perceive them as out of reach.

C. Sources of Ease & Difficulty

When making decisions, participants commonly used their generalizations about difficulty as discussed in the previous sections. However, in the post-task interviews participants identified what they believed to be the source of difficulty for that task. Participants' expectations often aligned with their actual experience of difficulty. In this section, we discuss the elements that participants felt contributed to the ease or difficulty of completing tasks (see Table IV). Overall, participants identified fairly common sources of difficulty. Participants felt that the instructional format, their prior programming experience, the interface's mechanics, and their own personal degree of struggle, were sources of task difficulty.

TABLE IV. SOURCES OF EASE & DIFFICULTY RESPONSE THEMES

	% of Tasks (*Tutorials* [a], *Puzzles* [a])	% of Participants (*Females*, *Males*)
Instructional Format	31% (58%, 15%)	87% (79%, 94%)
Prior Experience & Barriers	55% (37%, 66%)	100% (100%, 100%)
Prior Experience	23% (24%, 22%)	60% (50%, 69%)
Conceptual Barrier	37% (15%, 51%)	93% (86%, 100%)
Translation Barrier	2% (0%, 4%)	13% (7%, 19%)
Mechanics	21% (29%, 16%)	63% (57%, 69%)
Instructional Format	11% (10%, 11%)	30% (36%, 25%)
Programming Environment	13% (24%, 6%)	57% (50%, 63%)
Degree of Struggle	49% (32%, 59%)	90% (86%, 94%)
My Experience	37% (21%, 47%)	83% (79%, 88%)
Length	16% (13%, 18%)	57% (79%, 38%)
Other	25% (26%, 25%)	73% (71%, 75%)
No Reason / Neutral	25% (26%, 24%)	70% (71%, 69%)
Other	1% (2%, 1%)	7% (0%, 13%)

[a.] Percent of sublabels for all tutorials/puzzles; not percent across all tasks.

1) Instructional Format

Unsurprisingly, 87% of participants cited the instructional format as a source of ease or difficulty. The responses here were consistent with the overall perception that tutorials are easier as we have noted elsewhere. 77% of participants identified tutorials as easy or difficult because of the format while only 43% said the same for puzzles. We note however, that 74% of these responses for tutorials are cited for being easy, whereas the responses for puzzles are spread across easy and difficult.

2) Prior Experience & Barriers

For many users, content familiarity made tasks easier and a lack of it created the potential for barriers. If a participant had prior experience with a programming concept, they tended to perceive it as easier, "It was much easier than I thought it was, just using the stuff that I had already learned." In total, 60% of participants cited that their familiarity with a programming concept made that task easier for them. Participants' experiences of difficulty based on their background were consistent with their background-based expectations of difficulty. In contrast, 93% of participants cited a conceptual barrier, or not understanding a programming concept, as a source of difficulty in the tasks. One participant said that "Not knowing that it was actually possible to put *Do in Order* inside with *Do Together*" made a task difficult. Conceptual barriers are also strongly associated with puzzles. 87% of participants stated that a conceptual barrier caused them difficulty in puzzles, whereas only 23% of participants said the same for tutorials.

We also saw a translation barrier, connecting a programming statement with its output, as a source of difficulty for participants working on puzzles. 13% of participants stated that the translation barrier was a source of difficulty for them.

3) Mechanics

30% of participants cited the instructional format's mechanics as a cause for difficulty. In the tutorial, participants found watching and following the video difficult whereas in the puzzle, participants found watching the correct output difficult. Participants also found the programming environment's mechanics difficult. In total, 57% of participants cited the mechanics of the programming environment as a source of difficulty.

4) Degree of Struggle

Lastly, participants cited their own perceptions as a source of ease and difficulty. These perceptions were often very self-referential in nature: in essence, participants seemed to express "It was hard because I found it hard." Some participants talked about "having to do it over and over and over and over again." Others referenced the amount of time necessary to complete the task: "It was hard and long. It took me a really long time to do it." In 18% of puzzles and 13% of tutorials, participants cited length as a source of ease or difficulty. Using a multilevel model, we noted that participants spent significantly longer on puzzles (M = 8.2, SD = 6.6 minutes) than tutorials (M = 7.92, SD = 4.6 minutes), $\chi^2(27) = 67.39$, $p < .001$. Overall, 83% of participants explained their perception of difficulty through their own experience at least once during the study.

D. Experience Outcomes

Our final category, experience outcomes, explores the values that participants perceived while completing instructional tasks (see Table V). Participants received enjoyment from completing the tasks and also improved their programming skills. However, in many tasks participants were unable to state what outcome or benefit they received from completing the task.

1) Enjoyment

In the *decision rationales* section we noted that participants often made decisions based on their preference. When we asked participants in the post-task interviews what they got out of a task, one of the most common responses was that they simply enjoyed it. 63% of participants enjoyed the animation content, 73% enjoyed working on the task, and 90% enjoyed completing the task. 83% and 53% of participants stated that they enjoyed puzzles and tutorials, respectively. The most interesting responses described struggling to complete the task and taking strong satisfaction from that success. As one participant said, "The more you struggle at something the more exciting it is when you finish it." Echoing the other results, satisfaction from completing a task is a theme we saw more strongly with puzzles than tutorials. One possible explanation for the enjoyment disparity between puzzles and tutorials is that participants saw value in the freedom to solve the problem: "The best part was having to figure everything out." Our TEQ results suggest that participants perceived more choice in the puzzles (M = 6.4, SD = .7) than the tutorials (M = 6.0, SD = .9), $\chi^2(1) = 6.48$, $p < .05$.

TABLE V. EXPERIENCE OUTCOMES RESPONSE THEMES

	% of Tasks (*Tutorials* [a], *Puzzles* [a])	% of Participants (*Females*, *Males*)
Enjoyment	80% (79%, 80%)	97% (100%, 94%)
Liked Animation	27% (27%, 27%)	63% (64%, 63%)
Working on Task	26% (32%, 22%)	73% (71%, 75%)
Finishing Task	50% (45%, 53%)	90% (86%, 94%)
Improve Programming Skills	62% (61%, 63%)	97% (93%, 100%)
Learned Skill	44% (50%, 40%)	87% (71%, 100%)
Improved Competency	22% (23%, 21%)	63% (64%, 63%)
Practice	34% (32%, 35%)	87% (86%, 88%)
No Benefit	66% (71%, 64%)	97% (93%, 100%)
Other	92% (100%, 87%)	100% (100%, 100%)
No Reason / Neutral	87% (97%, 81%)	100% (100%, 100%)
Other	35% (44%, 31%)	87% (86%, 88%)

[a.] Percent of sublabels for all tutorials/puzzles; not percent across all tasks.

2) Improve Programming Skills

Participants stated improving their programming skills as an outcome. We note that this may be due in part to our priming to prioritize learning. Participants often distinguished between three concepts: learning, general competency, and practice.

When participants talked about learning, they focused on their introduction to a new concept or programming element. Overall, 87% of participants stated they learned a specific programming skill from completing a task. 70% of participants reported they learned new programming skills from the puzzles, compared to 60% of participants who reported the same for tutorials. This contrasts with participants' stated decisions around learning, where we found that they tended to select tutorials when citing a learning goal (see section V.A.2). 63% of participants reported improved computing competency. While 87% of participants also reported finding value in practicing using the instructional tasks. As one participant stated, "I enjoyed it. It was fun to practice on my own with the skills that I had just learned." 70% of participants reported that the puzzles helped them reinforce their skills, compared to just 50% of participants who said the same thing about tutorials.

We compared the reported programming experience from the pre-study and post-study interviews: 80% of participants felt more competent after completing the study, 17% reported no change, and a single participant reported feeling less competent.

3) No Benefit

In 66% of tasks participants responded to some interview questions that they did not benefit from the task. In most cases, this was in response to a question about what specific programming knowledge did they learn from the task. Recall that in only 44% of tasks was learning a specific skill cited. Frequently, participants did not know what they had just learned. Many recognized that the task was useful (i.e. practice) but could not name a specific programming concept that they had learned.

VI. Discussion

Below we discuss the implications this work has for novice programming environments.

A. Choice & Motivation

While direct evidence of learning suggests that puzzles are a better choice than tutorials [6], participants saw value in both formats beyond improving their skills. This is an exciting result that suggests that since participants primarily care about animation choice and balancing difficulty, we should consider providing multiple formats with a range of scaffolding that enables users to obtain adequate support while perusing their interests. This might include traditional formats like tutorials and puzzles. But it could also include formats designed for open-ended situations like remixing code [25], [35] or providing hints or suggestions while users work on their own projects [36], [37]. This would empower users to improve their skills while making decisions based on their own goals.

B. Challenge

In this study, almost every participant made a decision where they were seeking challenge. Given the prevalence of this decision rationale, we encourage novice programming environment developers to specifically seek out avenues to enable their users to challenge themselves. Additionally, based on the challenge fluctuation we saw in the study, we suggest that novice programming environments provide mechanisms to allow users to alternate between easy and challenging tasks.

C. Learning

A majority of participants made a decision based on trying to discover which tasks were appropriate for their skill level. Because this rationale appears to be both important for users and commonplace, programming environment designers should consider ways that enable participants to explore their abilities. We observed that in several MOOCs and puzzle-like systems, users are not allowed to progress until they have finished the current lesson or level. Our result suggests that designers of these systems should give users the choice and the freedom to choose their own tasks, even if they may not be ready for them. This comes with its own risk; users may feel discouraged after failing to complete a task. However, as we observed in this study, many of the participants who failed to complete a puzzle were driven to discover their mistake by using another instructional format or by selecting an easier task.

VII. Threats to Validity

In this study we used a semi-structured interview that we specifically geared towards learning programming. We also primed participants to improve their programming skills. Our specific focus on learning may skew our results away from other factors that also affect decisions and perceptions of value. However, even when we explicitly encouraged participants to prioritize learning, participants were more likely to make their decisions based on their preferences and desire for challenge.

This study sought to identify the perceptions of value that novice programmers see in two specific instructional formats. While the specifics of these perceptions may not hold to other formats, our general conclusions about the factors that participants feel are important, like challenge, will likely hold for other instructional formats. However, outside of middle school children, these results would likely differ for other demographics, like expert programmers. We also note that participants with a stronger interest in programming may have self-selected to participate in this study.

VIII. Conclusion

In this paper, we presented a qualitative study investigating which instructional formats users preferred and why they chose to use them. We discovered that enjoyment, challenge, and perceived value all play important roles in a user's decision to choose between instructional formats. We hope that this data can help inform improvements to novice programming environments that further advance their independent learning capabilities. We hope that with these decision factors identified, empirical results will soon follow which demonstrate whether these factors are helpful to our ultimate goal of helping novices learn programming independently.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grants No. 1054587 and 1440996. We would like to thank Michelle Ichinco for her valuable advice during analysis and Craig Anslow for his feedback on this paper.

REFERENCES

[1] "Scratch." [Online]. Available: https://scratch.mit.edu/.

[2] "MIT App Inventor." [Online]. Available: http://appinventor.mit.edu/.

[3] "Hour of Code," *CSEd Week*. [Online]. Available: http://csedweek.org/.

[4] "Gidget." [Online]. Available: http://www.helpgidget.org/.

[5] "Lightbot." [Online]. Available: http://lightbot.com/.

[6] K. J. Harms, N. Rowlett, and C. Kelleher, "Enabling independent learning of programming concepts through programming completion puzzles," in *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2015, pp. 271–279.

[7] R. L. Mack, C. H. Lewis, and J. M. Carroll, "Learning to Use Word Processors: Problems and Prospects," *ACM Trans Inf Syst*, vol. 1, no. 3, pp. 254–271, Jul. 1983.

[8] J. M. Carroll and S. A. Mazur, *LisaLearning*. Citeseer, 1985.

[9] J. M. Carroll, P. L. Smith-Kerker, J. R. Ford, and S. A. Mazur-Rimetz, "The Minimal Manual," *Human–Computer Interact.*, vol. 3, no. 2, pp. 123–153, Jun. 1987.

[10] M. Hamada, "Web-based Tools for Active Learning in Information Theory," in *Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education*, New York, NY, USA, 2007, pp. 60–64.

[11] A. Radenski, "Digital Support for Abductive Learning in Introductory Computing Courses," in *Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education*, New York, NY, USA, 2007, pp. 14–18.

[12] A. Radenski, "Freedom of Choice As Motivational Factor for Active Learning," in *Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education*, New York, NY, USA, 2009, pp. 21–25.

[13] J. A. Stone and E. M. Madigan, "The Impact of Providing Project Choices in CS1," *SIGCSE Bull*, vol. 40, no. 2, pp. 65–68, Jun. 2008.

[14] F. Ke and K. Xie, "Online Discussion Design on Adult Students' Learning Perceptions and Patterns of Online Interactions," in *Proceedings of the 9th International Conference on Computer Supported Collaborative Learning - Volume 1*, Rhodes, Greece, 2009, pp. 219–226.

[15] J. Boustedt, A. Eckerdal, R. McCartney, K. Sanders, L. Thomas, and C. Zander, "Students' Perceptions of the Differences Between Formal and Informal Learning," in *Proceedings of the Seventh International Workshop on Computing Education Research*, New York, NY, USA, 2011, pp. 61–68.

[16] "Codecademy." [Online]. Available: http://www.codecademy.com/.

[17] "Khan Academy." [Online]. Available: http://www.khanacademy.org.

[18] "edX." [Online]. Available: https://www.edx.org/.

[19] K. Benda, A. Bruckman, and M. Guzdial, "When Life and Learning Do Not Fit: Challenges of Workload and Communication in Introductory Computer Science Online," *Trans Comput Educ*, vol. 12, no. 4, p. 15:1–15:38, Nov. 2012.

[20] A. P. Rovai, "In search of higher persistence rates in distance education online programs," *Internet High. Educ.*, vol. 6, no. 1, pp. 1–16, st 2003.

[21] M. B. Rosson, J. Ballin, and J. Rode, "Who, What, and How: A Survey of Informal and Professional Web Developers," in *Proceedings of the 2005 IEEE Symposium on Visual Languages and Human-Centric Computing*, Washington, DC, USA, 2005, pp. 199–206.

[22] A. Head, C. Appachu, M. A. Hearst, and B. Hartmann, "Tutorons: Generating context-relevant, on-demand explanations and demonstrations of online code," in *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2015, pp. 3–12.

[23] P. J. Guo, "Online Python Tutor: Embeddable Web-based Program Visualization for Cs Education," in *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, New York, NY, USA, 2013, pp. 579–584.

[24] M. Gordon and P. J. Guo, "Codepourri: Creating visual coding tutorials using a volunteer crowd of learners," in *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2015, pp. 13–21.

[25] K. J. Harms, D. Cosgrove, S. Gray, and C. Kelleher, "Automatically Generating Tutorials to Enable Middle School Children to Learn Programming Independently," in *Proceedings of the 12th International Conference on Interaction Design and Children*, New York, NY, USA, 2013, pp. 11–19.

[26] D. Parsons and P. Haden, "Parson's Programming Puzzles: A Fun and Effective Learning Tool for First Programming Courses," in *Proceedings of the 8th Australasian Conference on Computing Education - Volume 52*, Darlinghurst, Australia, Australia, 2006, pp. 157–163.

[27] B. J. Ericson, M. J. Guzdial, and B. B. Morrison, "Analysis of Interactive Features Designed to Enhance Learning in an Ebook," in *Proceedings of the Eleventh Annual International Conference on International Computing Education Research*, New York, NY, USA, 2015, pp. 169–178.

[28] M. J. Lee and A. J. Ko, "Comparing the Effectiveness of Online Learning Approaches on CS1 Learning Outcomes," in *Proceedings of the Eleventh Annual International Conference on International Computing Education Research*, New York, NY, USA, 2015, pp. 237–246.

[29] "CodeCombat: Learn to Code by Playing a Game," *CodeCombat*. [Online]. Available: http://codecombat.com. [Accessed: 13-Mar-2016].

[30] "Looking Glass." [Online]. Available: http://lookingglass.wustl.edu/.

[31] J. J. G. Van Merrienboer and M. B. M. De Croock, "Strategies for Computer-Based Programming Instruction: Program Completion Vs. Program Generation," *J. Educ. Comput. Res.*, vol. 8, no. 3, pp. 365–394, Jan. 1992.

[32] S. E. Dreyfus and H. L. Dreyfus, "A Five-Stage Model of the Mental Activities Involved in Directed Skill Acquisition," Feb. 1980.

[33] S. Kalyuga, P. Chandler, and J. Sweller, "Managing split-attention and redundancy in multimedia instruction," *Appl. Cogn. Psychol.*, vol. 13, no. 4, pp. 351–371, 1999.

[34] "Intrinsic Motivation Inventory," *Self-Determination Theory*. [Online]. Available: http://www.selfdeterminationtheory.org/questionnaires/10-questionnaires/50.

[35] P. Gross, J. Yang, and C. Kelleher, "Dinah: an interface to assist non-programmers with selecting program code causing graphical output," in *Proceedings of the 2011 annual conference on Human factors in computing systems*, New York, NY, USA, 2011, pp. 3397–3400.

[36] C. Piech, M. Sahami, J. Huang, and L. Guibas, "Autonomously Generating Hints by Inferring Problem Solving Policies," in *Proceedings of the Second (2015) ACM Conference on Learning @ Scale*, New York, NY, USA, 2015, pp. 195–204.

[37] M. Ichinco, "Towards crowdsourced large-scale feedback for novice programmers," in *2014 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2014, pp. 189–190.