Teachers' Use of Video Reflections to Reinforce Computer Science Language and Concepts

Ha Nguyen*, Leiny Garcia†, Sharin Jacob‡, Debra Richardson§, and Mark Warschauer¶
University of California-Irvine
Irvine, CA
*thicn@uci.edu, †leinyg@uci.edu, ‡sharinj@uci.edu, §djr@ics.uci.edu, ¶markw@uci.edu

***Abstract*—This paper examines teachers' use of Flipgrid, a student-facing video platform, as a reflection tool to promote computer science language in upper elementary classrooms. We take a case-study approach with three fourth grade teachers: one had a high number of students with special needs, one had substantially more gifted and talented students, and one taught a dual immersion English-Spanish class. Data sources include teacher interviews, design meetings between researchers and teachers, and classroom observations. We find that teachers with different pedagogical visions adopted the tools for reinforcement, student engagement, and formative assessment. We document teachers' iterative improvement strategies and shifts in teacher noticing, particularly the objects (i.e., computer science vocabulary and concepts), level (i.e., whole-class versus individual students), and depth of how they noticed students learning through video reflections. This study contributes to the ongoing work that examines instructional approaches to promoting computing education in diverse K-12 classrooms, especially among teachers with no formal training in computer science education.**

***Index Terms*—computer science education, educational technology**

I. Introduction

Researchers have called for the introduction of CS in earlier grades to recruit interests and address the underrepresentation of women and minority students [5], [28]. An understanding of teachers' existing knowledge, experiences, and instructional contexts, as well as how these components influence teachers' adoption of CS tools and curriculum, is critical to promoting systematic integration of CS into formal education settings [31]. However, there is limited research on the instructional and learning processes that take place in the classrooms, especially in elementary grades [13], [14], [16]. We contribute to this growing research area by documenting how teachers in different instructional contexts adopted a video reflection tool to notice students' CS learning. The tool was introduced in a one-year CS curriculum implemented by teachers and university researchers in a large urban school district in California. Prompting students to reflect on what they just learn can make implicit knowledge explicit for students and improve their content understanding.

This work draws from the growing body of research around instructional strategies in CS education [14], [16] and teacher noticing of instruction for improvement [8], [22], [24], [26]. We employ a cross-case analysis to examine the potential affordances of the video reflection tool in three elementary classes taught by teachers with no formal training in CS. The teachers faced different classroom dynamics: one had a higher number of students with special needs, one had substantially more gifted and talented students, and one taught a dual immersion English-Spanish class. We explore the instructional strategies and beliefs that characterize the teacher adopters of the tool. We also examine ways in which teachers use the tool to make sense of students' development of CS-specific language.

This study has two main contributions. First, we explore how teachers in varied contexts experimented with a video reflection tool to reinforce CS discourse. A practical implication is that tools' alignment with teachers' knowledge, beliefs, and practices can be positively perceived by teachers who are new to CS education. Second, we examine the utility of student reflections in fostering teacher noticing of computational language and understanding. This is one of the first studies to apply teacher noticing frameworks to CS education. We find that teachers mostly focused on students' vocabulary instead of conceptual understanding, and teacher noticing was mostly descriptive and not yet focused. We discuss implications for designing professional development opportunities that employ student artifacts to guide teacher instructional improvements.

II. Related Work

A. Examining Instructional Strategies in CS through the Universal Design Learning Framework

The literature on teachers' perceptions when implementing CS curricula has been scarce. Studies have focused on teachers' perceived challenges and instructional moves that are effective in fostering students' computational thinking [14], [16], [21]. Multiple implementation models of CS curricula may emerge depending on intrinsic factors, such as teachers' perceptions of their professional efficacy, and extrinsic factors, namely time constraints and classroom contexts [14], [21]. A case study analysis [13] using observations and interviews with teachers and administrators across instructional contexts (e.g., library, art, general education classrooms) found that teachers employed different instructional strategies based on task open-endedness and classroom organization when teaching CS. For example, some teachers tend to employ more whole-class and explicit instruction, while others take a more open-ended

This work is funded by the National Science Foundation Grant 1738825.

approach [14]. These pedagogical choices were correlated with the amount of administrative support, teacher values, and teachers' perceived expertise [14], [15].

Studies have also employed the Universal Design Learning (UDL) framework to explore teachers' perceptions and implementation of CS curricula [14], [16]. The UDL framework encourages teachers to consider not only direct versus open-ended pedagogical approaches, but also ways in which instruction may broaden students' means of representation, action and expression, and engagement [20]. Applying the UDL framework to examining teachers' practices explores ways in which curricular planning and delivery can purposefully reduce barriers and increase meaningful entries for all students, including special education students and English learners (Els). The dimensions of the UDL framework can be summarized as follows:

- **Representation**: Provide multiple ways (e.g., texts, multimedia visuals, audio, discussion) for students to interact with content, communicate in disciplinary language, and generate understanding.
- **Action and Expression**: Provide ways for students to demonstrate comprehension in multiple modes (e.g., physical action, writing, speaking) and monitor their own progress (e.g., goal-setting, managing learning resources).
- **Engagement**: Recruit and sustain students' interests and collaboration.

B. Universal Learning Design in Student Reflections

UDL principles to promote student action and expression is the underlying theory for prompting students' self-reflection on content knowledge. Discussion and argumentative activities to clarify, reflect, and build on one's own and others' ideas have been promoted in mathematical, science, and English Language Arts (ELA) classrooms [1], [9], [17], [29]. Students may use multiple types of discourse–both everyday and scientific vocabulary–to present their understanding of scientific concepts based on situational demands [11]. Strategies in English Language Arts posit that multiple exposures and reinforcements are required before students begin to correctly use and understand new terms [1].

In the field of CS, students engage in discipline-specific language as they develop and test logical processes to address abstract computational problems. Students articulate the exploratory, explanatory, and elaborative steps involved in problem solving to construct understanding of key computational concepts and associated vocabulary knowledge [11]. Attending to student voices in reflection develops students' discursive identities and promotes identification with the field [6]. Research has shown that developing STEM identities at the elementary level promotes later interest in CS careers [30].

Teachers can utilize several UDL strategies to scaffold the reflection activities. Compared to simply stating the vocabulary's definition, combining instruction of language and content knowledge more effectively builds students' knowledge base for productive reflection [9]. Scaffolding with explicit prompts can be effective for supporting student argumentation and general reflection in the discipline [9]. Encouraging students to use multiple methods of action and expression to explain how they design their codes can also increase access and involvement in computing education, particularly for learners with special needs [7], [16].

C. Teacher Noticing of Student Discourse

Few studies have examined the tools and learning moments that teachers use to notice students' computational thinking in K-12 classrooms [14]. We thus draw on related work on teacher noticing of instructional moments and student discourse in mathematics [8], [22] and science [24], [25]. Several aspects are salient in teacher noticing framework [27]:

- Object of noticing: Strengths and weaknesses in student thinking, classroom interactions, lesson content, ...
- Focus: Specific concept versus general theme.
- Perspective: Description, interpretation, evaluation.
- Level: Whole-class versus individual student.

Although teacher noticing tends to vary across classroom contexts and teaching experience, it is *multifaceted*–attending to different dimensions of instruction and classroom management, *selective*–centering on certain events at one time, and *instrumental*–the topics of noticing lead to instructional adjustments [8].

To hone into teacher noticing, classroom artifacts such as student work and video segments of student interactions have been employed as locally grounded representations of teaching practice to help teachers reflect on instruction and student thinking more intentionally [1]. The current study frames teacher noticing of CS knowledge in terms of the contexts and artifacts that helped teachers assess task appropriateness and student understanding. An example of task appropriateness is teacher's assessment of whether the instruction on programming loops was effective. An illustration of student understanding is whether students can use the acquired vocabulary and everyday language to explain the concepts.

III. Research Questions

1) What instructional strategies and beliefs characterize the adopters of the reflection tool within a CS curriculum? To what extent do these strategies and beliefs influence how teachers adopted the tool?
2) How do teachers use reflection videos to make sense of students' development of CS-specific language?

IV. Method

A. Study Setting & Participants

This study follows teachers from three fourth and fifth grade (ages 9-11) classes in one of the largest, most linguistically and culturally diverse school districts in the U.S. (92.9% Hispanic or Latino, 80.4% on Free and Reduced lunch, 38.7% ELs). Each class had 25 students on average.

The teachers differed in classroom demographics and instructional routines. Ellen and Helen (pseudonyms) taught all subjects in fourth grade in the same school and co-planned the CS lessons. Meanwhile, Juanita (pseudonym) worked with groups of special education students in Math at a dual-immersion charter school. The one hour per week teaching CS was the only time Juanita had with a whole class. The teachers reported no prior experience teaching formal CS curricula.

The teachers also faced different constraints and affordances from their classes. Ellen had a class with mainly Gifted and Talented students. About 10% of students in Helen's class received various Individualized Education Programs (IEP). Juanita had a group of five special education students that she consistently worked with throughout the year. The students in this study had no prior computing courses, although students in Ellen and Helen's classes had participated in Hour of Code activities in the previous academic year.

B. Curriculum

Teachers were part of a district-wide initiative to integrate CS into elementary schools. The project team, consisting of teachers and researchers, developed a curriculum that built on UDL framework to target computational concepts and vocabulary appropriate for students aged 9-11 and meet the needs of the district's linguistically diverse student populations. The curriculum includes language frames intended to develop student's use of computational language and interweaves stories to promote identity towards the CS field. The linguistic scaffolds leverage both academic language and interaction functions (e.g., discussion, reflection).

The researchers and teachers met monthly to reflect on curriculum progress and devise instructional strategies to accommodate the needs of each classroom. During those inperson design meetings, supplementary pedagogical tools were introduced to further facilitate students' acquisition of CS disciplinary language. In particular, researchers did a demo of Flipgrid, a platform where students could videotape themselves presenting their codes and share with their peers. . The reflection activities align with the UDL frameworks to foster multiple student expressions and engagement. Ellen, Helen, and Juanita decided to pilot the tool.

The reflection activities occurred at the end of Unit 1 and 2 of the curriculum and took about 45 minutes each. Most student responses were brief, lasting 1.15 minutes on average.

For Unit 1, students pair-programmed in Scratch with only ten designated code blocks. At the end of the unit (lesson 5), students were asked to reflect in pair about their "10 Blocks Challenge" in response to the Flipgrid prompt: *"Walk us through your project. What blocks did you use? What was difficult? What did you find?"* Because the activity occurred quite early in the year, students had only been introduced to a few CS vocabulary (e.g., code) and concepts (e.g., algorithm).

At the end of Unit 2 (lesson 10), students reflected individually on a digital collage they created about themselves. The activity occurred about three months into the curriculum, when students had been introduced to more CS vocabulary and concepts, namely event, sequence, initialization, and parallelism. Example of the Flipgrid reflection prompt: *"BRIEFLY state some details about you that are in the program. Then, pick ONE SPRITE, open the code, and explain what you did. Be sure to use vocabulary words like "algorithm," "code," "program," "parallel program" or "initialization"*.

All teachers reported reviewing their student Flipgrid videos right after the reflection lessons. The design meetings (December 2018 and March 2019) gathered additional teacher insights about students' language. During the meetings, all the teachers piloting the curriculum watched a selected set of students' Flipgrid video responses explaining their codes from all the teachers' classes. Teachers were then asked to reflect in the whole-group setting on student videos based on two questions: *(1) "What do you notice about student language and CS understanding?*, and *(2) "What insights for instruction do you gather from the student Flipgrid responses?"*.

C. Data Sources

Data sources include teacher interviews, two design meetings (detailed above), and classroom observations. These data sources are selected to produce a more comprehensive view of teachers' beliefs and practices through their own and researchers' perceptions across settings (e.g., individual interviews, teacher-researcher, teacher-student interactions). Interviews and classroom observations were used to answer research question 1 about the characteristics of teacher adopters. Teacher reflections during the design meetings were used to answer research question 2 about teacher noticing.

The research team visited classrooms almost every week and took structured notes of instructional strategies using the UDL framework [22]. A subset of two of the same lessons from each teacher (lesson 4 and 6; conducted between November 2018 and January 2019) was selected for analyses because they were part of the units students were reflecting on. Semi-structured interviews lasting about 30 minutes were conducted at the end of the school year to understand each teacher's experiences with the CS curriculum more generally. The interviews included questions such as *"How would you describe your CS instructional approaches?"*

D. Cross-case Analysis

We conduct a cross-case analysis to examine teachers' instructional practices as they unfolded in time [3]. Cross-case analysis is chosen as a means to explore complex social units of interconnected variables to understand a focal phenomenon [19]. The units of analysis are the three teachers teaching the same lessons and adapting the same pedagogical tool. We identify the salient themes in each teacher's approach to teaching CS before examining common themes across cases.

Observation notes, interviews, and transcripts from the design meetings were analyzed using an initial set of codes for pedagogical visions, instructional goals, and teacher noticing of student thinking. Prior frameworks, namely UDL [19] and teacher noticing [7, 26] inform these codes. The final coding scheme pertains to instructional strategies and noticing patterns that appear in the data (Table 1). Data from each teacher were coded separately, and then compared for similarities and differences to generate themes.

Domain	Code	Sub-code	Description for teacher actions
Instructional Strategies *Data sources: Teacher interviews, classroom observations*	Representation	Display information in alternative forms	Display information in alternative forms (e.g., visual, auditory, verbal)
		Provide language and symbols	Clarify vocabulary and symbols by scaffolding questions
		Reactivate background knowledge	Activate student background knowledge by drawing connections to other subjects or previous activities
		Model	Model activities through step-by-step instruction
	Expression	Display comprehension using multimedia	Provide students opportunities to demonstrate understanding in multiple modalities (e.g., write, speak, code, etc.)
		Recruit student interests	Recruit student interests in multiple ways
	Engagement	Foster collaboration and communication	Foster group work, pair programming, communication, etc.
		Provide mastery-oriented feedback	Provide feedback to orient students to mastery
		Scaffold resources	Vary demands and resources to support learning according to students' ability
Teacher Noticing *Data source: design meeting transcripts*	Object (What teachers notice)	Vocabulary	Notice students' use of computer science vocabulary (e.g., code, algorithm) or everyday language
		Concept	Notice students' use of computer science concepts (e.g., abstraction, initialization, parallelism)
		Practice	Notice students' engagement, communication, collaboration
	Level (At what level)	Whole class	Make general comments about whole class environment, behavior, learning, and pedagogy
		Individual	Attend to particular students' thinking
	Elaboration (How teachers notice) ↓	Basic	Form general impressions of students' learning, or provide description and evaluation of their performance.
		Mixed	Highlight noteworthy episodes, or provide primarily evaluations with some interpretations of student learning patterns.
		Focused	Propose alternative pedagogical solutions citing specific learning interactions as evidence.

TABLE I
CODING SCHEME FOR TEACHER INSTRUCTIONAL STRATEGIES AND SENSE-MAKING OF STUDENTS' CS REFLECTION. ADAPTED FROM [2], [7], [22], [29]. THE ARROW INDICATES INCREASING LEVELS OF SOPHISTICATION IN TEACHER NOTICING

E. Validity

Instrumentation and researchers' biases may result in interpretations of teachers' instructional strategies and noticing in ways that significantly differ from their intentions. We triangulated observation notes with audio recording and teacher interviews and conducted member checking as a validity procedure. The interview and design meetings prompts include open-ended questions to avoid biasing teachers' responses.

F. Hypothesis

We hypothesized that classroom contexts and pedagogical visions about the curriculum, instruction, and student ability would inform teachers' adoption of Flipgrid and implementation of the reflection activities. For example, teachers with more structured instructional approaches may use the tool for reinforcement and employ more directive prompts [13].

V. FINDINGS

A. Characteristics of Teacher Adopters

1) Beliefs – Teacher as Learner: Teachers' belief in teaching CS emerged in all interviews. Teachers posited themselves as learners who were getting used to the curriculum alongside their students. They indicated an willingness to experiment with ideas and teaching methods to see what worked best. For instance, Ellen gave examples of going back and forth with the tools (e.g., notebook, language frames, Flipgrid). She reflected that she became more confident with teaching the curriculum as she practiced beforehand what the students did. She admitted having no experience in computer programming and feeling confused in the beginning, and emphasized the importance of positioning herself as ”learning with the students”. The teachers further reflected that adopting a new tool is always challenging, but it is a "learning curve and students would get something out of it in the end" (Juanita, interview).

2) Tool Purposes – Reinforcement, Engagement, & Formative Assessment: The three teachers possessed different views about students' ability to thrive in the CS curriculum in terms of computing knowledge, language and academic needs, and persistence. These views seemed to intertwine with their goals for adopting Flipgrid. First, teachers viewed the reflection tool as an opportunity to engage all students. For example, Helen frequently described her class in terms of varying student skills and needs in her interview. She framed her students in different groups—those who may be more comfortable exploring and those who may need additional encouragement and support. The teachers reportedly adopted Flipgrid to invite all students to share about their programming progress.

Second, teachers adopted the tool as a reinforcement mechanism. The teachers expressed the belief that students learned more effectively if teachers could slow down on certain parts of the lessons and gave students time to reflect and practice. This theme particularly stands out in Helen's and Juanita's classes, which have a high number of ELs and special education students. Both teachers referred to going at the pace that accommodates students with particular needs.

Third, teachers used the tool as a formative assessment to understand potential changes in student learning. Mapping student reflections to their programming projects provided teachers an opportunity to quickly evaluate the learning progress of the whole class, and adjust their instruction accordingly.

Juanita alluded to differentiating instruction:

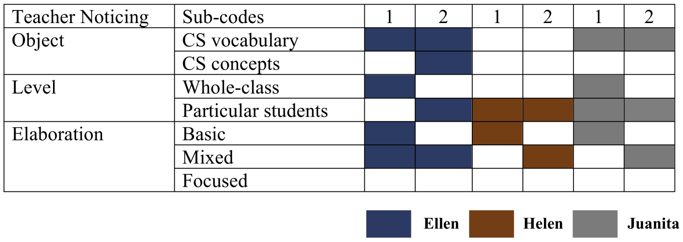
> You have a span - Special Ed students who need more time to process and students who run and are ready to go. What I'm finding is that I can't get to them all, and now that I know what they can do, I can push these kids to go higher and deeper.

Traditionally Juanita was used to working in small groups with special education students. Her reflection following the first use of Flipgrid reflected her hope to use the tool to assess students' CS understanding and language use simultaneously at the whole class level. She further noted in her interview that reflective tools such as Flipgrid could be a means to derive personalized instruction to accommodate different needs. Helen shared this vision, claiming that the reflection could show her "a side of students I have not seen before".

3) Instructional Strategies – Reinforcement & Multiple Representations: Analyses of classroom observation notes helped clarify how teachers' pedagogical strategies overlapped and diverged to adapt to specific classroom needs. All teachers employed explicit instruction, including reviewing prior knowledge and modeling tasks, before moving toward student-driven practices. The instructional routine was largely similar across the three classrooms across lessons. The teachers started the lessons by explaining the CS concepts, vocabulary, and tasks of the day. A common strategy to review content knowledge is to bridge students' background knowledge in other subjects with the new CS content. For example, teachers connected the concept of an "algorithm" to step-by-step algorithms in Math, or computer "events" to "events" in stories. Next, teachers provided step-by-step instruction before letting students explore coding individually or in pair for more than half of the lessons.

However, teachers employed different focus on language and representation. The teachers who identified special education and ELs as a focal point of instruction during interviews tended to clarify new vocabulary and symbols and repeat these vocabulary words and concepts more frequently. For Helen, it was connecting the visual representations with the CS concepts in printouts for students to refer to throughout the lessons. For Juanita, it was a routine of introducing the vocabulary with visual illustrations and definitions in the beginning of the lesson. The visual demonstrations of the vocabulary were put up in the Technology corner, accumulating into a display of new vocabulary and concepts over time. Juanita also pointed to these images to spot-check with students about the newly acquired vocabulary throughout the lessons.

Additionally, teachers differentiated task expectations to accommodate students' varied abilities. One example is the extent to which teachers encouraged students to use the language frames from the curriculum to explain CS concepts. All three teachers printed out the language frames and reminded students to use them during whole-class, group, and pair discussions. However, their expectations for students' usage of the frames differed, with teachers in classes with special education and ELs providing more explicit scaffolds and reinforcements of how students could practice the language. Helen and Juanita particularly designated the specific levels of language that they would like students to use, and gave example student responses based on this designated baseline.



Teacher Noticing	Sub-codes	1	2	1	2	1	2
Object	CS vocabulary	🟦	🟦			▨	▨
	CS concepts		🟦				
Level	Whole-class	🟦				▨	
	Particular students		🟦	🟫	🟫	▨	▨
Elaboration	Basic	🟦		🟫		▨	
	Mixed	🟦	🟦		🟫		▨
	Focused						

🟦 Ellen 🟫 Helen ▨ Juanita

Fig. 1. Teacher Noticing. Colored boxes indicate occurrences of codes. 1 & 2 denote teacher noticing at the first and second design meeting.

In sum, although employing quite similar representation and representation strategies, teachers adjusted these strategies to classroom needs. The two teachers who focused on learners with special needs and ELs particularly emphasized using the reflection tool to reinforce CS concepts and vocabulary, as well as assess students.

B. Teacher Noticing & Instructional Revision

The variation in pedagogical goals and approaches may also influence how teachers employed Flipgrid to improve CS instruction. We document teachers' reactions to students' video reflections at the two design meetings, particularly the object, level, and depth of their noticing. We highlight how their noticing of students' CS vocabulary and concepts shifted over time and informed insights for instructional improvements. Fig. 1 provides an overview of teachers' varied noticing.

1) First iteration: Focus on Vocabulary. Teachers mostly focused on students' vocabulary use, rather than CS concept (i.e., algorithm, abstraction) in the first design meeting. Juanita, for example, was baffled when pointing out that she did not see any of the vocabulary they were teaching in student reflections. Teachers commented that students were using a range of everyday language to describe their projects, but were not yet using the target vocabulary.

Whole-class versus Individuals. In addition, pedagogical visions influenced the level (whole-class versus individual students) that teachers identified with student data. The level of noticing in the interviews revealed important insights into how teachers anchored their goals and problems of practice. Consider teachers' reactions to student videos:

> Ellen: You could see some of them using the vocabulary but they got carried away with the acting.
>
> Juanita: You know, even though we get around, we don't get around to every single one. I am looking at Natalie. I had to go see her stuff because she looked a little lost. Very quiet. Very compliant. It doesn't mean that she doesn't need help on it.

Whereas Ellen made global comments about her students' presentation and vocabulary as a class, Helen and Juanita anchored their answers in specific students who they considered as needing the most attention and showing the most critical changes. For example, Juanita wanted to see how her more ”quiet, compliant” students interacted with the codes. These insights overlapped with the teachers' anchoring in individual cases when reflecting on lesson planning and tool adoption.

Descriptive, not yet Focused Elaboration. The depth of teachers' noticing pertains to the extent to which teachers provide interpretive comments of students' learning patterns, and propose alternative strategies. After commenting on students' vocabulary, two of the teachers immediately brainstormed ways to increase the use of the target vocabulary. Juanita pointed out that students' limited usage of CS vocabulary in the first set of videos may result from the open-ended nature of the task and not the lesson or teaching style. Upon this realization, Ellen and Helen co-planned and proposed instructional changes. For assessment, they incorporated the vocabulary into their assignment, reminding students to use words like “algorithm”, “parallel”, or “initialization”. For instruction, following the first design meeting, teachers explicitly taught the vocabulary and reinforced it regularly. Helen, for example, created her own vocabulary printouts, with visual illustrations and definitions, and handed these out to her students so they could follow throughout the lessons.

2) Second Iteration: Shift towards CS Concepts & Student Everyday Sensemaking. There appears to be a shift in teacher noticing of CS vocabulary and concepts. In the first design meeting, teachers focused more on seeing whether students employed the target vocabulary. In the second meeting, teachers still attended to students' vocabulary use, but with an increasing focus on precision. Consider the following reactions in the second design meeting after teachers watched a Flipgrid response, where a student was describing his project while explaining the programming concept “parallelism”. Part of the student's video stated:

> What I learned from my classmates' about me is that I could have multiple backgrounds, multiple sprites, and multiple parallelism in each step. So for example, let's say I have a parallelism for when I press M and it will move to x 25 y -15 and the background will change as well when I click M.

In response, Juanita noticed that “He called them backgrounds and not stages, sometimes kids say ‘it won't move’ and they are referring to the sprites, not the stages”, and Ellen commented on the specificity of his description and the use of the target concept, “parallelism”. In this instance, Juanita and Ellen were attending to the specific vocabulary (“stage”, “background”, “parallel”) in student talk. However, their comments also suggested evaluation of how correctly the student was using the vocabulary and the extent to which vocabulary usage was related to conceptual understanding.

In addition, there appeared to be a shift in vocabulary goals. When seeing that the number of CS vocabulary did not increase substantially from the first to the second unit, Juanita reasoned that it was ”natural for kids this age to use their everyday language”, and that despite the limited vocabulary, she saw the values in students' talking about code in relation to their personal experiences. The teacher had moved from noticing specific vocabulary to student everyday language and how they used this language to talk about their codes.

Anchored Noticing Levels. The teachers continued to anchor their responses on their instructional focus on student engagement. Helen, for example, directed her attention to a special education student she had been following closely. The student demonstrated emerging use of CS vocabulary and concepts (e.g., sprite, set size) when explaining his Scratch project. Part of his response states ”This is the fish. You click on sprite. Right there. And fish 5 escape by 25% size for one second”. After watching the video, Helen did not emphasize the content of the talk as much as the fact that the student was engaging.

> There's a personality thing. I have a student who struggles a lot with panic attacks. He has many labelled needs. We're at a pace when we're going a lot more than he could handle. And he has meltdowns. And in this Flipgrid just about himself he's like the happiest kid. And he is just talking and talking.

Deeper Elaboration toward Instructional Practices. Although primarily descriptive (such as Helen's response above), the elaboration in some teachers' comments has moved from general (e.g., ”Students did great”) to specific instances of student display of CS vocabulary and concepts. The depth of elaboration has developed from basic—forming general impressions without grounding observations in evidence, to mixed—providing interpretative statements of student learning. For example, Helen and Juanita stated that they wanted to embed the reflection activities early in the next iteration of the curriculum, based on how deeply engaged students were with the reflections. The teachers indicated that the reflections would provide an early assessment of students' CS understanding, as well as an opportunity to invite students to practice disciplinary language both individually and collaboratively. Although their suggestions imply pedagogical strategies, their proposals did not directly draw from specific evidence of student learning, and thus are not yet at the focused level.

VI. Discussion

A. Pedagogical Visions & Knowledge Influence What and How Teacher Notice

The design meetings and interviews provided teachers the opportunity to make sense of students' demonstration of CS vocabulary and conceptual understanding through reflections. In general, teachers were able to notice students' use of vocabulary and CS concepts, although they tended to attend to students' vocabulary in greater details than conceptual understanding. Within noticing of vocabulary, there was a shift in focus from the target CS vocabulary to everyday language.

A possible explanation is because teachers did not receive formal training in CS education, they more comfortably drew on a familiar domain (i.e., vocabulary and strategies for language development). As teachers gained more experiences with the CS curriculum, they began to notice the curriculum's conceptual focus and the multiple types of discourse–both everyday sensemaking and CS vocabulary—-that students employ to present their understanding. Attention to students' use of both everyday and disciplinary language to express students' sensemaking has been called for in science, math, and increasingly in CS [11], [13], [22], [26]. More structure for professional development should be implemented to support teachers in learning to notice the details of student computational thinking—-to learn to identify how they are reasoning and connecting concepts, rather than just vocabulary.

Analyses also revealed the nuances in teachers' levels of noticing. Even though Ellen and Helen taught at the same school with the same materials, Helen, who consistently related her classroom experiences and pedagogical strategies to specific student populations, appeared to anchor noticing in individual student cases. Having a stronger impression of individual and student pairs engaging in CS discussion, Helen wanted to foster these activities in her next iteration of the curriculum. This is an encouraging sign that student reflection provided a basis for teachers to gather evidence of their learning and derive strategies [3].

B. Align Tools with Teachers Beliefs to Promote Use

We seek to understand the characteristics of teachers who voluntary adopted reflective pedagogical tools to assess students' CS vocabulary and understanding of concepts. An overarching finding was the teachers' internalization as learners alongside their students. This aligned with prior frameworks on teachers' technology learning and adoption trajectory [23]. This framework posits that teachers move from being "learners" who acquire the knowledge and skills to perform tasks to "adopters" that experiment with the tasks in their classrooms. After gaining an understanding of task management, teachers become "co-learners" with more focus on the relationship between the curriculum and technology. While teachers in this study have consistently positioned themselves as learners of the curriculum, in testing out tools such as Flipgrid in their classrooms and developing instructional strategies for improvement, they shifted to "adopters" and "reaffirmers", increasingly gaining agency over the tool and curriculum.

Tool adoption was also associated with specific pedagogical goals. The tool appears to match with the three teachers' tendency to employ reinforcement and multiple representations. The reflection on pedagogical goals and strategies from Juanita, Helen, and Ellen reveals their perceptions of learning and teaching in terms of students' own experiences and knowledge base, and their willingness to adopt technology-mediated experiences to help students learn CS in their own terms. Prior research states that technology whose affordances align with a teacher's pedagogical styles is more likely to be positively received and employed [19].

C. Implications & Future Work

Findings from this study have implications for developing reflection tools in CS curriculum and supporting teachers' use of those tools. First, we found that teachers with different classroom contexts chose to adapt the reflection activity for an array of purposes. The tool fosters values (i.e., multiple forms of student engagement and formative assessment) that teachers identified with. A practical implication for broader uptake of CS curricular initiatives is to frame them in terms of shared values with teachers–such as promoting student reflective voice. Our experience shows that aligned pedagogical visions can promote positive reception from teachers who are not yet familiar with CS discipline and instructional tools.

Second, pedagogical stances and content knowledge influenced teachers' interpretation of student reflections. In particular, teachers tended to attend more to student vocabulary than conceptual understanding. A direction for future work is to create opportunities for teachers to develop more focused noticing of students' CS practices, beyond counts of vocabulary. Emerging models for teacher learning in CS education include instructional coaching [15], credential programs [10], and sustained professional learning communities [12]. A recent evaluation of CS coaching models indicates that co-planning and co-teaching are critical to providing support to teachers [15]. We are exploring the systematic integration of student reflection and programming artifact in design cycles with teachers to better understand how to guide teachers towards instructional decisions based on their insights. Co-planning models such as the case of Ellen and Helen may also encourage teachers to converse about student learning and propose instructional changes. Our future work will also examine the potential differences in teachers' noticing when they are reflecting in group (this study) or individually.

Third, teachers used insights from student responses to propose instructional modifications, particularly to employ more scaffolds, diversify grouping strategies, and embed the tool as a formative assessment early in the school year. Future work can leverage teachers' insights to adapt tool use to classroom contexts and track the impact of instructional adjustments on student learning and teacher noticing. Following larger samples of teachers with different experience may yield valuable information about a broader range of pedagogical stances, strategies, and noticing in computing education.

D. Limitations

Findings from this study should be interpreted in light of several limitations. First, the small sample size limits the findings' generalizability. Second, because the teachers were experiencing other aspects of the CS curriculum between the first iteration of the tool and the second design meetings/teacher interviews, other factors may have influenced how teachers came to analyze instruction and develop pedagogical strategies. Third, the individual interviews and design meetings prompted teachers to reflect in retrospect, and thus may not have fully captured their in-the-moment noticing of students' learning as they were watching the video reflections.

VII. Conclusion

This cross-case analysis reveals the characteristics of teacher adopters of a reflective pedagogical tool to reinforce CS vocabulary and concepts in elementary grades. Findings from this study reveal the affordances of using student video reflection on their programming projects as a form of authentic assessment, reinforcement, engagement, and facilitator of teacher noticing for instructional improvements. These insights are crucial in light of expanding computing education initiatives in K-12 education [13], [15], [16], accompanied by the need for professional development to prepare teachers who may not have been formally exposed to computing education.

Acknowledgment

We acknowledge the valuable feedback from the anonymous reviewers, and the teachers and students for their continued participation in this project.

References

[1] D. August, M. Carlo, C. Dressler and C. Snow, "The critical role of vocabulary development for English language learners," *Learning Disabilities Research Practice*, vol. 20, no.1, pp. 50-57, Jan. 2005. https://doi.org/10.1111/j.1540-5826.2005.00120.x

[2] D. L. Ball and D. K. Cohen, "Developing Practice, Developing Practitioners: Toward a Practice-Based Theory of Professional Education," in *Teaching as the learning profession: Handbook of policy and practice* , G. Skykes and L. Darling-Hammond, Ed. San Francisco: Jossey-Bass Inc., 1999, pp. 3–32. https://doi.org/10.1037/0022-3514.90.4.644

[3] T. Barnhart, and E. A, van Es, "Studying teacher noticing: Examining the relationship among pre-service science teachers' ability to attend, analyze and respond to student thinking," *Teaching and Teacher Education*, vol. 45, pp. 83–93, Jan. 2015. https://doi.org/10.1016/J.TATE.2014.09.005

[4] J. Creswell, and C. Poth, *Five qualitative approaches to inquiry. In Qualitative inquiry and research design: Choosing among five approaches*. Thoasand Oaks, CA: Sage Publications, 2016.

[5] J. Blickenstaff, "Women and science careers: leaky pipeline or gender filter?," *Gender and education*, vol. 17, no. 4, pp. 369-386, Oct. 2005.

[6] B. A. Brown, J. M. Reveles and G. J. Kelly, "Scientific literacy and discursive identity: A theoretical framework for understanding science learning," *Science Education*, vol. 89, no. 5, pp. 779–802, Sep. 2005. https://doi.org/10.1002/sce.20069

[7] Q. Burke, W. I. O'Byrne, and Y. B. Kafai, "Computational Participation: Understand coding as an extension of literacy instruction," *Journal of Adolescent & Adult Literacy*, vol. 59, no. 4, pp. 371–375, Jan. 2016. https://doi.org/10.1002/jaal.496

[8] F. Erickson, "On Noticing Teacher Noticing," in *Mathematics teacher noticing: Seeing through teachers' eyes*, M. Sherin, V. Jacobs and R. Philipp, Ed. New York, NY: Routledge, 2011, pp. 47–64. https://doi.org/10.4324/9780203832714-13

[9] National Academies of Sciences, Engineering, and Medicine. "English Learners in STEM Subjects: Transforming Classrooms, Schools, and Lives," in *The National Academies Press*, Dec. 2018. https://doi.org/10.17226/25182

[10] J. Gal-Ezer, and C. Stephenson, "Computer science teacher preparation is critical," *ACM Inroads*, vol. 1, no. 1, pp. 61-66, Mar. 2010. https://doi.org/10.1145/1721933.1721953

[11] K. Gomez, "Negotiating discourses: Sixth-grade students' use of multiple science discourses during a science fair presentation," *Linguistics and Education*, vol. 18, no. 1, pp. 41–64, Mar. 2007. https://doi.org/10.1016/J.LINGED.2007.03.002

[12] J. Goode, J. Margolis and G. Chapman, "Curriculum is not enough," in *Proceedings of the 45th ACM Technical Symposium on Computer Science Education - SIGCSE '14*, 2014, pp. 493–498. https://doi.org/10.1145/2538862.2538948

[13] S. Grover and R. Pea, "Using a discourse-intensive pedagogy and android's app inventor for introducing computational concepts to middle school students," in *Proceeding of the 44th ACM Technical Symposium on Computer Science Education - SIGCSE '13*, 2013, pp. 723-728. https://doi.org/10.1145/2445196.2445404

[14] M. Israel, J. N. Pearson, T. Tapia, Q. M. Wherfel and G. Reese, "Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis," *Computers Education*, vol. 82, pp. 263–279, Mar. 2015. https://doi.org/10.1016/J.COMPEDU.2014.11.022

[15] M. Israel, M. Ray, W. Maa, G. Jeong, C. Lee, T. Lash, ... and V. Do, "School-Embedded and District-Wide Coaching in K-8 Computer Science: Implications for Including Students with Disabilities," *Journal of Technology and Teacher Education*, vol. 26, no. 3, pp. 471–501, Jul. 2018. https://www.learntechlib.org/p/181938/

[16] M. Israel, Q. M. Wherfel, J. Pearson, S. Shehab and T. Tapia, "Empowering K-12 Students With Disabilities to Learn Computational Thinking and Computer Programming," *Teaching Exceptional Children*, vol. 48, no. 1, pp. 45–53, Sep. 2015. https://doi.org/10.1177/0040059915594790

[17] O. Lee and S. H. Fradd, "Science for All, Including Students From Non-English-Language Backgrounds," *Educational Researcher*, vol. 27, no. 4, pp. 12–21, May 1998. https://doi.org/10.3102/0013189X027004012

[18] S. B. Merriam, *Qualitative research and case study applications in education*. San Francisco, CA: Jossey-Bass Publishers, 1998. https://eric.ed.gov/?id=ED415771

[19] D. S. Niederhauser and D. L. Lindstrom, "Instructional technology integration models and frameworks: Diffusion, competencies, attitudes, and dispositions," in *Handbook of Information Technology in Primary and Secondary Education*, J. Voogt et al., Ed. Springer, 2018, pp. 1-21.

[20] D. H. Rose and A. Meyer, A practical reader in universal design for learning. Cambridge, MA: Harvard Education Press, 2006.

[21] S. Sentance and A. Csizmadia, "Computing in the curriculum: Challenges and strategies from a teacher's perspective," *Education and Information Technologies*, vol. 22, no. 2, pp. 469–495, Mar. 2017. https://doi.org/10.1007/s10639-016-9482-0

[22] M. Sherin, V. Jacobs and R. Philipp, Ed. New York, NY: Routledge, 2011.

[23] L. Sherry, S. Billig, F. Tavalin and D. Gibson, "New insights on technology adoption in communities of learners," in *Society for Information Technology Teacher Education International Conference*, Association for the Advancement of Computing in Education (AACE), 2000, pp. 2044-2049.

[24] V. Talanquer, D. Tomanek, D. and I. Novodvorsky, "Assessing students' understanding of inquiry: What do prospective science teachers notice?" *Journal of Research in Science Teaching*, vol. 50, no. 2, pp. 189–208, Feb. 2013. https://doi.org/10.1002/tea.21074

[25] V. Talanquer, M. Bolger, and D. Tomanek, "Exploring prospective teachers' assessment practices: Noticing and interpreting student understanding in the assessment of written work," *Journal of Research in Science Teaching*, vol. 52, no. 5, pp. 585–609, May 2015. https://doi.org/10.1002/tea.21209

[26] E. A. van Es and M.G. Sherin, "Mathematics teachers' "learning to notice" in the context of a video club," *Teaching and Teacher Education*, vol. 24, no. 2, pp. 244–276, Feb. 2008. https://doi.org/10.1016/J.TATE.2006.11.005

[27] E. A. van Es, "A framework for learning to notice student thinking," in *Mathematics teacher noticing* (pp. 164-181). M. Sherin, V. Jacobs and R. Philipp, Ed. New York, NY: Routledge, 2011, pp. 164-181.

[28] R. Varma, "Making computer science minority-friendly," *Communications of the ACM*, vol. 49, no. 2, pp.129-134, Feb. 2006.

[29] M. Walshaw and G. Anthony, "The Teacher's Role in Classroom Discourse: A Review of Recent Research Into Mathematics Classrooms," *Review of Educational Research*, vol. 78, no. 3, pp. 516–551, Sep. 2008. https://doi.org/10.3102/0034654308320292

[30] J. Wang, H. Hong, J. Ravitz, and S. Hejazi Moghadam, "Landscape of K-12 Computer Science Education in the U.S," in *Proceedings of the 47th ACM Technical Symposium on Computing Science Education - SIGCSE '16*, 2016, pp. 645–650. https://doi.org/10.1145/2839509.2844628

[31] A. Yadav, C. Mayfield, N. Zhou, S. Hambrusch and J. T. Korb, "Computational thinking in elementary and secondary teacher education," *ACM Transactions on Computing Education (TOCE)*, vol. 14, no. 1, pp. 5, Mar. 2014.