How “Wide Walls” Can Increase Engagement: Evidence From a Natural Experiment in Scratch

Sayamindu Dasgupta
University of Washington
Seattle, WA
sdg1@uw.edu

Benjamin Mako Hill
University of Washington
Seattle, WA
makohill@uw.edu

ABSTRACT
A core aim for designing constructionist learning systems and toolkits is enabling “wide walls”—a metaphor used to describe supporting a diverse range of creative outcomes. Ensuring that a broad design space is afforded to learners by a toolkit is a common approach to achieving wide walls. We use econometric methods to provide an empirical test of the wide walls theory through a natural experiment in the Scratch online community. We estimate the causal effect of a policy change that gave a large number of Scratch users access to a more powerful version of Scratch data structures, effectively widening the walls for learners. We show that access to and use of these more powerful new data structures caused learners to use data structures more frequently. Our findings provide support for the theory that wide walls can increase engagement and learning.

Author Keywords
learning; design; computers and children; creativity support tools; evaluation; constructionism; Scratch

ACM Classification Keywords
K.3.2 Computers and Education: Computer and Information Science Education; H.5.2 Information Interfaces and Presentation (e.g., HCI): User Interfaces; H.5.3 Information Interfaces and Presentation (e.g., HCI): Group and Organization Interfaces—*Evaluation/methodology*

INTRODUCTION
Inspired by the work of mathematician and epistemologist Seymour Papert, a large and growing body of research in HCI has focused on the design of constructionist toolkits for learning. Well-known examples of these toolkits include Logo [30, 1], StarLogo [34], NetLogo [43], Lego Mindstorms [39], Storytelling Alice [19], Scratch [36], and MIT App Inventor [44]. One of the core aims of constructionism, that of empowering the learner to use knowledge in personally meaningful ways, has driven the design of these toolkits. A learning system, Papert argued, should offer learners immediate opportunities to use their new knowledge in ways that have “real personal importance” to the learner [31] and as “a source of personal power” [30]. This principle of design is described as the “power principle” in constructionism. Because learners have a diverse set of interests and passions, it is a corollary of the power principle that designers should enable a wide range of possible uses of knowledge—a principle that is described as designing for “wide walls” (i.e. supporting a diverse range of creative possibilities and outcomes) [37, 38].

Designers of constructionist learning systems have often used the principle of wide walls to guide their design. Toward this end, designers have introduced affordances that enable learners to use new ideas in personally relevant ways. A large body of qualitative and quantitative work in HCI, computing education, and the learning sciences has shown that many of these toolkits have been enormously successful in engaging users and promoting learning. However, because most constructionism-inspired toolkits already incorporate the wide-walls principle, finding sufficient variation in “wideness” remains a challenge. We know of no effort to formally or quantitatively evaluate the theory that widening walls can promote participation, engagement, and learning.

In this paper, we offer an empirical test of the wide walls principle by utilizing a natural experiment in the Scratch online community. We estimate the effect of an exogenous change in Scratch policy that granted a large number of Scratch users access to more powerful Scratch data structure called a Scratch

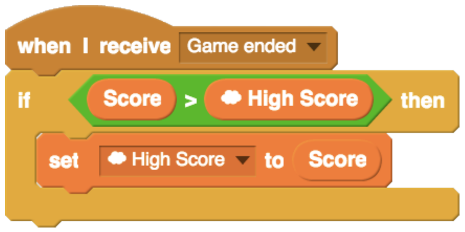


Figure 1. Scratch code for a program that uses an SCV to keep track of global, community-wide high-scores. This script compares the value of the SCV *High Score* to the score from the recently concluded game. If the score is higher, then *High Score* is set to the score. The cloud icon indicates that the variable *High Score* is an SCV.

CHI 2018 April 21–26, 2018, Montreal, QC, Canada
© 2018 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-5620-6/18/04.
DOI: https://doi.org/10.1145/3173574.3173935

Cloud Variable (SCV, see Figure 1). This change effectively widened the walls afforded to learners by Scratch variables. We show that access to and use of these data structures caused learners to increase their use of data structures in general—including the less powerful forms they had access to previously. We observed 13,967 users who were affected by the change and compared 45,809 projects that they created immediately before and after. Our findings represent empirical evidence in support of the wide walls principle and hold important lessons for designers of learning systems and creativity support tools.

BACKGROUND

Constructivist theories of learning suggest that learning happens through active building of knowledge structures in the mind rather than through passive transmission of knowledge from the teacher to the student [33]. Building on constructivism, Papert theorized that this process of building knowledge structures "happens especially felicitously in a context where the learner is consciously engaged in constructing a public entity" [32]. This framework of learning is the cornerstone of a larger body of theoretical and empirical research that constitutes what is now known as constructionism, a body of work spanning HCI, the learning sciences, and computing education [14, 18, 17].

In his book *Mindstorms: Children, Computers, and Powerful Ideas*, Seymour Papert put forth the argument that in order to support learning, an educational tool should be "appropriable" [30]. He went on to unpack appropriability into three principles. One of the principles was the "power principle" which was defined as:

> [An idea] must empower the learner to perform personally meaningful projects that could not be done without it [30].

Later, Papert's student and collaborator Mitchel Resnick built explicitly upon this idea to propose the principle of "wide walls." Resnick wrote:

> We know that kids will become most engaged, and learn the most, when they are working on projects that are personally meaningful to them. But no single project will be meaningful to all kids. So if we want to engage all kids—from many different backgrounds, with many different interests—we need to support a wide diversity of pathways and projects [35].

Resnick's principle of wide walls, which posits engagement as a pathway to learning, has seen significant adoption by communities that design for learning and creativity. One of the most prominent implementations of the principle is in the Scratch programming environment. The Scratch language was designed to support many pathways including a wide range of affordances [36]. As a result, Scratch users learn programming by expressing themselves in diverse ways. Scratch projects can include interactive animations, games, stories, simulations, music videos, and more.

Beyond Scratch and constructionist learning systems, the principle of wide walls has been widely adopted by designers of creativity support tools [41]. It has been frequently cited as a principle in, for example, the design of digital fabrication systems [2], 3D modelling software [5], web-based self-service systems [12], musical interaction [13], CAD software [20], systems for teaching statistics [23], and physical construction kits [38].

Although deeply influential and widely cited, the wide walls principle has never, to our knowledge, been posed as a hypothesis to be tested empirically. In this paper, we do so by breaking the principle into two distinct claims. First, we turn to the primary claim embedded in the wide walls principle: that widening walls increases engagement. Increased engagement might occur by eliciting participation from a broader range of potential users including individuals uninterested in using a similar system or feature with a narrower design space. It might also occur by encouraging learners already using a a narrower system to use the system more frequently and in a broader range of contexts. In both cases, increased engagement would manifest as increased use of a system or feature. Thus, we hypothesize that (H1) *widening the design space accessible by a feature will lead to more frequent use of that feature.*

The second claim of the wide walls principle is that greater engagement from wider walls leads to increased learning. This claim is rooted in the assumption that in order to take advantage of the new possibilities afforded by wider walls, users must engage with the original design space first. It also assumes that through that process of engagement, users will learn the underlying fundamental concepts that were necessary in the first place to use features in the original design. Based on this reasoning, we hypothesize that (H2) *widening the design space accessible by a feature will lead to more frequent use of that feature within the context of the original design space*. Although other mechanisms may be possible, we believe that the most plausible explanation for why the introduction of wider walls would cause an increase in the use of a feature within its original design space—a space that users had identical access to before—is that increased engagement caused by wider walls in turn increased learning about underlying fundamental concepts.

EMPIRICAL SETTING

Scratch

Scratch is a graphical, block-based programming language designed for children aged 8–15 [36]. Scratch programs are created by dragging and dropping graphical "blocks" of code and snapping them together to form "scripts," as shown in Figure 1. These scripts control the behavior of on-screen graphical characters. Scratch is situated within an online community where Scratch users can share their projects, comment on each others' work, bookmark and show appreciation for others' projects, and follow each other's user activity [25].[1] Since its launch in 2007, the Scratch online community has hosted more than 20 million registered users and more than 24 million shared projects.[2]

[1]https://scratch.mit.edu/ (https://perma.cc/U4AP-3FWF)
[2]https://scratch.mit.edu/statistics/ (https://perma.cc/4PBA-6MFJ)

Scratch Cloud Variables

Scratch users store and retrieve data using scalar variables and vector lists. More commonly used than lists, scalar variables are used to store scores in games, user input, or other data useful for a program. That said, variables are initialized each time a Scratch project is reloaded, which means variables cannot be used to store data that persists between visits to a Scratch project. It also means that data stored in a variable by one user is not visible to other users.

The SCV system was introduced as a new feature in the Scratch programming language in 2013 [6]. An SCV extends the standard scalar variable in Scratch by making it *persistent* and *shared*.[3] *Persistent* means that data in variables are not reset each time the Scratch project is run. Score data can, for example, be stored to create global leader-boards (as in in Figure 1). *Shared* means that that Scratch users running the same project simultaneously will each be able to read and update common data in an SCV. For example, user-input data can be shared to create projects such as real-time chat-rooms.

The SCV system's design was inspired directly by Papert's power principle and Resnick's call for wider walls. In particular, the motivation behind SCVs was to enable everyone to realize the "wide range of creative possibilities" enabled by the ability to store and access data online [6]. One of the two stated goals of the system was to "connect to [learners'] personal interests" [6]. Additionally, to ensure alignment with learners' personal interests, the design of SCVs drew from practices that were already prevalent in the Scratch online community. For example, the design took direct inspiration from the surveys children were designing with Scratch, which repurposed the comments section of the community website to collect response data. The SCV system sought to provide wider walls and greater power by allowing variables to be used in novel ways that were grounded in activities that learners in the community were already attempting to engage in. Since the public launch of SCVs in 2013, Scratch users have created a wide variety of projects, such as global high-score lists for games, surveys, collaborative art canvases, collaborative clicking games, chatrooms, and multiplayer games [7].

The SCV system stores the value of the variables in a central server and propagates changes to variable values to all connected Scratch users in real-time. While programming, a user can mark a variable as being a "Cloud Variable" through a checkbox (see Figure 2a), effectively setting an orthogonal property of the variable. Apart from the presence of this orthogonal property, SCVs have a programming grammar (`get()`, `set()`, `increment()`) that is identical to normal Scratch scalar variables (Figure 2b). Each SCV's scope is limited to the project it is used in—the value of the *High Score* SCV in Figure 1 would be visible to any user accessing that project, but not beyond (i.e., there is no between-project variable sharing). If multiple users are accessing a project at the same time, when one user causes a change to an SCV, all other users see the value of that variable updated in real time.

[3]A Scratch Cloud List was also developed as a prototype, but was never publicly deployed due to moderation concerns.

Because SCVs open up possibilities for powerful social affordances, such as real-time chatrooms, moderators of the Scratch website were concerned about potentially problematic uses of the system. As a mitigation measure, the SCV system was implemented so that only users with a certain level of prior activity on the Scratch website would be able to access and use them. This access control mechanism for SCVs took advantage of an existing status system in Scratch. Upon joining the community, every Scratch user is labeled as a *New Scratcher* on their user profile page. After a certain amount of activity, measured in project creation, social engagement, and other factors (the specifics have never been publicly disclosed), a *New Scratcher* becomes a *Scratcher* and is informed of this change through a notification on the Scratch website.

Users with *Scratcher* status have access to certain privileges, such as the ability to post in the forums at a high rate. This mechanism was extended so that only users with *Scratcher* status can access the SCV system. If a *New Scratcher* clicks the "See Inside" button on a project using SCVs, they receive a pop-up notification stating that SCVs are not usable by them; any SCVs act as normal Scratch variables in that they are neither persistent nor shared.

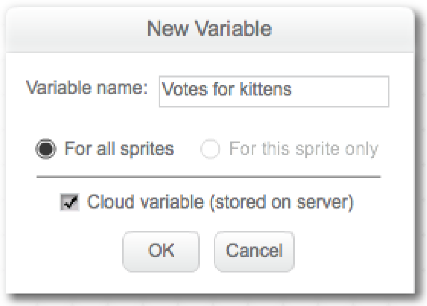
Natural Experiment

Our analysis seeks to provide causal evidence for the wide walls theory. The most common way to provide causal evidence in HCI research is through an experiment in which random assignment is used to provide a group of users with access to a new feature or affordance. User behavior is then tracked and the effect of the feature is identified as the set of differences in behavior between the users who received the feature (the treatment group) and the users that did not (the control group). In this paper, we take advantage of a natural experiment. Although access to the feature in question in our natural experiment was not assigned randomly, it was assigned in a way that was exogenous to users' behavior and can be considered "as if random."

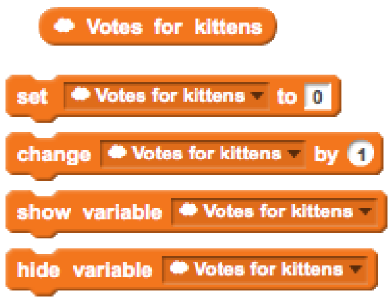
Our natural experiment occurred in the form of an unannounced change to the unpublished criteria used to grant users *Scratcher* status. This change, made on July 16, 2015, resulted in many users of Scratch being given access to SCVs through no action of their own. The change was made for reasons entirely unrelated to SCV and its use. Prior to July 16, users needed to have made 10 comments on the website to attain *Scratcher* status. On July 16, the requirement to have left comments was dropped. This change was made because website moderators felt that this requirement was too restrictive. There were many users who had met all the other thresholds for gaining *Scratcher* status except for the comment count. When this requirement was dropped, these users were all promoted to *Scratcher* status. Thus, due to this change, a number of users suddenly found themselves with *Scratcher* status—and hence, access to SCV.

DATA AND MEASURES

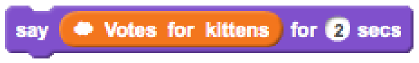
Our unit of analysis is the Scratch project and our dataset consists of projects shared by users who were affected by


New Variable
Variable name: Votes for kittens
For all sprites
For this sprite only
Cloud variable (stored on server)
OK
Cancel

(a) Creating a Cloud variable


Votes for kittens
set Votes for kittens to 0
change Votes for kittens by 1
show variable Votes for kittens
hide variable Votes for kittens

(b) Cloud variable grammar



(c) Using a Cloud variable

Figure 2. Scratch Cloud variables. (a) SCVs are specified when declaring a variable. (b, c) The grammar for using them is identical to normal Scratch variables [7].

the change described above and who were active in a way that would have let them take advantage of access to SCV. Drawing from the database of all Scratch users, we used three inclusion criteria. First, we considered only users who joined the Scratch online community between January 1st 2015 and July 1st 2015. We did not consider users before January 1st, in order to limit the scope of our data collection and because these users were relatively unlikely to be active. We did not consider accounts created after July 1st because one of the criteria for *Scratcher* status is that a user's account must be at least 2 weeks old. Second, we included only users who received *Scratcher* status due to the change. Third, we included only users who, after getting *Scratcher* status, had shared at least one project that was not a modified version of another users' project. A further 52 were excluded due to errors in our logging software that recorded incorrect information on SCV use or because users were found to have attempted to author projects with SCV via an upload workaround before being promoted to *Scratcher* status.[4] These inclusion criteria resulted in a dataset that included 13,967 users.

Once we had identified these users, we collected data on 180,065 *de novo* projects these users had publicly shared. We define *de novo* projects as those that were created from a blank-slate editor and were not remixed from an existing project. We focused on these works because we felt that the presence of data structures in a remixed project might not have indicated meaningful engagement or learning. For each project, we collected a series of measures capturing qualities of projects and their creators at the time they shared the projects in question. These measures included the cumulative number of projects (remixed and *de novo*) the learner had shared at the time they shared the project in question (*Share Count*). Although we excluded remixed projects in general from our dataset, we included remixes in *Share Count* because we sought to measure general experience. We also included a dichotomous variable that indicated whether the learner had received *Scratcher* status or not at the time that the project in question was shared (*Is Scratcher*?), and a dichotomous variable indicating whether the learner has used SCVs in a previously shared *de novo* project (*Used Cloud Data*). If the learner had used SCVs in this way, this variable was coded 1, otherwise 0. This variable was coded 1 even if the user subsequently removed the SCV before sharing the project.

[4]The attempts were ineffective because authentication tokens for engaging in Cloud data operations were generated on the server-side.

Our dependent variables sought to measure engagement and learning. Measuring learning quantitatively is always challenging. This difficulty is exacerbated by the fact that Scratch is an informal learning community with no fixed pathways for participation, no specific lesson plans, and no standardized forms of evaluation. We built on a method used in previous work that mapped certain Scratch blocks to computational thinking concepts (e.g., `forever` blocks map to the concept of loops) [4, 8]. Because the SCV system is focused on supporting engagement with and learning about data [6], we developed a dichotomous dependent variable (*Uses Data Structures*?) coded 1 if a project uses data structures and 0 otherwise. This measure includes any use of data structures, including SCVs.

Given a positive causal relationship between the use of SCVs and *Uses Data Structures*?, it is still difficult to disentangle the degree to which SCVs were simply more novel or more interesting to users from the degree they caused children to learn about data structures more generally. If users engaged with data structures more in the form of SCVs, this may have been because SCVs' wide walls encouraged them to learn and to become more comfortable using data structures. However, these users may also have understood data structures well before but were simply uninterested in using the narrow-walled variables.

In H2, we hypothesized that users would be motivated to learn about data structures in general. To test H2, we constructed a version of our dataset by removing all instances of shared projects using SCVs. Our second dependent variable, constructed the same way as the first, used this data. Because it did not include any use of SCVs, this variable represents the use of data structures within their original design space. We called this variable *Uses Data Structures*?$_{WOSCV}$ (for "without SCVs"). Although it is important to recognize that this variable is not a direct measure of learning, a positive relationship between using SCVs and *Uses Data Structures*?$_{WOSCV}$ means that using SCVs caused users to also use non-SCV data struc-

Variable	Proportion
Is Scratcher?	0.47
Used Cloud Data?	0.01
Uses Data Structures?	0.34

Table 1. Proportions for dichotomous variables of interest used in our analysis.

tures more frequently and in ways that we believe constitute evidence in support of the theory that that wide walls promote learning.

Our reasoning is as follows. First, many quantitative studies of Scratch have measured learning as the presence of certain blocks in projects [40, 26, 45, 24, 9, 27]—an approach that has been validated by expert assessments [27]. Second, because all users in our sample had access to non-SCV variables beforehand, an increase in non-SCV variable use is difficult to explain except through increased familiarity with data structures. Using similar reasoning, Dasgupta et al. [8] interpreted an increase in block use associated with previous exposure to the same blocks in remixes as evidence of learning through appropriation.

We also considered a subset of the projects in our dataset close to the point when the criteria for *Scratcher* status was changed (the rationale for the subsetting is described in more detail in the next section). The subset used in our primary analysis included a total of 49,982 projects created by 13,967 users—4,173 of which were dropped because of missing data. Table 1 shows the proportions of all our dichotomous variables for these projects. The only count variable used in our analysis, *Share Count*, had a range of [1, 293], a median of 5, a mean of 7.84, and a standard deviation of 9.4.

ANALYTIC PLAN

Our analysis drew heavily from a body of research in econometrics that has used natural experiments to establish causality in observational data where researchers have traditionally been limited to making claims about correlation [3, 28]. Recently HCI researchers have begun analyzing natural experiments as well [29]. Borrowing terminology from randomized controlled experiments in medicine, analyses of natural experiments typically describe their key independent variable as "the treatment." In our case, the treatment is having used SCVs (*Used Cloud Variable*?).

Although a simple correlation between *Used Cloud Variable*? and *Uses Data Structures*? might show projects by that *Scratchers* who used SCV also used data structures more frequently, evidence of this relationship does not necessarily imply causation. A positive correlation might simply be due to SCV users being more motivated, more curious, or more committed to exploring Scratch's functionality. Natural experiments take advantage of changes that, although not random, place individuals into a treatment condition (i.e. individuals using SCVs) in a way that is "as if random."

In our case, the change in criteria for *Scratcher* status was such a change. Because Scratch moderators' decision to change the criteria for *Scratcher* status was sudden and unannounced, we believe that projects created by these users immediately before and after the policy change should be expected to be equal in terms of any quality we might choose to measure—with the exception of changes caused by the policy change itself. In treating the policy change as a natural experiment, we are assuming that if they had not been given access to SCVs due to the policy change, the users in our sample would have been equally likely to use data structures in their projects immediately before and after the change. If this is true, we can assume that any difference was caused by the change in policy.

Like many analyses of natural experiments, our analytic strategy consisted of two steps. First, in what is called an "intent-to-treat" (ITT) analysis, we used the dataset of users who gained access to SCV due to the policy change and compared the likelihood that projects shared immediately before and immediately after used data structures (*Uses Data Structures*?). This analysis is referred to as "intent-to-treat" because, although users with access to SCVs could use the new system, most never did. In our ITT analysis, our main independent variable was a dummy variable coded as 1 if an individual had *Scratcher* status and 0 otherwise.

Second, in what is called a "treatment-on-the-treated" (TOTT) estimate, we sought to estimate the effect of using SCV. It is possible to calculate an unbiased parameter estimate of the effect of taking up a treatment using a mathematical method known as *two-stage least-squares* (2SLS).[5] TOTT estimates using 2SLS measure what is called a "local average treatment effect" (LATE) [28]. In our case, this means that our TOTT estimate is not an estimate of the effect of using SCVs among all Scratch users as one might assume. Instead, it is the estimate of the effect of using SCVs among users who did so because they were given access through the policy change.

The following example from education policy research illustrates the intuition behind ITT and TOTT estimates as well as some of its limitations. Several studies have shown that private school students have higher levels of achievement than public school students. This does not mean that private schooling causes higher achievement, because private and public school attendees are also different in many observed and likely unobserved ways (e.g., they are wealthier and their parents might be more committed to their education). One series of studies took advantage of a natural experiment arising from a private school voucher lottery in New York City [15, 21]. Because the lottery was random, one would expect winners and losers to achieve equally (on average) before the lottery.

Of course, some lottery losers chose to send their kids to private schools anyway, and some winners sent their kinds to public school. An ITT estimate of attending private school captures the simple effect of winning the voucher lottery. If many lottery winners did not use their vouchers, we would expect these ITT estimates to be much smaller than the effect of actually attending private school. A TOTT estimate

[5]Although a full explanation of 2SLS is outside the scope of this paper, the topic is discussed in depth by several excellent textbooks [11, 3, 28].

captures the effect of attending private school conditional on having won the lottery (i.e., the TOTT estimate is a LATE limited in scope to the group of students who entered the lottery). Although the effect is causal, the children who entered private school because they won the voucher lottery are likely different from the general population in many ways.

There are a series of important issues to consider when constructing ITT and TOTT estimates from natural experiments. In our case, one issue pertains to the possibility of a general upward or downward trend in the probability that users will use data structures as they gain experience. For example, as Scratch users share projects, they may become more likely to use data structures on average. As a result, a positive shift in our dependent variables might then reflect this general trend instead of a causal effect of users being granted access to SCVs. We controlled for the experience of the Scratch user in our model in terms of the number of previously shared projects (*Share Count*). As *Share Count* is highly skewed, we applied a started log transformation (i.e., $\log(x+1)$) to the variable to better satisfy the parametric assumptions of our model. We chose to use a "bandwidth" of 4 projects, (i.e., a maximum of two projects on each side of the transition point).

This approach is described as a regression discontinuity design (RDD) in econometrics [3, 28], and the technique has been used in HCI research as well [22]. Individuals in our sample gained *Scratcher* status at different points along our trend variable *Share Count*. Because the relationship between $\log(\textit{Share Count})$ and *Uses Data Structure*? might not be linear, we fitted a series of polynomial specifications of $\log(\textit{Share Count})$ using likelihood ratio tests to determine if increases in model goodness of fit justified the more complex specifications of the variables. These tests suggested that the model with a cubic term fit best. To address estimation issues related to collinearity in our polynomial terms, we used orthogonal polynomials in our models, as returned by R's `poly()` function.

Because a single user in our dataset will share more than one project, users are measured repeatedly in our dataset. To address this concern, we used a multi-level model [42] with a random intercept term for users. Because the dependent variables in both our ITT and TOTT models are dichotomous, our models are logistic regression models. The formula for our first logistic regression model (M1) that estimates the causal effect of a user being a *Scratcher* on the probability of using data structures (i.e., our ITT estimate) is:

$$\log\left(\frac{\hat{p}(\textit{Uses Data Structure?})}{1-\hat{p}(\textit{Uses Data Structure?})}\right) = \beta_0 + \beta_1 \textit{Is Scratcher?} + \beta_2 \log(\textit{Share Count}) + \beta_3 \log(\textit{Share Count})^2 + \beta_4 \log(\textit{Share Count})^3 + [u+\varepsilon]$$

As indicated earlier, our TOTT estimates were drawn from 2SLS regressions models, which require fitting regression models in—appropriately enough—two stages. In the first stage, a regression model was fitted to the hypothesized relationship between the potentially endogenous question predictor *Used Cloud Data*? and the intent-to-treat indicator *Is Scratcher*?. Thus, in our case, the first stage model (M2a) is represented by the following equation:

$$\log\left(\frac{\hat{p}(\textit{Used Cloud Data?})}{1-\hat{p}(\textit{Used Cloud Data?})}\right) = \beta_0 + \beta_1 \textit{Is Scratcher?} + \beta_2 \log(\textit{Share Count}) + \beta_3 \log(\textit{Share Count})^2 + \beta_4 \log(\textit{Share Count})^3 + [u+\varepsilon]$$

Next, predicted values of *Used Cloud Data*? were calculated using this model. In the second stage, we used the predicted value from the first stage (denoted $\textit{Used}\ \widehat{\textit{Cloud}}\ \textit{Data?}$) instead of the potentially endogenous observed values of *Used Cloud Data*?. Thus the model (M2b) for the second stage becomes:

$$\log\left(\frac{\hat{p}(\textit{Uses Data Structure?})}{1-\hat{p}(\textit{Uses Data Structure?})}\right) = \beta_0 + \beta_1 \textit{Used}\ \widehat{\textit{Cloud}}\ \textit{Data?} + \beta_2 \log(\textit{Share Count}) + \beta_3 \log(\textit{Share Count})^2 + \beta_4 \log(\textit{Share Count})^3 + [u+\varepsilon]$$

The result of fitting this model was our TOTT estimate.

The approach described above was the same for testing both H1 and H2—the only difference being that we used $\textit{Uses Data Structures?}_{WOSCV}$ as a dependent variable for testing H2 instead of *Uses Data Structures*?, which was used for H1.

RESULTS

As a first step, we simply compared all projects shared by users immediately before gaining *Scratcher* status to all the projects shared immediately after, in terms of the proportion of each that were coded 1 for *Uses Data Structure*? (H1) and for $\textit{Uses Data Structures?}_{WOSCV}$ (H2). The results of a 2-sample test for equality of proportions on these data are shown in Table 2. For both H1 and H2, we find that there is a statistically significant difference between the proportion of projects with data structures before and after learners gained *Scratcher* status ($\chi^2 = 28.047$ and $\chi^2 = 8.027$ respectively). As explained above, because this difference might be a function of the increase in experience of learners (i.e., reflecting a general upward trend), we proceeded to fitting our regression models.

First, we considered results from our ITT models. ITT model fits for both H1 and H2 are provided in Table 3a. For H1, the parameter estimate for *Is Scratcher*? is positive ($\beta = 0.131$, $p < 0.001$). The estimate is relatively small, suggesting that the odds of a Scratch user with *Scratcher* status sharing a

Hypothesis	Diff. in prop.	χ^2	p-value
H1	0.023	28.047	< 0.001
H2	0.012	8.027	0.005

Table 2. Results of tests estimating the difference in proportions of projects with data structures between learners before and after they gained *Scratcher* status. Tests were conducted with datasets for both H1 and H2.

	ITT	
	H1	H2
Is Scratcher?	0.131***	0.080***
	(0.0001)	(0.0001)
$\log(\textit{Share Count})$	-1.155	-1.166
	(1.980)	(3.061)
$\log(\textit{Share Count})^2$	-15.084^{***}	-15.282^{***}
	(2.447)	(3.084)
$\log(\textit{Share Count})^3$	5.057**	6.259
	(1.720)	(3.365)
Constant	-0.790^{***}	-0.787^{***}
	(0.0001)	(0.0001)
Observations	45,809	45,419
Log Likelihood	-28,650.290	-28,359.030
Akaike Inf. Crit.	57,312.590	56,730.070
Bayesian Inf. Crit.	57,364.980	56,782.410
Note:	*p<0.05; **p<0.01; ***p<0.001	

(a) Results of fitting model M1 (ITT)

	2SLS	
	H1	H2
Used $\widehat{\textit{Cloud}}$ Data?	2.248***	1.095***
	(0.173)	(0.154)
$\log(\textit{Share Count})$	2.078	0.906
	(2.024)	(2.483)
$\log(\textit{Share Count})^2$	-21.918^{***}	-20.192^{***}
	(1.706)	(2.268)
$\log(\textit{Share Count})^3$	6.949***	7.637***
	(1.760)	(2.302)
Constant	-0.744^{***}	-0.757^{***}
	(0.017)	(0.017)
Observations	45,809	45,419
Log Likelihood	-28,567.400	-28,336.810
Akaike Inf. Crit.	57,146.800	56,685.620
Bayesian Inf. Crit.	57,199.200	56,737.960
Note:	*p<0.05; **p<0.01; ***p<0.001	

(b) Results of fitting model M2b (stage 2 of 2SLS)

Table 3. Results of fitting mixed-effects logistics regression models with a random intercept term for users.

project with data structures were 1.14 times that for a user who has shared the same number of projects without *Scratcher* status. Of course, this represents the *average* effect for users who have *Scratcher* status. Small estimated ITT effects are typical for treatments with low uptake, like ours. For H2, we saw a similar result: the ITT estimate for *Is Scratcher*?, though lower than in H1, was also positive ($\beta = 0.08$, $p < 0.001$). We estimated that for a learner with *Scratcher* status, the odds of sharing a project with non-SCV data structures is 1.08 times the odds for a similar user without status.

To estimate the local effect of SCVs on users who used the system due to the change, we turned to our 2SLS models. Estimates from these models are shown in Table 3b. For both models, we obtained a positive estimate that was much larger than our ITT estimates ($\beta = 2.248$, $p < 0.001$ for H1 and $\beta = 1.095$, $p < 0.001$ for H2). For H1, we find that the odds of sharing a project with data structures by a learner who had used SCVs was 9.47 times those of a learner who had not used them. For H2, we found that the odds of sharing a project with data structures (excluding projects that use SCVs) is 2.99 times those of a similar learner who had not used SCVs.

To aid in interpreting these results, Figure 3 shows the model-predicted probabilities that a project will include data structures for a series of prototypical Scratch projects. The two panels constituting the left half of the figure visualize our ITT estimates and show relatively small differences between projects' likelihood of using data structures immediately before and after the change. The first panel shows that a prototypical project by a Scratch user without *Scratcher* status who has shared 5 projects (the median value of *Share Count*) had a probability of 0.32 of using data structures, while the probability for a project by an otherwise equivalent user with *Scratcher* status was 0.35. For projects which did not use SCVs, the corresponding probabilities were 0.33 and 0.34, respectively, as shown in the second panel. Because most users did not use SCVs, this small difference is not particularly surprising. While small, the estimates are statistically significant and offer evidence of a causal effect.

The right half of Figure 3 visualizes the results from our 2SLS models and shows our TOTT estimates. For a Scratch user who had 5 shared projects and had not used SCVs, the probability of their project including data structures was 0.34; for a user who had used SCVs, the probability was 0.83. This large difference is not entirely surprising, as the observed projects included SCV use itself in the dependent variable (H1). Excluding projects with SCVs (H2), our estimate of the probability of a project including data structures for a user who had not used SCVs was 0.33; for a user who had used SCVs, the corresponding probability was 0.6. These results reflect large and meaningful changes and support of a causal effect in terms of both of our hypotheses.

THREATS TO VALIDITY

The validity of our findings may be affected by a number of threats and limitations. A first threat is due to the fact that only 3.7 percent of the users in our sample took up the treatment within our bandwidth of 4 projects. We do not believe that this reflects a methodological problem. Econometricists have established that, while it is better to have a larger difference, as long as the functions on the two sides of a cut-off or transition point are different, the analysis can be valid [3].

Another series of methodological threats stems from the possibility that the change at the center of our analysis might not have been exogenously related to our outcomes. If the change was not exogenous, it compromises our ability to draw causal conclusions. In a traditional experiment, this might happen if individuals were able to move between treatment or control groups. In our case, a similar issue might have arisen

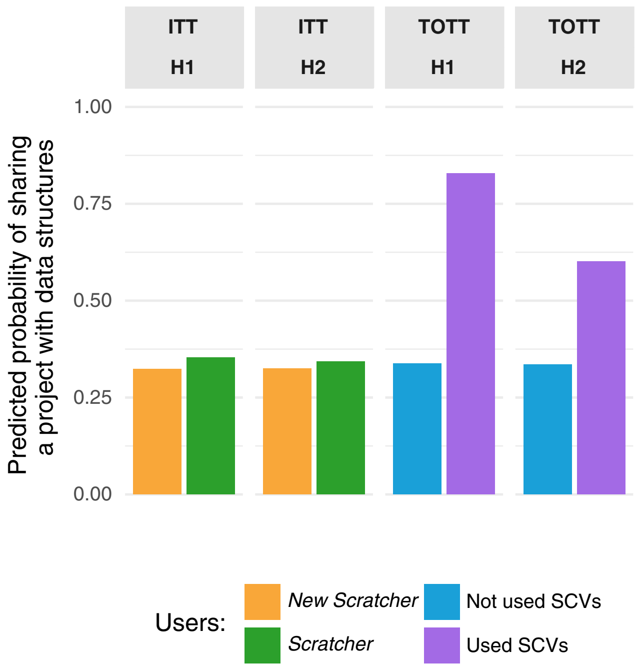


Figure 3. Model-predicted probabilities of sharing a project with data structures. The values of *Share Count* are held at 5 throughout.

if the users knew that the change was coming, because they might then have shifted their activity and used data blocks differently beforehand. Because the change was completely unannounced, we are reasonably confident that this did not occur.

There is a series of specific threats to a researcher's ability to draw causal inference in RDD studies and a set of associated robustness checks [16]. For example, researchers often fit alternative regression models with the cut-off or transition point shifted to another location (a "placebo" cut-off point) [16]. In a placebo test, the fitted regression should have no statistically significant estimate for the alternative treatment variable. For our analysis, we conducted a placebo test by shifting the cut-off point by two projects to the right of the actual cut-off. We could not reject the null hypothesis that there was no effect of the placebo cutoff ($p = 0.75$).

Another form of placebo test involves replacing the dependent variable with an outcome variable that theory or substantive knowledge suggests should not be affected by the change. In our case, this test was more difficult because access to SCVs might have encouraged users to create projects that were different in ways that went beyond simply including data structures. For example, although the SCV system was not designed to promote the exploration of loops or events, the use of data structures might have created the need in some users to explore these concepts as well. Our inability to conduct a placebo test of this kind means that we cannot address this threat as fully as we might like.

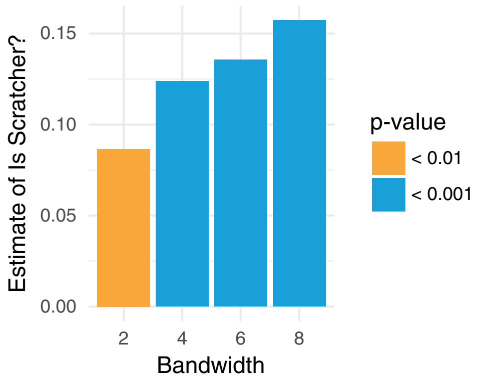


Figure 4. Estimates for *Is Scratcher?* from fitting M1, using the data for H1 across a range of bandwidths. The estimate remains positive and statistically significant over a range of bandwidths.

Another important methodological issue to consider in the context of an RDD study is the choice of bandwidth (i.e., the number of projects around the transition point). The challenge in choosing an appropriate bandwidth is to achieve a balance between having a large enough sample and minimizing the effect of factors far away from and unrelated to the change. To address this threat, we conducted our analyses with smaller and larger bandwidths and found that the positive estimate for *Is Scratcher?* remained. Figure 4 shows the estimates for *Is Scratcher?* from Model M1 across a range of bandwidths. Although estimates were consistent across large bandwidths, our placebo test failed with some larger bandwidths. This is not entirely unexpected as our results with these wider bandwidth models might have been influenced by a variety of factors as we included more projects further from the transition point. Nevertheless, this represents a potential threat to the validity of our results.

There is also a series of more conceptual threats to validity. Though we treat the SCV system as widening walls, it ultimately represents a very specific vision of what wide walls could be. SCVs introduce a wider range of opportunities for those who are already relatively experienced with coding in Scratch. It is possible to imagine other forms of wide walls that speak to very specific interests and passions that lie outside of traditional programming. It is entirely possible that the dynamics around the adoption and use of such hypothetical types of wider walls could be very different from what we found in our study.

Finally, we recognize that all quantitative measures of learning are limited. For one, measures such as ours can only observe outcomes of learning. In our analysis, we only detected the presence of data structures; we did not know the purpose of those data structures or even if they were effectively unused in the projects that we observed. We believe that this represents a significant limitation of our approach. We hope that other researchers will engage with, critique, and build on our methods.

DISCUSSION

The primary contribution of our work is its finding support for the theory that wide walls lead to increased engagement and learning. We found that providing more creative possibilities and affordances to learners caused them to engage more with underlying concepts. We also found that this increased engagement was not restricted to the new possibilities, but also applied to the possibilities that were available earlier. As children took advantage of SCVs' new possibilities, they engaged more with the "vanilla" forms of data structures as well. We believe that these results represent causal evidence for Papert's power principle and Resnick's concept of wide walls.

Our work also makes several methodological contributions. These contributions lie in our use of a natural experiment to establish a causal relationship using observational data. Although new to HCI, we believe that these techniques can find application in many online platforms and learning systems. It is not uncommon for policies governing who gets access to features to change in ways similar to the case we studied. In such cases, the strategy that our analysis followed could be utilized to measure the causal effect of those changes.

Despite our finding evidence in support of the wide walls theory, it is worth keeping in mind that only a small proportion of learners in our sample used SCVs. A possible reason for this—and a challenge for designers attempting to widen walls in an informal environment—is the difficulty that learners face in understanding a large number of functional possibilities arising out of a comparatively simple structure. This may represent an important limitation of the wide walls approach. Although the structural model of the SCV system is simple and consistent (i.e., there are only a limited number of possible operations on SCVs, and these operations are independent of any specific use), the functional model covers a vast number of uses and is dependent on users' intent. For example, SCVs can be used to keep persistent count of votes or users, to keep track of high scores, to store user preferences, to implement a message exchange system, and so on.

DiSessa [10] argued that one needs a "repertoire" of functional uses and that building such a repertoire may require "specific tutoring." As an informal learning environment, Scratch offers very little "specific tutoring." When walls are widened, it may also be necessary for designers to articulate and design specific plans to help learners understand how to make use of the new possibilities. Which strategies will work best for this (e.g., tutorials, sample projects, community events focused on the new possibilities) remains an open question.

Echoes of this conundrum can be heard within existing discourse around wide walls. For example, Resnick and Silverman suggest that there may be a potential tension between designing for wide walls and designing for low floors (i.e., a low barrier to entry for novice learners):

> The design challenge is to develop features that are specific enough so that kids can quickly understand how to use them (low floor), but general enough so that kids can continue to find new ways to use them (wide walls) [38].

Thus, though SCVs widened the walls for users of variables in Scratch, it may have, in effect, raised the floor for learners as well. As Resnick and Silverman point out, balancing out the effects associated with the raised floor remains an open design challenge.

CONCLUSION

We believe that our work represents the first large-scale quantitative test of the "wide walls" design principle. These results are excellent news for designers who apply this popular and influential principle, as well as for proponents of constructionism. Although not without limitations or caveats, we hope that our estimates of a positive causal effect of wider walls will encourage more designers of educational technology and creativity support tools to adopt the principle in their practice.

ACKNOWLEDGEMENTS

This work is partly based on the first author's PhD dissertation work and some of the images and text in the paper appear in the dissertation. We would like to acknowledge Mitchel Resnick, Natalie Rusk, Hal Abelson, Abhishek Nagaraj, members of the Community Data Science Collective, and our anonymous reviewers for their feedback and support. Financial support for this project came from the National Science Foundation (NSF) grants DRL-1417663 and DRL-1417952, as well as from a Moore/Sloan and Washington Research Foundation Innovation in Data Science postdoctoral fellowship. Work on the Scratch Cloud Data system was supported by NSF grants IIS-1002713 and DRL-1027848.

REFERENCES

1. Hal Abelson, Nat Goodman, and Lee Rudolph. 1974. *LOGO Manual*. Memo 313. Massachusetts Institute of Technology, Cambridge, MA. http://dspace.mit.edu/handle/1721.1/6226

2. Harshit Agrawal, Rishika Jain, Prabhat Kumar, and Pradeep Yammiyavar. 2014. FabCode: Visual Programming Environment for Digital Fabrication. In *Proceedings of the 2014 Conference on Interaction Design and Children (IDC '14)*. ACM, New York, NY, 353–356. DOI: http://dx.doi.org/10.1145/2593968.2610490

3. Joshua D. Angrist and Jörn-Steffen Pischke. 2008. *Mostly Harmless Econometrics: An Empiricist's Companion*. Princeton University Press, Princeton, NJ.

4. Karen Brennan and Mitchel Resnick. 2012. New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American Educational Research Association*. AERA, Vancouver, Canada. http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf

5. Siddhartha Chaudhuri and Vladlen Koltun. 2010. Data-driven Suggestions for Creativity Support in 3D Modeling. *ACM Trans. Graph.* 29, 6 (Dec. 2010), 183:1–183:10. DOI: http://dx.doi.org/10.1145/1882261.1866205

6. Sayamindu Dasgupta. 2013a. From Surveys to Collaborative Art: Enabling Children to Program with Online Data. In *Proceedings of the 12th International Conference on Interaction Design and Children (IDC '13)*. ACM, New York, NY, 28–35. DOI: http://dx.doi.org/10.1145/2485760.2485784

7. Sayamindu Dasgupta. 2013b. Surveys, collaborative art and virtual currencies: Children programming with online data. *International Journal of Child-Computer Interaction* 1, 3–4 (Sept. 2013), 88–98. DOI: http://dx.doi.org/10.1016/j.ijcci.2014.02.003

8. Sayamindu Dasgupta, William Hale, Andrés Monroy-Hernández, and Benjamin Mako Hill. 2016. Remixing as a pathway to computational thinking. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing (CSCW '16)*. ACM, New York, NY, 1438–1449. DOI: http://dx.doi.org/10.1145/2818048.2819984

9. Sayamindu Dasgupta and Benjamin Mako Hill. 2017. Learning to code in localized programming languages. In *Proceedings of the Fourth ACM Conference on Learning @ Scale (L@S '17)*. ACM, New York, NY, 33–39. DOI: http://dx.doi.org/10.1145/3051457.3051464

10. Andrea A. diSessa. 1986. Models of computation. In *User-Centered System Design: New Perspectives in Human-Computer Interaction*, Donald A. Norman and Stephen W. Draper (Eds.). Lawrence Erlbaum Associates, Hillsdale, New Jersey, 201–218.

11. Andrew Gelman and Jennifer Hill. 2007. *Data analysis using regression and multilevel / hierarchical models*. Cambridge University Press, New York, NY.

12. Elizabeth Mara Gerber and Caitlin Kennedy Martin. 2012. Supporting Creativity Within Web-based Self-services. *International Journal of Design; Vol 6, No 1 (2012)* (2012). http://www.ijdesign.org/ojs/index.php/IJDesign/article/view/911

13. Michael Gurevich, Paul Stapleton, and Peter Bennett. 2009. Designing for Style in New Musical Interactions. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Pittsburgh, PA, 213–217. http://www.nime.org/proceedings/2009/nime2009_213.pdf

14. Idit Harel and Seymour Papert (Eds.). 1991. *Constructionism: Research reports and essays, 1985-1990*. Ablex Pub. Corp., Norwood, NJ.

15. William G. Howell, Patrick J. Wolf, David E. Campbell, and Paul E. Peterson. 2002. School vouchers and academic performance: Results from three randomized field trials. *Journal of Policy Analysis and Management* 21, 2 (2002), 191–217. DOI: http://dx.doi.org/10.1002/pam.10023

16. Guido W. Imbens and Thomas Lemieux. 2008. Regression discontinuity designs: A guide to practice. *Journal of Econometrics* 142, 2 (Feb. 2008), 615–635. DOI:http://dx.doi.org/10.1016/j.jeconom.2007.05.001

17. Yasmin B. Kafai. 2006. Constructionism. In *The Cambridge Handbook of the Learning Sciences* (1st ed.), Keith R. Sawyer (Ed.). Cambridge University Press, Cambridge, UK, 35–46.

18. Yasmin B. Kafai and Mitchel Resnick (Eds.). 1996. *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*. Lawrence Erlbaum Associates, Mahwah, NJ.

19. Caitlin Kelleher, Randy Pausch, and Sara Kiesler. 2007. Storytelling Alice Motivates Middle School Girls to Learn Computer Programming. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07)*. ACM, New York, NY, 1455–1464. DOI:http://dx.doi.org/10.1145/1240624.1240844

20. A Knörig. 2008. *Design Tools Design: How to design tools for designers, and a proposal of two new tools for the design of physical interactions*. Ph.D. Dissertation. University of Applied Sciences Potsdam. http://fritzing.org/media/uploads/publications/Knoerig08_DesignToolsDesign.pdf

21. Alan B. Krueger and Pei Zhu. 2004. Another look at the New York City school voucher experiment. *American Behavioral Scientist* 47, 5 (2004), 658 –698. DOI: http://dx.doi.org/10.1177/0002764203260152

22. Momin Malik and Jürgen Pfeffer. 2016. Identifying Platform Effects in Social Media Data. In *International AAAI Conference on Web and Social Media*. AAAI, Palo Alto, CA. http://www.aaai.org/ocs/index.php/ICWSM/ICWSM16/paper/view/13163

23. Jean-Bernard Martens. 2014. Interactive Statistics with Illmo. *ACM Trans. Interact. Intell. Syst.* 4, 1 (April 2014), 4:1–4:28. DOI:http://dx.doi.org/10.1145/2509108

24. J. Nathan Matias, Sayamindu Dasgupta, and Benjamin Mako Hill. 2016. Skill progression in Scratch revisited. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM Press, New York, NY, 1486–1490. DOI: http://dx.doi.org/10.1145/2858036.2858349

25. Andrés Monroy-Hernández and Mitchel Resnick. 2008. FEATURE: Empowering Kids to Create and Share Programmable Media. *Interactions* 15, 2 (March 2008), 50–53. DOI:http://dx.doi.org/10.1145/1340961.1340974

26. Jesús Moreno-León and Gregorio Robles. 2015. Dr. Scratch: A web tool to automatically evaluate Scratch projects. In *Proceedings of the Workshop in Primary and Secondary Computing Education (WiPSCE '15)*. ACM, New York, NY, 132–133. DOI: http://dx.doi.org/10.1145/2818314.2818338

27. Jesús Moreno-León, Marcos Román-González, Casper Harteveld, and Gregorio Robles. 2017. On the automatic assessment of computational thinking skills: A comparison with human experts. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human*

Factors in Computing Systems (CHI EA '17). ACM, New York, NY, 2788–2795. DOI: http://dx.doi.org/10.1145/3027063.3053216

28. Richard J. Murnane and John B. Willett. 2011. *Methods matter: Improving causal inference in educational and social science research*. Oxford University Press, New York, NY.

29. Sneha Narayan, Jake Orlowitz, Jonathan Morgan, Benjamin Mako Hill, and Aaron Shaw. 2017. The Wikipedia Adventure: Field Evaluation of an Interactive Tutorial for New Users. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW '17)*. ACM, New York, NY, 1785–1799. DOI: http://dx.doi.org/10.1145/2998181.2998307

30. Seymour Papert. 1980. *Mindstorms: Children, computers, and powerful ideas*. Basic Books, New York, New York.

31. Seymour Papert. 1996. An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning* 1, 1 (1996), 95–123. DOI:http://dx.doi.org/10.1007/BF00191473

32. Seymour Papert and Idit Harel. 1991. Situating constructionism. In *Constructionism*. Vol. 36. Ablex Publishing, New York, NY, 1–11.

33. Jean Piaget. 1970. *Genetic epistemology*. Columbia University Press, New York, NY.

34. Mitchel Resnick. 1996. Beyond the Centralized Mindset. *Journal of the Learning Sciences* 5, 1 (Jan. 1996), 1–22. DOI:http://dx.doi.org/10.1207/s15327809jls0501_1

35. Mitchel Resnick. 2016. Designing for Wide Walls. (Aug. 2016). https://design.blog/2016/08/25/mitchel-resnick-designing-for-wide-walls/ Archived at https://perma.cc/9N3P-48E3.

36. Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, and Yasmin Kafai. 2009. Scratch: Programming for all. *Commun. ACM* 52, 11 (Nov. 2009), 60–67. DOI: http://dx.doi.org/10.1145/1592761.1592779

37. Mitchel Resnick, Brad Myers, Kumiyo Nakakoji, Ben Shneiderman, Randy Pausch, Ted Selker, and Mike Eisenberg. 2005. *Design Principles for Tools to Support Creative Thinking*. Technical Report. Institute for Software Research, Carnegie Mellon University, Pittsburgh, PA. http://repository.cmu.edu/isr/816 (Working paper).

38. Mitchel Resnick and Brian Silverman. 2005. Some reflections on designing construction kits for kids. In *Proceedings of the 2005 conference on Interaction design and children (IDC '05)*. ACM, New York, NY, 117–122. DOI: http://dx.doi.org/10.1145/1109540.1109556

39. James Randal Sargent. 1995. *The programmable LEGO brick : ubiquitous computing for kids*. Thesis. Massachusetts Institute of Technology. http://dspace.mit.edu/handle/1721.1/34694

40. Christopher Scaffidi and Christopher Chambers. 2011. Skill progression demonstrated by users in the Scratch animation environment. *International Journal of Human-Computer Interaction* 28, 6 (June 2011), 383–398. DOI: http://dx.doi.org/10.1080/10447318.2011.595621

41. Ben Shneiderman. 2007. Creativity Support Tools: Accelerating Discovery and Innovation. *Commun. ACM* 50, 12 (Dec. 2007), 20–32. DOI: http://dx.doi.org/10.1145/1323688.1323689

42. Judith D. Singer and John B. Willett. 2003. *Applied Longitudinal Data Analysis: Modeling Change and Event Occurrence*. Oxford University Press, New York, NY.

43. Uri Wilensky. 1999. Netlogo. (1999). http://ccl.northwestern.edu/netlogo/

44. David Wolber, Hal Abelson, Ellen Spertus, and Liz Looney. 2011. *App Inventor*. O'Reilly, Sebastopol, CA.

45. Seungwon Yang, Carlotta Domeniconi, Matt Revelle, Mack Sweeney, Ben U. Gelman, Chris Beckley, and Aditya Johri. 2015. Uncovering trajectories of informal learning in large online communities of creators. In *Proceedings of the Second (2015) ACM Conference on Learning @ Scale (L@S '15)*. ACM, New York, NY, 131–140. DOI: http://dx.doi.org/10.1145/2724660.2724674