Structured Input Improves Usability and Precision for Solving Geometry-based Algebraic Problems

Bo Kang
University Of Central Florida
Orlando, FL
bkang@cs.ucf.edu

Joseph J. LaViola Jr.
University Of Central Florida
Orlando, FL
jjl@eecs.ucf.edu

Pamela Wisniewski
University Of Central Florida
Orlando, FL
pamwis@ucf.edu

ABSTRACT

Previous research has shown that sketch-based input is efficient and preferable in the context of algebraic equation solving. However, research has not been conducted to evaluate whether this holds true when involving geometry input to facilitate quantitative problem-solving. We developed a bimodal (graphing geometric shapes and writing algebraic expressions) user interface, in order to conduct a within-subject, controlled experiment with 24 college students and varied two types of geometry input: 1) sketch-based input and 2) structured input. The sketch-based input was significantly faster than the structured input, but there were no significant differences based on perception and cognition. However, after a post-hoc analysis, we found a significant interaction effect on perception between prior knowledge and geometry input. Novice students preferred the sketch-based input, but advanced students preferred the structured input. Our study implies that natural sketch-based input may be less preferable than structured input for geometry-based interfaces toward math problem-solving.

ACM Classification Keywords

H.5.2. Information Interfaces and Presentation (e.g. HCI): User Interfaces - User-centered design; K.3.1. Computers and Education (e.g. HCI): Computer Uses in Education - Computer-assisted instruction

Author Keywords

Sketch Input; Gesture Input; Geometry Editing; Math Learning Environment

INTRODUCTION

Sketch input has been advocated in the HCI community due to its perceived "naturalness" and application of the principle of direct manipulation [19]. This input method has demonstrated its effectiveness and usefulness in multiple applications, such as note taking [38], user interface prototyping [22, 32], 3D drawing [5, 26] and intelligent systems [16, 39, 49]. At the same time, due to the prevalence of ubiquitous computing across desktop, mobile, tablet, and other devices, gesture input has attracted HCI researchers to explore such novel interaction techniques across multiple domains [4, 6, 29, 31]. Applying sketch input in mathematics education, Oviatt et al. compared different user interfaces (Anoto-based digital stylus and paper interface, pen tablet interface, graphical tablet interface) for algebraic math problem-solving. They found that the naturalness property of various pen-based input reduced students' cognitive load and facilitated math problem-solving compared to using a mouse and keyboard [41, 40]. Anthony et al. built upon this work by implementing a sketch-based interface within the context of an algebra tutoring system. They found that the sketch-based approach was effective in helping students efficiently input algebraic expressions and guided their problem-solving [3].

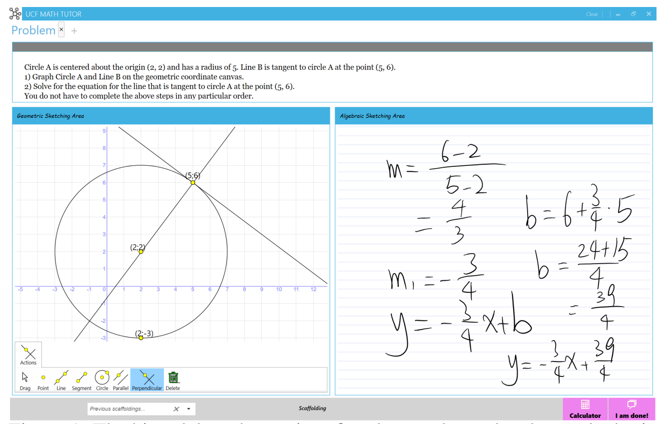


Figure 1: The bimodal math user interface lets students sketch out algebraic expressions on the right algebraic canvas and enter geometric shapes on the left canvas. The above image shows the structured visual widget geometry input method on the geometric canvas.

In addition to writing algebraic expressions, drawing geometric shapes and plotting them on a Cartesian plane are essential skills required to facilitate quantitative reasoning during math problem-solving [30]. Currently, a number of dynamic geometry environments let users enter and manipulate geometric shapes. Many have implemented Window, Icon, Menus, Point and Click (WIMP) interfaces, such as Geogebra[1], while others have developed gesture-based approaches for graphing shapes, such as Sketchometry[2]. Even though gesture-based and structured WIMP-based geometry interfaces already exist, research lacks regarding the efficacy of these interfaces in the context

[1]http://www.geogebra.org
[2]http://sketchometry.org

CHI 2017, May 6-11, 2017, Denver, CO, USA.

http://dx.doi.org/10.1145/3025453.3025468

of math problem-solving. Further, since students might need to enter both geometric shapes and algebraic expressions to advance quantitative reasoning, previous systems have not explored the potential of bimodal algebraic and geometry inputs for math problem-solving. This gap in the research motivated us to conduct a formal study to evaluate different geometry input interaction techniques within analytical geometry math problem-solving.

This paper presents an in-depth evaluation of a bimodal (algebraic and geometric) math problem-solving user interface. Since prior research has already established that pen-based approaches are preferable for algebraic equation solving, we place our emphasis instead on the less explored interaction of geometric graphing. We conducted a controlled experiment to evaluate two different geometry input methods: sketch-based vs. visual widget. We initially hypothesized that a sketch-based interface would out-perform the visual widget-based interface for geometric graphing. However, our initial hypotheses were not supported. We then conducted a post-hoc analysis by dichotomizing our sample based on participants' prior knowledge of math concepts, and found an interesting interaction effect: novice students preferred to use the sketch-based input to explore geometric concepts, whereas advanced students (who also outperformed the novices) significantly preferred the structured visual-widget input to enter geometric shapes more precisely. This finding informs us when designing math-learning environments that incorporate graphing tasks, providing geometry input methods that adapt to the student may help to support the users' goals of concept exploration versus successful problem-solving.

We evaluate our sketch-based geometry input method in comparison to a more traditional WIMP-based interface. More specifically, we implemented a structured, visual widget for graphing geometry input. As such, our work contributes to the existing literature in the following ways:

- We develop a bimodal user interface that extends sketch capabilities from algebraic expression input to additionally graphing geometric shapes (Figure 1).
- We conduct an empirical study embedded in the context of college-level math problem-solving.
- We evaluate a sketch-based input geometry interface in comparison to a structured visual widget interface.
- We discuss the design implications of our findings and the potential for building adaptive math interfaces to meet the goals of exploration and problem-solving for novice and advanced students.

RELATED WORK

Gesture-based interactions include recognizing various human motions, i.e., 2D trajectories drawn by users with their finger on a touchscreen or with a pen, so that a computer system can act based on user input [33]. Original gesture recognizers utilized domain specific features to train classifiers to differentiate drawn shapes [47, 44, 53]. Recent dollar gesture recognizers showed a lightweight template matching method to help developers quickly prototype gesture-based user interfaces [33, 51, 2]. Wobbrock et al. presented a high level taxonomy to guide gesture design for surface computing [50]. Applying existing gesture recognition technology and design taxonomy into math learning environments, Sketchometry included several symbolic gestures and used template-matching recognizers to analyze touch input to determine the geometric shapes drawn by users [17].

Sketch recognition also focuses on analyzing large number of strokes to support semantic interpretation in different scenarios, such as written text [7], geometric diagrams [1] and mathematical expressions [52]. For math problem-solving, Anthony et al. presented a sketch-based tutoring environment, which supported math expression recognition for algebraic problem-solving [3]. Jiang et al. presented a sketch-based geometry theorem proving system that recognized handwritten shapes and textual scripts in different structured regions [28]. While researchers continue to attempt to use sketch recognition techniques to differentiate between users' sketches, such as trying to detect text versus shapes drawn in a single writing space [9], the application of recognition-based solutions for distinguishing between sketched text, shapes, and math expressions is limited due to the accuracy of these recognition algorithms.

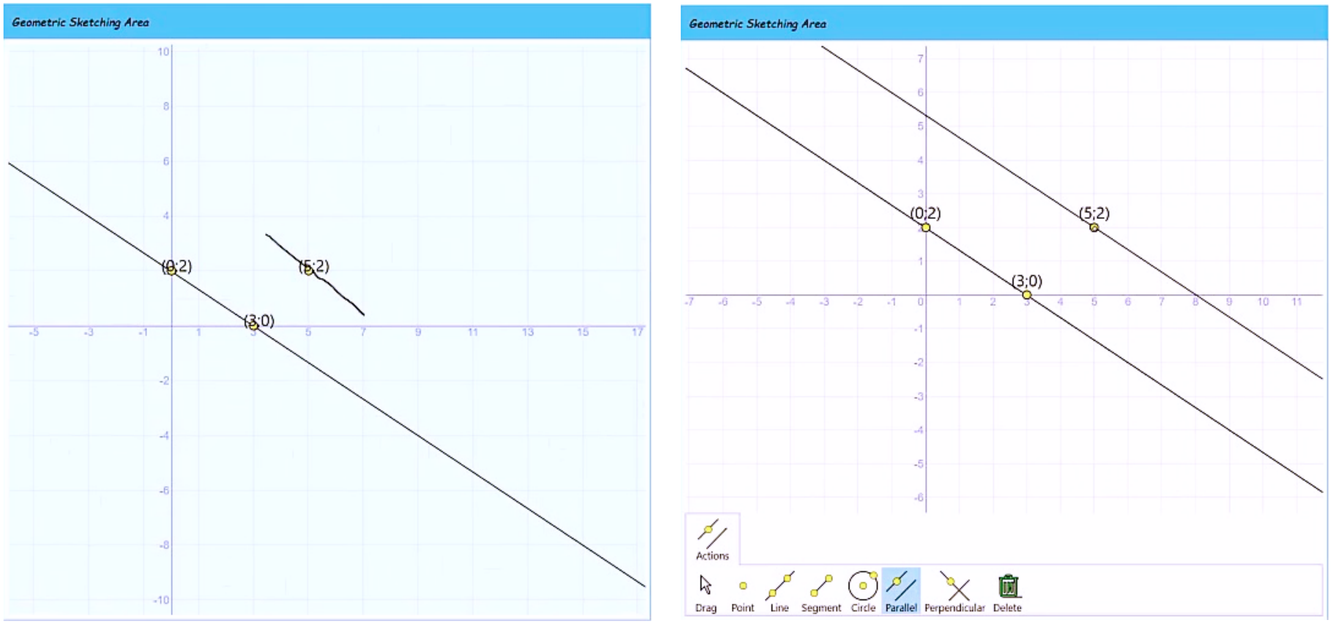
Building upon this previous work, our bimodal user interface has two distinct canvases to distinguish geometry shape input (geometric canvas) from other input, such as algebraic expressions (algebraic canvas). The sketch-based geometry input method specifically refers to the gesture input that uses a template matching approach to recognize symbolic gestures [51]. Since relations between geometric shapes exist, such as two parallel lines, after recognizing certain geometric shapes, inferring geometric relations may be essential when users do not provide sufficient input information. Igarashi et al. developed geometric constraints to let systems infer geometric relations [25]. Cheema et al. further expanded geometric constraints and revised the constraint solving algorithm to support the relation checking procedures [15]. Since our work only modeled a specific set of geometric relations, such as connection to a point, connection to a line segment, two parallel lines, two perpendicular lines and line-and-circle tangent, instead of using a constraint solver, we developed and pruned geometric relations heuristically using a case by case approach.

METHODS

Experimental Design

Our study utilized a within-subjects design that varied geometry input type: 1) sketch-based gestural input and 2) structured visual widget input. The sketch-based gestural input lets students enter geometric shapes using direct manipulation (Figure 2a). In contrast, the structured visual widget input lets students enter geometric shapes by selecting visual widgets and further pointing onto the geometric canvas (Figure 2b).

This study specifically examined problem-solving in the analytical geometry math domain that requires students to perform quantitative reasoning by graphing geometric shapes and writing algebraic expressions. For the stimuli design, we selected math problems that covered four analytical geometry math concepts: perpendicular lines, parallel lines, distance



(a) After sketching out out a line, an approximate line is rendered. The user further draws another line that is approximately parallel to the first line. The system infers geometric relations between the current drawing and existing geometric shapes, renders the parallel line.

(b) The user first clicks the line widget, and points twice as two visual point coordinates, a line is rendered. The user further clicks the parallel line widget, points once onto the existing line, points to another point coordinate off the existing line. The parallel line is rendered.

Figure 2: Both images show the same scenario on how to use two geometry input methods to input two parallel lines.

between a point and a line, and a tangent between a line and a circle. By solving problems within these four concepts, we were able to maximally cover the different geometry interactions that students perform during problem-solving. Such problems required students to do quantitative reasoning using both geometric and algebraic knowledge representations. Four problems were selected and pruned from a high school geometry textbook [46]. The parallel lines math problem was *"Line C passes through the point* $(5,2)$*. Line C is parallel to the line* $4x+6y-12=0$*. Graph Line C on the geometric coordinate canvas. Solve for the equation of line C on the algebraic canvas. You do not have to complete the above steps in any particular order."*. The other three problems are listed in the Appendix. We cloned the four problems within each experimental condition by retaining the wording but replacing the numerical values to provide enough problems for the experiment.

Research Hypotheses

Previous research in the algebraic equation solving context has shown that sketch-based pen input is preferable and more efficient than mouse-and-keyboard input [3]. Based on these findings, we derived four initial hypotheses that posit the sketch-based geometry interface will also out-perform the more traditional visual widget interface:

- **Hypothesis 1, Efficiency:** Sketch-based input will be significantly more efficient (i.e. faster) than visual widget input when executing the same geometric plotting tasks.
- **Hypothesis 2, Usability:** Participants will significantly prefer sketch-based input over the visual widget input.
- **Hypothesis 3, Cognitive Load:** The sketch-based input will significantly reduce the amount of cognitive load on participants compared to the visual widget input.
- **Hypothesis 4, Performance:** The sketch-based input will significantly increase problem-solving performance of participants compared to the visual widget input.

Dependent Measures

To test our hypotheses and evaluate the two geometry input methods, we measured four dependent variables, which are described below. All perceived measures were operationalized using pre-validated constructs from previous literature and measured on 7-point Likert scales.

- **Solving Time:** The time in which users first interacted with the interface to when they notified the system of their completion of a math problem.
- **Perceived Usability:** Operationalized using Lund's Usefulness, Satisfaction, and Ease of Use (USE) questionnaire [35], which includes the following dimensions: 1) usefulness, 2) ease of use, 3) ease of learning, and 4) satisfaction. Each dimension contained four items or questions.
- **Cognitive Load:** Measured the mental effort toward the extrinsic environment [43] and was operationalized using NASA-TLX [23]. A high-ranking on this scale implied low mental effort; thus, was negatively correlated with cognitive load.
- **Solving Performance:** Based on an existing grading rubric [24], we customized our solution grading on a 100% scale that assessed the various components of each problem given the following equal weights: algebraic problem

translation, geometry problem translation, graphing accuracy, algebraic problem-solving, and arriving at the correct answer.

To calculate solving performance, we recruited two math tutors to grade participants' solutions through recorded videos and averaged the scores between the two raters. To assess inter-rater reliability, we asked them to cluster participants into high or low performing groups by comparing their mean grade score to the sample mean. Two graders clustered all participants in a similar manner (Cohen's kappa = 1.0). For all other subjective measures (e.g., perceived usability, cognitive load), we calculated Cronbach's alpha to assess construct reliability.

System Design

We arranged our user interface with two equally sized but separate input canvases, which appear below a problem description area. The right side of the interface (Figure 1) is a sketch-based canvas, that lets users write math expressions or other texts freely. On this canvas, users can perform a scribble erase gesture to erase their drawings. Though previous work included the sketch-based math expression recognition [3], sketch recognition is not incorporated in our work for interpreting any writings on the right side algebraic canvas. The reason is that our research focuses on the geometry input methods with minimal interference on other factors, such as sketched math expression recognition accuracy. Touch input also allows users to manipulate the canvas. The algebraic canvas remained constant across both conditions as only the geometric canvas was varied for the experiment.

The left canvas of the interface provides a geometry canvas, which allows users to plot and manipulate geometric shapes on a visualized Cartesian coordinate system. This user interface was built upon a dynamic geometry engine where users are allowed to create geometric shapes [3]. When manipulating a geometric shape, this engine can use the shape's relation dependencies to re-compute the other dependent shapes' information. On the bottom of the screen, there is a Done button that students can click after they finish a problem. The system then directs the user to the next problem.

Visual Widget Input Condition

The structured, visual widget input method (Figure 2b) lets users select buttons on a visual widget bar that was located beneath the graphing interface. Aligned with the later experimental stimuli, the visual widget bar contains graphical icons that represent as point, line, line segment, circle, parallel lines, and perpendicular lines, respectively in sequence. In addition, drag and delete visual widgets are also present. Users could create geometric shapes or relations by selecting the corresponding visual widgets and performing one or more continuous pointing tasks. Users could manipulate existing geometric objects by clicking the drag widget. In addition, in drag mode, single contact touch manipulation gestures allow users to move the coordinate system, whereas the two contact touch manipulation gesture lets them zoom in-and-out the coordinate. Last, in order to delete a geometric object, users can first choose the delete widget, and then click that geometric object.

Sketch-based Input Condition

The sketch gestural input method (Figure 2a) let users directly sketch out symbolic shapes (points, lines and circles), which could be further manipulated by pointing and dragging. Single and double contact touch manipulation input was provided to translate and scale the visible area. Users could delete any shape by drawing a scribble gesture on top of the object. When drawing relational geometric shapes, such as parallel lines, users could deterministically plot two points and draw a line to connect two points so as to draw a parallel line. Otherwise, they could draw the second line to approximately parallel to the first line. The system will perform the constraint solving procedure that checks the new drawings with existing shapes to detect certain geometric relations between shapes [25]. Using relation detection makes the system able to generate relation-based geometric shapes, such as line segments, and parallel and perpendicular lines. In addition to the above heuristics, there are other relation checking cases, such as plotting a line or a line segment upon two existing points, dragging an existing point or drawing a new point. We carefully model such constraint heuristics case by case. All constraint heuristics were adjusted based on feedbacks collected during a pilot study. All gestures were recognized using the $1 template based gesture recognizer [51].

Apparatus

We developed the interactive learning environment using the C# and .NET framework, integrated the existing dynamic geometry environment live-geometry and $1 template based gesture recognizer. The developed system was deployed onto a Microsoft Surface Pro 3 with a digitizer. The experiment window was set in full-screen mode. Participants used the stylus and touch to interact with this bimodal interactive math environment. During problem solving, they were asked to use both algebra and geometry canvases without any input sequence constraints.

Procedure

The study took place in the user experience research lab on our university's campus. Participants were first asked to give their informed consent to participant in the study and took a brief web-based pre-survey. We assessed students' prior knowledge of the math concepts being tested. Participants were asked to rate their knowledge of the four math concepts on a 4-point Likert scale (1-Never learned, 2-Learned in the past but do not remember now, 3-Vaguely remember, and 4-Definitely Remember).

Next, participants were randomly assigned to one of the two conditions (i.e., sketch-based or visual widget input). However, the study design was counter-balanced to avoid ordering effects. In either condition, participants first followed interaction instructions to complete a step-by-step practice session on how to use the current settings to input geometric shapes and algebraic expressions. Participants were encouraged to practice multiple times to familiarize themselves with the system prior to officially starting the experiment. For the experimental study, participants were required to solve four problems in

[3]https://livegeometry.codeplex.com/

Table 1: Scale Reliabilities, Descriptive Statistics and Paired t-test Results

Measure	Cronbach's α	sketch-based input		structure-based input		Result
		M	SD	M	SD	
Solving Time	N/A	**178.30s**	**76.74s**	**221.68s**	**89.36s**	$t(23) = 2.25, p < 0.05$
Usefulness	0.80	5.96	0.87	5.96	0.72	$t(23) = 0.00, p = 1.0 > 0.05$
Ease of Use	0.72	5.74	0.84	6.06	0.74	$t(23) = 1.67, p = 0.11 > 0.05$
Ease of Learning	0.83	6.28	0.67	6.39	0.56	$t(23) = 0.87, p = 0.39 > 0.05$
Satisfaction	0.84	5.75	1.04	5.88	0.80	$t(23) = 0.62, p = 0.54 > 0.05$
Cognitive Load	0.79	6.06	0.93	5.77	1.19	$t(23) = 1.22, p = 0.23 > 0.05$
Solving Performance	N/A	66.88	18.85	66.12	20.45	$t(23) = 0.35, p = 0.73 > 0.05$

Note: The higher score of cognitive load indicates lower mental effort that users need to perform the tasks.

one condition (cover each math concept per problem), and the other four math concept problems in the other condition. Two problems per math concept were randomly assigned to two conditions. For each condition, four assigned math problems was also presented in a random order. Participants were asked to talk aloud during problem solving, and they could ask for minimal guidance from the experimenter. We urged them to "try their best" to solve each problem and show their work. Once a problem was completed, participants could click the "Done" button, which will direct to the next problem. After problem-solving was complete, participants asked to assess their experience with the given interface. Then, they were assigned the other condition, which was completed using the same procedure as described above.

In the post survey, a questionnaire was administered to gather participants' explicit preference between the two geometry input methods and their rationale for such preference. The questionnaire also asked for participants' demographic information (age and gender). At the conclusion of the study, participants were given a $10 gift card to compensate them for their time. The entire session was video/audio recorded. The whole study took around 60 minutes.

Participants
Prior to recruiting participants, we conducted a priori power analysis to determine our target sample size. Using G*Power, to detect a medium effect size with a power of 0.80, we needed a total of 24 participants [18]. We recruited participants by contacting the math department within our university for participating in the study. Participants had to have taken Algebra 1 and Geometry 1 in high school. Prior to participating in the study, participants had to agree to the terms of our IRB approved informed consent document. Twenty-four adults, 12 females and 12 males, aged 19 to 21-years-old, participated in our experiment. All participants were college freshmen or sophomores from our university.

RESULTS
Table 1 summarizes descriptive statistics for timing data, all perceived constructs, and performance data. Prior to conducting statistical analyses for our hypotheses tests, we validated the internal consistency (calculated as Cronbach's alpha [48]) of our constructs, which were computed as averages across the Likert scale-items that comprised each measure. Multi-item constructs measured using Likert scales are designed to create indices that represent an underlying continuous variable [10, 12, 34]. We also confirmed that all dependent measures were normally distributed [27]. Given that our data met the prerequisite assumptions, we used standard parametric tests to validate our hypotheses [14, 21].

Hypothesis Testing
We performed paired t-tests based on our within-subjects design to determine significant effects of our treatment on the dependent variables. As shown in Table 1, we only detected significant differences for solving time (Cohen's d= 0.52), such that sketched-based input was significantly faster than visual widget-based input. Therefore, only hypotheses one for efficiency of our initial hypotheses was supported. Based on the post-survey results, participants' explicit preference between the two interfaces were also evenly split (12 for sketch-based and 12 for visual widget).

Post-Hoc Analysis
Surprised by our initial results, we went back to the literature to better understand why we did not find a significant difference between the sketch-based and visual widget interfaces. Motivated by Oviatt et al.'s work [41, 40], we further examined how both participants' prior knowledge and the input method affected their perceptions and outcomes. For the prior knowledge independent variable, we averaged these items and performed a mean split to create two distinct groups: 1) novice students and 2) advanced students. Coincidentally, there were exactly half novice and half advanced students in our sample. We checked the reliability of using the prior knowledge factor as an independent variable by evaluating if two groups' solving performance significantly different between each other. We employed paired/dependent tests to account for the individual differences in our within-subjects design. The result showed that advanced students had significantly higher solving performance score ($M = 81.75, SD = 10.37$) than novice students ($M = 53.27, SD = 13.64$), $t(22) = 5.76, p < 0.001$.

Next, we conducted a mixed factorial analysis between the two levels of prior knowledge (novice vs. advanced) and two-levels of geometry input methods (sketched-based vs. visual widget). For solving time, there was a significant main effect of the between-subject factor based on students' prior knowledge ($F(1,22) = 7.67$, $p < 0.05$, $\eta_p^2 = 0.26$). A post-hoc Tukey's pairwise comparison revealed that novice students took less time ($M = 164.79s$, $SD = 40.66s$) compared to advanced students ($M = 227.76s$, $SD = 71.46s$), $t(22) = 2.65, p < 0.05$. By closely examining recorded videos of each participant, we

Construct	ANOVA
Interaction Effects of Input Style *X* Prior Knowledge	
Usefulness	$F(1,22) = 5.26,\ p < 0.05,\ \eta_p^2 = 0.19$
Ease of use	$F(1,22) = 5.25,\ p < 0.05,\ \eta_p^2 = 0.19$
Satisfaction	$F(1,22) = 9.94,\ p < 0.01,\ \eta_p^2 = 0.31$
Cognitive load	$F(1,22) = 4.20,\ p < 0.05,\ \eta_p^2 = 0.16$
Simple Effects of Visual Widget Input	
Usefulness	$F(1,22) = 6.31,\ p < 0.05,\ \eta_p^2 = 0.22$
Ease of use	$F(1,22) = 4.85,\ p < 0.05,\ \eta_p^2 = 0.20$
Satisfaction	$F(1,22) = 5.58,\ p < 0.05,\ \eta_p^2 = 0.20$
Cognitive load	$F(1,22) = 5.51,\ p < 0.05,\ \eta_p^2 = 0.20$
Simple Effects of Advanced Students	
Usefulness	$F(1,22) = 4.89,\ p < 0.05,\ \eta_p^2 = 0.18$
Ease of use	$F(1,22) = 8.46,\ p < 0.01,\ \eta_p^2 = 0.28$
Satisfaction	$F(1,22) = 7.53,\ p < 0.05,\ \eta_p^2 = 0.26$

Table 2: Interaction effects, significant simple effects.

realized this was because novice students gave up and continued to the next problem before they successfully solved it. In contrast, advanced students took more time because they worked toward achieving the solution before they moved on to the next problem. Therefore, the original support for our hypothesis one was equivocated; the sketch-based interface did not necessarily make user more efficient in problem-solving. It may have been that participants were more likely to move on without solving the problem.

For the four dimensions of perceived usability and cognitive load, we did not detect any significant main effects of prior knowledge or input method. However, for three of the four perceived usability dimensions (all except ease of learning), we found a significant interaction effect between the two factors (in Table 2). Figure 3 illustrates the interaction effect for the perceived usefulness, which was similar to the pattern for the other dependent variables. For the cognitive load, we did found the same significant interaction effect between two factors. Novices students significantly preferred the sketch-based input method over the visual widget interface. Meanwhile, advanced students significantly preferred the visual widget input over the sketch-based interface.

We performed further analysis to calculate the simple effects across the two levels of the two variables (four cases) in order to explore the differences in the dependent variables in more depth. For visual widget input, a post-hoc Tukey's pairwise comparison revealed significant differences between novice and advanced students all four constructs (e.g., usefulness, ease of use, satisfaction, and cognitive load) (in Table 2). Figure 4 displays the mean rating comparisons. Overall, advanced students perceived the visual widget interface as significantly more useful, easy to use, and satisfying. Advanced students took significantly less mental efforts to use the visual widget input other than novice students. For sketch-based input, no significant differences were found between the prior knowledge groups. Similarly, advanced students provided significantly different ratings on three of the usability constructs (in Table 2). A post-hoc Bonferroni's pairwise comparison revealed the significant differences between visual widget input

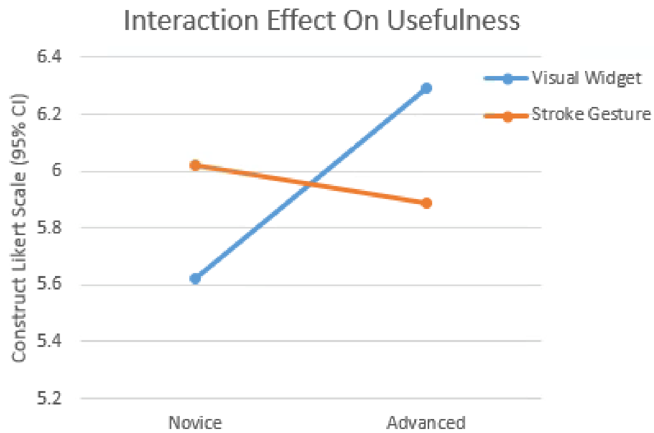


Figure 3: Interaction Effect of Usefulness mean ratings across prior knowledge (two levels) and input method (two levels).

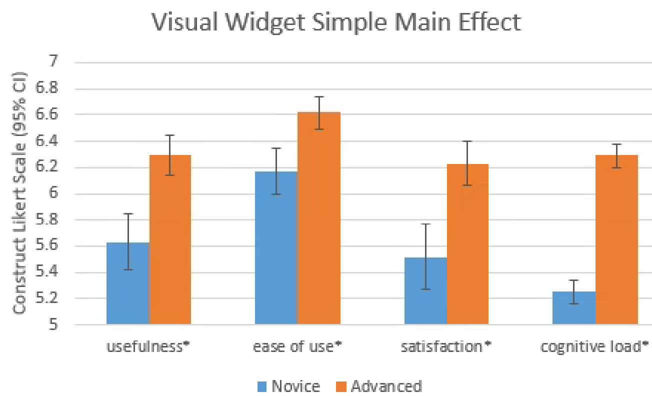


Figure 4: Simple Main Effect Post-Hoc Mean Ratings: Construct mean ratings across two prior knowledge levels on the visual widget input level. (Construct with * means statistical significance was found.)

and sketched-based input on two dimensions among the three. They felt that the visual widget input method was more ease of use ($M = 6.62, SD = 0.43$) than the sketch gesture input ($M = 6.35, SD = 0.54$), $t(11) = 3.48, p < 0.01$. They satisfied the visual widget input method ($M = 6.23, SD = 0.58$) more than the sketch gesture input method ($M = 5.56, SD = 0.95$), $t(11) = 2.79, p < 0.05$. Novice students did not exhibit a significantly different rating on two input methods on perceived constructs.

Based on the $2x2$ contingency table (prior knowledge *x* input method) of participants' explicit preference between the interfaces, we conducted a chi-squared test to see if there were statistical differences. Since some of the counts in the table were small, we used the Fisher's exact test. We found that the input method preference significantly ($p < 0.05$) differed based on prior knowledge level. Nine out of twelve novice participants preferred sketched-based gestural input. Nine out of twelve of the advanced students preferred visual widget input.

Why Prior Knowledge Influenced Preference

To further interpret the unintended results of our study and to understand why participant outcomes varied based on their prior knowledge of math concepts, we conducted a thematic content analysis of the qualitative comments from the post-survey that explained users' preferences [11]. The first author coded the qualitative data. The last author assisted in interpretation and checked the coding for face validity. We first identified themes within the comments that aligned with our dependent variables (efficiency, ease of use, usefulness, cognitive load, etc.). Then, we dichotomized the themes based on input method preference and noted if the participant was a novice or advanced student. As we originally hypothesized, participants who preferred the sketch-based interface often commented on aspects aligned with our dependent variables, as illustrated by the responses below:

Ease of Use: *"The free-form based Geometry User Interface was easier to use. All of the commands were made through simply drawing what I wanted."* - Novice Student

Ease of Learning: *"Quicker and simpler to use than the widget. The widget took some more time to use because you had to switch between modes."* - Novice Student

Efficiency and Cognitive Load: *"The free form based interface lets me work much faster because I don't have to think about pressing the right button before I draw, I can move as fast as I'm thinking."* - Novice Student

However, we also realized that these responses were primarily from novice students who did not successfully solve the problems before they gave up and moved on. When analyzing the comments from participants who preferred the visual widget interface, the pre-defined themes related to our dependent variables were also present in the data. For instance, some users thought the interface was easier to use and more useful:

Usefulness: *"The given tools were much more helpful in the wedge as compared to the free form which forced you to be on your own when graphing and gave less assistance."* - Advanced Student

However, new themes emerged and dominated the sentiments expressed in these comments, which were primarily made by advanced students who were able to solve the math problems. These emergent themes are illustrated in the comments below:

Familiarity: *"I preferred the widget based interface because I already know these shapes well."* - Advanced Student

Precision: *"The visual widget interface allowed me to get perfect geometric lines and circles without focus on making the best guess to how perpendicular a line is or how a circle should be formed."* - Advanced Student

Task Capability: *"While the Free-form interface is more user-friendly and takes full advantage of the pen/tablet interface, I see the Visual Widget being much more capable."* - Advanced Student

Error Control: *"The visual widget input didn't leave room for mistakes (such as not drawing the free form line as parallel when you wanted to or deleting)."* - Advanced Student

Based on these comments, we realized that novice students preferred the sketch-based interface because it was easier to use. In contrast, advanced students preferred the visual widget interface because it enabled them to graph more precisely, increasing their capability to perform the task without errors and, thereby facilitated actual problem-solving. The ability to achieve their goal of solving the problem seemed to outweigh the benefits of natural input for advanced students.

DISCUSSION

In our initial analysis, we found that the sketch-based gestural input was faster than the visual widget input. Overall, this finding is unsurprising and consistent with Fitt's Law, which states that the time to click on an object is a function of the size of the object and the distance away the cursor is from that object [36]. The visual widget input required users to click on icons located on the toolbar, which were distance away from the canvas to switch between different commands. In contrast, the sketch-based gestural input allowed users to directly manipulate the geometry canvas. While our first hypothesis regarding the efficiency of sketch-based input was supported, we later found out that it was with a caveat because our post-hoc analysis discovered a main effect where novice students took less time to complete the task simply because they gave up more quickly.

We were initially surprised when our three additional hypotheses were not supported. However, our post-hoc analysis helped to explain some nuance in the data that was not apparent based on the original results. We were able to gain valuable insights into the relationships between the two geometry input methods (sketch-based input vs. visual widget input), student types (novice vs. advanced), and the participants' primary goals (exploratory vs. problem-solving). We found that novices generally preferred the sketch-based input. Though novices were able to use the sketch-based input to finish problem solving quickly, these novice students were also not successful in mastery of the math problems. Because they weren't actually solving the problem, they liked the sketch-based gestural input because it was more natural for their exploratory interaction with the interface. A key implication from this finding is that sketch gestural input may be preferable as an early learning tool so that novice students can easily interact with the geometry canvas for learning basic, more foundational concepts, such as how to graph a line.

More importantly, advanced students preferred the structured visual widget and rated it significantly better in terms of usefulness, ease of use, and satisfaction. This was true even though the visual widget input took significantly longer for solving similar problems. It was initially difficult for us to understand why this was the case as we did not include a formal measure for perceived accuracy or precision in our dependent variables. Luckily, this theme clearly emerged from our qualitative data helping us understand why advance students preferred the structured interface. It was because precision and accuracy became more important to the task at hand - actually solving the math problems. The advanced students had the knowledge to solve the problems and preferred the tool that helped them accomplish this most effectively. In this way, the utility of being able to draw precisely using more structured input was more valuable than the ease of using the more naturalist sketch gestural input.

Implications for Design

Based on the above results, we suggest that future research on designing interactive math system with graphing tasks should consider individual differences for their prior concept mastery levels. For instance, recent Massively Open Online Courses (MOOCs) continue to improve user experience by letting students conduct math problems solving other than watching video only. French et al. illustrated an interaction graphing component that can be integrated into the MOOCs. The interaction graphing component allows students to construct graphs using the similar sketch-based method that we compared with in this paper [20]. Applied our findings, novice students might prefer this sketch input into the interactive tutoring environment to explore math concepts instead of solving problems. When such students increase their concept mastery to become advanced students, they are prone to lean on the structured input method to improve input precision toward math problem-solving tasks.

In addition, learning science researchers have considered to adapt instructions, problems and learning styles into ILEs so as to improve students learning efficiency [13, 37, 45], we believe interactive math system may employ an intelligent user interface that could personalize the input method based on the level of the student and/or the task at hand within the context of geometry-based algebraic math learning and problem-solving. These intelligent user interface that account for student's prior knowledge could serve as a remediation strategy for students entering college who do not have the necessary background knowledge to excel in their college algebra classes [8]. However, such adaptive systems would have to be implemented cautiously as to not appear inconsistent or disruptive if adapted from one interface to the other over the course of time.

Limitations and Future Work

One limitation of our study is that we did not formally measure for accuracy or perceived accuracy of the sketch input method. Sketch-based gestural input for geometry graphing tasks relies heavily on recognition accuracy. Constraint-checking procedures to determine geometric shape relations is another critical component that affects users' perceptions and cognition. To account for these constraints, we trained our template-based gesture recognizer with a large set of training data and ran numerous pilot studies. During the study practice session, we also observed how participants interacted with the system in order to decide if we need to let them enter additional training sample gestures. During the study, we did not get any explicitly negative feedback on the sketch-based gesture triggering mechanisms nor the accuracy of the relation detection between shapes. Thus, we can assume with some confidence that the recognition and relation checking accuracy did not adversely affect users' perception. However, future work should use machine learning algorithms to further improve the geometric relation checking procedure used in our bimodal interface. Additionally, our evaluation relied heavily on self-reported subjective ratings. In future work, we plan to conduct a detailed writing feature analysis of how the users actually interacted with the system [42, 54].

Our results are generalizable to the context of algebra and geometry math problem-solving using interactive systems designed for student students. While this is an important and fairly broad context, we cannot say that we can generalize our findings beyond this scope. We propose that our findings might be applicable within learning contexts where adults use 2D interactive graphing interfaces that require some level of precision. However, future research should empirically test this hypothesis. Further, our participants were university who had already learned geometry and algebra concepts in high school. An implication for future research is to repeat this study with the high school to validate if our results hold true for different user populations. Otherwise, when performing 3D modeling tasks [5], sketch-based system might guide novice users to create 3D curve models in the initial learning phase while advanced users may prefer structured input methods to edit curve models more accurately. Therefore, we recommend further research studying structured versus naturalistic input methods for precision-based tasks.

CONCLUSION

HCI researchers and practitioners embrace the challenge of designing systems across different contexts by making them as user-friendly and natural to use as possible. This paper explores the notion that more naturalistic (i.e., "human-like") interaction may be prone to more human error; thus less appropriate for precision-based tasks that require a high level of accuracy, such as geometry input in analytical geometry math problem-solving. Unlike past research, which was solely the algebraic domain, we found that sketched-based gestural input may not be an optimal interaction technique in the context of math problem-solving when geometry graphing is involved. In addition, providing personalized input methods for geometry graphing tasks may be essential for meeting the needs of different students.

APPENDIX

The other original three math problems that were used as stimuli in our experiment:

Perpendicular lines: *"Line A is perpendicular to the line* $2y = 4x + 8$ *and passes through the point* $(-2;-8)$*. Graph Line A on the geometric coordinate canvas. Solve for the equation of line A on the algebraic canvas. You do not have to complete the above steps in any particular order."*.

Distance between a point and a line: *"Find the minimum distance between the point* $(1,3)$ *and the line* $y = -x + 6$*. Graph this line, point* $(1,-3)$*, and the line segment between them onto the geometric coordinate canvas. Solve for the value of the distance, d, in the algebraic canvas. Round your answer to two decimal places."*.

Tangent between a line and a Circle: *"Circle A is centered about the origin* $(0,0)$ *and has a radius of 5. Line B is tangent to circle A at the point* $(-3,4)$*. Graph Circle A and Line B on the geometric coordinate canvas. Solve for the equation for the line that is tangent to circle A at the point* $(-3,4)$*. You do not have to complete the above steps in any particular order."*.

REFERENCES

1. Christine Alvarado and Randall Davis. 2004. SketchREAD: a multi-domain sketch recognition engine.

In *Proceedings of the 17th annual ACM symposium on User interface software and technology*. ACM, 23–32.

2. Lisa Anthony and Jacob O Wobbrock. 2010. A lightweight multistroke recognizer for user interface prototypes. In *Proceedings of Graphics Interface 2010*. Canadian Information Processing Society, 245–252.
3. Lisa Anthony, Jie Yang, and Kenneth R Koedinger. 2012. A paradigm for handwriting-based intelligent tutors. *International Journal of Human-Computer Studies* 70, 11 (2012), 866–887.
4. Caroline Appert and Shumin Zhai. 2009. Using Strokes As Command Shortcuts: Cognitive Benefits and Toolkit Support. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 2289–2298. DOI: http://dx.doi.org/10.1145/1518701.1519052
5. Seok-Hyung Bae, Ravin Balakrishnan, and Karan Singh. 2008. ILoveSketch: As-natural-as-possible Sketching System for Creating 3D Curve Models. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology (UIST '08)*. ACM, New York, NY, USA, 151–160. DOI: http://dx.doi.org/10.1145/1449715.1449740
6. Olivier Bau and Wendy E. Mackay. 2008. OctoPocus: A Dynamic Guide for Learning Gesture-based Command Sets. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology (UIST '08)*. ACM, New York, NY, USA, 37–46. DOI: http://dx.doi.org/10.1145/1449715.1449724
7. Eveline J. Bellegarda, Jerome R. Bellegarda, David Nahamoo, and Krishna S Nathan. 1994. A fast statistical mixture algorithm for on-line handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16, 12 (1994), 1227–1233.
8. Eric P Bettinger and Bridget Terry Long. 2009. Addressing the Needs of Underprepared Students in Higher Education Does College Remediation Work? *Journal of Human resources* 44, 3 (2009), 736–771.
9. Christopher M Bishop, Markus Svensen, and Goeffrey E Hinton. 2004. Distinguishing text from graphics in on-line handwritten ink.. In *IWFHR*, Vol. 4. 142–147.
10. Harry N Boone and Deborah A Boone. 2012. Analyzing likert data. *Journal of extension* 50, 2 (2012), 1–5.
11. Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative research in psychology* 3, 2 (2006), 77–101.
12. James Dean Brown. 2011. Likert items and scales of measurement. *Shiken: JALT Testing & Evaluation SIG Newsletter* 1 (2011), 10–14.
13. Peter Brusilovsky and Christoph Peylo. 2003. Adaptive and intelligent web-based educational systems. *International Journal of Artificial Intelligence in Education (IJAIED)* 13 (2003), 159–172.
14. James Carifio and Rocco J Perla. 2007. Ten common misunderstandings, misconceptions, persistent myths and urban legends about Likert scales and Likert response formats and their antidotes. *Journal of Social Sciences* 3, 3 (2007), 106–116.
15. Salman Cheema, Sumit Gulwani, and Joseph LaViola. 2012. QuickDraw: improving drawing experience for geometric diagrams. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1037–1064.
16. Salman Cheema and Joseph LaViola. 2012. PhysicsBook: a sketch-based interface for animating physics diagrams. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*. ACM, 51–60.
17. Matthias Ehmann, M Gerhauser, Carsten Miller, and Alfred Wassermann. 2013. Sketchometry and jsxgraph-dynamic geometry for mobile devices. *South Bohemia Mathematical Letters* 21, 1 (2013), 1–7.
18. Franz Faul, Edgar Erdfelder, Albert-Georg Lang, and Axel Buchner. 2007. G* Power 3: A flexible statistical power analysis program for the social, behavioral, and biomedical sciences. *Behavior research methods* 39, 2 (2007), 175–191.
19. Kenneth D Forbus, Ronald W Ferguson, and Jeffery M Usher. 2001. Towards a computational model of sketching. In *Proceedings of the 6th international conference on Intelligent user interfaces*. ACM, 77–83.
20. Jennifer French, Martin A Segado, and Phillip Z Ai. 2016. Sketching Graphs in a Calculus MOOC: Preliminary Results. (2016).
21. Gene V Glass, Percy D Peckham, and James R Sanders. 1972. Consequences of failure to meet assumptions underlying the fixed effects analyses of variance and covariance. *Review of educational research* 42, 3 (1972), 237–288.
22. Tracy Hammond and Randall Davis. 2007. LADDER, a Sketching Language for User Interface Developers. In *ACM SIGGRAPH 2007 Courses (SIGGRAPH '07)*. ACM, New York, NY, USA, Article 35. DOI: http://dx.doi.org/10.1145/1281500.1281546
23. Sandra G Hart and Lowell E Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. *Advances in psychology* 52 (1988), 139–183.
24. Mary Ann Huntley, Robin Marcus, Jeremy Kahan, and Jane Lincoln Miller. 2007. Investigating high-school studentsÕ reasoning strategies when they solve linear equations. *The Journal of Mathematical Behavior* 26, 2 (2007), 115–139.
25. Takeo Igarashi, Satoshi Matsuoka, Sachiko Kawachiya, and Hidehiko Tanaka. 1997. Interactive Beautification: A Technique for Rapid Geometric Design. In *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology (UIST '97)*. ACM, New York,

NY, USA, 105–114. DOI: http://dx.doi.org/10.1145/263407.263525

26. Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. 1999. Teddy: A Sketching Interface for 3D Freeform Design. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 409–416. DOI: http://dx.doi.org/10.1145/311535.311602

27. Susan Jamieson and others. 2004. Likert scales: how to (ab) use them. *Medical education* 38, 12 (2004), 1217–1218.

28. Yingying Jiang, Feng Tian, Hongan Wang, Xiaolong Zhang, Xugang Wang, and Guozhong Dai. 2010. Intelligent understanding of handwritten geometry theorem proving. In *Proceedings of the 15th international conference on Intelligent user interfaces*. ACM, 119–128.

29. Shaun K. Kane, Jacob O. Wobbrock, and Richard E. Ladner. 2011. Usable Gestures for Blind People: Understanding Preference and Performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 413–422. DOI: http://dx.doi.org/10.1145/1978942.1979001

30. Kenneth R Koedinger, John R Anderson, William H Hadley, and Mary A Mark. 1997. Intelligent tutoring goes to school in the big city. (1997).

31. Per Ola Kristensson and Shumin Zhai. 2007. Command Strokes with and Without Preview: Using Pen Gestures on Keyboard for Command Selection. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07)*. ACM, New York, NY, USA, 1137–1146. DOI: http://dx.doi.org/10.1145/1240624.1240797

32. James A Landay and Brad A Myers. 1995. Interactive sketching for the early stages of user interface design. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press/Addison-Wesley Publishing Co., 43–50.

33. Yang Li. 2010. Protractor: a fast and accurate gesture recognizer. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2169–2172.

34. Rensis Likert. 1932. A technique for the measurement of attitudes. *Archives of psychology* (1932).

35. Arnold M Lund. 2001. Measuring usability with the USE questionnaire. *Usability interface* 8, 2 (2001), 3–6.

36. I Scott MacKenzie. 1992. Fitts' law as a research and design tool in human-computer interaction. *Human-computer interaction* 7, 1 (1992), 91–139.

37. Erica Melis, Eric Andres, Jochen Budenbender, Adrian Frischauf, George Goduadze, Paul Libbrecht, Martin Pollet, and Carsten Ullrich. 2001. ActiveMath: A generic and adaptive web-based learning environment. *International Journal of Artificial Intelligence in Education (IJAIED)* 12 (2001), 385–407.

38. Pam A Mueller and Daniel M Oppenheimer. 2014. The pen is mightier than the keyboard advantages of longhand over laptop note taking. *Psychological science* (2014), 0956797614524581.

39. Tom Y. Ouyang and Randall Davis. 2011. ChemInk: A Natural Real-time Recognition System for Chemical Drawings. In *Proceedings of the 16th International Conference on Intelligent User Interfaces (IUI '11)*. ACM, New York, NY, USA, 267–276. DOI: http://dx.doi.org/10.1145/1943403.1943444

40. Sharon Oviatt. 2013. Interfaces for Thinkers: Computer Input Capabilities That Support Inferential Reasoning. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction (ICMI '13)*. ACM, New York, NY, USA, 221–228. DOI: http://dx.doi.org/10.1145/2522848.2522849

41. Sharon Oviatt, Alex Arthur, and Julia Cohen. 2006. Quiet interfaces that help students think. In *Proceedings of the 19th annual ACM symposium on User interface software and technology*. ACM, 191–200.

42. Sharon Oviatt, Kevin Hang, Jianlong Zhou, and Fang Chen. 2015. Spoken Interruptions Signal Productive Problem Solving and Domain Expertise in Mathematics. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*. ACM, 311–318.

43. Fred Paas, Juhani E Tuovinen, Huib Tabbers, and Pascal WM Van Gerven. 2003. Cognitive load measurement as a means to advance cognitive load theory. *Educational psychologist* 38, 1 (2003), 63–71.

44. Brandon Paulson and Tracy Hammond. 2008. Paleosketch: accurate primitive sketch recognition and beautification. In *Proceedings of the 13th international conference on Intelligent user interfaces*. ACM, 1–10.

45. Elvira Popescu, Costin Badica, and Lucian Moraret. 2010. Accommodating learning styles in an adaptive educational system. *Informatica* 34, 4 (2010).

46. Peter D. Frisk R. David Gustafson. 1991. *Elementary Geometry, 3rd Edition*. John Wiley & Sons, Inc.

47. Dean Harris Rubine. 1991. *The automatic recognition of gestures*. Ph.D. Dissertation. Citeseer.

48. J Reynaldo A Santos. 1999. CronbachÕs alpha: A tool for assessing the reliability of scales. *Journal of extension* 37, 2 (1999), 1–5.

49. Jeremy Scott and Randall Davis. 2013. Physink: sketching physical behavior. In *Proceedings of the adjunct publication of the 26th annual ACM symposium on User interface software and technology*. ACM, 9–10.

50. Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. 2009. User-defined Gestures for Surface Computing. In *Proceedings of the SIGCHI*

Conference on Human Factors in Computing Systems (CHI '09). ACM, New York, NY, USA, 1083–1092. DOI: http://dx.doi.org/10.1145/1518701.1518866

51. Jacob O. Wobbrock, Andrew D. Wilson, and Yang Li. 2007. Gestures Without Libraries, Toolkits or Training: A $1 Recognizer for User Interface Prototypes. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology (UIST '07)*. ACM, New York, NY, USA, 159–168. DOI: http://dx.doi.org/10.1145/1294211.1294238

52. Richard Zanibbi and Dorothea Blostein. 2012. Recognition and retrieval of mathematical expressions. International Journal on Document Analysis and Recognition (IJDAR) 15, 4 (2012), 331–357.

53. Robert C Zeleznik, Andrew Bragdon, Chu-Chi Liu, and Andrew Forsberg. 2008. Lineogrammer: creating diagrams by drawing. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*. ACM, 161–170.

54. Jianlong Zhou, Kevin Hang, Sharon Oviatt, Kun Yu, and Fang Chen. 2014. Combining empirical and machine learning techniques to predict math expertise using pen signal features. In *Proceedings of the 2014 ACM workshop on Multimodal Learning Analytics Workshop and Grand Challenge*. ACM, 29–36.