An Intelligent System to Analyze Sketched Solutions to Open-Ended Truss Problems

Matthew Runyon
Texas A&M University
mattrunyon@tamu.edu

Seth Polsley
Texas A&M University
spolsley@tamu.edu

Blake Williford
Texas A&M University
bwilliford@tamu.edu

Sin-Ning Cindy Liu
Texas A&M University
sinning.cindy.liu@tamu.edu

Josh Hurt
Georgia Institute of Technology
jhurt30@gatech.edu

Julie Linsey
Georgia Institute of Technology
julie.linsey@me.gatech.edu

Tracy Hammond
Texas A&M University
hammond@tamu.edu

ABSTRACT

Engineering students need practical, open-ended problems to help them build their problem-solving skills and design abilities. However, large class sizes create a grading challenge for instructors as there is simply not enough time nor support to provide adequate feedback on many design problems. In this work, we describe an intelligent user interface to provide automated real-time feedback on hand-drawn free body diagrams that is capable of analyzing the internal forces of a sketched truss to evaluate open-ended design problems. The system is driven by sketch recognition algorithms developed for recognizing trusses and a robust linear algebra approach for analyzing trusses. Students in an introductory statics course were assigned a truss design problem as a homework assignment using either paper or our software. We used conventional content analysis on four focus groups totaling 16 students to identify key aspects of their experiences with the design problem and our software. We found that the software correctly analyzed all student submissions, students enjoyed the problem compared to typical homework assignments, and students found the problem to be good practice. Additionally, students using our software reported less difficulty understanding the problem, and the majority of all students said they would prefer the software approach over pencil and paper. We also evaluated the recognition performance on a set of 3000 sketches resulting in an f-score of 0.997. We manually reviewed the submitted student work which showed the handful of student complaints about recognition were largely due to user error.

IUI '21, April 14–17, 2021, College Station, TX, USA

ACM ISBN 978-1-4503-8017-1/21/04…$15.00
https://doi.org/10.1145/3397481.3450651

CCS CONCEPTS

• **Human-centered computing** → **Web-based interaction**; • **Computing methodologies** → **Artificial intelligence**; • **Applied computing** → **Interactive learning environments**; Engineering.

KEYWORDS

sketch recognition, engineering education, intelligent tutoring system

ACM Reference Format:
Matthew Runyon, Seth Polsley, Blake Williford, Sin-Ning Cindy Liu, Josh Hurt, Julie Linsey, and Tracy Hammond. 2021. An Intelligent System to Analyze Sketched Solutions to Open-Ended Truss Problems. In *26th International Conference on Intelligent User Interfaces (IUI '21), April 14–17, 2021, College Station, TX, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3397481.3450651

1 INTRODUCTION

On March 15, 2018, a pedestrian bridge by Florida International University (FIU) collapsed onto the road below, killing six and injuring nine [3, 66]. The investigation determined this deadly collapse was due to faulty design and failure to properly analyze nodes of the trusses under appropriate loads [6]. The FIU accident is not an isolated incident; such errors are surprisingly common. In fact, studies examining hundreds of bridge collapses have determined natural events (e.g., flooding) and human error as the two main sources of bridge failure [11, 65], and others have stated a lack of knowledge and/or carelessness as a contributing factor to many, if not most, of all such collapses [5, 13, 48].

It is critical that engineers have practical experience with core concepts [54], yet there is some disconnect between engineering education and the typical skills engineers will be using on the job [12]. One survey of professional engineers found that around 25% of participants relied on on-the-job experience to overcome weaknesses in their formal educations [16]. Educators and researchers are aware of the need to tie industry-related experience into engineering coursework [38], and many approaches have been developed to update curricula and programs to encourage more hands-on problems and creative thinking [9, 41, 56, 67].

Unfortunately, it can be difficult for teachers and schools to provide sufficient resources for more hands-on approaches as large class sizes and the time necessary to provide feedback can be burdensome [7]. As an example, consider free body diagrams, a concept integral to many fields of engineering and the basis of problems related to bridge design. These diagrams are illustrations used to visualize how forces and reactions occur on a given body. Students should have thorough feedback about their diagrams in entry classes so that they do not form misconceptions and unnecessarily struggle with future classes or worse, graduate into a job for which they are unprepared [32]. However, many university courses can be quite large, especially at the introductory level. The increased number of students and homework submissions lead to difficulty for those grading to provide the necessary feedback as even simple problems can require substantial work, and a class may have over one hundred students solving five such problems per week as homework.

To provide more timely feedback, some instructors have turned to web-based homework systems. These automated tools can help reduce reliance on instructor feedback by giving students their scores immediately [44], but typically, they only check the final answers and do not require diagrams. Feedback on diagrams is particularly important [4]. Furthermore, most online learning software lacks adaptability [63] and relies on instructors to design each assignment which adds another resource limitation.

To bridge the gap between enhancing engineering students' knowledge through more practical exercises while decreasing the burden for instructors and assistants, we turn to the adaptability of intelligent tutoring systems. These systems combine principles of machine learning into educational interfaces to provide students with more intelligent interactions than typical online learning tools [1, 24, 47]. Our solution is a web application that utilizes sketch-recognition algorithms to assess a sketched free body diagram and automatically grade student responses to homework questions. Returning to our discussion of bridge design, the software includes a *Creative Design* mode in which instructors only specify a few constraints. For example, they may ask students to design a bridge of a certain length capable of bearing a given load. From these constraints, students are encouraged to interact with the canvas freely, trying different approaches and getting feedback on their designs. The system is able to automatically grade their answers, even though there are numerous answers which can satisfy all the requirements. This allows instructors to assign these problems which would normally be too time-intensive to grade by hand, thus giving students the opportunity to apply their core knowledge to a scenario which is more applicable to the problem solving they would see in engineering jobs.

2 RELATED WORK

2.1 Creativity and Problem-Solving

From a pedagogical perspective, instilling the ability to critically think about problems and create solutions is an essential aspect of engineering education [10, 67]. Employers agree, as studies have shown that imagination and creativity are the most desired qualities they seek in candidates [35]. Moreover, the widely-used Bloom's Taxonomy positions creativity as the pinnacle of learning, placing it higher than skills like analysis and evaluation [26, 33].

The main finding regarding the FIU bridge incident was that the design was flawed [6]. In conjunction with the widely-recognized disconnect between what is taught in classrooms and what engineers need to know on the job [12, 16], these types of accidents illustrate the need for more open-ended and creative design problems in engineering coursework. Accordingly, efforts have been made to update curricula to account for job relevance and hands-on experience [38, 41, 56], but assessing this type of learning can be difficult and time-consuming [23]. Online tools, especially intelligent tutoring systems, are an appealing solution to these issues because they can incorporate elements of human intelligence into automated tools [1].

2.2 Existing Statics Grading Systems

Several major textbook publishers provide commercial online grading systems for statics problems; these include *Mastering Engineering*, *WileyPlus*, and *McGraw-Hill Connect*. However, they typically evaluate based on the final answer of a problem, with limited or no support for diagrams and the algebraic comparisons in some problems [18, 53], but these aspects are an important part of the learning process [28]. On the other hand, some systems may support diagrams, but do not typically involve numbers. Examples of such tools include the Andes physics tutoring system [58], the Free-Body Diagram Assistant [49], and the Conceptual Helper [2]. Their interactions are more similar to computer-aided drawing (CAD) systems, which is also different from freehand drawing. Similarly, *SkyCiv* is a powerful online bridge construction and analysis tool that combines diagramming and numeric solving, but it is built around a restrictive coordinate system and CAD-style interface.[1] A fairly comprehensive survey of common online homework systems is presented by Wilson and Kennedy [63]. In general, they find distinct advantages to online tools, although these options lag behind human teaching in areas of adaptation and intelligence, which is especially problematic in terms of supporting open-ended design questions.

2.3 Sketch Recognition

In this work, we seek to apply the natural interaction of sketching to creative design problems and focus on sketch recognition as the basis for an intelligent tutoring system. The selection of sketching is advantageous for design problems because sketch interaction has been strongly linked to creativity and learning [19, 42, 45, 59, 62]. Furthermore, sketch-based algorithms have been developed for a variety of applications related to gesture and diagram recognition. The seminal work in gesture recognition was produced by Rubine [50] in which he defined 13 important drawing features. These have been expanded to primitive shape recognition tools like *PaleoSketch* [46] and geometric systems such as *LADDER* [20] which uses domain-specific knowledge to define shape relationships and complex shapes by breaking them into combinations of primitive shapes. Another domain-specific tool is *nuSketch* [14], focusing on the geometric relationships between shapes such as "above" or "below." This was extended to *COGSketch* [15] to provide a tutoring

[1] https://skyciv.com

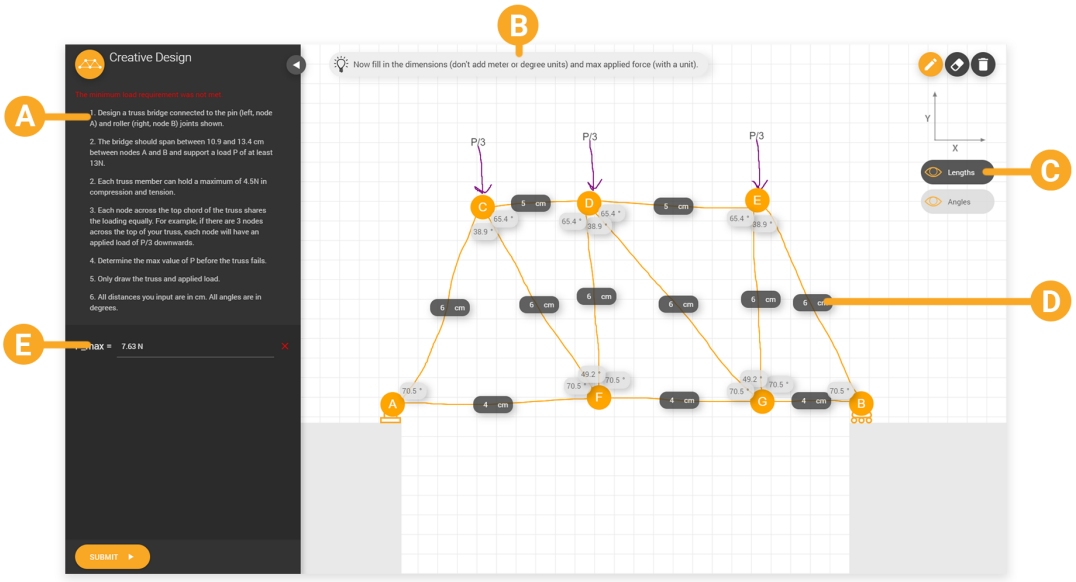

Figure 1: The user interface for creative design. (A) Instructions. (B) Step-by-step guidance. (C) Toggle for edge lengths and angles. (D) Sketch surface. (E) Numerical answer input.

system for students that can grade relationship-based problems, such as plate tectonics [17].

Sketch recognition is an important tool for intelligent tutoring systems as the natural form of interaction is supported by multiple studies that demonstrate students learn more and retain knowledge better when drawing and writing versus using a keyboard [40, 61]. Some related examples are *Newton's Pen* [31] and *Mechanix* [43, 57] which utilize early digital stylus computers to allow students to sketch diagrams and equations in physics problems while providing some feedback about their diagrams. Tools like *Newton's Pen II* have sought to overcome the technical limitations of the original *Newton's Pen* with wider support and capability [30]. A very interesting sketching systems for physics-based problems is Physics Book by Cheema and LaViola [8]. Both systems are well-rated, and while they are good options for specific kinds of FBD problems, such as those involving springs and pulleys, our system is focused around supporting creative design for trusses.

3 SYSTEM OVERVIEW

The system is a web-based tool for completing an open-ended truss design problem. Students are tasked with designing a bridge that meets the instructor's specifications. The instructor sets the minimum load the bridge must support, maximum tension and compression allowed for the beams, and distance range for the straight path from one side to the other. An example of the interface is shown in Figure 1.

Unlike generic truss problems, and distinct from most other online learning tools, these questions have no single answer. Students may come up with any number of solutions for a given set of constraints, so the system must have a robust algorithm for automatically analyzing submissions. Students must draw a valid truss which spans the gap, draw the applied load across the top of the truss, label the lengths and angles of their beams, and provide the maximum load their truss can support. The goal of this software is to provide immediate feedback to students which can help them learn through iterative design. We did not want to provide design recommendations since students are learning how to design around constraints; rather, we prefer to validate the student's solution and provide them with some general feedback such as where the truss will fail.

While the system can recognize trusses live, there is an added "Recognize" button for students to mark the state at which the full truss has been entered. This avoids providing unnecessary feedback to students as sometimes a truss may be recognized when it is a substructure of a larger system.

After the truss is complete, the students draw the applied forces at the appropriate nodes. In the initial problem, forces are split among the top chord of the truss, which we define as the nodes within 30 pixels height of the highest node. This number was chosen as it is the size of our feedback circles.

The next step is to input details about the lengths and angles of the beams in the truss. It is true that some of the relationships could be guessed from the sketched truss; however, drawing the truss to exact specifications would be difficult if not nearly impossible, so we choose to have user inputs to make the user experience better. A separate toggle exists to control whether or not the lengths and angles are shown so that students can determine how much detail they want to see. Also, to further improve user experience at this stage, the system auto-fills lengths and angles in a triangle once sufficient values have been defined to calculate the rest. Typically once three lengths/angles are provided in a triangle the other three can be calculated.

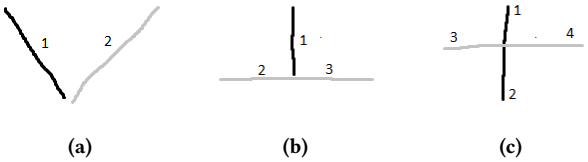

Figure 2: Three cases of substrokes intersecting

When students have completed their diagram, they enter their maximum supported load and can submit for feedback. The system combines recognition on the sketched elements with mathematical analysis of the inputs to evaluate their solution. If there is a miscalculation, they will be informed. If some truss members would break under the required load, the system returns a warning to report which beams would fail. The system concurrently checks that the other requirements of the problem are met such as minimum length and minimum supported load. If all problem constraints are met and the student analysis is correct, then the problem is marked correct.

4 ALGORITHMS

4.1 Sketch Segmentation

The first step of recognition is to segment the stroke created between a single pen-down and subsequent pen-up action. This step involves breaking the stroke into its separate substroke components. In this application, we split strokes based on corners using the Shortstraw corner finding method [64]. This method provides the locations of the corners within the stroke. Including the first and last points as the first and last corner, the points between each pair of corners is then converted into a substroke.

For each substroke created, we check if it intersects any other substrokes within a threshold of 30 pixels via Euclidean distance. The 30 pixel value was selected based on the node graphical element later drawn to the screen as visual feedback. If two endpoints intersect, such as in Figure 2a then no additional steps are taken. If one substroke intersects the middle section of another, such as in Figure 2b, then the intersected substroke is split in two, resulting in three substrokes. If the substrokes cross each other, such as in Figure 2c, then each is split, resulting in four substrokes.

4.2 Truss Recognition

Trusses are an important topic for engineering students to learn due to their strength, stability, and prevalence in structural design. The definition of a truss is extremely broad and more complicated than just a structure made of only triangles; any structure which has only two-force members (i.e., along the axis of the member) is considered a truss. This allows for trusses containing arcs or squares (e.g., queen post truss). This system focuses on a specific subset of trusses called planar trusses which occur in a two-dimensional space. Although trusses can contain non-triangles, the problems explored in this system and many introductory courses all contain only triangles, so we limit the system to recognizing trusses composed solely of triangles.

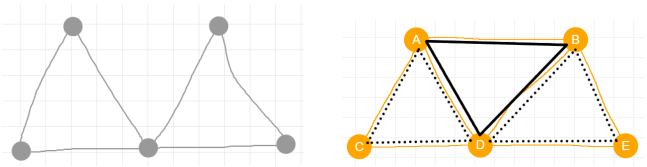

Figure 3: (a) Example of a non-truss with all triangles, but not all sharing a side. (b) Illustration of the triangles and sharing edges in a truss

For the purposes of this project, we define a truss to have the following properties:

(1) A connected graph
(2) Only made up of triangles
(3) Each triangle shares at least 1 side with another

A truss which follows this definition will need three external reaction forces to be statically determinate. These are commonly provided by a pin support (horizontal and vertical reaction force) and a roller support (vertical reaction force). Trusses are recognized in the following four steps:

4.2.1 Graph conversion. The first step in recognizing a truss is to convert the substrokes into a graph. Each substroke end is a possible node, and the substroke itself is an edge in the graph. If the ends are within a threshold Euclidean distance of 30 pixels from an existing node in the graph, then the substroke is considered to have one end connected to the existing node. This threshold was chosen to match the diameter of the circles shown to the user, so the visual feedback matches the internal representation of the sketch.

4.2.2 Connected graph check. After the substrokes have been converted to a graph, the next step is to check that the graph is connected. This is achieved through a standard depth-first search starting from any node while keeping track of the nodes visited. If there are any disjointed sections, then the sketch cannot be a truss because trusses are connected by definition.

4.2.3 Triangle check. The third step is to check that a path of length 3 exists for traveling from each node to itself. This ensures that each node is a component of a triangle. This is done via a breadth-first search that tracks the number of steps taken from the beginning node to the current node. We keep track of distinct triangles based on their node labels in alphabetical order (e.g. the triangle containing nodes A, B, and C is ABC, not BAC).

4.2.4 Adjacent triangle check. The final step is to check that all triangles share an edge with at least one other triangle, unless there is only one triangle. This prevents sketches that may be composed entirely of triangles but are not trusses such as in Figure 3a. This is achieved by counting the number of times each edge occurs based on the distinct triangle names. Then, each distinct triangle is checked to confirm at least one of its edges occurs in two triangles.

4.3 Internal Force Calculation

In order to verify the solution provided by the student, all of the internal forces must be calculated. The following explanation will

use the simple truss of a single, equilateral triangle with side length 3cm shown in Figure 4 to help illustrate the process.

4.3.1 Logical coordinates. The first step to calculating the internal forces is converting the sketch into what we call the logical x-y space. It would be nearly impossible to sketch the truss to perfect scale, and such a requirement would make for a terrible user experience. As a result, we cannot use the coordinates of the sketch canvas but instead must convert the relative positions, labeled side lengths, and angles to create an accurate representation of the truss.

We start by assigning the coordinates $(0, 0)$ to node A.Then, the edge with the angle closest to 0 is found, and that node is labeled as horizontal from node A. The instructor specified that the truss needed to have a horizontal path from A to B. Future problems can support any style path by requesting students input the angle from the x-axis to the nearest node if no node is horizontally across from A. In the example, node B. is assigned the position $(3, 0)$ based on the length of $\overline{AB}$.

Now that the coordinates of two nodes are determined, the remaining node coordinates can be found through a rotation matrix. A node without logical coordinates is solvable is the other nodes in the triangle already have coordinates. Starting with node A, we perform a breadth-first search (BFS) where any solvable node without logical coordinates is solved, and any node with coordinates is added to the BFS queue. In the example, the coordinates for C can be determined using the following equations where $\hat{AB}$ is the unit vector from A to B and θ_{BAC} is the labeled angle at node A. The sign of the cross product of the two vectors is used to determine if the angle is positive or negative.

$$\hat{AC} = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} \hat{AB}$$

$$\theta = sign(\overrightarrow{AB} \times \overrightarrow{AC}) * \theta_{BAC}$$

Once the unit vector is found for $\overline{AC}$, the coordinates are determined by the labeled length multiplied by the unit vector, and then added to the coordinates of A as shown in the equation below. The coordinates of C are (1.5, 2.6).

$$\begin{bmatrix} C_x \\ C_y \end{bmatrix} = \begin{bmatrix} A_x \\ A_y \end{bmatrix} + \|AC\| * \hat{AC}$$

4.3.2 Matrix structural analysis. Now that the truss has logical coordinates, the next step is to perform a matrix analysis to determine if the truss will meet the loading requirements. Matrix structural analysis is a broad topic which covers methods for analyzing statically determinate trusses (the type in this problem) by creating a matrix of equilibrium equations. Since the software is used in introductory classes, the students are not performing analysis of bending or materials, so a more complicated method such as finite-element analysis is not needed and the simpler matrix approach will calculate all internal forces. Our matrix approach is a modification of those found in Matrix Structural Analysis [34].

The system of equations created will need to follow the format $Ax + B = 0$ where A is a square coefficient matrix, x is a column matrix of the unknown values, and B is a column matrix of any externally applied forces on the system.

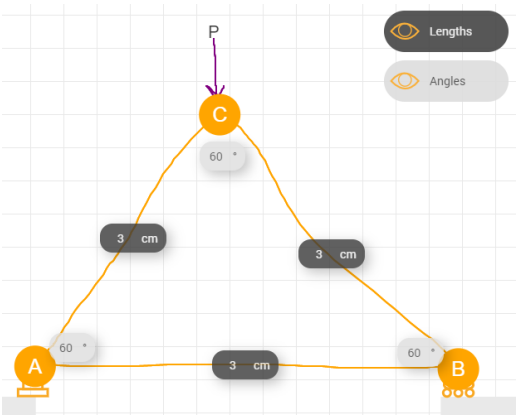

Figure 4: Example of a simple truss with side lengths of 3cm

In order to create a system of equations, we use a method of joints analysis on the truss. Each joint in the truss can provide two static equilibrium equations: one in the x-direction and one in the y-direction. In order to have a solvable system, we must have an equal number of equations and unknowns. The first triangle in the truss provides 3 nodes and thus 6 equations. The 6 unknowns are the force within each of the 3 beam and the 3 reaction forces. Every additional triangle added will add only 1 node to the truss. This results in 2 additional equations to solve for the 2 additional beams that were added. Therefore, there will always be an equal number of equations and unknowns in the trusses accepted for this problem.

Since the problem does not need the value of the reaction forces, we ignore any of the three equations that have the reaction force present. Then, at each node the static equilibrium equations are added to a matrix. The logical coordinates are used to get the angle for determining the components of each force at a node. In this example, both equilibrium equations are skipped at A since they involve support reaction forces. Then, the x-direction equilibrium equation at B is added to the matrix. Finally, both equilibrium equations at node C are added to the matrix, and the applied force, P, is added to matrix B. The matrix for the equilibrium equations in the example are shown below.

$$\begin{bmatrix} -1 & -cos(60°) & 0 \\ 0 & cos(60°) & -cos(60°) \\ 0 & -sin(60°) & -sin(60°) \end{bmatrix} \begin{bmatrix} F_{AB} \\ F_{BC} \\ F_{AC} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -P \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

We then move the constants to the right side and utilize lower-upper (LU) decomposition and the superposition principle to solve.

The superposition principle is used to simplify solving any systems with multiple applied forces. Instead of solving in variable terms, we solve the system once per externally applied force. When solving for that force, we use the value 1 for it and 0 for any other external forces. In this case, only P is applied, so we just substitute 1 for P and solve with LU decomposition.

The solution gives us the coefficients for how the applied force affects each beam. Since statically determinate trusses are linear systems which follow the principle of superposition, adding the

solutions of the answers for each applied force gives the total force in each beam. In this case, we get the following.

$$\begin{bmatrix} 0.288 \\ -0.577 \\ -0.577 \end{bmatrix} P = \begin{bmatrix} F_{AB} \\ F_{BC} \\ F_{AC} \end{bmatrix}$$

Note that we would simply have an additional term added on the left side for every applied force. Negative numbers indicate the beam is in compression.

Now we can easily check if the system will meet the loading requirements by finding the greatest positive and negative coefficients in the equation to determine which beams are under the largest tensile and compressive load. Then we can calculate the value of P which creates the maximum allowed force in tension and compression to determine the maximum allowable force.

5 METHODOLOGY

5.1 Study Design

The creative design problem was tested in an introductory statics course with a total of 58 students and was part of a larger multi-university study evaluating different aspects of the software. The students were randomly assigned to experimental and control groups with an equal number of participants in each. Students were given instructions to design a bridge with a range of acceptable lengths, the maximum amount of tension and compression each beam can hold, and a minimum amount of load that the bridge should be able to support. The experimental group was given a short demonstration on how to use the creative design mode software, and students in the control group were required to use paper and pencil since there were no existing systems used in classrooms that support grading this type of problem with sketching.

In this paper, we examine the user experience and algorithmic implementation of the creative design system. An analysis of student scores on the homework showing that the software group trended towards performing slightly better than the pencil and paper group is discussed in [22], including other pedagogical implications. More detail about the broader multi-university study and the software interface itself for more traditional, not open-ended homework problems is discussed in [60] and [51].

5.2 Focus Groups

At the end of the semester, students were offered extra credit in exchange for participating in a one hour focus group session. Focus groups were selected to allow participants to share ideas among each other and in an effort to gain more candid responses from students than they may feel comfortable than in one-on-one interviews [25, 29]. Overall, four focus groups were administered containing four students each. Two of the groups contained only students from the control group who did the problem with paper and pencil, and two of the groups contained only students from the experimental group who did the problem with our software. The focus groups covered more topics than just the creative design problem, and the questions specific for our analysis are listed below.

(1) What did you think about the creative design problem? Did you find it useful? Why/why not?
(2) What did you think about the creative design interface? What did you like/dislike about it?
(3) Would you have preferred to use the opposite method (paper/pencil or software) to complete the problem? Why/why not?

5.3 Conventional Content Analysis

The student focus groups were transcribed using Otter.ai[2] and analyzed using conventional content analysis (CCA) on ATLAS.ti Cloud[3]. CCA was conducted in three phases: 1) creating a hierarchy of codes, 2) confirming the hierarchy, and 3) finalizing it, as detailed in [21]. In this context, a *code* is "a word or short phrase that symbolically assigns a summative, salient, essence-capturing, and/or evocative attribute" to the data [52]. In the first phase, two trained coders individually skimmed all transcripts and identified relevant keywords and phrases as potential codes [36, 37, 39, 55]. After meeting to reach consensus and organizing the codes into a hierarchy of terms, two additional coders confirmed the hierarchy by using it to apply codes to the transcripts themselves [39, 52]. In the third phase, the four coders had one final meeting to discuss any amendments and finalize the hierarchy.

6 RESULTS AND DISCUSSION

The focus groups yielded several important points of discussion, which we have organized around three primary aspects: the design problem itself, a comparison of the assignment methods, and concerns specific to our system. For simplicity, we will refer to the paper and pencil group as P/P and those who used our software as SW.

Table 1: Number of occurrences for codes related to student experience with the design problem.

Reported Experiences	SW	P/P
Solving the Problem		
Iterative Design	12	8
Real-World Applications	2	1
Encouraged Critical Thinking	3	0
Encouraged Creativity	4	0
Encouraged Research	1	1
Good Practice	6	2
Different from Homework	3	1
Thoughts on the Problem		
Liked Problem	6	4
Problem was Helpful/Useful	14	1
Problem was Difficult	2	0
Problem was Frustrating	1	2
Problem was Not Helpful	1	6
Other Factors		
Confusion Regarding Problem	0	7
Difficulty Reaching Professor	0	3
Used Point & Click Truss Analysis	0	2
Time and Travel Interference	5	0

[2]https://otter.ai/
[3]https://atlasti.com

6.1 Student Experiences with the Design Problem

When considering students' experiences with the creative design problem, we found that there were some similarities between the reported experiences of those who used our system (SW) and those using paper and pencil (P/P) (see Table 1 for detailed comparisons). Here we report some key codes, along with illustrative quotes from which we derived these codes.

Students from both experimental groups described using an **iterative design process**. One student in SW said, "I feel like getting to apply that knowledge [and . . .] having to learn from the mistakes that we've made in that design and improving it based on that, it was pretty cool to apply a lot of the topics that we had taken rather than just problem solving itself. I guess it felt like a breath of fresh air compared to everything else."

Students in both groups mentioned that the problem was useful to **real-world applications**. One student in P/P remarked, "I'm studying engineering to be able to do design work like this [. . .] Even though it was more work, [I. . .] actually enjoyed doing that because I felt like I was actually applying what I learned. And I like designing stuff. So yeah, that was interesting."

We also found notable differences between the experiences of those in SW and P/P. Students who used the software described the problem as encouraging **critical thinking** and **creativity**. One student shared, "I really like projects and stuff because it makes you feel like more critically thinking about what you're doing." Another noted that the problem "just exposed me to different designs and [. . .] it gave me experience in solving other shaped designs besides the normal triangle bridge. So that's why I liked it."

Additionally, more students in SW described the problem as being **helpful** and **good practice** for truss analysis. One student stated "we're pretty much all studying to be engineers. The fact that we were able to apply the knowledge that we had in a very important engineering class in the class itself [. . .] rather than having to wait to me was a pretty cool thing." Another student shared this very positive thought: "At that time, I had been struggling with using method of joints and methods of sections, with trusses. And the fact that I had to do so many iterations, I guess, gave me good practice for that. And so I feel like I can confidently say that I became better at truss analysis and using method of joints and sections because of that problem."

6.2 Comparing Software to Paper and Pencil

Overall, 8 students reported that they preferred the software approach (5 in SW and 3 in P/P), 4 students preferred paper and pencil (2 in SW and 2 in P/P), 4 students had no preference (1 in SW and 3 in P/P). Students in SW who preferred that method explained that they valued the **real-time feedback** provided by the system. Similarly, the students in P/P who would have liked to try the software cited several reasons, including the desire for **real-time feedback**, a perceived improvement in **efficiency**, and support for **iterative design**. In fact, two students in P/P reported looking up other online tools to help them approach the problem. Conversely, those in P/P who did not want to switch explained that they did not want to deal with a **software learning curve**, and students in SW who preferred P/P did not provide any particular reasons beyond familiarity with using paper and pencil to solve their work.

Another critical point of comparison between these two approaches regarded external factors. Due to COVID-19, the students involved in the study transitioned to online courses partway through their semester. This was especially problematic to some in P/P, as they felt **confusing wording** of the problem and lack of **professor availability** during the transition made it difficult to understand the task.

No one in SW cited such concerns, as the software provided them instructions and feedback on-demand; however, they did remark on the impacts of **travel** and **time management** during this period. That contrast was particularly interesting to see; in a way, students in P/P complained that their problems were beyond their control, stemming from the assignment itself or their professor, while those with access to an online tool recognized factors they can influence, like time management.

6.3 Experiences with the Software

During the focus groups, students in SW were asked about their experiences with the system. Three sub-categories of codes arose from the data: a) advantages, b) constraints, and c) disadvantages. Many of the students in SW reported that the **real-time feedback** was a major advantage. One student explained that the program "was really open-ended and it gave you specific feedback on your open-ended design. So, I really liked the nature of it." This was also the main cited advantaged for why students would prefer to use the software again if they had more creative design problems. Similarly, several students said that they felt using a specially-designed interface such as this was a very **efficient** approach.

There were some challenges as well. The most commonly reported technological constraint was **intermittent recognition issues**, and on the side of disadvantages a few students felt that the interface was **less streamlined than paper**, either because of their familiarity with that approach or confusion using parts of the interface: "I didn't know that, that [the software] even gave like feedback at that time."

It is worth noting that at least one student who had recognition issues did not attend the class where a brief overview of the system was given. Some of the issues may also have been related to the students' input devices (some only had trackpads for sketching). We hope to address some potential issues with better canvas manipulation tools, and the next section discusses algorithmic concerns in more detail.

6.4 Recognition Considerations

A couple students reported encountering intermittent recognition issues in the focus groups, and some students were concerned about a software learning curve associated with moving to a digital system from pencil and paper. This led us to examine common error cases regarding truss and arrow recognition specific to the creative design problem.

Of the students in the software group, we received 36 problem submissions. To identify students who encountered recognition issues, we searched for anyone who had to erase any parts of their

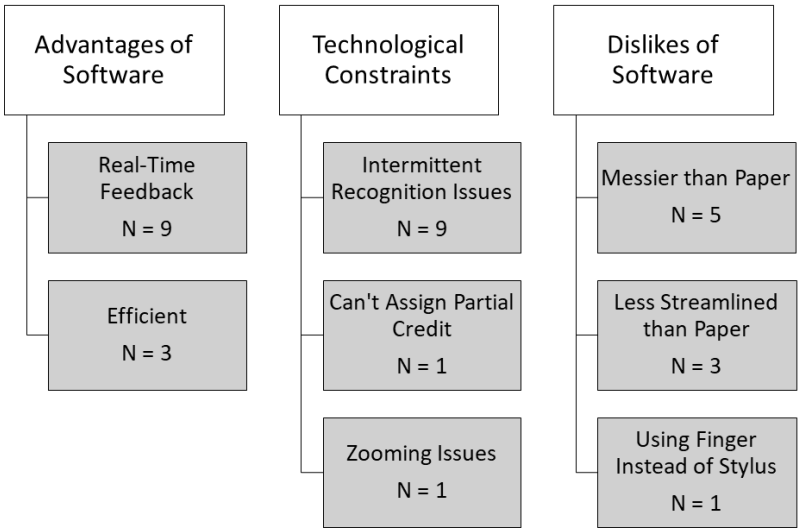

Figure 5: Number of occurrences for codes related to the software experience

sketches. This resulted in 26 sketches, and a manual review showed that only 9 of these had encountered some kind of issue.

We determined that most of the errors were the result of users struggling with certain parts of the interface. For instance, one cause of failed recognition was if a student drew an arrow for a force before marking the truss as complete. The software requires students to mark the truss diagram complete before adding forces. This is because students can enter any truss configuration, so although a valid truss may be present, the student may still be modifying it. We enforce this constraint because students will tend to take the easiest path if the system automatically presents it [27], and it is important that they be solving the problem and learning as much as possible themselves.

The lack of multi-stroke support for single truss members also affected a couple of students. The system assumes each stroke is a beam rather than a partial beam. Users are given a visual cue when this error occurs by the appearance of an erroneous node along the member, but they still must correct it manually. Figure 6 demonstrates an example truss where the user indicated an extra node but did not immediately realize it until after attempting to correct the recognition by redrawing parts of the truss. In the future, we may automatically combine strokes along a single member if the nodes of multiple strokes are collinear; however, the choice is not simple as there were multiple students who intentionally drew several collinear members in a row so they could use the nodes to align their truss.

Lastly, we noticed a couple cases involving thresholding. On occasion, a student may have to redraw a truss member if the nodes fall outside the threshold of the expected joint location; however, they did not complain of these as a recognition issue because the interface provides visual feedback of each node's threshold with a grey circle. Arrows were the other concern because the arrow recognizer considers the length of the shaft versus the sides of the

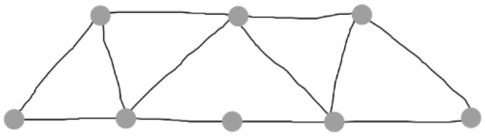

Figure 6: The student drew the bottom member in multiple strokes, leading the system to expect a member attached to the lower middle node. The student believed there was a recognition error until they remembered to add the middle support.

arrowhead when determining validity and orientation. The existing algorithm rejected short, wide arrows which we believe was the cause for some recognition complaints. We have addressed this by relaxing constraints to allow for arrows with head lengths that are almost as long as the shaft.

6.5 Algorithm Accuracy

To evaluate the performance of the truss recognition algorithm, we applied the recognizer to a dataset of 3000 sketches. The majority of these sketches came from a previous implementation of this software which used different recognition algorithms. In total, with 1500 truss and 1500 non-truss examples, our recognition algorithm correctly identified 1481 trusses and rejected all non-trusses. Upon visual inspection of the missed trusses, we were surprised to find mislabeled trusses in the dataset based on an older truss recognition algorithm. This means that the new algorithm was actually more correct; 10 of the trusses it rejected were not valid trusses despite being in the larger collection of 1500 "truss" shapes. Of the 9 trusses that were truly missed, they were found to contain squares, single

members comprised of multiple strokes, or very messy linework. Accounting for the mislabled trusses in the dataset, there were only 1490 truss shapes, yielding an overall f-score of 0.997.

7 FUTURE WORK

During focus group discussions, students identified a few technological issues they faced. In terms of user experience, it is vital that the recognition be seamless and accurate, and we believe there are several steps to take that will enhance the algorithm. First, providing the option to zoom or scale the canvas could help mitigate space issues, especially on smaller screens.

There will always be some risk of misrecognition as student work can be messy and sometimes the solutions may be difficult for even a human grader to understand, but enhancements to the underlying machine learning algorithms could improve recognition accuracy. We could also enhance the visibility of errors and feedback, such as the case where users have an invalid truss or non-truss elements interfering with recognition, such as arrows.

We would also like to use the algorithms and interfaces developed to improve general problem creation for instructors. Currently, instructors who want to create their own problems for a regular homework must calculate and validate their trusses before assigning the problems. However, the algorithm which solves and checks student work could be reversed to automatically do the calculations for professors after they draw the truss they envisioned. This could enable professors to deploy many custom problems while reducing their workload and encouraging students to further practice core concepts.

8 CONCLUSION

In this work, we present an online system for automatically grading open-ended truss design problems. The system leverages novel sketch recognition algorithms and a robust linear algebra approach for analyzing the truss to accept custom solutions to constraints-based homework questions. Instructors need not provide answers when creating homeworks, only the requirements, and students can create any number of designs to fulfill these requirements. Our system helps fill the need for engineering students to receive more real-world design experience in their courses through intelligent tutoring, and reduces the workload on teachers while providing students a deeper understanding of core concepts.

This paper considered qualitative results from four focus groups including students in the paper and pencil control and software experiment groups. Overall, students in both groups were able to complete the problem and found that it encouraged iterative design and critical thinking. Students using our software were more likely to enjoy the problem and consider it helpful. Conversely, students in the paper and pencil group complained of confusing wording and difficulty reaching the professor, illustrating the hardship it places on teachers to manually manage these types of problems for large classes. The majority, including paper and pencil students, said they would prefer to use the online system for these types of problems while often citing the real-time feedback and efficiency of iterating on the online system.

The user interface could be further improved through more detailed feedback to the students; however, as an instructional tool we do not want to suggest design recommendations and instead want to provide assessment of the design correctness. Reviewing the recognizer performance, we found that the system performed very well, surpassing even our previous generation of truss recognizers. Intelligent user interfaces like this system have the potential to enhance engineering curriculum at all levels by providing students on-demand learning through applied exercises.

ACKNOWLEDGMENTS

We would like to acknowledge the NSF for their support via grants 1726306, 1725423, 1725659, 1726047, and 1725785 as well as our other collaborators Dr. Ben Caldwell, Dr. Kristi Shryock, Dr. Kim Talley, Dr. Vimal Viswanathan, and Morgan Weaver for their help with this project.

REFERENCES

[1] Alaa N Akkila, Abdelbaset Almasri, Adel Ahmed, Naser Al-Masri, Yousef S Abu Sultan, Ahmed Y Mahmoud, Ihab S Zaqout, and Samy S Abu-Naser. 2019. Survey of Intelligent Tutoring Systems up to the end of 2017. *International Journal of Academic Information Systems Research* 3, 4 (2019), 36–49.

[2] Patricia L Albacete and Kurt VanLehn. 2000. The Conceptual Helper: An Intelligent Tutoring System for Teaching Fundamental Physics Concepts. In *International Conference on Intelligent Tutoring Systems*. Springer, Montreal, Canada, 564–573.

[3] Crispin Andrews. 2018. A bridge too far? *Engineering & Technology* 13, 4 (2018), 70–74.

[4] Robert L Bangert-Drowns, Chen-Lin C Kulik, James A Kulik, and MaryTeresa Morgan. 1991. The Instructional Effect of Feedback in Test-like Events. *Review of Educational Research* 61, 2 (1991), 213–238.

[5] María Victoria Biezma and Frank Schanack. 2007. Collapse of steel bridges. *Journal of Performance of Constructed Facilities* 21, 5 (2007), 398–405.

[6] Ran Cao, Sherif El-Tawil, and Anil Kumar Agrawal. 2020. Miami Pedestrian Bridge Collapse: Computational Forensic Analysis. *Journal of Bridge Engineering* 25, 1 (2020), 04019134.

[7] Ceilidh Barlow Cash, Jessa Letargo, Steffen P Graether, and Shoshanah R Jacobs. 2017. An Analysis of the Perceptions and Resources of Large University Classes. *CBE—Life Sciences Education* 16, 2 (2017), ar33.

[8] 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, Lisbon, Portugal, 51–60.

[9] Nuno Cosme, Michael Z Hauschild, Christine Molin, Ralph K Rosenbaum, and Alexis Laurent. 2019. Learning-by-doing: experience from 20 years of teaching LCA to future engineers. *The International Journal of Life Cycle Assessment* 24, 3 (2019), 553–565.

[10] Shanna R Daly, Erika A Mosyjowski, and Colleen M Seifert. 2014. Teaching creativity in engineering courses. *Journal of Engineering Education* 103, 3 (2014), 417–449.

[11] Lu Deng, Wei Wang, and Yang Yu. 2016. State-of-the-art review on the causes and mechanisms of bridge collapse. *Journal of Performance of Constructed Facilities* 30, 2 (2016), 04015005.

[12] Patrícia Fernanda dos Santos and Alexandre Tadeu Simon. 2018. An evaluation of the competences and abilities of the production engineer in the industrial environment. *Gest. Prod* 234 (2018), 250.

[13] Ziad A Eldukair and Bilal M Ayyub. 1991. Analysis of recent US structural and construction failures. *Journal of performance of constructed facilities* 5, 1 (1991), 57–73.

[14] K Forbus, Kate Lockwood, Matthew Klenk, Emmett Tomai, and Jeffrey Usher. 2004. Open-Domain Sketch Understanding: The nuSketch Approach. In *AAAI Fall Symposium on Making Pen-based Interaction Intelligent and Natural*. AAAI Press, Arlington, VA, 58–63.

[15] Kenneth Forbus, Jeffrey Usher, Andrew Lovett, Kate Lockwood, and Jon Wetzel. 2011. CogSketch: Sketch Understanding for Cognitive Science Research and for Education. *Topics in Cognitive Science* 3, 4 (2011), 648–666.

[16] Joaquín Fuentes-Del-Burgo and Elena Navarro-Astor. 2016. What is engineering education for? Listening to the voices of some Spanish building engineers. *Journal of Engineering, Design and Technology* 14, 4 (2016), 897–919.

[17] Bridget Garnier, Maria Chang, Carol Ormand, Bryan Matlen, Basil Tikoff, and Thomas F Shipley. 2017. Promoting sketching in introductory geoscience courses: CogSketch geoscience worksheets. *Topics in cognitive science* 9, 4 (2017), 943–969.

[18] Abigail S Gertner. 1998. Providing Feedback to Equation Entries in an Intelligent Tutoring System for Physics. In *International Conference on Intelligent Tutoring*

Systems. Springer, San Antonio, TX, 254–263.
[19] Tracy Hammond. 2015. Dialectical creativity: Sketch-negate-create. In *Studying Visual and Spatial Reasoning for Design Creativity*. Springer, Dordrecht, England, 91–108.
[20] Tracy Hammond and Randall Davis. 2005. LADDER, A Sketching Language for User Interface Developers. In *Computers & Graphics*, Vol. 29–4. Elsevier, Amsterdam, The Netherlands, 518–532. https://doi.org/10.1016/j.cag.2005.05.005
[21] Hsiu-Fang Hsieh and Sarah E Shannon. 2005. Three approaches to qualitative content analysis. *Qualitative health research* 15, 9 (2005), 1277–1288.
[22] J. Hurt, M. Runyon, T. A. Hammond, and J. S. Linsey. 2020. A Study on the Impact of a Statics Sketch-Based Tutoring System Through a Truss Design Problem. In *2020 IEEE Frontiers in Education Conference (FIE)*. IEEE, Uppsala, Sweden, 1–7. https://doi.org/10.1109/FIE44824.2020.9274208
[23] Victoria A Jideani and IA Jideani. 2012. Alignment of assessment objectives with instructional objectives using revised Bloom's taxonomy—The case for food science and technology education. *Journal of Food Science Education* 11, 3 (2012), 34–42.
[24] Igor Jugo, Bozidar Kovacic, and Vanja Slavuj. 2016. Increasing the adaptivity of an intelligent tutoring system with educational data mining: A system overview. *iJET* 11, 3 (2016), 67–70.
[25] Bridget Turner Kelly. 2003. Focus group interviews. In *Research in the college context: Approaches and methods*. Rutledge, London, UK, 49–62.
[26] David R Krathwohl and Lorin W Anderson. 2009. *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. Longman, London, England.
[27] Steve Krug. 2000. *Don't make me think!: a common sense approach to Web usability*. New Riders, San Francisco, California.
[28] Diarmaid Lane, Niall Seeery, and Seamus Gordon. 2009. The understated value of freehand sketching in technology education. *Engineering Design Graphics Journal* 73, 3 (2009), 13–22.
[29] Linda Costigan Lederman. 1990. Assessing educational effectiveness: The focus group interview as a technique for data collection. *Communication education* 39, 2 (1990), 117–127.
[30] C Lee, Josiah Jordan, Thomas F Stahovich, and James Herold. 2012. Newtons pen ii: an intelligent, sketch-based tutoring system and its sketch processing techniques. In *Proceedings of the International Symposium on Sketch-Based Interfaces and Modeling*. ACM, Annecy, France, 57–65.
[31] WeeSan Lee, Ruwanee de Silva, Eric J Peterson, Robert C Calfee, and Thomas F Stahovich. 2008. Newton's Pen: A Pen-Based Tutoring System for Statics. *Computers & Graphics* 32, 5 (2008), 511–524.
[32] Anastasiya A Lipnevich and Jeffrey K Smith. 2009. "I Really Need Feedback to Learn:" Students' Perspectives on the Effectiveness of the Differential Feedback Messages. *Educational Assessment, Evaluation and Accountability* 21, 4 (2009), 347.
[33] Matthew C Makel. 2009. Help us creativity researchers, you're our only hope. *Psychology of Aesthetics, Creativity, and the Arts* 3, 1 (2009), 38.
[34] William McGuire, Richard H Gallagher, and Ronald D Ziemian. 2000. *Matrix Structural Analysis*. CreateSpace Independent Publishing Platform, Scotts Valley, California, 9–49.
[35] Erica McWilliam. 2009. Teaching for creativity: from sage to guide to meddler. *Asia Pacific Journal of Education* 29, 3 (2009), 281–293.
[36] Matthew B Miles and A Michael Huberman. 1994. *Qualitative data analysis: An expanded sourcebook*. Sage Publications, Thousand Oaks, CA.
[37] David L Morgan. 1993. Qualitative content analysis: a guide to paths not taken. *Qualitative health research* 3, 1 (1993), 112–121.
[38] Sophie Morin, Jean-Marc Robert, and Liane Gabora. 2018. How to train future engineers to be more creative? An educative experience. *Thinking Skills and Creativity* 28 (2018), 150–166.
[39] Janice M Morse, Peggy-Anne Field, et al. 1995. *Qualitative research methods for health professionals*. Vol. 2. Sage Publications, Thousand Oaks, CA.
[40] Pam A Mueller and Daniel M Oppenheimer. 2014. The pen is mightier than the keyboard: Advantages of longhand over laptop note taking. *Psychological science* 25, 6 (2014), 1159–1168.
[41] Jesús Muñuzuri, Pablo Cortés, Rafael Grosso, and Luis Onieva. 2016. Teaching heuristic methods to industrial engineers: A problem-based experience. *INFORMS Transactions on Education* 16, 2 (2016), 60–67.
[42] Kumiyo Nakakoji, Atau Tanaka, and Daniel Fallman. 2006. "Sketching" nurturing creativity: commonalities in art, design, engineering and research. In *CHI'06 extended abstracts on Human factors in computing systems*. ACM, Montreal, Canada, 1715–1718.
[43] Trevor Nelligan, Seth Polsley, Jaideep Ray, Michael Helms, Julie Linsey, and Tracy Hammond. 2015. Mechanix: a sketch-based educational interface. In *Proceedings of the 20th International Conference on Intelligent User Interfaces Companion*. ACM, Atlanta, Georgia, 53–56.
[44] Elizabeth Odekirk-Hash and Joseph L Zachary. 2001. Automated Feedback on Programs Means Students Need Less Help From Teachers. In *ACM SIGCSE Bulletin*, Vol. 33:1. ACM, New York, NY, 55–59.
[45] Martin Pache, Anne Römer, Udo Lindemann, and Winfried Hacker. 2001. Sketching behaviour and creativity in conceptual engineering design. In *Proceedings of the International Conference on Engineering Design (ICED'01)*. Springer, Berlin, Germany, 243–252.
[46] Brandon Paulson and Tracy Hammond. 2008. Paleosketch: accurate primitive sketch recognition and beautification. In *Proceedings of the 13^{th} International Conference on Intelligent User Interfaces*. ACM, Gran Canaria, Spain, 1–10.
[47] Seth Polsley, Jaideep Ray, Trevor Nelligan, Michael Helms, Julie Linsey, and Tracy Hammond. 2016. Leveraging trends in student interaction to enhance the effectiveness of sketch-based educational software. In *Revolutionizing Education with Digital Ink*. Springer, Seattle, WA, 103–114.
[48] Dirk Proske. 2018. *Bridge collapse frequencies versus failure probabilities*. Springer, New York, NY.
[49] Robert J Roselli, Larry Howard, and Sean Brophy. 2006. A Computer-Based Free Body Diagram Assistant. *Computer Applications in Engineering Education* 14, 4 (2006), 281–290.
[50] Dean Rubine. 1991. Specifying Gestures by Example. In *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '91)*. ACM, New York, NY, USA, 329–337. https://doi.org/10.1145/122718.122753
[51] Matthew Runyon, Blake Williford, Julie Linsey, and Tracy Hammond. 2020. An Intelligent Interface for Automatic Grading of Sketched Free Body Diagrams. In *Proceedings of the 25th International Conference on Intelligent User Interfaces Companion*. ACM, Cagliari, Italy, 136–137.
[52] Johnny Saldaña. 2015. *The coding manual for qualitative researchers*. Sage Publications, Thousand Oaks, CA.
[53] Joel A Shapiro. 2005. An Algebra Subsystem for Diagnosing Students' Input in a Physics Tutoring System. *International Journal of Artificial Intelligence in Education* 15, 3 (2005), 205–228.
[54] AK Suresh. 2020. The 'Making'of the Mind of an Engineer—Some Thoughts on Engineering Education in India. In *The Mind of an Engineer: Volume 2*. Springer, New York, NY, 65–73.
[55] Renata Tesch. 1990. *Types of qualitative analysis*. RoutledgeFalmer, London, England, 77–102.
[56] Selin Üreten, Dieter Krause, Gregor Beckmann, et al. 2017. PLANTING THE SEEDS OF FUTURE MECHANICAL DESIGN ENGINEERS–LEARNING SKILLS. In *DS 88: Proceedings of the 19th International Conference on Engineering and Product Design Education (E&PDE17), Building Community: Design Education for a Sustainable Future*. The Design Society, Oslo, Norway, 698–703.
[57] Stephanie Valentine, Francisco Vides, George Lucchese, David Turner, Hong-Hoe (Ayden) Kim, Wenzhe Li, Julie Linsey, and Tracy Hammond. 2012. Mechanix: A Sketch-Based Tutoring System for Statics Courses. In *Proceedings of the Twenty-Fourth Innovative Applications of Artificial Intelligence Conference (IAAI)*. AAAI, Toronto, Canada, 2253–2260.
[58] Kurt Vanlehn, Collin Lynch, Kay Schulze, Joel A Shapiro, Robert Shelby, Linwood Taylor, Don Treacy, Anders Weinstein, and Mary Wintersgill. 2005. The Andes Physics Tutoring System: Lessons Learned. *International Journal of Artificial Intelligence in Education* 15, 3 (2005), 147–204.
[59] Ilse M Verstijnen, Cees van Leeuwen, G Goldschmidt, Ronald Hamel, and JM Hennessey. 1998. Sketching and creative discovery. *Design studies* 19, 4 (1998), 519–546.
[60] Vimal Kumar Viswanathan, Josh Taylor Hurt, Tracy Anne Hammond, Benjamin W Caldwell, Kimberly Grau Talley, and Julie S Linsey. 2020. Impact of a Sketch-Based Tutoring System at Multiple Universities. In *2020 ASEE Annual Conference*. ASEE, Maryland, USA, 12.
[61] Caroline Widjaja and Stefanus Satria Sumali. 2020. Short-Term Memory Comparison Of Students Of Faculty Of Medicine Pelita Harapan University Batch 2015 Between The Handwriting And Typing Method. *Medicinus* 7, 4 (2020), 108–111.
[62] Blake Williford. 2017. Sketchtivity: Improving creativity by learning sketching with an intelligent tutoring system. In *Proceedings of the 2017 ACM SIGCHI Conference on Creativity and Cognition*. ACM, Singapore, 477–483.
[63] Erin E Wilson and Sarah A Kennedy. 2017. Modern "homework" in general chemistry: an extensive review of the cognitive science principles, design, and impact of current online learning systems. In *Online Approaches to Chemical Education*. ACS Publications, Washington, D.C., 101–130.
[64] Aaron Wolin, Brian Eoff, and Tracy Hammond. 2008. ShortStraw: A Simple and Effective Corner Finder for Polylines. In *Proceedings of the Fifth Eurographics Conference on Sketch-Based Interfaces and Modeling (SBIM)*. Eurographics Association, Annecy, France, 33–40. ISBN: 978-3-905674-07-1.
[65] Fu You Xu, Ming Jie Zhang, Lei Wang, and Jian Ren Zhang. 2016. Recent highway bridge collapses in China: review and discussion. *Journal of Performance of Constructed Facilities* 30, 5 (2016), 04016030.
[66] Xuhong Zhou, Jin Di, and Xi Tu. 2019. Investigation of collapse of Florida International University (FIU) pedestrian bridge. *Engineering Structures* 200 (2019), 109733.
[67] Susan M Zvacek. 2016. Building Creativity into Engineering Education: Practical Strategies. *The International Journal of Creativity and Problem Solving* 26, 2 (2016), 141–156.