STRAT: A Sketched-Truss Recognition and Analysis Tool

Joshua M. Peschel
Hydrology Research Laboratory
Zachry Department of Civil Engineering
Texas A&M University
College Station, Texas 77843-3136
peschel@tamu.edu

Tracy A. Hammond
Sketch Recognition Laboratory
Department of Computer Science
Texas A&M University
College Station, Texas 77843-3112
hammond@cs.tamu.edu

Abstract

The statically-determinate, pin-connected truss is a basic structural element used by engineers to create larger and more complex systems. Truss analysis and design are topics that virtually all students who study engineering mechanics are required to master, many of whom may experience difficulty with initial understanding. The mathematics used to analyze truss systems typically requires lengthy hand calculations or the assistance of proprietary computer-aided design (CAD) programs. To expedite work in this domain, we propose: STRAT (Sketched-Truss Recognition and Analysis Tool), a freehand sketch recognition system for solving truss problems. The STRAT system allows users to rapidly determine all of the unknown forces in a truss, using only a hand-drawn sketch of the truss itself. The focus of this article covers the design methodology and implementation of the STRAT system. Results from a preliminary user study are also presented.

Introduction

Structural engineers rely upon simplified mathematical descriptions of physical systems for analysis and design. Whether it is for an underground tunnel, a high-rise office building, or the wing of an airplane, engineers utilize these simplified models to work with larger and more complex systems. One basic structural element used by engineers is the statically-determinate, pin-connected truss (Figure 1). Formally, a pin-connected truss can be defined as a rigid structure that consists of slender members interconnected by joints. The practical use of the truss is to create a support system for distributed forces over relatively large distances. Examples of distributed forces carried by trusses may include the weight of a roof or the pressure underneath an airplane wing. The internal configuration of a truss is that of well-defined triangular subunits, which provide strength and stability. The employment of truss systems in design necessitates an understanding of engineering mechanics, as well as the additional use of computer-aided design (CAD) programs that allow engineers to evaluate multiple solution scenarios.

Some common CAD-based programs used for truss analysis include: Visual Analysis [1], Truss 2000 [2], and RISA-2D [3]. Each of these CAD programs, however, utilizes a standard drag-and-drop graphical user interface. To create a truss representation in these types of CAD programs, the user typically selects pre-defined truss elements, such as a joint or member, from a tool palate and then places each item on a pre-defined grid in the desired configuration. Member and joint connections are often made on-the-fly, along with force placements; properties such as member length and force magnitude are usually associated with the truss elements once the system has been manually assembled. After the assembly process is complete, a mathematical analysis for all of the unknown forces can then be performed.

In this article we propose: STRAT (Sketched-Truss Recognition and Analysis Tool), a freehand sketch recognition system for working with truss problems. Sketch recognition may be formally defined as the automated identification of shapes and objects that are drawn by freehand sketching alone. Sketch recognition is uniquely different from other drawing applications, including CAD, because it does not rely on predefined objects to create text or drawings – only freehand strokes are used as inputs. The recognition process usually
consists of multiple levels, ranging from low-level recognition that may include simple lines and polygons, up through higher-level recognition usually involving more complex elements such as engineering schematics. Research in the area of sketch-based truss analysis can be traced back to Ivan Sutherland. In his original 1963 work, Sutherland used the drawing and computational capabilities of Sketchpad to determine optimal geometric solutions for truss member lengths [4]. But this work lay mostly dormant for decades until later research efforts began to address freehand sketch recognition. In the general context of engineering mechanics, Stahovich developed an application to use freehand sketching for interacting with certain related classes of mechanical design problems [5]. LaViola and Zeleznik created a sketch-based interactive solver capable of understanding mathematical symbols and basic mechanics problems [6]. However, in both research efforts, neither system specifically addresses recognizing and solving truss problems. Hutchinson et al. used an existing freehand sketch recognition interface for finite element model (FEM) assembly [8]. Truss problems were addressed in this work but the FEM approach is an advanced technique involving additional layers of computational complexity not needed for standard truss analysis.

The primary rationale for the proposed STRAT application is that virtually all students of engineering mechanics spend a significant amount of time learning to solve truss systems ‘by hand’. Therefore, freehand sketching of a truss system is a common skill learned by all of these students, well before CAD software packages are introduced to them. Learners could use the STRAT application to quickly verify answers to hand-worked problems; likewise, they could use the application to evaluate incremental solution steps. The mathematics used to analyze trusses often requires lengthy hand calculations and, working with pre-defined tool palate objects may involve a steep learning curve – much more so than freehand sketching alone. Because truss elements, e.g. members, joints, pins, etc. have reasonably well-standardized hand drawn forms in both academic and professional practice, the expected variability of freehand sketching would be quite low; thus, a simple yet reliable sketch recognition system could be developed to expedite and improve work in this domain.

The focus of this research article covers the application methodology of the proposed STRAT system, which includes primarily, recognition and analysis capabilities. Results and discussion from a preliminary user study are presented, along with the conclusions and future directions for this work.

Application Methodology

A framework for both recognizing and solving a robust set of truss problems, whether CAD or programmatically-based, necessarily involves several layers of information processing. When only freehand sketching is used as an input, additional layers must be considered. The STRAT application consists of two main information processing layers, with three sub-layers in each primary layer (Figure 2). The following sections outline the specific processing layer details.

**Sketched-Truss Recognition**

The initial information processing layer for STRAT is the sketched-truss recognition layer, representing the human-computer interface. Within this processing layer, user data are transformed from basic pen strokes into a recognized truss schematic. There are three sub-layers to the sketched-truss recognition layer: (1) stroke parsing, (2) low-level recognition, and (3) recognized associations.

**Stroke Parsing.** When a user draws, they will either form simple or complex strokes. Simple strokes in STRAT are defined as either single lines or circles – they are singular strokes that can be recognized initially as geometric primitives. Complex strokes, as defined in STRAT, are all other non-simple polylines segments. An example of a complex stroke would be a triangle or an arrow (Figure 3a). When a stroke in STRAT is recognized as complex, a corner-finding algorithm is employed to locate parsing points [8]. The parsing is done to address the stroke and drawing order variability that often occurs among users. Once the parsing sequence is complete, for a given complex stroke, each of its component sub-strokes will have been recognized as either a single line or circle. When all complex strokes have been simplified, low-level recognition can take place.

**Low-Level Recognition.** The low-level recognizer used in STRAT is a geometric rules-based algorithm. Simple
strokes (lines or circles) are now put into context to form other geometric primitives, such as triangles or arrows (Figure 3b). This approach to recognition is taken for two reasons. First, by simplifying each stroke prior to recognition, the variability that may exist in comparison to correctly indentifying more complex gestures directly (such as an arrow) is likely reduced. Second, with a composite sequence of recognized strokes, the user is given the flexibility to remove one or more possibly unwanted segments without deleting the entire original segment. After all of the existing strokes are recognized, they move to the recognized association layer for assembly into a truss schematic.

**Recognized Association.** Geometric primitives that have been recognized possess context within a truss system. To give the appropriate context to recognized primitives, we again employ a rules-based algorithm where object types are assigned. For example, circles represent joints, arrows represent forces, etc. More complexity arises when reaction supports (small triangles) exist because larger triangles are also formed by the structural members (Figure 3c). To mitigate confusion between these two object types, our algorithm compares the relative sizes and spatial orientations using common-sense reasoning (i.e. we would not expect to find a reaction support in the center of the truss). A final step taken for recognized association is an examination of determinacy. To be statically-determinate, a two-dimensional truss system must contain zero degrees-of-freedom (DOF). The actual DOF can be numerically determined using the following expression:

\[ DOF = 3(n - 1) - 2j \]  \hspace{1cm} (1)

where \( n \) is the number of structural members and \( j \) is the number of joints. A non-zero result for the calculation would indicate that the truss system was statically-indeterminate. At present, the STRAT application can only evaluate statically-determinate truss systems. Once a truss schematic is correctly assembled, it moves to the next information processing layer.

**Sketched-Truss Analysis**

The second information processing layer for STRAT is the sketched-truss analysis layer. Here, recognized objects from the truss schematic are given numerical meaning and the unknown values desired are solved for. There are three sub-layers to the sketched-truss analysis layer: (1) element valuing, (2) assembly of equilibrium equations, and (3) solution of equilibrium equations.

**Element Valuing.** When truss systems are analyzed, engineers are interested in the forces that are carried by the structural members in response to applied external forces. These internal forces result in members being in either tension or compression, and they change in a predictable manner with changing external forces, as well as changing geometric configurations. To capture these configurations, numeric values must be assigned to the recognized truss elements. This is accomplished by assigning lengths to members and magnitudes to external force arrows. In the STRAT application, users may enter these values directly onto the truss schematic (Figure 4a). When this step is complete, equation assembly can begin.

**Equation Assembly.** All truss systems can be solved using an assemblage of static equilibrium equations. To
determine the unknown forces in the members, a system of linear equations is developed. In the STRAT application, the ‘method of joints’ is used to create this system of equations. The method of joints technique simply examines each truss joint, and creates two separate static equilibrium equations that are the summation of x- and y-direction forces, respectively (Figure 4b). The standard positive x- and y-direction orientation for forces is assumed. Forces on angles are decomposed into x- and y-components using their relationship to the assumed positive x-axis; decomposed forces are recombined after the solution process. Unknown reaction forces are always assumed to act into the joints and unknown member forces are always assumed to act away from the joints. Positive or negative numeric values obtained from solving the system of equations determines the final force orientation. Once all of the static equilibrium equations are determined for each joint, they are assembled into an array for solving.

Equation Solving. When the system of linear equations is assembled for solving, it takes the form $Ax = b$, where A is the numeric array of truss element coefficient values, x is the vector of force identities, and b is the vector populated with external force values, if any are present at the individual joint (Figure 4c). To solve this expression, we simply employ an algorithm that inverts the array A and multiplies it by b (i.e. $x = A^{-1}b$). Determined values are then associated with all unknown forces. Sign also plays an important role in the solution process. A positive numeric value for an unknown reaction force indicates that its orientation is into the joint; a negative value would indicate that the reaction force acts away from the joint. Likewise, a positive member force implies that the member is in tension, while a negative value means that the truss member is in compression. All of the unknown forces solved for are written to a table of values displayed next to the truss schematic.

**User Study Results**

To evaluate the recognition and analysis capabilities of the STRAT application, a preliminary user study was performed. Ten users consisting of structural engineering faculty, staff, and students were asked to complete two separate sketch recognition application tasks on a Tablet PC (Figure 5). The user study initially began with a warm-up for each participant that lasted approximately two minutes. This was to familiarize each user study participant with the Tablet PC; brief explanations of the sketch recognition tasks were given and users were allowed to experiment with drawing basic shapes before the study began. The first part of the user study was to draw ten geometric primitives typically found in truss systems (e.g. lines, triangles, circles, etc.) (Figure 5a). In this task, users were instructed to reproduce the geometric primitive they were shown on the screen. Each user submitted three samples for each geometric primitive; therefore, a total of 300 samples were taken for evaluation. The second part of the study involved drawing complete truss systems (Figure 5b). Two separate trusses were shown on-screen and each user was asked to submit two samples for each truss, resulting in a total of 40 samples being taken for evaluation. For the second part of the study, users were instructed to reproduce the shown truss in the manner that they would normally draw it. This was done to capture the variability among users who may draw certain aspects of the truss system with subtle differences, such as reaction supports...
and force arrows. The following section discusses the results obtained from the preliminary user study.

Discussion

The first part of the preliminary user study resulted in a 100-percent recognition rate for all of the geometric primitive samples taken. This meant that all of the sample gestures were parsed correctly, then recognized, and reassembled into the correct recognized associations. This outcome was somewhat expected because stroke inputs were generally simple and clean, without much complex stroke parsing involved. This part of the user study was done to evaluate the low-level recognition capabilities for the STRAT application. Results from the second part of the user study were considered on an all-or-none basis, meaning that either every element of the truss system was correctly identified, or the truss system as a whole was not recognized. An overall accuracy of 70-percent was obtained from the samples in the second part of the user study. This lower level of accuracy can likely be attributed to one of the following three causes, following the examination of the original input data (Figure 6). First, some users when instructed to draw the truss system how they would normally do so provided reaction supports in the form of circles (Figure 6a). The consequence to this action, along with the overlapping location of the joints resulted in not all reaction supports being recognized for these truss sketches. Therefore, the truss system was not fully recognized. The second issue observed from the sample data was excessive overtracing and scribbling (Figure 6b). In the context of drawing a truss, scribbling to darken in the joint locations is not an uncommon occurrence. However, in the context of most sketch recognition applications, scribbling over a gesture means that the gesture should be deleted. Thus, when users performed this action, the recognized joints would be erased. The final occurrence observed that is believed to have contributed to lower accuracy was extended strokes (Figure 6c). An example of this would be when a user sketched the collective bottom chords of the truss in one stroke. The stroke itself was recognized as a simple line and therefore not parsed into additional members. Although, other strokes were connected to it at interior locations and even if the user indicated that joints were at these locations, the long stroke was not segmented.

Evaluation of the second information processing layer, the sketched-truss analysis, was not included in the original collection of user study data. We have tested the solver by manually adding data to the recognized associations and performance has been at 100-percent. There are obvious consequences to not fully correctly recognizing the truss system; chief among these is an inability to continue to the solver stage or the returning of an incorrect result. Therefore, we have focused primarily to date on the sketched-truss recognition portion of the STRAT application.

Future Work

There are several directions for the continuance of this work. Most pressing among these directions is improving the overall low-level recognition (and recognized association) accuracy of the STRAT application. We see this going in one of two possible ways. First, we add and evaluate additional rules to the low-level and recognized association layers. The only issue in doing so would be overcoming existing sketch recognition dogma, such as using scribbles to delete strokes. The alternative is to employ an additional object-based interface, such as
LADDER, for the low-level recognition and recognized associations [9]. LADDER does not provide the same degree of stroke segmentation as STRAT, but that could be considered a trade-off in favor of improved accuracy. Also, the solver component has worked well manually but an additional user study that takes into account element valuing also needs to be performed, as well as an overall usability study. Additional areas of advancement for the STRAT application include extending the recognition and analysis to statically-indeterminate trusses. Statically-determinate trusses only rely upon equilibrium equations for their solution, while statically-indeterminate truss systems require additional constitutive equations, such as those based on material properties.

Conclusions

This paper presented the application methodology and implementation of STRAT, a sketched-truss recognition and analysis tool. A preliminary user study was conducted to evaluate the recognition component of the application. Basic geometric primitives were all perfectly recognized when individually input; complete truss systems were recognized at an all-or-none accuracy of 70-percent. Work on the STRAT application is being continued and techniques to improve the overall accuracy are being investigated.

Acknowledgments

Joshua Peschel would like to recognize the Zachry Department of Civil Engineering for its financial support through the Tommy E. Knight ’61 Endowed Fellowship. This work was funded by NSF IIS grant 0744150: Developing Perception-based Geometric Primitive-shape and Constraint Recognizers to Empower Instructors to Build Sketch Systems in the Classroom.

References