Visual Structure Editing of Math Formulas

by

Gavin Nishizawa

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science
in Computer Science

B. Thomas Golisano College of Computing and Information Sciences
Rochester Institute of Technology

August 2020

ROCHESTER INSTITUTE OF TECHNOLOGY
ROCHESTER, NEW YORK

CERTIFICATE OF APPROVAL

MS DEGREE THESIS

The MS degree thesis of Gavin Nishizawa
has been examined and approved by the
thesis committee as satisfactory for the
thesis required for the
MS degree in Computer Science

Dr. Richard Zanibbi, Advisor

Dr. Kenny Davila Castellanos, Advisor

Dr. Zack Butler, Reader

Dr. Matthew Fluet, Observer

Date

Visual Structure Editing of Math Formulas

by

Gavin Nishizawa

Submitted to the
B. Thomas Golisano College of Computing and Information Sciences
Department of Computer Science
in partial fulfillment of the requirements for the
Master of Science Degree
at the Rochester Institute of Technology

Abstract

Math formulas can be large and complex resulting in correspondingly large and complex LaTeX math strings for expressing them. We design operations to visually edit the typeset LaTeX formulas. The operations are invoked via the formula's control points, which are created as a way to specify an operation associated with the point's location relative to a symbol in the formula. At the control points, formulas can be extended in multiple ways, LaTeX can be inserted locally by typing, an existing formula can be inserted, or part of the formula itself can be moved to that point. Parts of formulas can be selected by clicking on a symbol or dragging a rectangle over an area in the formula, and the subtree for the selection can be replaced, deleted, moved to another point in the formula, or lifted out of the formula into a chip floating above the canvas. Formula chips can be used as arguments to operations, including a set of existing formulas provided in a symbol palette. Operations can be performed either by making a selection, selecting a control point operation, and then specifying an argument, or by dragging an argument to one of the control points in the formula. We perform an online formula editing experiment to examine if these visual editing operations can be used to reduce the time and actions spent in order to make edits to formulas. With 35 participants completing 18 formula editing tasks split between 3 input conditions of LaTeX only, Visual only, or LaTeX and Visual, we find that on average participants spend the least amount of time on the editing tasks when both editing capabilities are available.

Acknowledgments

There are so many people who have supported me, and I would not have been able to come close to completing this without all of your support. To everyone who has supported me, Thank you.

I would like to extend a few special thanks and first, I would like to thank my advisors Dr. Richard Zanibbi and Dr. Kenny Davila Castellanos for the incredible amount of support and guidance they have provided me along this journey, for teaching me new things, and for helping me move past the obstacles I encountered. I would also like to say that I am very grateful for the opportunity to work in the Document and Pattern Recognition Lab.

Thank you Dr. Zack Butler and Dr. Matthew Fluet for the helpful feedback on my thesis, providing me with new things to think about, and for serving on my thesis committee.

I would like to thank the DPRL members for working with me on MathDeck and the fun and interesting discussions. I would also like to thank Yancarlos, Behrooz, Mahshad, Abishai, and Alex in particular for testing out the interface and experiment protocol and providing me with helpful feedback on improving it.

I would like to thank my advisor Christina Rohr for the helpful conversations and in helping me plan my graduate studies with my thesis. Thank you Dr. Marti Hearst for reviewing the usability of the MathDeck interface and sharing ways to improve it. Thank you Dr. Agarwal for providing helpful feedback on MathDeck and sharing formulas to consider for the interface.

I would like to thank the National Science Foundation and the Alfred P. Sloan Foundation for supporting my work. I would like to thank the Golisano College of Computing and Information Sciences and College of Science faculty and staff for supporting me and helping me find participants for my experiment.

Thank you Ginda for supporting me throughout this journey.

Finally I would like to thank my mom, my dad, and my sister for always being there for me and supporting me.

This is dedicated to making it easier to edit math formulas on computers.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Mathematical formulas can be large, be of a complex structure, and use a wide variety of symbols and notation. Using LaTeX math is a standard way to enter typeset formulas. This is a good way to be able to enter a formula by typing, but there is sometimes a non-trivial difference between the LaTeX string entered and the typeset formula visualized. If someone wishes to make changes to an existing LaTeX formula, one may need to locate the position in the LaTeX string which corresponds to the location in the visual formula where the change is needed. If one has the LaTeX as their mental model of the formula which they wish to change, this is more straightforward and the visualization of the formula is not as relevant at that time. However, if one has the visual formula as their mental model on which they wish to make a change, they must translate that change from one on the visual formula to one on the LaTeX string which they are editing. We wish to provide a way to perform these changes without this cost by removing the need to translate this change into a LaTeX string modification and replacing it with a translation to changing the formula with a physical metaphor. To do so we design operations on a formula's visual structure tree so that certain editing operations can be performed more efficiently. We develop a prototype implementation of the operations we design, and perform an online experiment with human participants to attempt to evaluate our hypothesis.

Our visual formula editing operations include being able to move around a symbol or a set of symbols within the visual representation of a formula. The primary mechanism for a user of the system would be to use a mouse pointer or other touch device to select a symbol or a set of symbols in the formula and move them around the formula. Keyboard shortcuts for performing these operations/edits is considered a possible extension of the operations in the future.

The research question is related to whether or not these imagined editing operations can help people perform formula editing operations more easily. That may be by requiring less time, or simply a user preferring to edit formulas this way. The hypothesis is that these operations will be a preferred method to perform certain types of editing operations such as editing a specific part of a formula that may be hard to locate or identify in a LaTeX based string representation of the formula.

Part of the inspiration for the imagined editing operations were from attempting to make corrections to formulas entered through the use of a formula recognition-based system where users can enter formulas via handwriting or images [23]. One issue with this method of formula input is that sometimes the recognition results are not what was intended by the user, although in our experience when they are not exactly the intended formula, they are often somewhat close with only a few errors that need fixing. Usually there are a small set of symbols which are different or in different positions in the formula than intended. In the system the only methods to reach the desired "target" formula are to try to edit the result's LaTeX string or to try to re-enter or adjust the input slightly to see if the recognition system can produce the intended result.

For the LaTeX editing option it can be difficult at times (especially in large formulas) to locate where in the string you need to make changes. Additionally, this may require that the user has knowledge of LaTeX. In our experience, it is easier to identify where there is something wrong in the visual representation of the formula than it is in the corresponding LaTeX string. In this system the formula is displayed in a rendered form visually next to the LaTeX string.

For the re-entry method there is the issue of uncertainty regarding whether or not the system will be able to produce the correct formula from your handwriting input. It is possible that even with entering it again, the system may not produce the correct result or even a closer result.

For this reason combined with the fact that there is a time and effort cost associated with re-entry, a user may decide against trying to enter it again and opt for the more certain method of editing the LaTeX string. The idea of editing the formulas using their visual representation where it may sometimes be easier to locate specific parts of the formula compared to finding the relevant position in the LaTeX string was partially motivated by this and the fact that not everyone is knowledgeable in LaTeX.

We find that on average participants spend more time during formula editing tasks when they only have the Visual operations for editing compared to LaTeX, but less time when they have both available. With known issues in our implementation of the visual operations, we think that an improved version of the visual editing operations could significantly help in reducing the time for editing formulas in some cases.

Chapter 2

Background

To aid in solving the issues involved with using math online, there are several approaches that have appeared including template based editors, LaTeX's math mode strings, and formula recognition techniques for handwriting and images. MathML emerged as an XML format for clearly expressing either the presentation (visual appearance) or semantics (meaning) of math, but even if you were familiar with the tag names, it would still be very tedious to manually type an expression in MathML due to its verbosity. In comparison, LaTeX's math mode strings are much shorter and would be relatively easier to type out once you were familiar with LaTeX.

Math search engines have appeared many of which accept queries with math-content provided as LaTeX-strings or entered using a template editor [34]. Several commercial applications have systems for recognizing math expressions in handwriting such as the MyScript[1] systems and the Microsoft Equation Editor. We found a lot of helpful information to build a LaTeX formula editing interface in a search context from multiple sources [2, 8, 9, 13, 18, 19, 28].

2.1 Formula Search for Autocomplete

Zhong [33] discusses how autocompletion functionality can aid in query formulation for math if it makes use of the mathematical information rather than just text-based suggestions. Autocompletion functionality can be provided by

[1] https://www.myscript.com/

finding a set of formulas similar to a query formula from a collection of popular or common formulas. This similarity can be based either on the formula similarity, text encoding similarity, or visual similarity. The edit distances in formula space, text space, and visual space can be used to describe the corresponding notion of similarity between points in the space.

Even if editing operations had no cost to perform, there is still the mental cost of planning the operations. The well studied relationship between recognition and recall [1, 20] provides evidence that presenting people with suggestions which they can recognize, is easier for them than asking them to recall their target formula. Zha et al. [32] use this to provide more helpful image based queries and suggestions. The idea that recognizing content is easier than recalling it relates to the relationship between browsing and querying in online interfaces [10].

Visual search can be used successfully especially in contexts where more exact methods of searching are difficult. Davila et al. [4,5] present the Tangent-V system for visually searching math formula images. This enables searching of formulas based on the relative positions of symbols without the constraints of a specific structure. In MathDeck, symbols can be freely moved on the canvas and be recognized based on their placement, but they can not yet be searched this way. Eitz et al. [7] present a method for visually searching a 3D model collection using handwritten sketches using a bag of features approach. This is an example of how visual search can be an effective method for finding an object in a collection where it is difficult to specify a query of the object type. Visual search is another good approach for easier searching of math formulas. Where we attempt to make entering structured formula queries easier, visual search systems such as Tangent-V can make this easier by also working on unstructured or semi-structured data which may be easier to enter.

2.2 Math Formula Recognition

One way of supporting math input is through the use of math formula recognition in handwriting or images. Mahdavi, et al. [21] present their LPGA recognition model which will be used in the evaluation of math formula recognition. Blostein et al. [3] discuss the need for effective interfaces for users to correct errors in recognition systems. One of the goals of our system is to provide usable methods for creating and editing formula structures produced from recognition systems. Zanibbi and Blostein [30] cover math recognition and mention how math notation is a “graphical language for representing complex interactions between primitive objects”. This is a nice description related to how the proposed editing operations attempt to capture some of that language (roughly the portion that LaTeX captures) while making it easier for people to ‘speak’ that portion of the language.

2.3 Visual Editing

Many approaches have appeared aiming to expand people’s capabilities while avoiding learning language syntax where it may not be necessary.

Kölling et al. [14,15] discuss block-based programming environments which reduces the need to learn the language syntax or deal with syntax related errors. Our work shares the goal of supporting people in working with complex structures without forcing them to learn a special language syntax (LaTeX math in our case). Their Stride editor also uses both visual editing for high-level operations and text editing for low-level ones.

Russell-Rose et al. [24] introduce a system for visually inputting boolean queries. This interface effectively uses visual editing and construction to make it easier for people to create and interpret boolean structures which has is an equivalent goal that we have for math formulas. Both are also similar in how the visual input of structured information is designed for use in a search context.

Khuong et al. [27] present a user interface for inputting handwritten math which becomes recognized and allows for gesture operations for correcting segmentation errors, a symbol list for correcting symbol recognition errors, and region boxes for modifying the structure of a formula. This interface and our interface are related in providing non-recognition based ways for users to edit formulas, and the region boxes can perform some of the same functions of the proposed visual editing operations for editing formula structure. They also conducted a user experiment with 20 users to evaluate their interface.

Zanibbi et al. [31] introduce a method to visually aid users with style preserving morphs when the user’s handwritten input is recognized and repositioned. This demonstrates how users might check their recognized formulas visually and can have a better experience with helpful visual feedback.

Smithies et al. [25] introduce a handwriting-based equation editor, with simple operations to correct errors in the recognition. The correction operations described include methods for splitting or combining handwritten strokes which were erroneously split or grouped. This is similar to how the visual structure editing operations allow the user to correct incorrect recognized structures. They use multiple modes for different types of operations.

Kimura [12] describes how using an abstract document model can be combined with commands to structurally edit the document. Our work also has a model for symbols to be connected to each other which allows for operations to be defined and used for creating and editing formula structures. They mention how by using a directed acyclic graph (DAG) it is possible to 'share' an object between multiple parents which can be helpful for tables or matrices where the row and column both could be considered the parent to an individual cell. Additionally, the structure editing is separate from the editing of the node contents in the structure. They also have the concept of ghost windows which are empty but locations where new content can be added which is similar in how the connection points represent empty relations where new connections can be made.

Math boxes [11, 26] is a user interface for editing math expressions. They found their math boxes method is preferred over another 'Offset' system for complex expressions.

The MathBrush [16, 17] system is designed for working with math on pen-based devices. They approach formula entry in three steps: draw, correct symbols, check structure. However, it appears the only way to correct structure is through their system for structural analysis, which may misinterpret the intended structure.

The MST system [22] is designed for teaching algorithmic problem solving with manipulations on uninterpreted formulas. It uses gestures to apply algebraic rules on expressions. This is similar in how it is editing structures, but is working with structures that are more than just visual and requires information about symbol types, operators, and rules.

2.4 Summary

We attempt to examine possible solutions to make it easier to edit LaTeX math formulas. We based our prototype of our operations for our experiment on the

MathDeck[2] system which has developed from the MathSeer interface [23]. In the interface formula chips are used to present formulas as entities with operations that can be performed on them. Additionally, we use a set of common symbols and formulas collected together in categories as a 'symbol palette', which is a feature that many math entry systems share. In our system, the symbol palette contains a set of existing formula chips as well as those the user creates. When a formula chip is created, it is added to a 'My Formulas' category in the symbol palette where it can be reused as a building block in other formulas. In the experiment system we discuss in Chapter 4, there is only one formula being edited at a time, but formula chips can still be lifted out of a formula, used from the symbol palette, copied, deleted, and inserted elsewhere in the formula. In MathDeck, formula chips can also be edited, downloaded as an image, added to the canvas for editing, and searched for online (directly or as part of a query). We further discuss the proposed operations, how they work in the interface, as well as how they are evaluated in the following chapters.

[2] https://mathdeck.cs.rit.edu

Chapter 3

Visual Editing Operations

Our visual editing operations are based on a tree model of a formula which is based on the symbol layout tree (SLT) model. This tree is partially structured on the spatial layout of symbols of in a formula, an example of which can be found in Figure 3.1.

3.1 Design of Operation Behavior

We would like to examine operations for visually editing the visual structures of math formulas.

Suppose you have a formula: $y = (a + b - c + d)^{(a^2+b^2-c^2+d^2)}$, and you would like to change it to become $y = (a + b - c + d)(a^2 + b^2 - c^2 + d^2)$ or $y = (a+b(a^2+b^2-c^2+d^2)-c+d)$. Natural responses might be similar to "move the expression in the exponent down" for the first scenario, and "move the expression in the exponent to the right of the b" in the second scenario. These are the types of editing operations that we would like to provide and see if they are as natural, intuitive, and helpful as we hope. A person can understand the operation without needing to know what the formula means, what the symbols are, or much else other than being able to recognize the similar(same other than size) group of symbols at one position in the first formula has moved to another position in the second formula. A person performing the operation just needs to know how to select an expression and how to move the expression somewhere else.

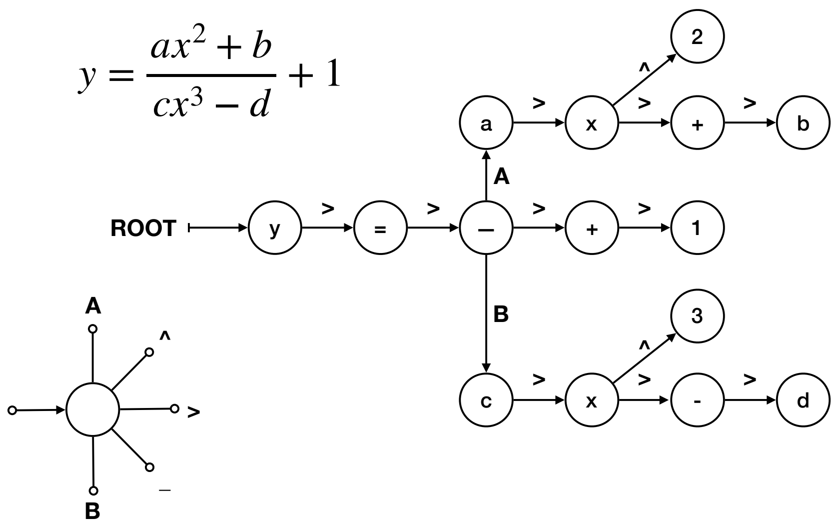


Figure 3.1: An illustration of the tree corresponding to the visual structure of a formula and the corresponding spatial relations.

Use cases include a user correcting the output from a recognition system, editing an existing formula, or combining two expressions with a relation.

Motivations for visual editing over text-based editing: Many people are already familiar with visual metaphors for moving things around. Many operating systems include a desktop environment where files and folders are displayed and can be interacted with using visual metaphors. The same is true for many mobile devices with app icons and folders. People who wish to explore math concepts may not want to learn how to write formulas using LaTeX, just to be able to use them.

3.2 Design of Specifying and Invoking an Operation

Visual Indication of Possible Operations

When a person using the system would like to perform one of these operations, how should the system communicate to them their options, and how should the user communicate to the system their desired operation? Let us consider a simple example where we have an existing formula xy and a new expression z which the user would like to connect to the existing formula in some way.

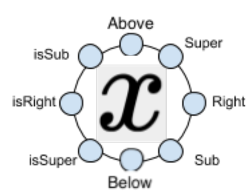


Figure 3.2: Example of possible relations available on a single unit.

Each symbol can have multiple types of relations defined, and those can be visually communicated to a user with small circles in locations associated to the spatial position for each relation as they are in Figure 3.2. However, adding the z expression to the xy expression can result in many outcomes, only some of which are shown in Figure 3.3.

If we display a circle for every possible relation in the expression, this will quickly become visually cluttered due to the 7 (in our implementation) relations for each of the S symbols. One solution to this is to automatically filter the relation candidates shown as the user moves the z expression towards their target, by only displaying the closest relation circle an example of which is shown in the left image in Figure 3.4. If only displaying one closest option is too restrictive it may be more helpful to find either a count k of the k-closest options to display, or a distance d where only options within a radius of d from the user's cursor are displayed. With any of these options, as the user moves their expression towards the location where they wish to insert it, they will be able to see their options for locations to insert.

A user can select to perform the insert operation by 'dropping' the new expression into the desired relation circle in the existing expression. The whole process (visualized in Fig. 3.3) involves the user selecting their expression

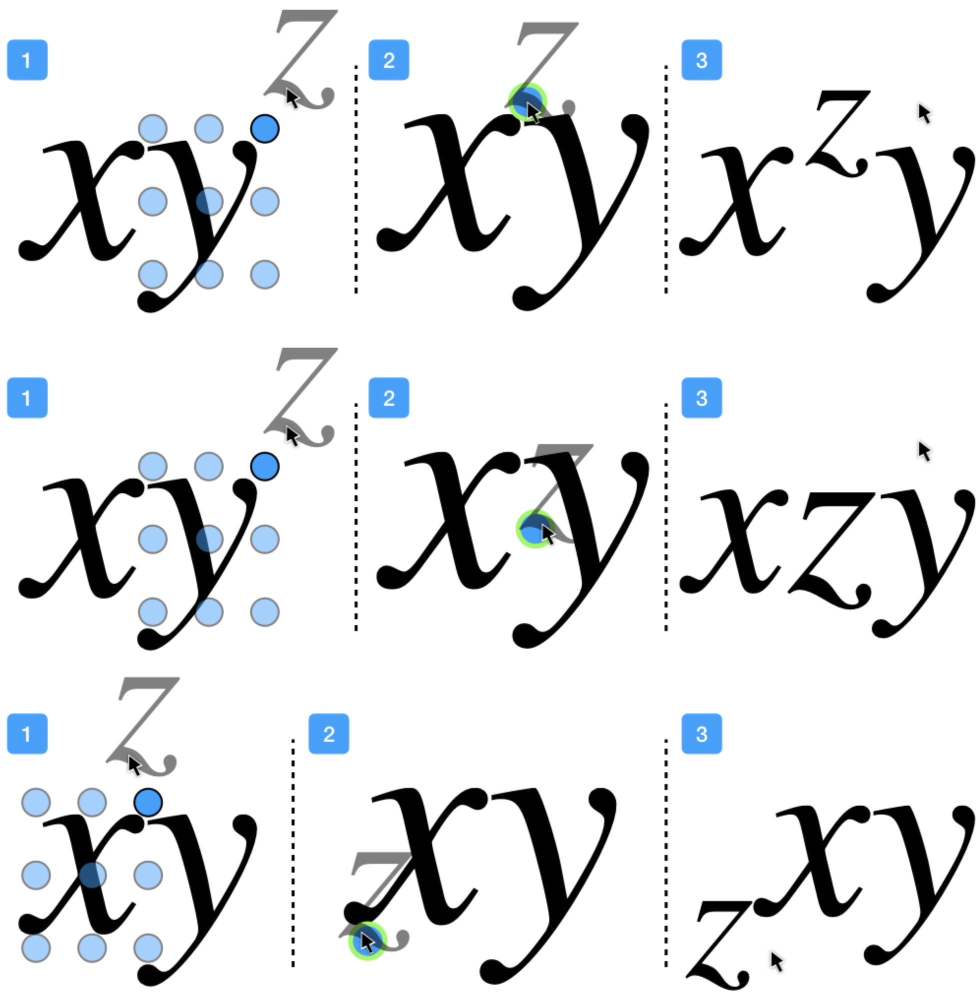


Figure 3.3: Examples of insertion of z into xy.

to insert (e.g. by clicking and dragging the z), the filtered relation circles being updated and shown dynamically as the user drags the z to their desired position, and finally the user releasing their mouse button over the circle for the relation. The operation is performed as previously described, and the resulting expression is displayed to the user.

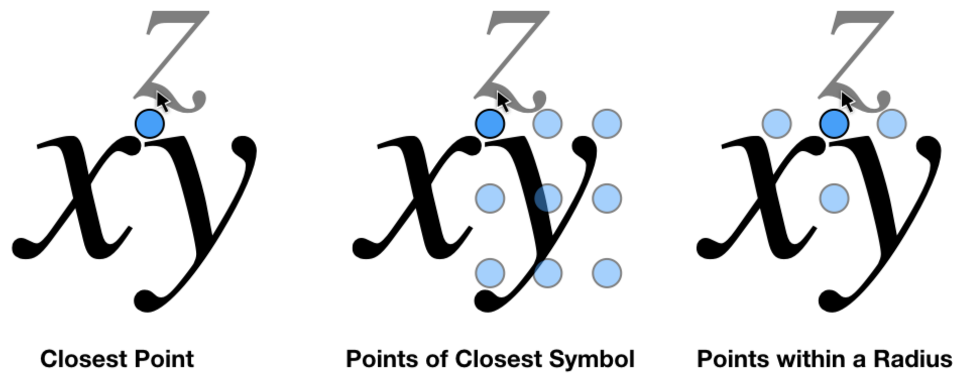


Figure 3.4: Insert operation relation circles with examples of different display options shown while moving an expression.

Another option for improving the user experience is to provide them with a preview of what the operation will result in when hovering over a relation circle but before releasing the mouse button. This would involve computing the result of the operation and either displaying the result temporarily in-place (with a visual indication it is a preview such as transparency), or in another way (e.g. a preview tool-tip appearing).

Additionally a relation circle in the center of each symbol can exist, associated with the operation of replacing the symbol with the inserted expression.

Matrices and other exceptions

It may be useful to distinguish between a set of standard relations and other relations for special types of entities such as matrices. For matrices, users may wish to insert an expression within an empty cell, into a specific relation in an expression in a non-empty cell, or into a cell on a new row or column. With all the same relation circles for expressions previously mentioned, new relation rectangles can be added for operations involving new rows or columns in a matrix as displayed in the right side of Figure 3.5. These would also be filtered to only show the option(s) most likely given the current cursor position, an example of which is the left side of Figure 3.5.

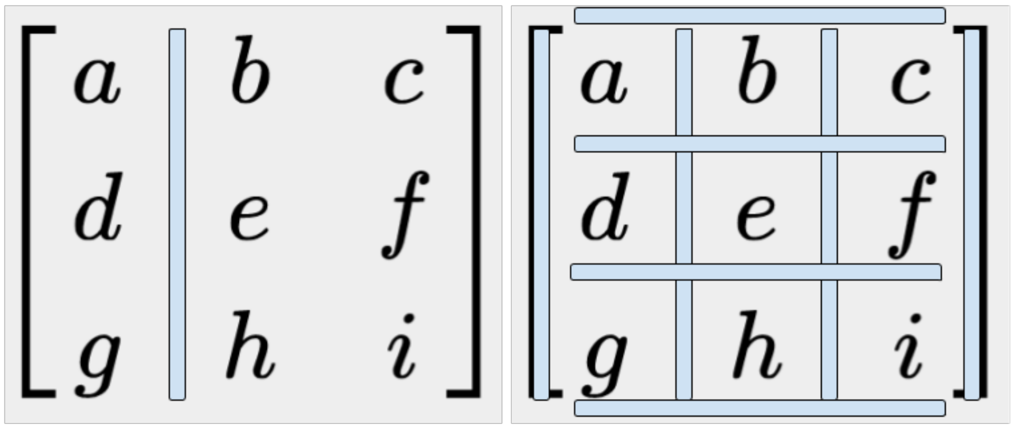


Figure 3.5: We initially designed custom controls for matrix editing operations to introduce new rows or columns (Filtered: left and All: right). However, we did not end up implementing these due to time constraints.

We decided to try to save time in our implementation by making use of the excellent MathJax[1] renderer which we were already using to display the formulas. This saved a lot of time but does have the drawback of not having as much control and because we are using it for something it wasn't designed for, this introduces issues where the original LaTeX string is lost in the process, and such as \alpha being converted into a Unicode α character.

3.3 Visual Operations

These operations support replacing or deleting a selection in a formula, and inserting at or moving a symbol or subtree within a formula to any of the control points.

3.4 Tree Replacement

The tree replacement operation is based on the formula context, the subtree to be replaced, and the replacement tree. We wish to specify our use of the

[1] https://www.mathjax.org/

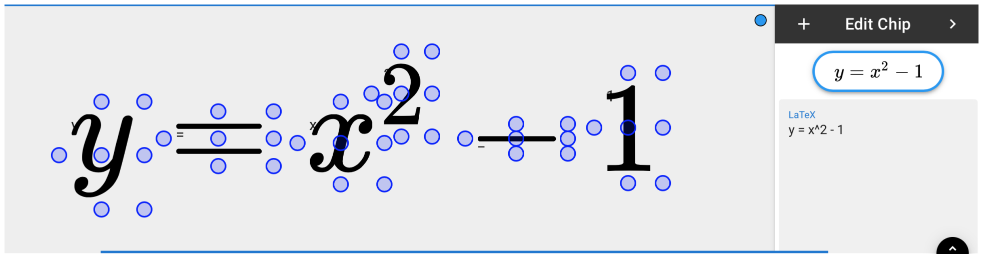


Figure 3.6: An early version of the model showing all of the control points of a formula for visualization purposes. When in use only control points for a single symbol or set of symbols are shown at one time.

word subtree to not be limited to a node in a tree extending to all of its descendant leaf nodes, which may be the more common concept associated to this word. Often in a formula's visual structure tree, we wish to select a subset of connected symbols which have descendants which are not part of the selection. These symbols are connected as a tree within the larger formula tree, but they do not have the typical property of extending all the way to leaf nodes.

When replacing trees using the visual tree structure, it is not often that we wish to replace a whole subtree as in all of the nodes, and intermediate nodes that are a descendant of a root node. This means tree replacement in this situation is not as simple as exchanging the two root nodes. There are potentially many subtrees that are connected to leaf or non-leaf nodes in the subtree to be replaced which we wish to move over onto the replacement tree. This is not always possible to do (e.g. moving N subtrees from an initial subtree a new subtree which only has N-1 outgoing edges) and therefore we allow for a 'floating forest' portion of the results of an operation where any subtrees which we cannot move over are placed into this portion of the result. To the user, these become formulas lifted in chips on the canvas, and they can manually re-connect them as they desire. However, we wish to save the user the effort of manually making these reconnections and attempt to reconnect these parts across subtrees as much as we can, without performing re-connections which may not make sense to the user. We expect it will be easier for the user if we give them control in an ambiguous scenario, so that they may make the connection themselves rather than having the reconnection dramatically alter the result in an unexpected way.

To implement these automatic reconnections of subtrees between an initial subtree and replacement subtree, we develop a model based on the relative outgoing edge paths of the subtrees. The simplest scenario is when the subtree to be replaced has no outgoing edges (it extends to all leaf nodes). In this case the only thing which needs to happen is the placement of the new subtree at the position of the old subtree in the formula context. When a selected subtree has outgoing edges to other subtrees, we consider the position of the outgoing subtree based on its outgoing edge path relative to the root of the selected subtree. In the replacement subtree, we examine its outgoing edge paths and if there is a match, we can simply connect the outgoing subtree at that path in the replacement subtree.

3.5 Selection

Selections can be made in main two ways: clicking on a symbol or dragging a rectangular region over a set of symbols in the formula. In these cases the selection is displayed to the user, and further specification of the operation must occur. When a user begins to drag a symbol, this is also a selection of the symbol, but will be used as an argument to an operation.

In order to implement the selection mechanism, we needed to find out which symbols are covered by the selection rectangle. This involves looking up the bounding boxes for each of the symbols in the formula, for each node in the tree. This way we can have a set of nodes in the tree which correspond to the symbols on the canvas which are covered by the selection rectangle. The criteria of whether the bounding box of a symbol is covered by the rectangle of the selection is just a simple check to see if the boxes intersect.

Once we have the set of nodes, which meet the criteria of being covered by the selection rectangle, we need to group them into subtrees so that the selection is connected and not a separate set of symbols. It is still possible for multiple disconnected subtrees to be in the same selection rectangle so we use a heuristic to identify a primary selection subtree which is used for most operations except for delete which uses all selected subtrees. This distinction between the primary selection subtree and secondary subtrees is communicated to the user via an orange bounding box covering the primary subtree as well as a grey bounding box covering all of the subtrees in the selection.

Additions can be made at any control point. The implementation is based on the formula context, the node of the control point, the relation for the insertion, and the tree argument.

3.6 Examples

Consider a simpler type of tree with relations: right, super, and sub.

3.6.1 Tree Replace Example 1

With: $x^2 + 1$, replace selected x with y. The result is: (tree: $y^2 + 1$, floating forest: $\{\}$). This is because the outgoing edge paths for $x = \{$ right, super, sub $\}$, the outgoing edge paths for x (with subtrees) $= \{$ right, super $\}$, the outgoing edge paths for $y = \{$ right, super, sub $\}$, and the intersection $=$ $\{$ right, super $\}$. The difference/remainder $= \{ \}$ and the selected subtree's outgoing path, subtree pairs are $\{$ (right, +1), (super, 2) $\}$. The result tree is formed by connecting the subtree y with subtrees at paths in the intersection, and moving (the remaining) subtrees at paths in the difference to the floating forest portion of the result.

When performing the path intersection check we actually want more flexibility over exact path matching, so that we can have replacements along a baseline reconnect more naturally. To do so we implement a modification of the path intersection which ignores any right relations at the end of the paths. This allows for a variety of additional automatic reconnections the including connected symbols left and right of a set of symbols deleted in a baseline.

3.6.2 Tree Replace Example 2

With: $x_{a^{b+3}+5}^{y^{z+1}+4} + 2$, replace $x_{a^b}^{y^z}$ with α. The result is: (tree: $\alpha_{+5}^{+4} + 2$, floating forest: $\{+1, +3\}$). This is because the outgoing edge paths for $\alpha = \{$ right, super, sub $\}$, the outgoing edge paths for $x_{a^b}^{y^z}$ (with subtrees) $= \{$ right, super:right, super:super:right, sub:right, sub:super:right $\}$, and the intersection with flexible matching $= \{$ right, super, sub $\}$. The difference/remainder $=$ $\{$ super:super, sub:super $\}$ and the selected subtree's outgoing path, subtree pairs are $\{$ (right, +2), (super:right, +4), (super:super:right, +1), (sub:right, +5), (sub:super:right, +3) $\}$.

3.7 Implementation Behavior

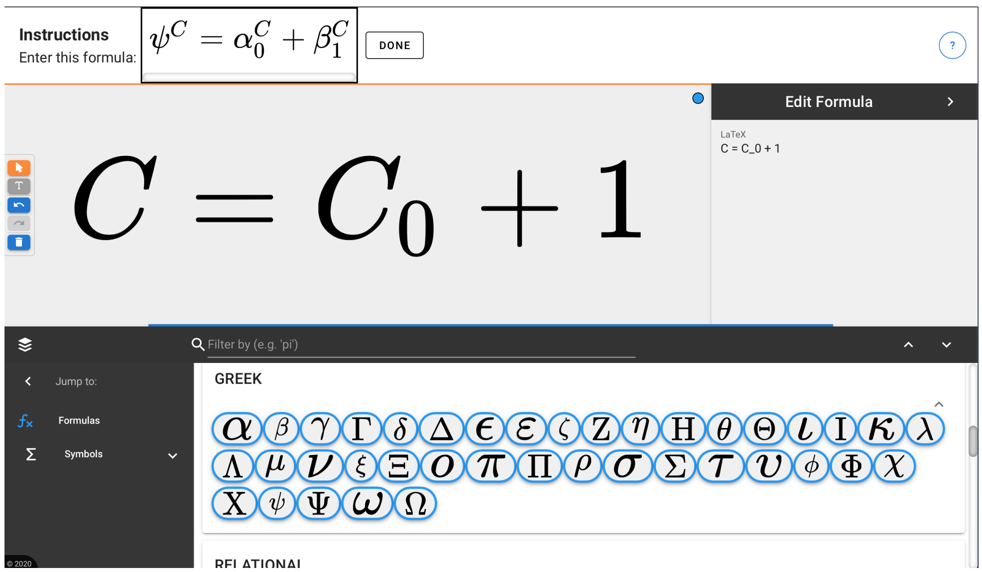


Figure 3.7: The interface for the experiment.

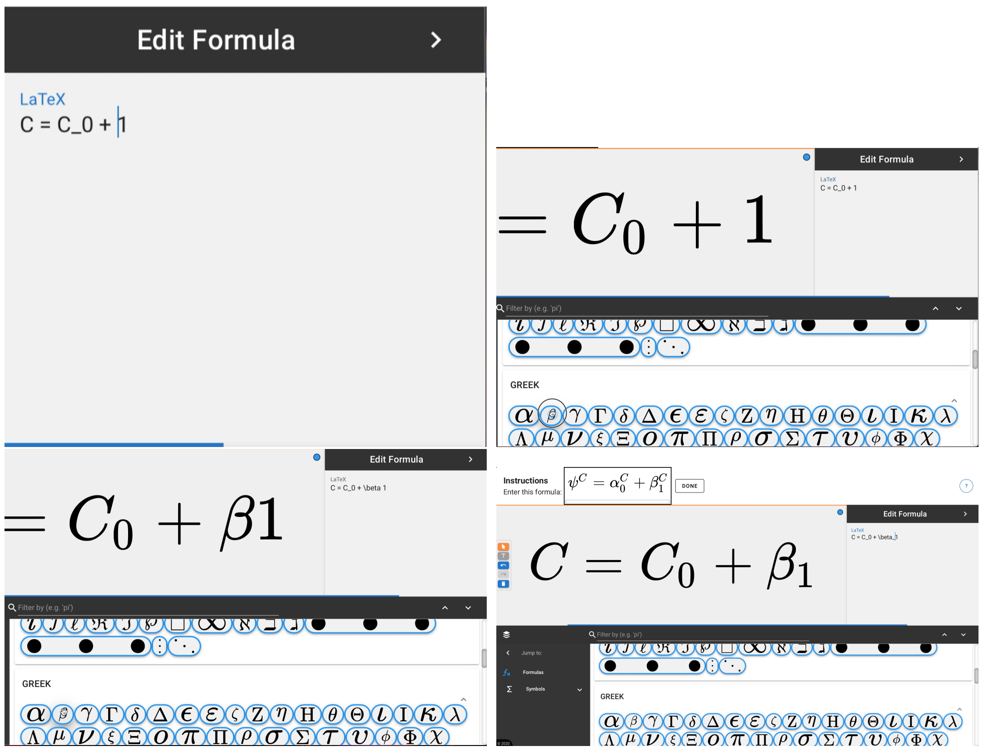


Figure 3.8: Example of editing with LaTeX in the interface and using the symbol palette to insert into the string. Notice the cursor position, the click on the β formula chip in the deck, followed by the insertion and the user entering '_' to move the 1 into the subscript position.

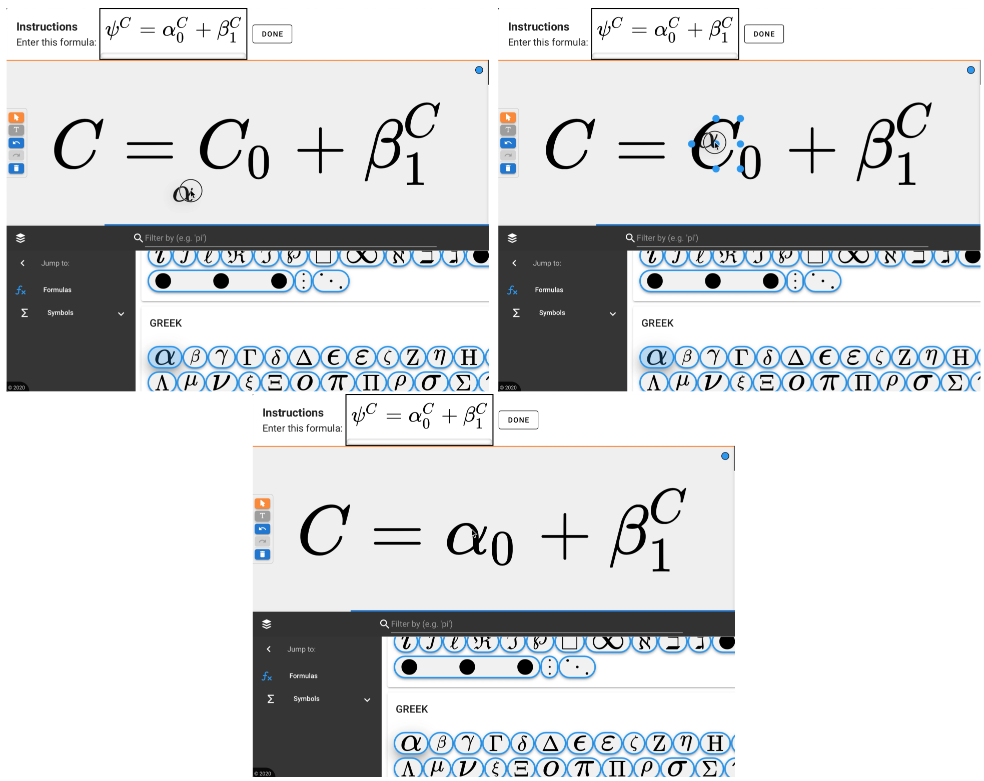


Figure 3.9: Step by step example of editing with visual operations by dragging a formula chip from the symbol palette. The formula chip for α is dragged to the canvas near the C_0, since the C symbol is closest, its control points appear, and when it is dropped nearest to the center control, the default operation of replace is performed. This example is demonstrated in a practice video shown to participants.

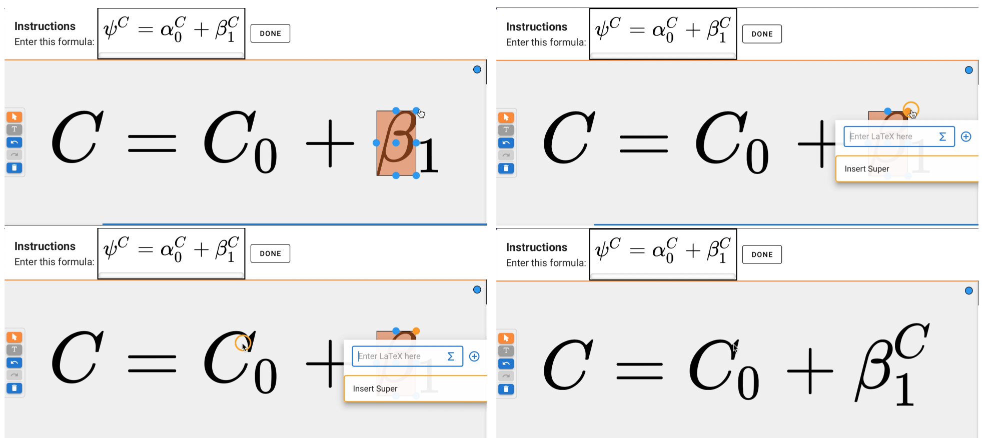


Figure 3.10: Step by step example of editing with visual operations without dragging where the β symbol is selected, its super control is selected, and then the C symbol is selected as the argument to the operation. Arguments can be any symbol or selection in the formula, any chip, or the LaTeX entered in the text box which appears for the control. This example is demonstrated in a practice video shown to participants.

Chapter 4

Results

In this chapter we present the design of our experiment for testing and evaluating our new visual editing operations. The main measurement tools were questionnaires (demographic and post-experiment) and timed formula editing tasks, in which participants convert a provided initial formula to a target formula using three different editors: a ‘plain’ LaTeX formula editor, an editor using our visual operations, and a third combining the ‘plain’ LaTeX and visual operations. A symbol palette providing LaTeX shortcuts for symbols and structures such as fractions are provided in all conditions.

In the remainder of this Section we present the experimental design, followed by results for the questionnaire and editing task results. Additional results may be found in Appendix A.

4.1 Experimental Design

In order to evaluate whether the hypothesis that our visual operations would provide an easier way to make formula edits, a prototype system was developed to implement these operations. There are some issues and limitations of the prototype implementation of the operations, so if possible we would like to distinguish between problems with the implementation and problems with the concept.

4.1.1 Experimental Tasks

Our experiment is designed to measure the **difference in editing time** and **number of operations and actions** performed by the user as they complete a formula editing task using the visual operations, LaTeX editing on its own, or with both options available. The editor mode (i.e., availability of the visual operations and the LaTeX editing area) is the primary independent variable, and the editing time and number of operations and actions performed are the primary dependent variables.

Ideally we would like to know the interactions between these variables for all formulas and all edits. However both of these have too many possibilities for this to be practical for realistic scenarios. In order to try to understand the behavior on formulas, we attempt to account for some variable properties of formulas and edits which we expect to have an influence on our dependent variables.

To observe behaviors for different editing tasks, we select tasks to cover different editing operation types needed in each formula editing task, such as only addition (of symbols), only deletion, or some combination. Additionally, whether or not there is a large difference between the starting formula and the target formula is expected to impact editing time, and the number of operations and actions needed to complete a task. To control for these variables of edit type and edit distance, we attempt to classify editing tasks by these two variables with either a high or low editing distance (using tree edit distance on the formula's visual tree structure), and editing types of either addition only, deletions only, or a combination (which includes moving within a formula).

We created 18 formula editing tasks referencing formulas from the NTCIR12 [29] and formula entity cards [6] data sets. Due to limitations of the prototype operations, we created tasks which we think are both interesting and within the capabilities of the prototype, while attempting to balance for editing operations and distance across the 18 formulas. The tasks can be found in Table 4.1 and Table 4.2.

4.1.2 Participants and Blocking Design

We designed the experiment to have at least 30 participants, ideally more than 36, and we invited 48 of the participants who signed up to try to reach that number. We also attempted to control for order effects across the formulas and

Table 4.1: The first 9 formula editing tasks. Tasks are identified as requiring only adding symbols, only deleting symbols, or both (a 'change').

Task	Start formula Target formula
1 (change)	$x = b_0 + \frac{a_1}{b_1 + \frac{a_2}{b_2 + \frac{a_3}{b_3 + \frac{a_4}{b_4 + \ddots}}}}$ $b_0 + \frac{a_1}{b_1 + \frac{a_2}{b_2 + \frac{a_3}{b_3 + \ddots}}}$
2 (change)	$P_{mag} = \frac{B^2}{2\mu_0}$ $F = \frac{B^2 A}{2\mu_0}$
3 (add)	$f(x) =$ $f(\lambda x) = \lambda^{\Delta} f(x)$
4 (add)	$\int f(x)dx$ $\int_{-1}^{1} f(x)(1-x)^{\alpha}(1+x)^{\beta}dx$
5 (change)	$x^n + c_{n-1}x^{n-1} + \cdots + c_2x^2 + c_1x + c_0$ $c_0 + c_1x + c_2x^2 + \cdots + c_{n-1}x^{n-1} + x^n$
6 (delete)	$b^2c^2x^2 + a^2c^2y^2 + a^2b^2z^2 - a^4yz - b^4xz - c^4xy = 0$ $b^2x^2 + a^2y^2 + a^2b^2z^2 - a^4yz - b^4xz - xy = 0$
7 (change)	$k^2A + \nabla^2 A = 0$ $\nabla^2 A + k^2 A = 0$
8 (add)	$F(X) = 0$ $R^i F(X) = 0 \forall i > 0$
9 (delete)	$\frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = \frac{\alpha}{k}u(1-u^q)$ $\frac{\partial u}{\partial t} - \frac{\partial u}{\partial x}$

Table 4.2: Formula editing tasks 10-18.

Task	Start formula Target formula
10 (change)	$\frac{x^2}{a^2} - \frac{y^2}{b^2} - z = 0$ $\frac{x^2}{a^2} = z + \frac{y^2}{b^2}$
11 (change)	$2\varphi - 1 = \sqrt{5}$ $\varphi = \frac{1+\sqrt{5}}{2}$
12 (add)	$A^+AA^+ \quad =$ $(AA^+)^* \quad =$ $(A^+A)^* \quad =$ $A^+AA^+ \quad = A^+$ $(AA^+)^* \quad = AA^+$ $(A^+A)^* \quad = A^+A$
13 (add)	$\overrightarrow{F} =$ $\overrightarrow{F} = -\frac{c}{12\pi\sigma}\overrightarrow{\nabla}U$
14 (change)	$L_o(x, \overrightarrow{w}) = L_e(x, \overrightarrow{w}) + \int f_r(x, \overrightarrow{w}, \overrightarrow{w}')(\overrightarrow{w}' \cdot \overrightarrow{n})d\overrightarrow{w}$ $L_o(x, \overrightarrow{w}) = L_e(x, \overrightarrow{w}) + \int_\Omega f_r(x, \overrightarrow{w}', \overrightarrow{w})L_i(x, \overrightarrow{w}')(\overrightarrow{w}' \cdot \overrightarrow{n})d\overrightarrow{w}'$
15 (delete)	$[\rho^a(\overrightarrow{x}), \rho^b(\overrightarrow{y})] = \delta(\overrightarrow{x} - \overrightarrow{y})\rho^c(\overrightarrow{x})$ $\rho^a(\overrightarrow{x})\rho^b(\overrightarrow{y})\rho^c(\overrightarrow{x})$
16 (change)	$\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_n x_n$ $\gamma_1 x_1 + \gamma_2 x_2 + \cdots + \gamma_n x_n$
17 (add)	$a \triangleleft (b \triangleleft c) = (a \triangleleft b) \triangleleft (a \triangleleft c)$ $a \triangleleft (b \triangleleft (c \triangleleft d)) = (a \triangleleft b) \triangleleft (a \triangleleft (c \triangleleft d))$
18 (delete)	$Q = -k\frac{dT}{dz}$ $\frac{T}{z}$

our 3 input configurations of LaTeX only, Visual only, and both by changing the order they are presented across participants. To do so we split the 18 formulas into 3 blocks of 6 formulas each which are roughly even in terms of high/low edit distance, and add/delete/change editing types for the tasks. We then randomized the initial order of the tasks in each of these blocks.

The presentation of different input modes is alternated between one order of LaTeX only, then Visual only, then both, and a second order of Visual only, then LaTeX only, then both. We rotated the 3 formula blocks across participants, and after 3 participants, this loops back to the original block order. Then this switches to the Visual-LaTeX-Both order for input methods associated to each block of editing tasks. This alternating between LaTeX-Visual-Both and Visual-LaTeX-Both is repeated every 3 participants.

After every 6 participants where we have had both input orderings for each block rotation, there is a rotation within each block so that if a block of formula editing tasks was ordered 123456, it becomes 612345 (note that the original order was randomized). This pattern repeats after every 6 participants, and the within block order returns to the original order after 6 rotations, at the 37th participant.

Our blocking design is intended to minimize order effects across the participants, but uniformly rotating formula editors, editing task blocks, and tasks within each block (in that order).

4.1.3 Experiment Protocol (online)

Participants begin the experiment by completing a demographic questionnaire (see Appendix A). After this, participants are then provided with a URL linking to a page providing an overview and practice trials. To introduce available editing operations in LaTeX and our visual editor, and to reduce learning effects where a participant's behavior changes as they become more familiar with the system, we include 7 familiarization (practice) tasks after participants watch a video providing an overview of the system. We provide a demonstration video for each practice task to help participants become familiar with the system so that their behavior is more stable across the tasks. The practice tasks are generally easier and aimed at introducing system capabilities to the participants (see the Appendix).

Upon reaching the experiment system URL, participants are presented

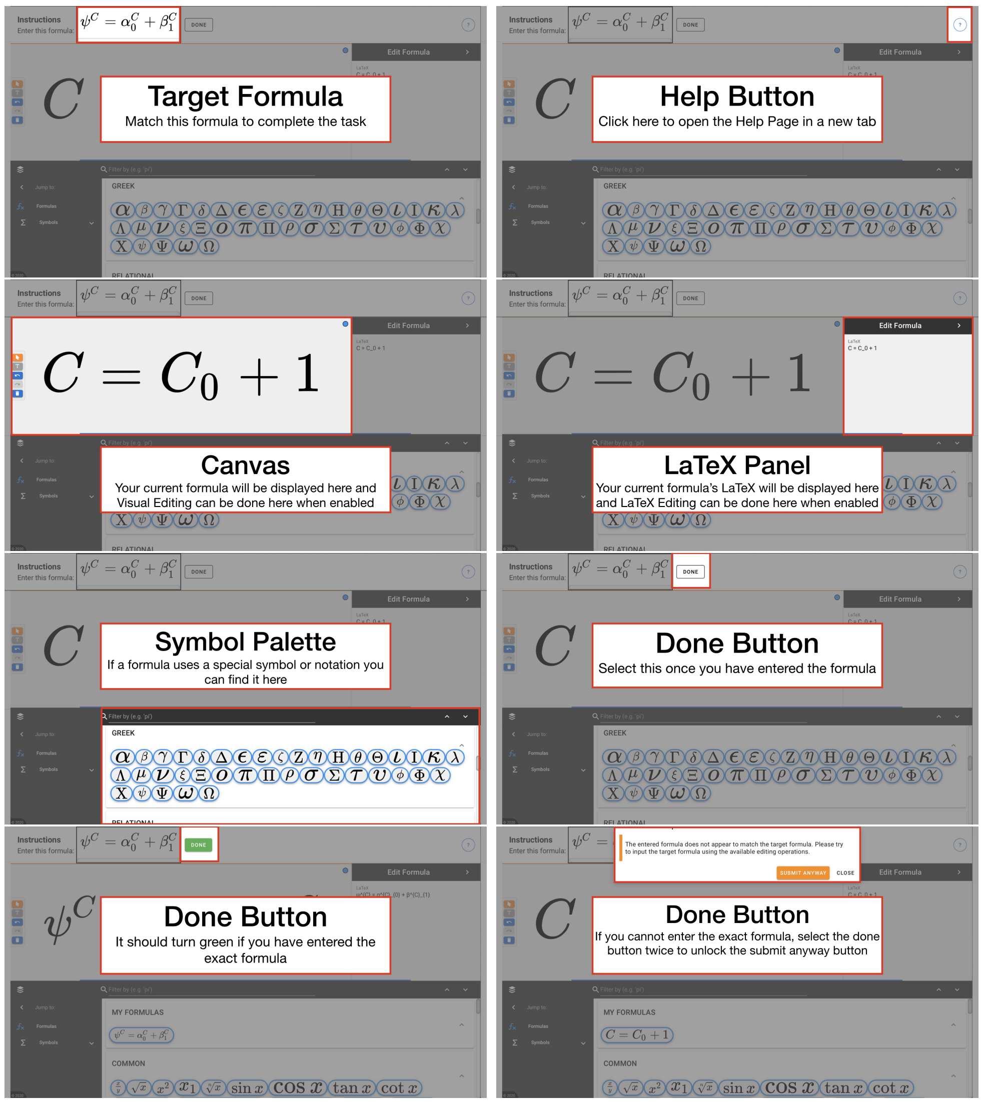


Figure 4.1: Images from the system overview video shown to participants.

with the experiment instructions which describe how the experiment will work, that there will be 7 practice tasks and a video explaining the main parts of the experiment interface and how they will be used. At the bottom of the page is a link to the first practice task. Each practice task has a video showing how to complete it, the input type available, and a link to begin. After the 7 practice tasks, there is a short message letting the participant know they have completed the practice tasks and that the experiment tasks will begin, and a reminder that they can reference the practice videos using a provided help button at any point. They are presented with the input type for the first task and a link to begin the task. After each task is completed using the 'Done' button in the interface, they are taken to the page displaying the task number, the input type for the next task, and the link to start it. After the final (18th) task, they are presented with a page saying that the tasks are complete and a link to the post experiment questionnaire is provided (see Appendix A).

In the post experiment questionnaire, participants can comment on the strategies they used for performing the tasks, provide any comments, and rate the difficulty of the tasks. After submitting the post questionnaire they are provided with a message thanking them for their participation and contact information for the lab.

4.1.4 Data Collection

The experiment system logs whenever the current formula changes, and the current state of the formula including the LaTeX string, MathML, and tree representation for performing the operations. This data will be used to create a plot of the tree edit distance between the target formula and the current formula over time. With this we can observe how close the formula gets to the target at any time during the task, and see if there are instances when it becomes further before becoming closer, or if the edit distance is strictly decreasing until it reaches zero.

The other events that are logged include whenever the participant selects the help button, and the done button and whether or not the current formula is an exact match at that point. Additionally, whenever a visual operation is performed that operation's information is logged and typing LaTeX is also logged. Finally click events are also logged to be combined with the other events to see how many actions are performed on average to complete the

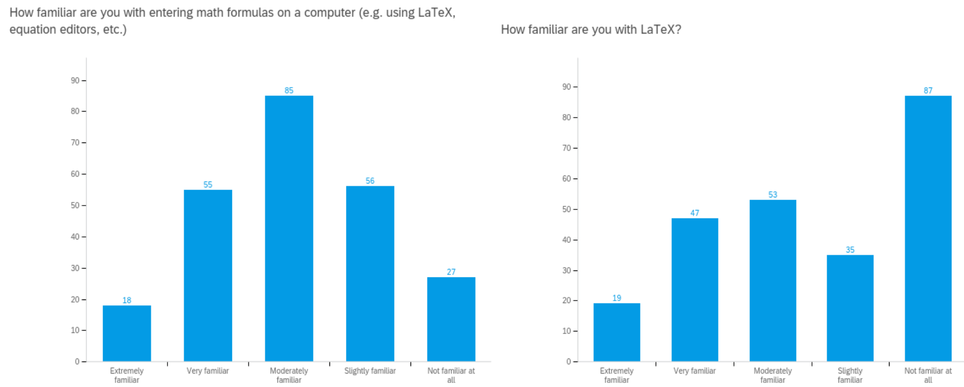


Figure 4.2: Distribution of responses to Sign Up Form Question 5: How familiar are you with entering math formulas on a computer (e.g. using LaTeX, equation editors, etc.) and Question 6: How familiar are you with LaTeX?

tasks. These are all timestamped events, so this will be used to compute the total time to complete each task as well as examine the relationship between these and changes in the edit distance to the target formula over time.

Additionally, the Flask-based system provides an alternative way to identify when participants start and end tasks - task completions are identified by time and anonymous id-stamped page requests corresponding to the beginning and completion of tasks.

4.2 Recruitment

Participants are recruited by emails sent to the GCCIS and COS. In the email is general information regarding the experiment (Appendix) and a link to the sign up form. The sign up form and other questionnaires are created with Qualtrics. Here participants provide their name, email, LaTeX and formula editing experience, and major area of study. By completing the sign up form, individuals indicate their interest in participating in the experiment. In the recruitment email, participants are told they will receive a $10 Amazon gift card for their participation.

Over a week after sending out the email, we examine the responses to the sign up form of which there are roughly 240. We observe the breakdown of responses to the questions regarding formula editing experience and LaTeX experience, each of which are on a 5 point scale as seen in Figure 4.2. We observe the formula question appears to have responses closely following a bell shape with most responses in the middle of the 5 options. The question on LaTeX experience has responses which can be placed into 3 groups of approximately the same size: responses under no experience, responses under some or moderate experience, and responses under very or extremely experienced. We decide selecting along these 3 groups would provide a good balance of candidates with varying amounts of LaTeX experience.

We select a total of 48 participants from the roughly 240 responses, 16 from people who responded with one of the two highest LaTeX experience answers, 16 from the middle two, and 16 from the no experience responses. To do so we split the responses into 3 groups based on their answers to this question and generate a random number between 0 and 1 for each candidate in a spreadsheet. We then sort each group of candidates by their random number and select the first 16 candidates in each group. This method also has the advantage of having a fair way to select addition participants if necessary by continuing down the list.

All 48 selected participants are emailed indicating they have been selected, and are given information about the experiment along with the link to the consent form. The consent form in Qualtrics ensures participants are provided with the consent form statement and they must select between 'I consent' and 'I do not consent' options before moving on. If they do not consent the form ends there.

For selected participants that do consent, they continue to the name, signature and email page to electronically sign the consent form and provide their email address to which the gift cards are emailed. The consent form also provides a method for creating anonymous participant IDs which are used through the demographic questionnaire, experiment system, and post questionnaire. Once the participant completes the consent form they are automatically navigated to the demographic questionnaire with their participant id. These questions are all optional, and once completed participants are automatically navigated to the experiment system and complete the experiment tasks as described above in Section 4.1.3.

Of the 48 people who were selected and invited to participate, 35 completed the experiment through the post questionnaire.

4.3 Participant Demographics

Ages and Education. The 35 participants were mostly within the ages of 18-24, with only 4 out of 35 being in the age range of 25-34. More than half (20) of participants selected High School as the highest level of education completed, 13 selected Bachelor's, and only 1 selected Master's. Note that these responses are for completed education, and given that we recruited by emailing current students we guess that these may correspond to current undergraduate, graduate, and PhD students.

Math Information Use. More than half of all (35) participants reported needing to look up mathematical information at least once a week (14 weekly + 6 daily), and 21 reported taking 3 to 5 courses in Mathematics in college. For needing to express mathematical information on a computer there were only 10 responses of once a week or daily, with 14 participants selecting the most common answer of once a month.

Discipline. The most common answers for major or area of study include Computer Science (10), Game Design and Development (5), Applied Mathematics (3), Computational Mathematics (3), and Software Engineering (3). The full distribution can be seen in Figure 4.3 which includes Bioinformatics, Physics and 7 other unique answers.

Gender. To the question 'What is your gender?' the responses were 60% male (21), 37% female (13), and 3% non-binary (1).

4.4 Post Questionnaire

Task Difficulty. In the post experiment questionnaire, no participants responded that the tasks were very difficult, and only participants who answered Yes to question 3 (regarding having used LaTeX for formulas before) selected 'somewhat easy' or 'very easy' as their response. This break down of responses can be seen in Figure 4.4. In question 3, the only 2 participants who reported not having used LaTeX for formulas but having used LaTeX before also reported using LaTeX for less than 1 year (Figure 4.5).

LaTeX familiarity. Of the selected participants, one of the three groups of 16 was randomly selected from responses with 'Not familiar at all' to the LaTeX familiarity sign up question. It is possible that there is an imbalance between the number of participants in each group who completed the exper-

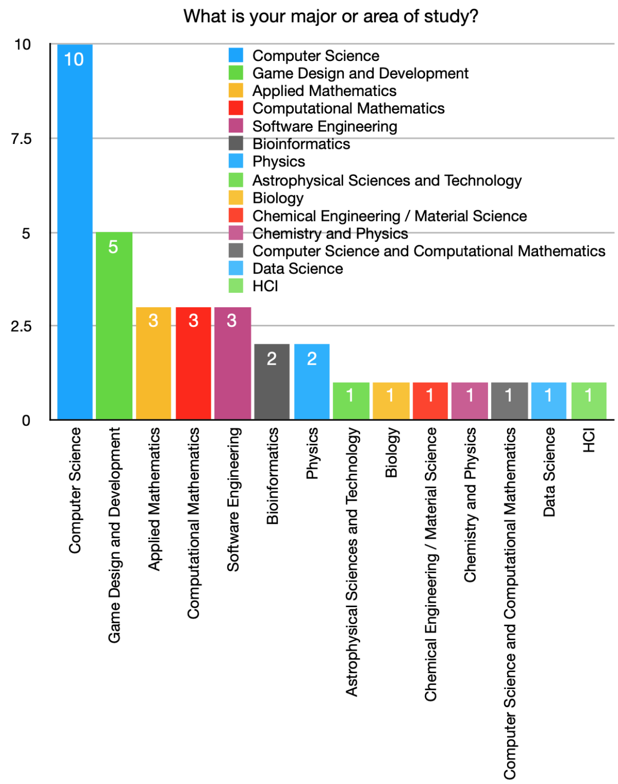


Figure 4.3: A breakdown of responses to the demographic questionnaire question: What is your major or area of study? There are 14 different answers across the 35 participants and the most common answer is Computer Science which was entered by 10 participants.

iment compared to those invited. This is due to the fact that 13 of the 48 participants selected either did not start or did not complete the experiment. If all are from a single group of 16, this would leave only 3 to represent this group.

There were 11 ‘No’ and 24 ‘Yes’ responses to ‘Have you ever used LaTeX before this study?’ which is only 1 ‘No’ participant away from being exactly the same 1/3 to 2/3 ratio as seen in the sign-up questionnaire. This appears to be evidence that the participants who completed the experiment are balanced

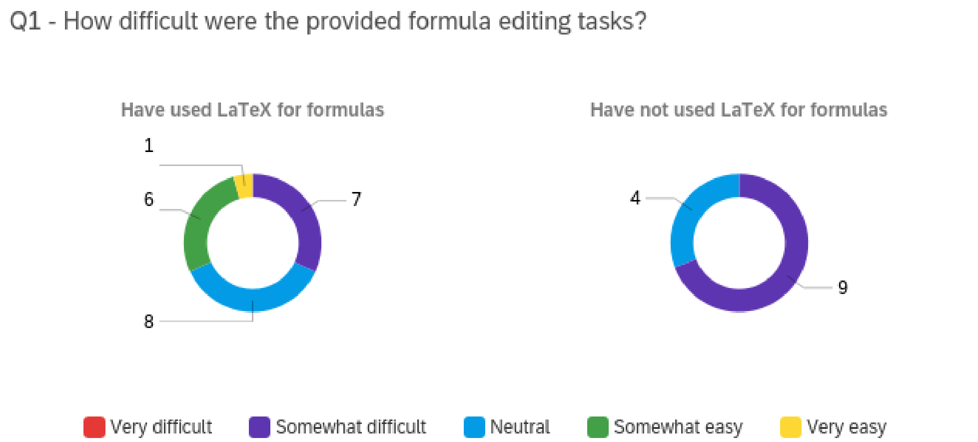


Figure 4.4: Post Questionnaire Q1 broken down by participants' answer to Q3: Have you ever used LaTeX for formulas before this study?

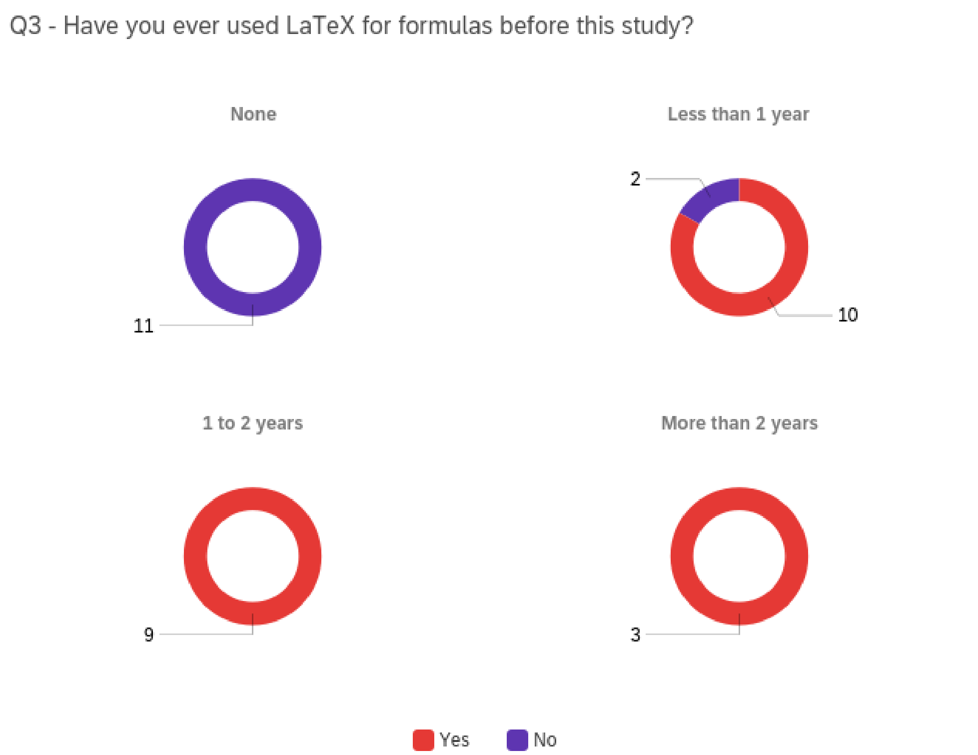


Figure 4.5: Post Questionnaire Q3 broken down by participants' answer to Q4: Before this study, how much experience did you have using LaTeX?

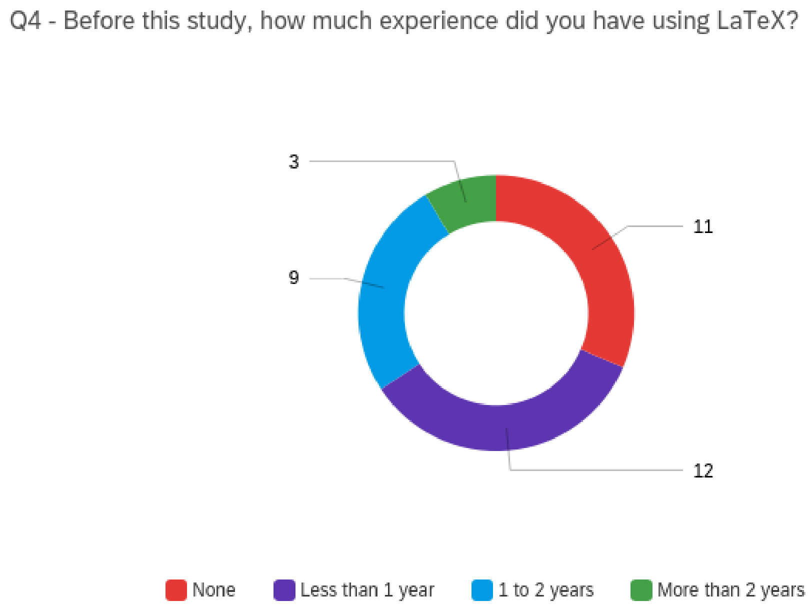


Figure 4.6: Post Questionnaire Q4 showing participants' responses to Q4: Before this study, how much experience did you have using LaTeX?

according to how they were invited (at least for the ratio between 'Not familiar at all' group compared to the Slightly/Moderately and Very/Extremely familiar groups). In Figure 4.6, we can also see that 12 participants had responses for 1 year or more of LaTeX experience (3 of which have more than 2 years). If these answers for 1+ years of experience correspond with the Very/Extremely familiar LaTeX familiarity answers, then the set of participants would be nearly exactly the balance that we were aiming for with our selection. Taking these things into account and the low chance of extreme imbalance if it were decided randomly, we think it is unlikely that the participants who completed the experiment are of a very different balance than we invited.

Open Response Questions. Participants were asked to describe the strategies they used to complete the tasks and to provide any additional comments in questions 5 and 6. One thing we are interested in is seeing if there is any similarity in the responses when grouped by the participant's answer to the question of having used LaTeX before. All of the responses to these questions can be seen split into groups by the participant's answer to this question in Appendix A. There were several comments regarding issues with the interface relating to missing keyboard shortcuts (primarily delete), a right click or double click to open the menu, sizing issues, or other things which we think can be fixed in the implementation. There were comments which clearly stated a preference for either LaTeX only, visual over LaTeX, or having both available. One nice observation is several participants commented that they had fun while completing the experiment (e.g. "I found the tool very user friendly and fun to use." and "That was fun!"). There are several comments that the visual operations are helpful or preferred in some way (either on their own or in combination) which is a good sign, but we need to look at the quantitative data to see the effects on the key dependent variables of our hypothesis: time and number of actions.

4.5 Time Results

We examine the total time spent on each Task as our time metric. In our analysis, we observe the time spent on each Task for all 18 tasks performed by each participant with the exception of one outlier which was omitted. This was a time of 13361 seconds on Task 14 under the LaTeX input mode by a participant with no LaTeX experience.

We originally planned on using the detailed event data to examine the time and number of actions performed for each task. However, when we were examining the data collected there were instances where some participants did not have any events for some tasks. Of the 35 participants who completed the post questionnaire, 26 of them had at least one detailed event being logged for each of their 18 tasks. However, one participant had only 2 of 18 tasks with any detailed events logged and upon closer inspection, there was an instance where a participant had events logged for all 18 tasks, but for one task had only a single event at the task start. We think reasons for this may include the delete button accidentally bringing the participant back, networking issues for some of the events, and perhaps some browsers did not send the last set of events when the done button is selected before navigating to the next step. Due to the certain incompleteness of the detailed event data for specific tasks and the uncertainty of the completeness for the remaining tasks we had to fall

Task	Min	Q1	Median	Q3	Max	Mean
T1	7	89	161	250.5	705	205.3
T2	14	20	30	46.5	128	34.4
T3	5	50	68	93	770	95.77
T4	20	68	97	131.5	324	118.6
T5	7	107	128	197	846	170.9
T6	5	31	39	49.5	158	44.54
T7	5	21	26	33	151	32
T8	4	48	71	98	377	90.94
T9	5	27	35	51	113	41.6
T10	7	38	44	65	149	54.2
T11	6	46	66	124.5	377	100.5
T12	28	43.5	50	85	435	84.63
T13	12	114.5	172	343	616	224.7
T14	107	211.5	285.5	396	1118	370.6
T15	14	34	41	59	355	56.63
T16	20	29	42	59.5	336	57.14
T17	8	58	72	102	722	115.17
T18	11	19	23	27	192	34.94
All	4	34	58	119	1118	107.0

Table 4.3: Statistics for time in seconds spent on each task type.

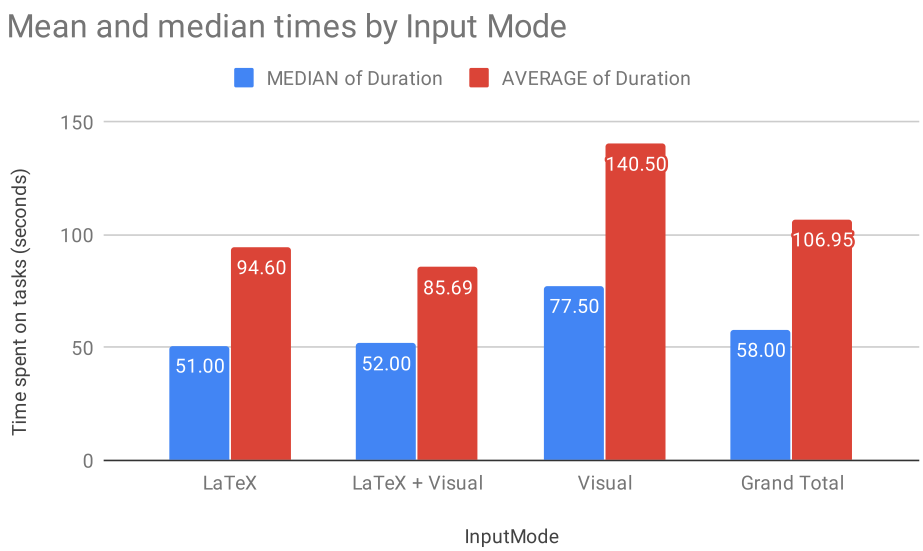


Figure 4.7: Mean and median time spent on tasks with each input method.

back to our less detailed timing data.

Instead of using the detailed event data we still examine the time participants spent on each task, but we don't have the information (in all cases) regarding whether or not the participant completed the task with an exact match.

4.6 Comparing by LaTeX Experience

The charts in Figure 4.10 and Figure 4.10 showing the data for time spent on each task broken down by participants responses to the post questionnaire question 'Have you ever used LaTeX for formulas before this study?' are interesting. While participants spend the most time on the visual only tasks, and the participants with LaTeX experience generally spend less time than the participants without LaTeX experience, the LaTeX + Visual input mode brings the time spent by participants with no LaTeX experience much closer to the times of the participants with LaTeX experience. In the LaTeX only

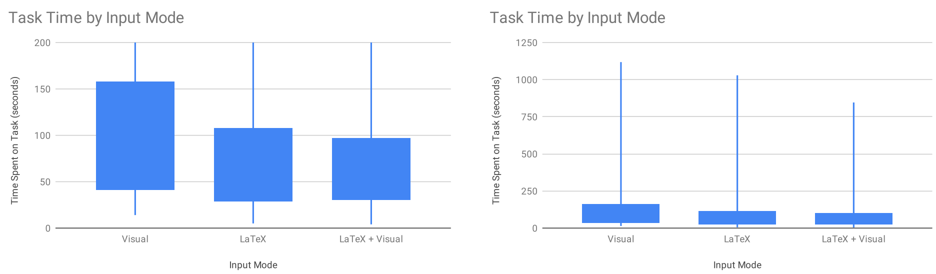


Figure 4.8: Time spent on each task for each input mode. Only times under 200 seconds are shown in the version on the left in order to more easily see the differences between the middle 50% of data. The full version with the max values is shown in the version on the right.

input mode tasks, participants without LaTeX experience on average spent 28.4 more seconds (33% more time) on each task than participants with LaTeX experience who averaged 84 seconds per task. However, in the LaTeX + Visual input mode tasks, participants without LaTeX experience on average spent only 10.4 more seconds (13% more time) on each task than participants with LaTeX experience who averaged 81.8 seconds per task. Median times were 40% higher (than 44 seconds) for participants without LaTeX experience in the LaTeX only input mode tasks, and only 9% higher (than 48.5 seconds) in the LaTeX + Visual input mode tasks.

In order to determine if there are any interesting patterns between the multiple variables of Input Mode, LaTeX experience of the participant, and the type of edits required for the task, we examine the data displayed in Figure 4.12 and Figure 4.13. A noticeable trend is the higher time spent on the add type tasks in the Visual only input mode. When the input is Visual only and the tasks are add type tasks, the average time is more than twice as high as the average time with LaTeX only for participants without LaTeX experience, and 60% higher for participants with LaTeX experience. This may have to do with the how adding to a formula with the visual operations requires the extra steps of opening the control menu (without shortcuts such as right click) and then selecting an argument for an operation or focusing the local text field. Removing these extra steps would likely be a good way to improve the visual operations for additions.

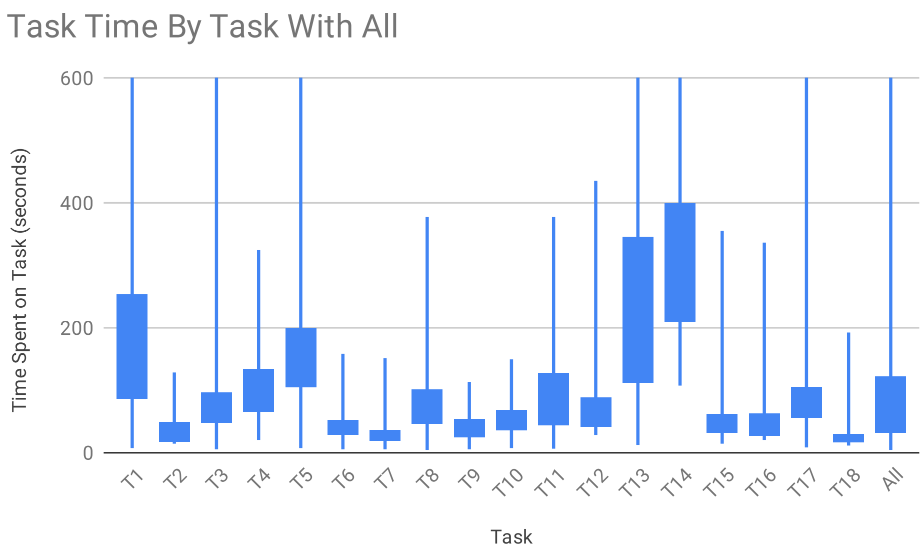


Figure 4.9: Time spent on each task. The box ends are at the 1st and 3rd quartiles. The chart is capped at 600 seconds in order to more easily see the differences between tasks.

There are also some interesting patterns in the times for delete type tasks. One notable thing is the average times for delete type tasks are almost equal between Visual only and LaTeX only modes regardless of previous LaTeX math experience. Interestingly, the participants with no LaTeX math experience spent less time than the other participants on the delete type tasks in both the Visual only and LaTeX only input modes. However, in the LaTeX + Visual condition the participants with LaTeX math experience spent less time on average. Additionally, in the LaTeX + Visual condition for delete type tasks, the average times are approximately half (50.5% and 51.5%) of what they were in the other two modes for participants with LaTeX math experience. Participants with no LaTeX experience also spent 16-18% less time on average in the LaTeX + Visual condition. It is also interesting to consider this keeping in mind that several participants had comments about adding a keyboard shortcut for the visual delete operation.

Input Mode	Min	Q1	Median	Q3	Max	Mean
Visual (210)	14	42	77.5	156.75	1118	140.5
LaTeX (209)	5	30	51	107	1029	94.60
LaTeX + Visual (210)	4	31	52	96	846	85.69
No experience						
Visual (78)	23	46	96	194	1118	161.3
LaTeX (78)	16	37.25	61.5	136	1029	112.4
LaTeX + Visual (78)	18	33.5	53	106	734	92.23
Yes experience						
Visual (132)	14	40.75	71.5	136	829	124.8
LaTeX (131)	5	27.5	44	91	556	84.01
LaTeX + Visual (132)	4	31	48.5	91.25	846	81.82

Table 4.4: Statistics for time spent on each task broken down by Input Mode and participants' reported LaTeX math experience.

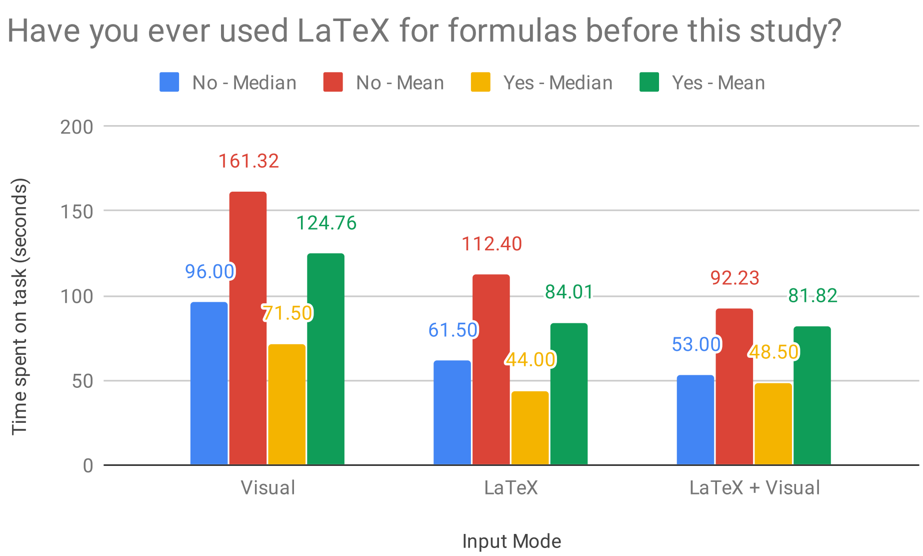


Figure 4.10: Median and Mean time spent on each task for each input mode broken down by participants response to having LaTeX experience with formula before.

For change type tasks, we notice that the average time spent on LaTeX only

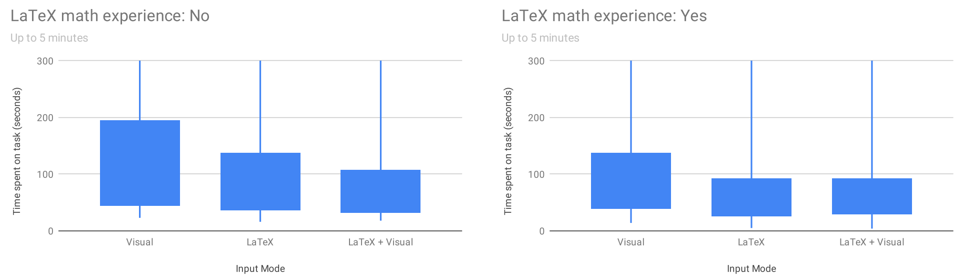


Figure 4.11: Time spent on each task for each input mode broken down by participants response to having LaTeX experience with formulas before.

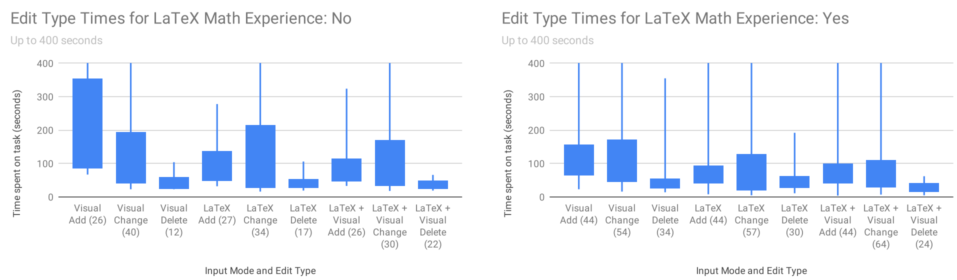


Figure 4.12: Time spent on each task for each input mode and edit type broken down by participants response to having LaTeX experience with formulas before.

input mode tasks was 67.4% higher for participants without LaTeX experience compared to participants with LaTeX experience. In the LaTeX + Visual input mode, their average time spent is only 34% higher.

4.7 Summary

If we examine the average time spent on a task under the Visual condition (140s), it is 47% higher than the average time under the LaTeX condition (95s).

While we included instructions with practice tasks and practice videos,

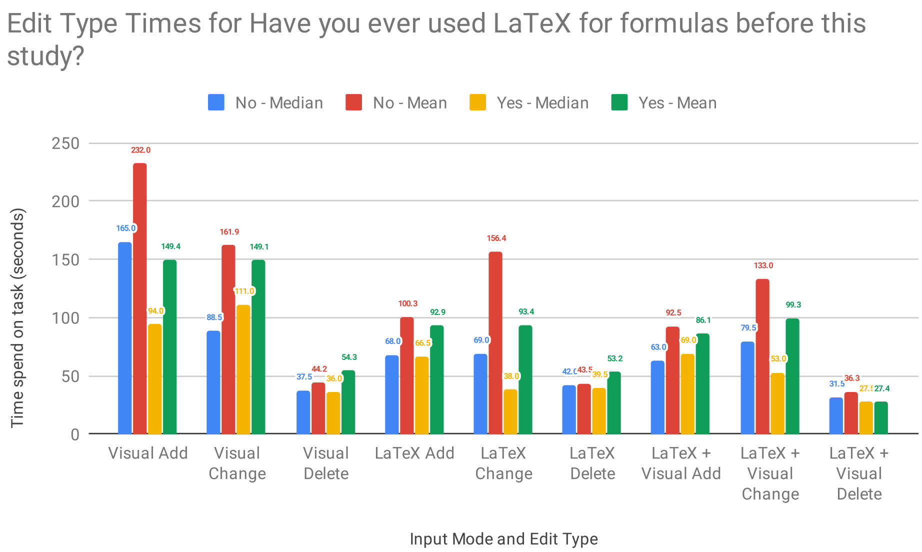


Figure 4.13: Edit types are broken down with the input mode and the participant's answer to having previous LaTeX experience.

all participants are seeing the visual operations for the first time while approximately 2/3 of participants have used LaTeX before. If we break down times by participants responses to the post questionnaire on LaTeX experience we observe. This and the known issues with our implementation may be factors contributing towards this difference. However, in the LaTeX + Visual condition, the median time (52s) is only one second longer than the LaTeX condition (51s) and the average time (85.7s) is 8.9 seconds lower than the LaTeX condition (94.6s). This supports our hypothesis than the visual operations can reduce editing time when combined with LaTeX editing. Additionally, for tasks which only required deletions, participants with LaTeX experience spent almost half the time on average in the LaTeX + Visual input mode compared to the LaTeX only mode. This might suggest that participants can find specific times during their editing when they benefit by using the visual operations, while also using on the LaTeX editing.

Chapter 5

Conclusion And Future Work

There are several improvements of the implementation of the visual editing operations which can be made. This includes adding standard keyboard shortcuts for copying and pasting, and mapping the backspace and delete keys to delete the current selection which was a common request in responses. In addition supporting a double click to open the menu and focus the LaTeX box for the replace operation, and a right click to select a symbol and open the menu can each save time by eliminating extra steps in the existing process. There are also several sizing issues which can be improved such as the controls being too close to one another for narrow symbols and symbols in large formulas. The ability to control the size of the formula by adding a zoom feature or allowing the formula to extend past the canvas edges with scrolling could also help keep the symbols at a large enough size where there is sufficient spacing between controls.

Adding full keyboard-based navigation of the formula structure with a cursor positioned at a control point in the tree could be a powerful method to interact with the visual operations and might be a faster way to perform multiple editing operations in sequence. This could mean users could type immediately at the location of the cursor in the tree, and move their cursor position with arrow keys or other key bindings. With a key binding to toggle between modes of moving within the LaTeX string at a position and moving around the positions in the tree, this could become a very efficient method and allow for usage with the keyboard as the only input device.

Extending the number of LaTeX constructions the model supports and finding a solution such as adding a remapping process after the MathJax rendering step to convert unicode symbols back into LaTeX commands, could be used to preserve the parts of the LaTeX string which are not relevant to the operation so they should in stay unchanged. Extending the implementation to have control points on groups with the ability to group and ungroup would allow for the operations to be performed in more ways for trees that are less 2-dimensional with more hierarchy.

Despite many of these issues with the visual operations, the average time participants spent on tasks was almost 9 seconds faster in the LaTeX + Visual conditions (85.7s) compared to just LaTeX (94.6s). For participants with LaTeX experience this was only a difference of about 2 seconds (84.01s LaTeX, 81.82s LaTeX + Visual), but for participants without LaTeX experience this was a difference of about 20 seconds (112.4s LaTeX, 92.23s LaTeX + Visual). This is evidence that associates the combination of visual operations and LaTeX editing with a lower average time spent on formula editing tasks, particularly for participants with no LaTeX experience. Additionally while some participants expressed a preference for LaTeX on its own, several participants also had positive comments regarding the visual operations even with its flaws, so we think these operations can be a useful addition to LaTeX formula editing especially with a more robust implementation.

Bibliography

[1] A. Allin. The "Recognition-Theory" of Perception. *The American Journal of Psychology*, 7(2):237, January 1896.

[2] Anne Aula and Tampereen yliopisto. *Studying user strategies and characteristics for developing web search interfaces*. Tampere University Press, Tampere, 2005. OCLC: 847376412.

[3] Dorothea Blostein, Edward Lank, and Richard Zanibbi. Treatment of Diagrams in Document Image Analysis. In G. Goos, J. Hartmanis, J. van Leeuwen, Michael Anderson, Peter Cheng, and Volker Haarslev, editors, *Theory and Application of Diagrams*, volume 1889, pages 330–344. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.

[4] Kenny Davila, Ritvik Joshi, Srirangaraj Setlur, Venu Govindaraju, and Richard Zanibbi. Tangent-V: Math Formula Image Search Using Line-of-Sight Graphs. In Leif Azzopardi, Benno Stein, Norbert Fuhr, Philipp Mayr, Claudia Hauff, and Djoerd Hiemstra, editors, *Advances in Information Retrieval*, volume 11437, pages 681–695. Springer International Publishing, Cham, 2019.

[5] Kenny Davila and Richard Zanibbi. Visual Search Engine for Handwritten and Typeset Math in Lecture Videos and LATEX Notes. In *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 50–55, Niagara Falls, NY, USA, August 2018. IEEE.

[6] Abishai Dmello. Representing mathematical concepts associated with formulas using math entity cards. Master's thesis, Rochester Institute of Technology, 2019. https://scholarworks.rit.edu/theses/10238.

[7] Mathias Eitz, Ronald Richter, Tamy Boubekeur, Kristian Hildebrand, and Marc Alexa. Sketch-Based Shape Retrieval. page 10.

[8] Marti A. Hearst. *Search User Interfaces*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[9] Reinhold Heckmann and Reinhard Wilhelm. A functional description of T e X's formula layout. *Journal of Functional Programming*, 7(5):451–485, September 1997.

[10] Morten Hertzum and Erik Frøkjær. Browsing and querying in online documentation: a study of user interfaces and the interaction process. *ACM Transactions on Computer-Human Interaction*, 3(2):136–161, June 1996.

[11] Bo Kang, Joseph J. LaViola Jr., and Pamela Wisniewski. Structured Input Improves Usability and Precision for Solving Geometry-based Algebraic Problems. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI '17*, pages 4692–4702, Denver, Colorado, USA, 2017. ACM Press.

[12] Gary D. Kimura. A structure editor for abstract document objects. *IEEE Transactions on Software Engineering*, SE-12(3):417–435, March 1986.

[13] Donald Ervin Knuth. *The texbook*, volume 3. 1984.

[14] Michael Kölling, Neil Brown, and Amjad Altadmri. Frame-Based Editing. *Journal of Visual Languages and Sentient Systems*, 3(1):40–67, July 2017.

[15] Michael Kölling, Neil C. C. Brown, Hamza Hamza, and Davin McCall. Stride in BlueJ – Computing for All in an Educational IDE. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education - SIGCSE '19*, pages 63–69, Minneapolis, MN, USA, 2019. ACM Press.

[16] G. Labahn, E. Lank, S. MacLean, M. Marzouk, and D. Tausky. Mathbrush: A system for doing math on pen-based devices. pages 599–606. IEEE, 2008.

[17] George Labahn, Edward Lank, Mirette Marzouk, Andrea Bunt, Scott MacLean, and David Tausky. Mathbrush: a case study for pen-based interactive mathematics. pages 143–150. Eurographics Association, 2008.

[18] David Landy and Robert L. Goldstone. Formal notations are diagrams: Evidence from a production task. *Memory & Cognition*, 35(8):2033–2040, December 2007.

[19] David Landy and Robert L. Goldstone. How abstract is symbolic thought? *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 33(4):720–733, 2007.

[20] Robert MacDougall. Recognition and Recall. *The Journal of Philosophy, Psychology and Scientific Methods*, 1(9):229–233, 1904.

[21] Mahshad Mahdavi, Michael Condon, Kenny Davila, and Richard Zanibbi. LPGA: Line-Of-Sight Parsing with Graph-based Attention for Math Formula Recognition. page 8.

[22] Alexandra Mendes, Roland Backhouse, and Joao F. Ferreira. Structure editing of handwritten mathematics: Improving the computer support for the calculational method. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces*, ITS '14, pages 139–148, New York, NY, USA, 2014. ACM.

[23] Gavin Nishizawa, Jennifer Liu, Yancarlos Diaz, Abishai Dmello, Wei Zhong, and Richard Zanibbi. Mathseer: A math-aware search interface with intuitive formula editing, reuse, and lookup. In Joemon M. Jose, Emine Yilmaz, João Magalhães, Pablo Castells, Nicola Ferro, Mário J. Silva, and Flávio Martins, editors, *Advances in Information Retrieval*, pages 470–475, Cham, 2020. Springer International Publishing.

[24] Tony Russell-Rose, Jon Chamberlain, and Farhad Shokraneh. A Visual Approach to Query Formulation for Systematic Search. In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval - CHIIR '19*, pages 379–383, Glasgow, Scotland UK, 2019. ACM Press.

[25] Steve Smithies, Kevin Novins, and James Arvo. A Handwriting-Based Equation Editor. page 8.

[26] Eugene M. Taranta, Andrés N. Vargas, Spencer P. Compton, and Joseph J. Laviola. A Dynamic Pen-Based Interface for Writing and Editing Complex Mathematical Expressions With Math Boxes. *ACM Transactions on Interactive Intelligent Systems*, 6(2):1–25, July 2016.

[27] V. Tran Minh Khuong, M. K. Phan, and M. Nakagawa. Interactive user interface for recognizing online handwritten mathematical expressions and correcting misrecognition. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, pages 26–30, Sep. 2019.

[28] Abdou Youssef. Roles of Math Search in Mathematics. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Dough Tygar, Moshe Y. Vardi, Gerhard Weikum, Jonathan M. Borwein, and William M. Farmer, editors, *Mathematical Knowledge Management*, volume 4108, pages 2–16. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[29] Richard Zanibbi, Akiko Aizawa, Michael Kohlhase, Iadh Ounis, Goran Topic, and Kenny Davila. Ntcir-12 mathir task overview. In *NTCIR*, 2016.

[30] Richard Zanibbi and Dorothea Blostein. Recognition and retrieval of mathematical expressions. *International Journal on Document Analysis and Recognition (IJDAR)*, 15(4):331–357, December 2012.

[31] Richard Zanibbi, Kevin Novins, James Arvo, and Katherine Zanibbi. "Aiding Manipulation of Handwritten Mathematical Expressions through Style-Preserving Morphs". page 8.

[32] Zheng-Jun Zha, Linjun Yang, Tao Mei, Meng Wang, Zengfu Wang, Tat-Seng Chua, and Xian-Sheng Hua. Visual query suggestion: Towards capturing user intent in internet image search. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 6(3):1–19, August 2010.

[33] Wei Zhong. Independent study report on query autocompletion for math retrieval.

[34] Wei Zhong and Richard Zanibbi. Structural similarity search for formulas using leaf-root paths in operator subtrees. In Leif Azzopardi, Benno Stein, Norbert Fuhr, Philipp Mayr, Claudia Hauff, and Djoerd Hiemstra,

editors, *Advances in Information Retrieval*, pages 116–129, Cham, 2019. Springer International Publishing.

Appendices

Appendix A

Questionnaire Data

Note: questions are shown in the order seen by participants. Question numbers were assigned by the Qualtrics system; as a result, question numbers do not start and then increment from 1.

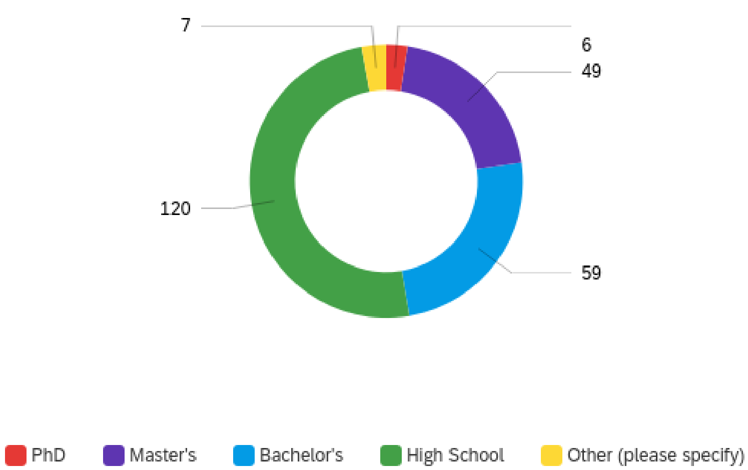
A.1 Sign Up Form

A.1.1 Question 3 - What is your major or area of study?

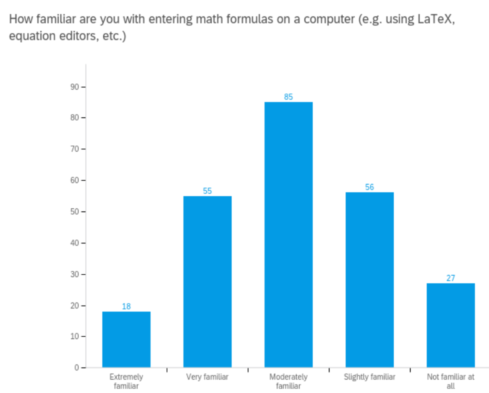
Q3 - What is your major or area of study?

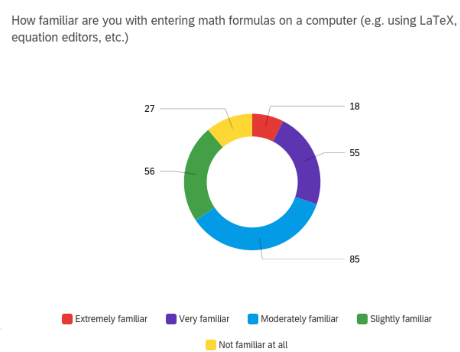
A.1.2 Question 4 - What is the highest level of education that you have completed?

Q4 - What is the highest level of education that you have completed?

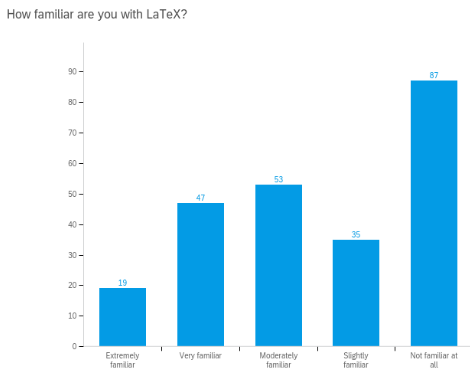


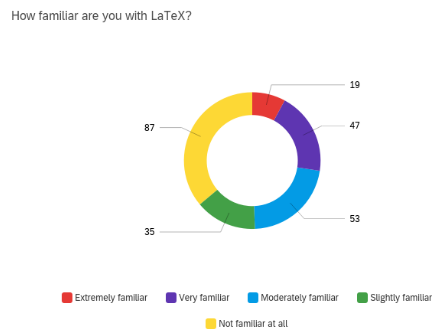
A.1.3 Question 5 - How familiar are you with entering math formulas on a computer (e.g. using LaTeX, equation editors, etc.)





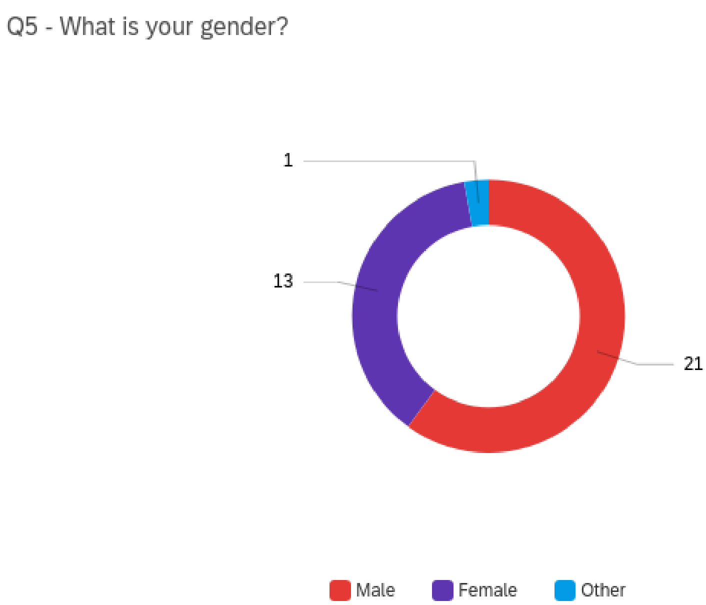
A.1.4 Question 6 - How familiar are you with LaTeX?



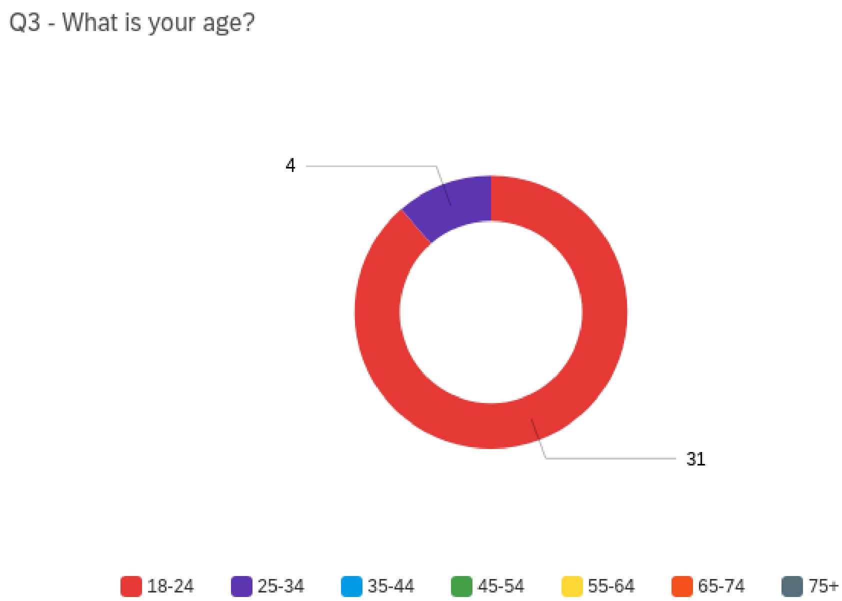


A.2 Demographic Questionnaire

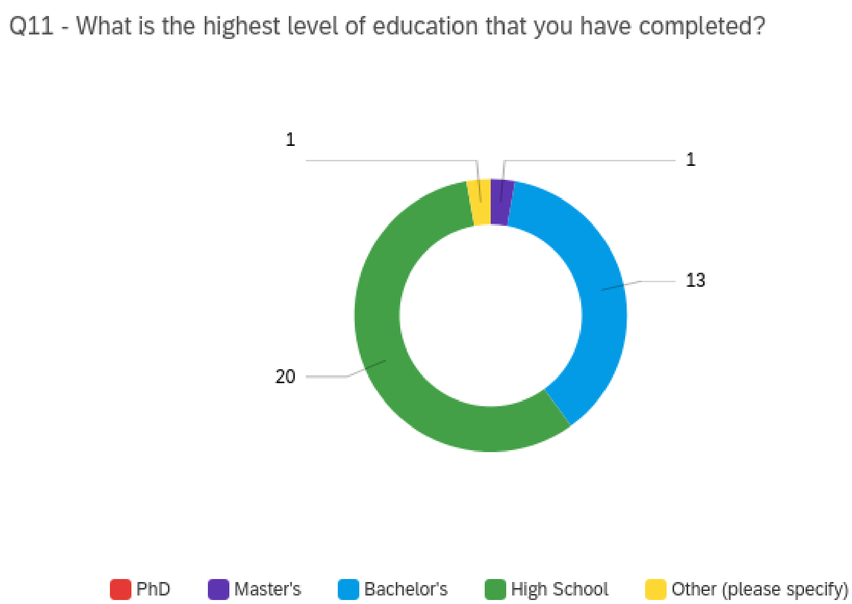
A.2.1 Question 5 - What is your gender?



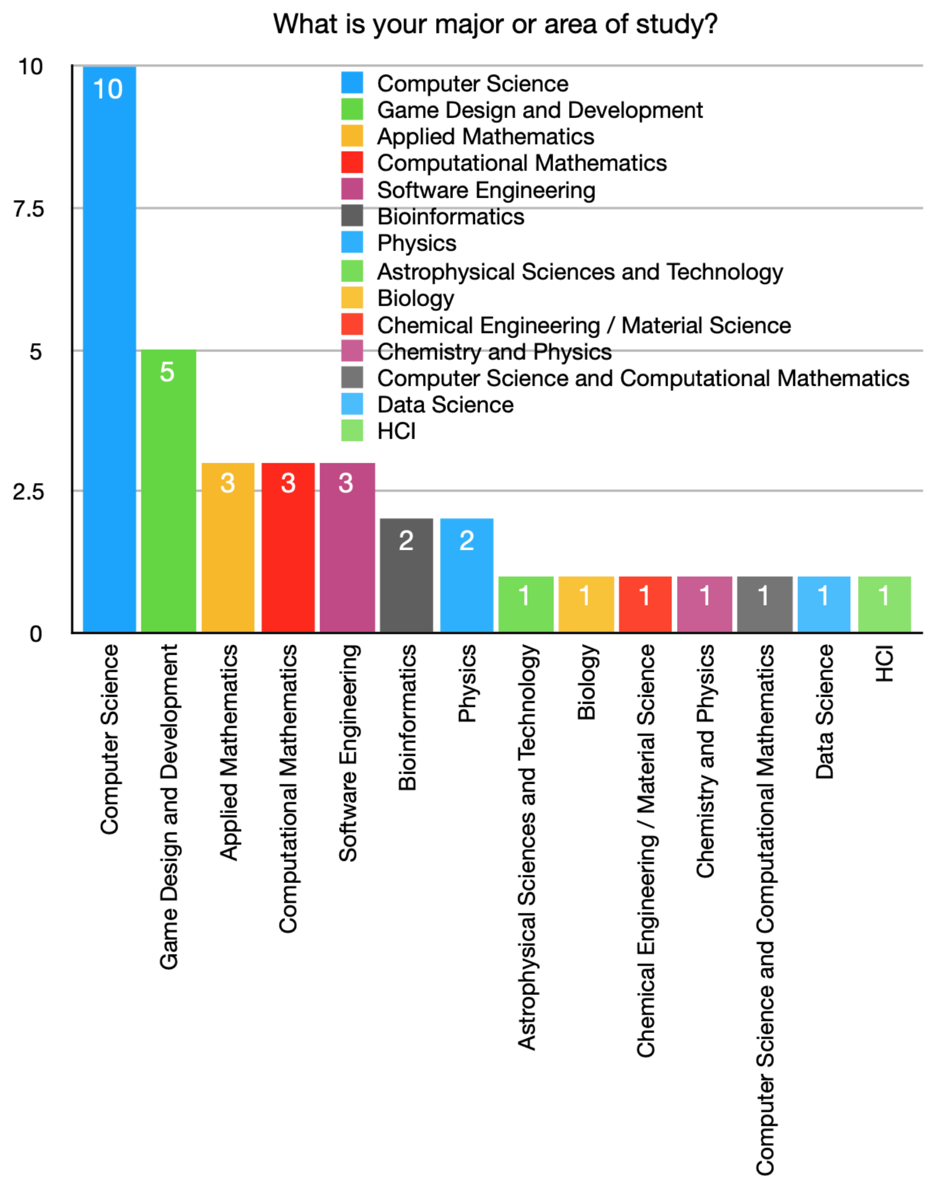
A.2.2 Question 3 - What is your age?



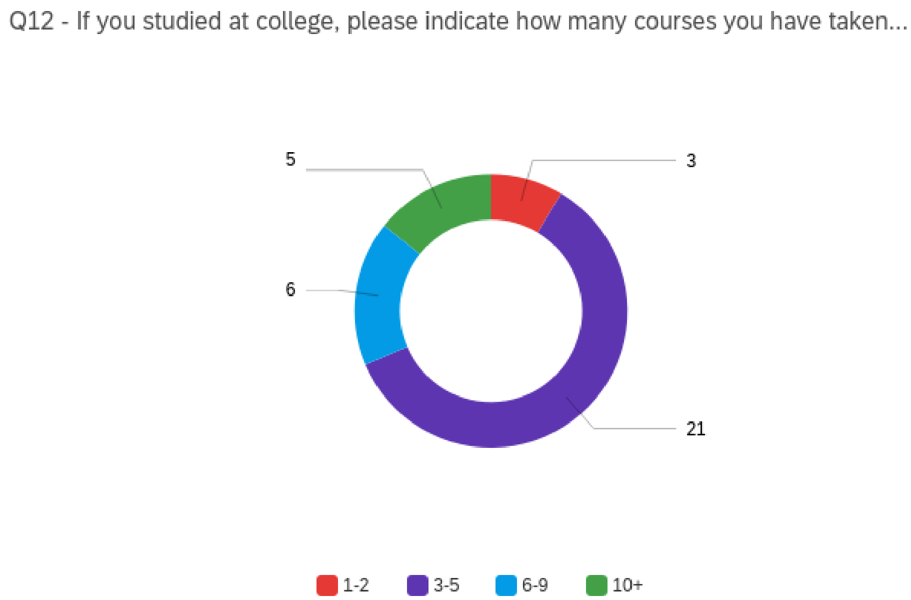
A.2.3 Question 11 - What is the highest level of education that you have completed?



A.2.4 Question 9 - What is your major or area of study?

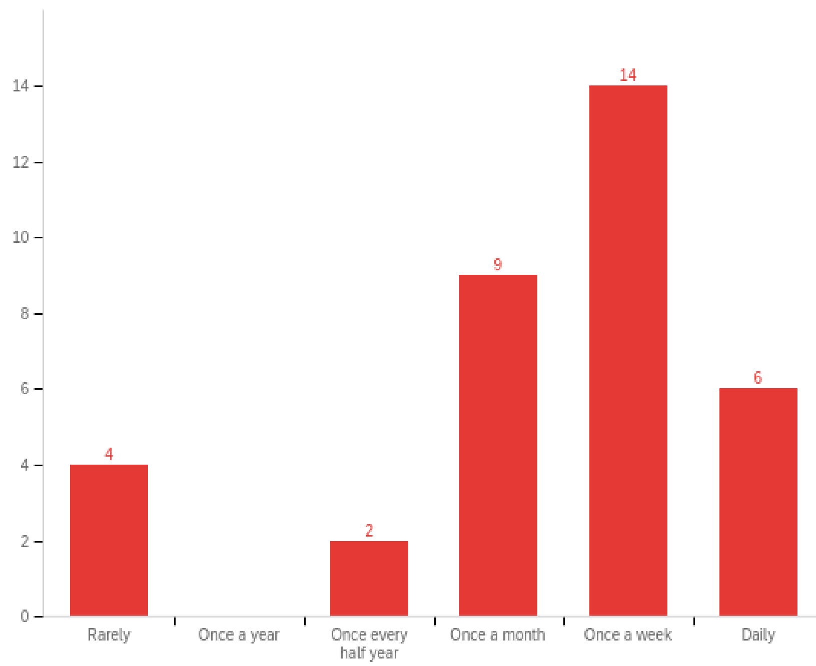


A.2.5 Question 12 - If you studied at college, please indicate how many courses you have taken in Mathematics:

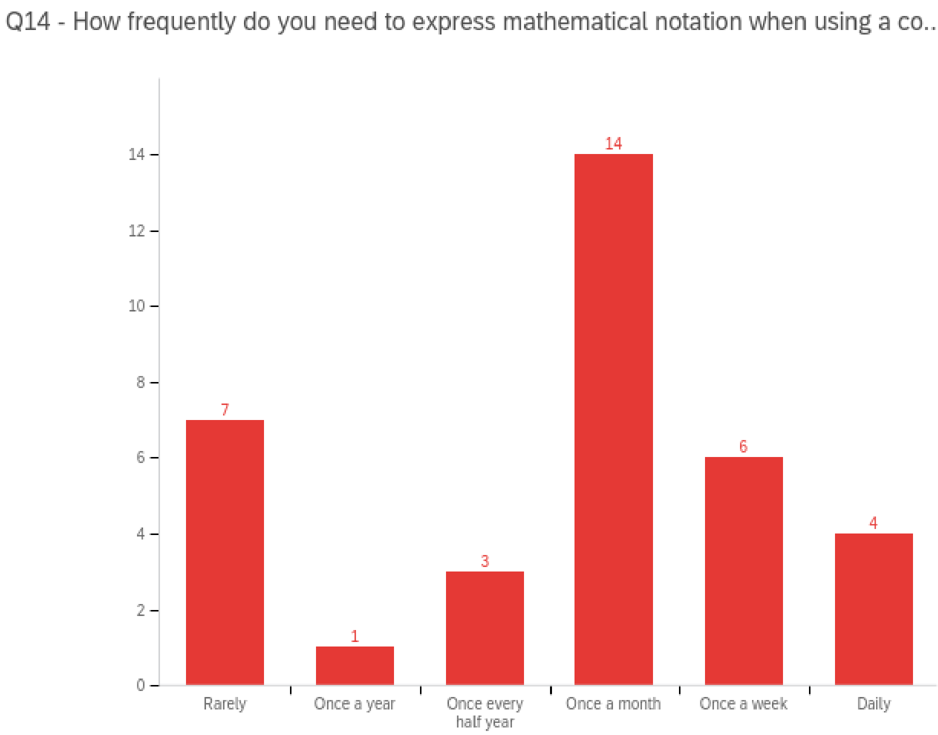


A.2.6 Question 13 - How frequently do you need to look up mathematical information? Examples of mathematical information include function definitions (e.g. trigonometric and statistical functions), definitions for mathematical symbols, function plots, mathematical models (e.g. environmental or physical models), theorems, and proofs.

Q13 - How frequently do you need to look up mathematical information? Examples of...

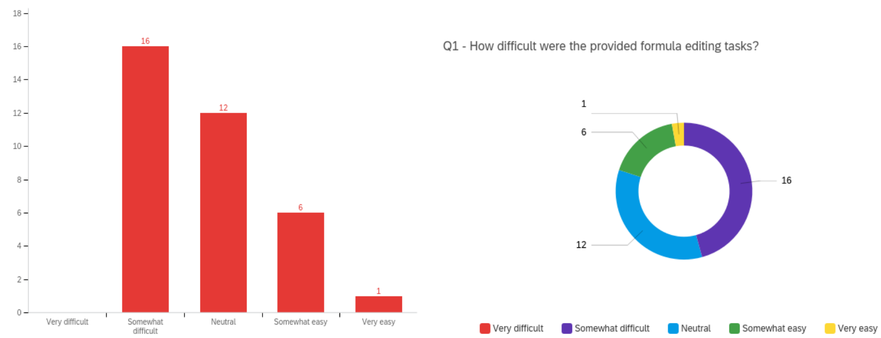


A.2.7 Question 14 - How frequently do you need to express mathematical notation when using a computer, such as for writing technical documents or in using computer programs such as Matlab, Mathematica, or Maple?

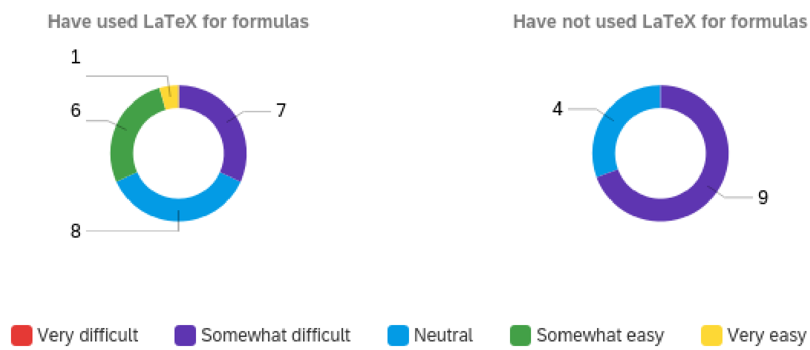


A.3 Post Questionnaire

A.3.1 Question 1 - How difficult were the provided formula editing tasks?

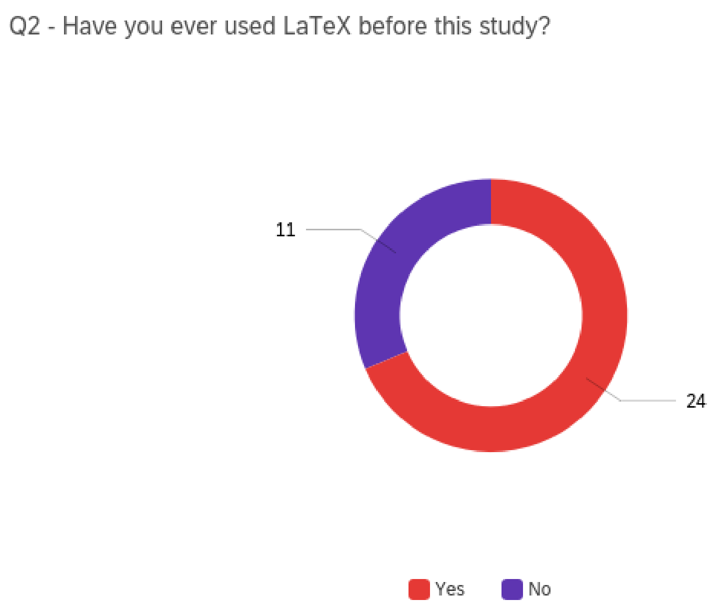


Q1 - How difficult were the provided formula editing tasks?

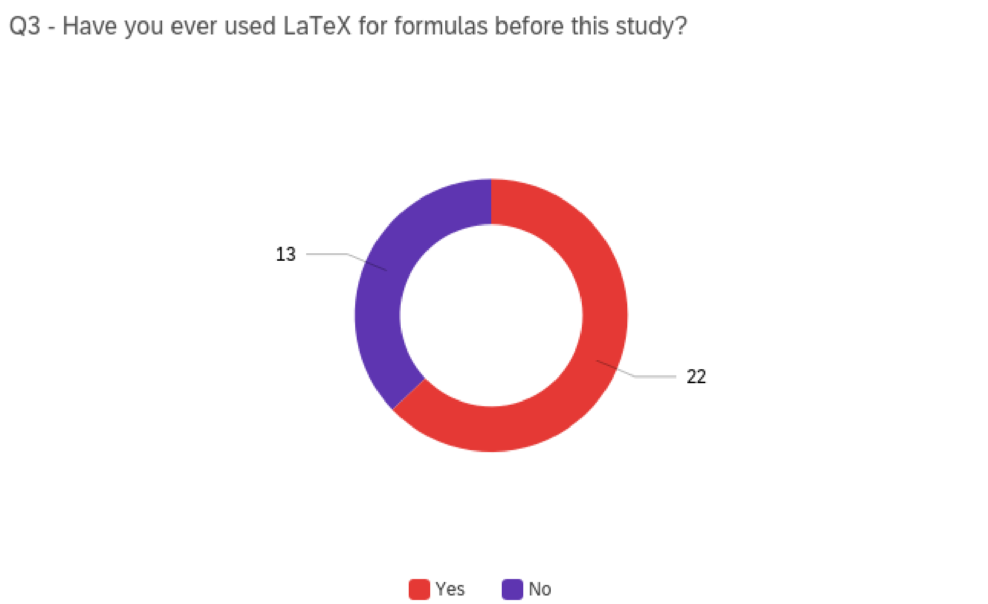


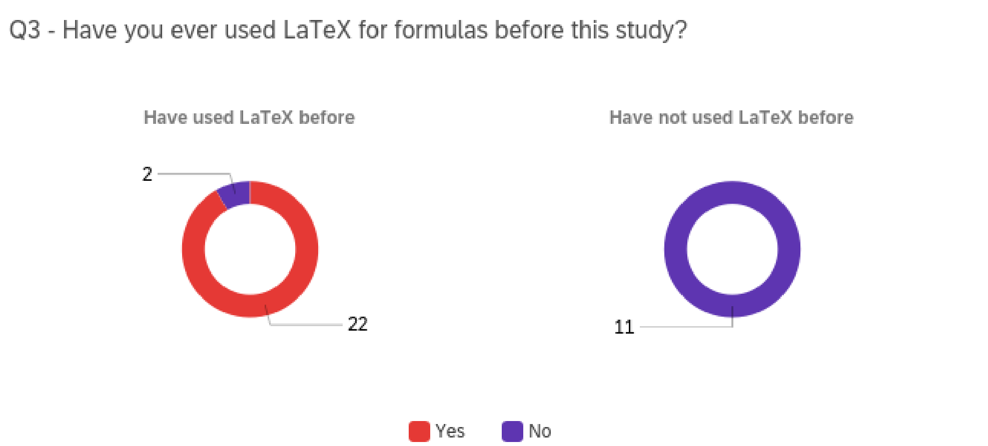
Q1 - How difficult were the provided formula editing tasks?

A.3.2 Question 2 - Have you ever used LaTeX before this study?

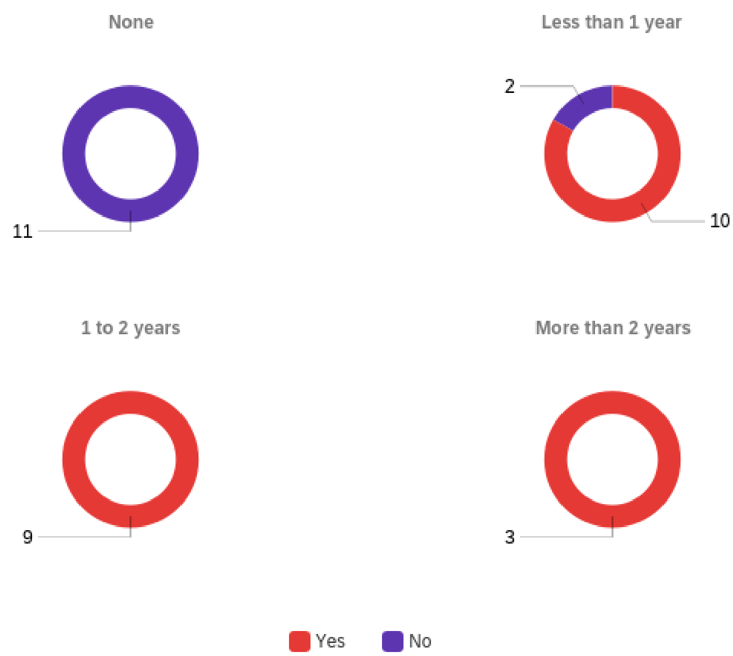


A.3.3 Question 3 - Have you ever used LaTeX for formulas before this study?

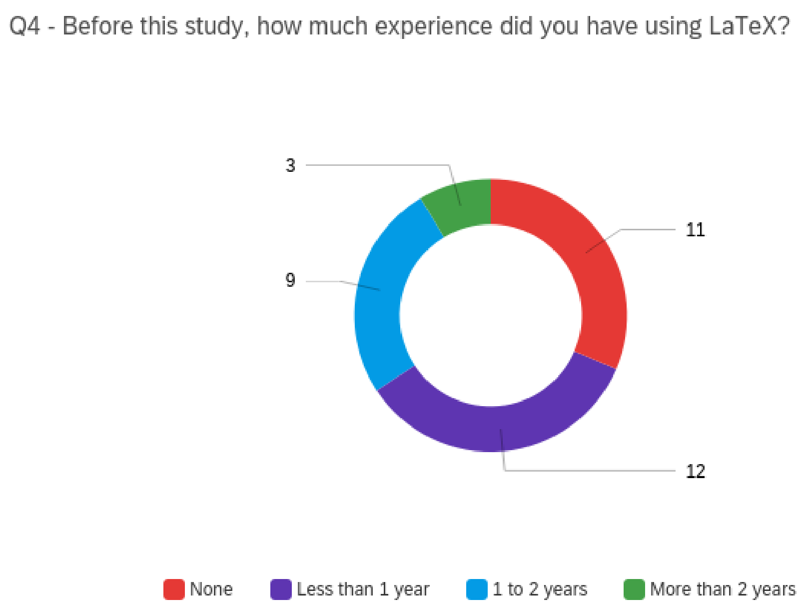




Q3 - Have you ever used LaTeX for formulas before this study?



A.3.4 Question 4 - Before this study, how much experience did you have using LaTeX?



A.3.5 Question 5 - Please describe the strategies you used to edit the starting formula into the target formula.

Please describe the strategies you used to edit the starting formula into the target formula.

Responses by: Have you ever used LaTeX for formulas before this study? (Yes)

```
"Mostly LaTeX, the Visual editing was a little off.  Used the visual
only for getting the math symbols.  Rest I edited using the visual-inbuilt
LaTeX editor"
   "If it was already in the formula I would shift the code around.
The drag and drop only was more annoying than helpful most of the
time with a few exceptions.  "
   "At 1st I tried typing it down like in latex.  But then as we moved
further I realized it was much more convenient to visually edit it
than type it in latex."
   "I would begin by looking for sections of the formula that could
be used in the target, and then seeing what needed to be done for
```

them. If sections needed to be deleted, I would usually do that with the gui. If small changes needed to be made, like switching a symbol or a copy/paste, normally I would also do that with the gui. When it came to entering larger blocks, it usually made more sense to just type everything that I could, whether that was in the box on the right or in the gui"

"I relied on latex more in the beginning, a lot of the time clicking on different elements of the formula to open up the latex editor. As I gained experience with the visual editor, I used lift a lot to move things around, in conjunction with clicking on components to add or subtract additional, basic components. I used latex commands I was familiar with a lot (e.g., \frac, _, ^, all the greek letters) but found myself turning to the list of symbols at the bottom so I didn't have to look up the commands for symbols I didn't know. "

"I first looked for places where I could drag-and-drop things into place, then looked for easy replacements. I tried to use the panel to introduce symbols, which came in handy for some that I didn't know. Modifying existing symbols, esp. adding super- and sub-scripts and overbars, I did using the panel. Making fractions and anything too complicated I did in LaTeX. Sometimes there were things that were easy to copy/paste in LaTeX."

"I first used the text input solely because that was what I was most comfortable with as a result of y time using matlab and Latex. Towards the end I got more used to this program so I tried to use this programs copy feature and the delete features. Towards the end I was trying to find shortcuts. I feel like for big equations I found myself using the visual features more because I didn't want to have to go through the struggle of finding my placement in the text."

"Copy-paste from the already given formula. "

"If I was deleting anything and had the visual option, I would select the area and delete. If I was adding an element that was already there, I would copy paste it. If I was moving things around, I would start at the left and work my way to the right."

"I first searched for patterns in the target formula that were already present in the starting formula, and moved them if necessary (generally copy-paste in LaTeX when available)

I then got rid of any extraneous characters

I then added in any necessary extras"

"For the formulas where I knew terms, it was easy to type those in latex.. deleting a symbol was easy on visual.. adding symbol to left and right or sub/super was okay but latex felt quick and easy"

"I used a mix of both LaTeX and the interface However, if I were more knowledgable in what the symbols were called I would likely do everything in LaTeX and fix mistakes with the interface"

"I much preffered the visual"

"Practice videos were very helpful and understandable. The interface also is very easy to access."

"If I wanted to delete/alter things at the very beginning or end of a formula, I often chose to use latex. However, to edit things in the middle, it was often easier to use the visual editing."

"I looked up specific syntax for the equation I need to complete the formula "

"Checked which option was available. It's easiest with both visual and LaTeX available. I compared the current formula to the target and moved around things as needed. There were symbols that I was not familiar with and that took the longest time. "

"I would try and use the LaTeX part as much as I could, however if I could not remember the name of a given Greek letter, inserting it in the visual part using the icons below proved helpful."

"For the LaTeX formulas, I cut/copied and pasted when I could. For the Visual formulas, I used the drag and drop to rearrange elements when I could."

"Usually tried to circumvent the GUI, and just overwrite the LaTeX source wherever possible."

"My style of formula editing involves copy and pasting wherever I can, even if it eventually means more work for me. For some of the formulas, using the built in editor was easier, as I could just drag over what I wanted to delete or copy, and paste it elsewhere (very useful for fractions). However, if there was multiple copy and pastes that need to be done, especially if it was one copy into multiple pastes, using the LaTeX editor was significantly easier."

Responses by: Have you ever used LaTeX for formulas before this study? (No)

"lots of removing, moving, and writing/dragging from scratch"

"At first, I tried to keep the identical symbols as it is and only modified or added any additional symbols required. Then at times I also edited the formula directly in latex."

"The LaText functionality reminded me of using Desmos, so if there were any letters, I'd use LaTex. But if there was a large amount that needed to be deleted, I'd select and drag over the stuff in the visual space and then delete it.

If there was a subscript, I'd use the visual dots where I'd click on the center blue dot, then delete what's in the text box, then I'd select insert sub I think and then I'd go to the symbols or type whatever was needed, then finally I pressed enter I think."

"Use visual whenever possible, if visual was disabled I would use type"

"Tried to find similar patterns from the provided initial formulas and adapted to them..mostly preferred the visual editing than latex."

"I pretty much tried my best to duplicate the target formula"

"Combining both the visual and the text editors. Some features are easier with one."

"Go left to right along the formula looking for differences between the starting formula and the target formula, making the necessary changes to each part as I went."

"I watched the videos and figured out how to insert different things or delete stuff."

"I started by replacing any existing letters\numbers\symbols that were incorrect with the correct one, then removing superfluous operations, then finally adding in the needed operations. All from left to right"

"It was a series of pattern recognition based on the starting formula's code to figure out which code created what pieces, and from there it was simple manipulations of that base to get the target formula, utilizing the visual database of symbols when needed."

"Depending on the complexity of the LaTeX (the way the formula was written out on the right side), I would choose one of a few things. If the formula was relatively simple, i.e. not having many curly braces or complex parts like fractions, I would start by deleting it in LaTeX, then depending on the complexity of what I needed to add, either directly type it in (for very simple), type in then add special characters on the visual part (for less simple), or just add elements on the visual half (for very complex, fractions, or only greek symbols). If the starting formula was more complex, I would try to highlight and delete whatever I could in the visual part, to avoid having to figure out where specific parts started or ended in the text field."

"I compared every part of the formulas that mismatched and then focused on those areas one by one."

A.3.6 Question 6 - Please provide any additional comments that you have below.

Q6 - Please provide any additional comments that you have below.

Responses by: Have you ever used LaTeX for formulas before this study? (Yes)

"Drag and drop feature was not very intuitive, because of keyboard use, latex editor seemed the easiest and fastest way. "

"The multiple fraction box was too small to see what was being asked. "

"The tutorials before the task were very helpful in solving the tasks"

"When I wasn't allowed to use the the text editing, occasionally there would be an empty \underset which I couldn't figure out how to get rid of. Usually I would have to do some jank re ordering to get it to go away."

"For the task, I forget which, that used \ddots, (the one that had b_0, b_1, etc) it looked to me (something about the way it showed up on my computer? There was a small white rectangle at the bottom of the formula obscuring some details) like \ddots only had 2 diagonal dots, which left me very confused until I tried it with the 3 dots and realized the rectangle was blocking out the last dot. "

"I feel like I f I had a couple more questions I could definitely see myself becoming more fluent with something like this. I am also much more adept with matlab so it took me a second to remember the the proper notation for Latex. I also feel like the visual symbols for me to pick and choose helps me here because if I were trying to cite another mathematical equation (which uses lesser known greek letters) onto latex then I would have to go through the struggle of identifying the letter and then I would have to find the appropriate command to print that symbol onto latex. This program definitely helped me to get through notations I wasn't confident with."

"This was very user-friendly. My only issue was sometimes when I selected anything from the previously written syntax from the 'My Formula' tab, I was not able to edit it. It just showed it within a blue circular box. I don't know if it was intentional. For example, if I selected f(x), f(x) would appear within a blue circular box that could not be edited. except for the option of copying it or deleting it."

" 1. Backspacce goes back to the previous page, I thought it would delete whatever was selected. 2. delete button doesnt delete either. 3. sometimes right click works sometimes it doesnt. When I copied something visually, the right click would give options like copy and one other thing. But in general I could not right click on the formulas, I had to click on the dots to bring up the menu. 4. I could not find a way to do overhead arrow indicating vector in the symbols list."

"greek letters were sometimes difficult to search for, especially ones where I knew the symbol but not the name of it"

"Copying symbols and pasting them on visual felt difficult. "

"A find and replace feature would help out a lot with some of the more complicated formulas

Delete button should work to delete characters

Right button click should work to open the selection menu on the characters

Color code or some more meaningful differentiation between the popup selection menu after a character is clicked

Defining why sometimes, when a character is clicked, it highlights the character in a box with one central dot, and other times the character is highlighted with multiple dots

No noticeable zoom function for if the equation got too long

As far as the study itself, the hints were only useful in the first half, and it was difficult to do everything and then try and remember all my critiques without writing them down. Suggest giving a section then asking questions."

"I think it would be useful if I could use my delete key on my keyboard to delete whatever I have selected instead of having to right click and than press the delete option. Also I couldn't find the method to make fractions without latex"

"Very good interface and easy to access difficult formula"

"I really like the visual editing for editing the middle parts of very messy formulae like the last one!"

"The visual was very difficult than the text. "

"The visual aspect was really nice but it can get a little hard when there's a long formula and the text become small. Dragging things around and seeing which corner of the other text it will attach to gets difficult. It might be nice to provide an option to change the screen sizes of the text and formula field so that for people who are more accustomed to LaTeX, there's a lot of room to see. I wish there was a way to undo more. I tried to undo and it only went up to a few steps back. If undo worked such that it reverted all the way to the original unedited formula, that would be best. It would have been easier to follow the instructions if there was some voiceover or text explaining what it's doing. It would be nice if you can double click to change the text in the visual view (instead of actually having to click the icon in the center)."

"When lifting and moving parts of the formula in the visual part, sometimes the lining of the box the part of the formula I was moving would make it difficult to see and tell where precisely I was placing it, which made it annoying when trying to sub or superscript it."

"I found deleting and copy/pasting to be more tedious in the visual editor than the LaTeX one, since I am more used to using keyboard buttons to do that rather than right-clicking on what I want to edit. However, it was much easier with the visual editor when I could drag a symbol or expression from the provided symbol bar directly onto the equation. It was also easier not to have to worry about keeping track of brackets in the visual editor."

"GUI is very slow and clunky, but it's a formula editor, and that's a universal trait for formula editors. Misses expected keyboard shortcuts like Delete, which would have been helpful at times. I'm not your target audience, though, as I'll continue writing LaTeX in Emacs with a keyboard decades after everybody else is controlling their computers with brain scanners.

The example with nested fractions was very frustrating. Wasn't clear what it wanted from the small thumbnail. Spent about 10 minutes fidgeting with it, thinking there was something wrong with where I put the $\ddots$, until I realized the LHS of the equation needed to be deleted."

"One thing I noticed was the lack of non-left click support in the built in editor. I found myself trying to use the delete/backspace key to try and remove whatever was highlighted. I also tried right clicking in an attempt to bring up the central click menu. I don't recall trying, but it would be great if Ctrl+C and Ctrl+V copied and pasted whatever was highlighted. If these features are added, it would greatly improve the workflow.

Another thing I noticed, which was minor, was the contextual dots around symbols sometimes didn't show up. Not sure why, but sometimes the dots around the symbol didn't appear, so I couldn't click the rightmost dot and add some more text. In the case of the minus symbol, the dots appeared, but they were so vertically close together that it was hard to place anything to the left or right. Would definitely recommend having a fixed spacing on the dots, regardless of the symbol. One last thing was that when dragging a floating equation (which is a feature that I absolutely love), it should become semi-transparent, so one can see dots underneath and position it without having to reposition the floating equation first.

Overall, it was a fantastic app, and I look forward to any improvements made in the future!"

Responses by: Have you ever used LaTeX for formulas before this study? (No)

"overall interactive method was more user friendly"

"I found the tool very user friendly and fun to use."

"When using the visual option, and I click on the blue dot in the center, a long vertical window appears, but sometimes the window COVERS some of the symbols I need, since it's long vertically. So I had to scroll up a little and there was just a sliver of the symbol I needed to press, so maybe if there was some way to adjust the width or ""resize"" the ""space"" that holds all the symbol buttons. Like in the software ""IntelliJ"" or ""Pycharm"", there are multiple ""sub-windows"" ingrained in the ""main"" window, but the smaller ""sub-windows"" are resizable."

"At some points dragging to the dots was too hard for example the subtraction sign was very hard to drag something to the middle"

"after I lifted or copied a symbol and put it in the wrong place it was hard to correct that error"

"Something like this would be beneficial for my math classes."

"Some small suggestions for improving Visual:

-Implement standard keyboard hotkeys for cut, copy, paste, and delete. -Implement a reverse symbol lookup, where the user can drag a symbol into the search bar (without disrupting the formula) and receive the same symbol below."

"It was a bit confusing, I have never used this software before so I wasn't expecting to know what to do. Overall, it was a good thing to try and I learned some new skills today. "

"I found that the visual display was easier to use for replacing and deleting existing information, while the LaTeX tool was easier to use for inserting data"

"That was fun!"

"Sometimes it felt like the UI bugged out whenever I accidentally clicked/click-dragged on a certain part, and I would usually hit undo whenever that happened, until it fixed."

"This was fun! I feel like the visual part was easier when needing to delete or move big parts but I definitely prefer typing it all out."