User-Centered Design of Automatic and Rapidly Generated Tangible Visualizations of Coordinate Spaces for the Visually Impaired

Craig Brown
University of Maryland Baltimore County
1000 Hilltop Circle, Baltimore, MD 21250
cbrown16@gwmail.gwu.edu

Amy Hurst
University of Maryland Baltimore County
1000 Hilltop Circle, Baltimore, MD 21250
amyhurst@umbc.edu

NOTE: THIS PAPER IS AN UNPUBLISHED, ROUGH DRAFT

ABSTRACT
Visual mathematical concepts have long been challenging to access for people with little to no vision. Given that these visualizations are typically authored for the seeing public, there do not exist many fast and easy solutions for low vision people to interpret data in this format. With a number of powerful, low cost technologies beginning to hit the market, we developed software which could leverage these technologies to rapidly and automatically generate tangible derivatives of these visualizations. In this paper, we describe our initial theories in developing this software, as well as the user-centered approach we took to refining it.

Author Keywords
Keywords are your own designated keywords.

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous. This is just an example, please use the correct category and subject descriptors for your submission. The ACM Computing Classification Scheme: http://www.acm.org/class/1998/

General Terms
Design, Economics, Reliability, Human Factors, Standardization, Legal Aspects.

INTRODUCTION
Visualization is at the heart of building the initial understanding of a number of mathematical concepts. For low and no vision individuals, the visualization problem is an additional major problem, inherent in this task. Some systems, such as Nemeth braille, have been developed as an attempt to tangibly visualize certain concepts; in this case, handling arithmetic problems and equations. Greater challenges arise, however, when handling concepts that are not easily represented by alphanumeric characters and symbols.

To demonstrate with an example, consider the basic form for describing a linear function as the relationship between its variables, where y = m*x + b. for those of us who learned it, we know that "for any point y in the equation, we can find its value by taking the corresponding value of x, multiplying it by the slope of the line, and finally adding the the point where the line intersects with the y axis." But what does all this mean?

It is usually at some point around here where an instructor will introduce his or her students to the visual concepts of this mathematical relationship. Students can see and comprehend the "rise over run" characteristic of a line's slope by counting the rise on the y axis and the run on the x axis, and computing the ratio. Additionally, a student can compare two separate slopes by looking at the plot of two different lines. Even more concepts emerge from this relationship when you start talking about lines which are parallel, perpendicular, or inverse, and all of these linear relationships have distinct visual representations which enhance our understanding. Naturally, a similar scenario unfolds when considering distinct representations of the y-intercept.

It is clear that an immediate problem emerges from an educational standpoint, when making the connection between the numerical representation of equations and data plots, and the their corresponding visual representation. But we also see this problem persist past initial phase of comprehension. Graphs have become an integral part of our means for interpreting data in our everyday lives. In politics, news publishers might use a graph to illustrate trends in public sentiment on policy issues over time. In economics, professionals might consult financial news agencies or government websites which use visualizations to demonstrate the latest development on economic trends and business cycles.

Any potential solution to these challenges is by its nature,

TEI 2012, Feb 19-22, 2012, Kingston, Ontario, Canada.

both a hardware problem and a software problem. On the software end, we need to develop technology which can automatically generate these objects from minimal user input. On the hardware end, we need technology which can reliably communicate with our software solution, while producing fast, cheap, and satisfactory products for our end users. In the past, some researchers have attempted to go at the hardware end of this problem by using technologies such as braille embossers, haptic devices, and more. The community, however, has found such solutions to high in both cost and user overhead, while also being limited in versatility and scaleability. We suggest using new technologies such as laser cutters and 3D printers (see Figure 1) to both minimize on the cost and maximize on the potential uses and benefits.

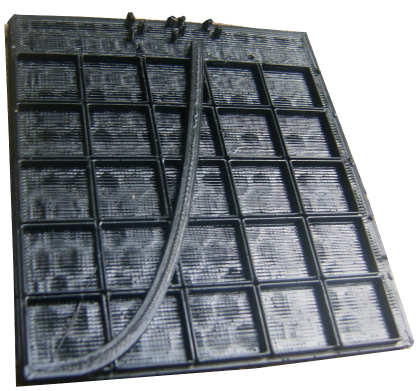

Figure 1. Laser printed prototype of the first quadrant of the basic cubic function.

The structure of this paper will begin by taking a more detailed look at related work in the area, while examining important lessons learned from technological capabilities and limitations. After that we will discuss both the hardware and the software of our alternative solutions, as well as the initial design decisions we made during development. Next, we take a look at our user-centered design process and what benefits we got from rapid prototyping approach. We conclude with topics for future work, and acknowledgments.

RELATED WORK

Similar existing technologies can be generally classified under the three categories of tangible devices, haptics, and audio devices. We examine each category below.

Tangible Devices

The most popular existing solution would probably be the use of braille embossers. Embossers are essentially, tactile printers which can impress dots and lines upon a piece of malleable plastic paper to form braille lettering in addition to tactile approximations to graphics. Some of the major limitations to such designs have been seen when trying to distinguish the grid lines from the lines of the functions [1]. Reasons for this include that the fact that while some embossers can emboss up to seven levels of height, all seven levels cannot be easily distinguished form one another. Also, the fact that braille like dots were used as the texture to represent both line and grid left room for confusing the two entities. There were also a number of alternative solutions suggested, such a underlaying the grid on a separate sheet of embossed paper or using different materials to give different textures to grid lines and function lines. The underlay solution proved to be difficult for users to adapt to, while the additional materials solution add a significant amount of overhead in the assembly of the graphs.

A similar software approach using braille printers attempted to use a mixture of image processing techniques and human intervention to make tactile graphic translations [2]. An advantage of this approach is that complicated diagrams can realistically be translated into tactile derivatives. Also, researchers estimated that on average, only 15 minutes would be needed to translate any one graph. The major drawbacks however come from the fact that highly skilled human intervention would be needed to aid the process, as well as the fact that the technique uses very expensive braille printers which run upwards of $2000. Additionally, users themselves do not have the ability to customize their content or generate it one the fly, as all manufacturing would have to be done by skilled professionals using a number of software applications such as Photoshop, in addition to the expensive braille printers.

Thirdly, there have been cork board solutions produced, most notably the perhaps is the product from the American Printing House for the Blind [3]. This solution proposes using a combination of rubber bands and thumb tacks to build graphs on a physical cork board. Immediate limitations are the type of graphs which can be produced (it seems to work best with standard bar graphs) as well as the significant amount of user overhead which must be employed to produce such representations. There is also no way to represent large and complicated data sets from, say, excel files.

Haptics

Attempts have been made to use Phantom Omni haptic pens to trace the trends of lines and functions represented on a graph. One approach attempted to use different frictional feedback cues to indicate the presence of multiple lines on a single graph [4]. Users, however, found this difficult to distinguish, as they would often confuse the different cues for signaling the different slopes on a single line, as opposed to indicating the presence of two separate lines. Additionally, users of this haptic device had s difficult time tracing the shape of the graph, as there were many instances (particularly at the corners) where the haptic pen would slip off of the path unexpectedly. As a general concern, like other approaches, this technique also requires significant user training.

Audio

Some attempts have also been made to integrate tactile methods with audio cues. In one project, users can build their own electronic data sets by placing markers into plastic cells aligned across a clear table with a web camera underneath [5]. Users can then slide a special marker across the bottom of the table, which serves to both calculate the values of data points on the graph, as well as signaling their relative value in the y direction with an appropriately attuned musical note. The difficulty with this approach is that it does not easily translate existing data sets, as any representation of any graph first requires a user to build the appropriate plot with the markers on the plastic grid. Also, because the location of data points is limited to the construct of the plastic grid cells, it is impossible to represent complex data sets with tens or even hundreds of data points. Similarly, we see that this approach is very much limited to simple bar graphs and simple line graphs, as an attempt to plot complicated functional forms such as cubic or logarithmic would immediately present as an impossible task.

EXPLOITING A NEW GENERATION IN RAPID PROTOTYPING

Given the many limitations of existing solutions, we propose a new system which will streamline the process on both the software and hardware side of the tangible graphing issue.

A Recap of Existing Limitations

Lets again look at some of the major limitations of existing solutions. Braille printers, while they can print up to seven different levels of height, can only print to a degree where the differences in height are difficult to distinguish from one another. They also tend to represent both text and lines as a series of raised dots, which limit the type of textures they can produce. There might also be concerns about the sturdiness of any particular embossed plastic sheet, as sheets need to be thin enough to emboss.

On the other hand, the performance of haptic devices can be somewhat unpredictable, as the use of a haptic pen leaves a lot of room for inaccurate interpretations of the data series at hand.

Audio technologies provide some assistance to low vision users, but they still require a significant amount of support of users themselves who must still build the graphs with markers. Additionally, the set up rig is cumbersome and complicated, and virtually impossible for a blind individual to assemble.

All of these solutions provide no easy way for blind users to interact with complicated data sets or functional forms on the fly. Set up of these systems is often cumbersome, and requires significant assistance for third party users to can both help them assemble the devices as well as help instruct them as to the particular use. There is an additional issue of standardization, as different graph types sometimes require radically different set ups, as is the case with trying to represent different scales on the axes and different functional forms.

3D Printers and Laser Cutters

A new generation of rapid prototyping has arisen from the introduction of 3D printers and laser cutters into the market place. We explored using both technologies to generate different concepts for tangible graphs. In this particular study we worked with the Makerbot 3D printer, which is currently a consumer level 3D printer and the most affordable of its kind on the market. For laser cutters, we used something a little more industrial strength in Universal Laser System's VLS3.60.

The Makerbot provided the exciting potential or home users to be able to make and print their own tangible graphs. At $1,299, the Makerbot comes in at less than most braille printers. As a relatively new technology, one can also assume that the pricing of the device will follow a Moore's Law type curve, getting cheaper and cheaper as the device becomes more powerful and as its demand in the market becomes more numerous. Other benefits of the Makerbot include the that it can produce printed structures relatively quickly, and in our case we found that it could print a quadrant of a graph in about an hour. The Makerbot also has the benefit that it can easily communicate with electronic data via a very well defined CAD (Computer Aided Design) pipeline for producing 3D geometry.

The VLS3.60 more expensive than the Makerbot, and requires more professional assistance to assemble than the Makerbot, which is designed for self assembly at home. While it may be less realistic to expect users to own and operate their own laser cutters, the laser cutter still provides a number of manufacturing benefits. Laser cutters typically operate very quickly. None of the prototypes we created in house took more than 30 minutes to produce, at fairly large sizes. Also, laser cutters can produce etchings and cuts on a number of cheap materials such as plastic, hardboard, cardboard, wood, and even paper. For a software perspective, laser cut images are also easy to produce, as all the user needs to create are simple SVG (Scaleable Vector Graphic) images for the software to read.

Our Automatic Software System

We developed a software system to integrate three separate major open source projects, as a means to consistently produce deliverables for our 3D printer and our laser cutter. This involved developing a python script which would allow all three programs to communicate with each other effectively, across a necessary multitude of intermediate file types.

The three software packages we integrated were Veusz, Inkscape, and OpenSCAD, with some intermediary assistance also coming form pstoedit. Veusz was leveraged as the first step in our pipeline (see Figure 2) which allows us to take any function or data set and plot it in two dimensions. After plotting the function and computing a few logical transformations in Veusz, we then move on to Inkscape which takes the .svg file exported from Veusz, and performs further transformations in order to clean up troublesome white space, eliminate arbitrary object groups, and convert all polyline objects to simple paths, which can be read by OpenSCAD. Once Inkscape produces this simplified .svg file format, it then converts that file to a .ps (Postscript) file type, which is an intermediary file type for reading by OpenSCAD. At this point, we use pstoedit as an intermediary take this .ps file and convert it to a .dxf (Desktop Cutting Plotter) file type, is a filetype that can be read by OpenSCAD. What .dxf files do is trace all of the path object types in our original file, to outlines which can be interpreted by OpenSCAD and extruded into 3D geometry.

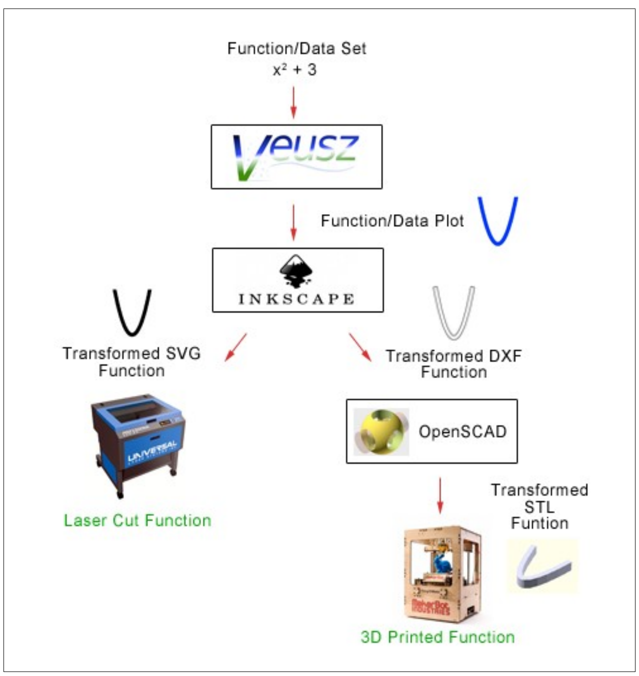

Figure 2. Algorithmic pipeline of our automatic software system.

At this point, we also wrote a few scripts in the OpenSCAD language in order to create the supporting geometry of our graphs (grid, axis numbers, graph titles). The OpenSCAD code is ultimately the code which writes a .stl (Stereolithography) file that is printable through the Makerbot software.

If one wanted to use a laser cutter rather than 3D printer, the process would simply stop with Inkscape saving to a .svg filetype and exporting the file the VLS3.60 cutting software. Future user studies however, left us to leave this functionality our of the final program.

Additional Features of Our Software System

All of the functionality of the previous section is integrated by a stand alone python script. Python was used as it is the robust scripting language, which also happened to be the chief scripting language of Veusz, which allows programs to manipulate the program through a series of libraries implemented in python.

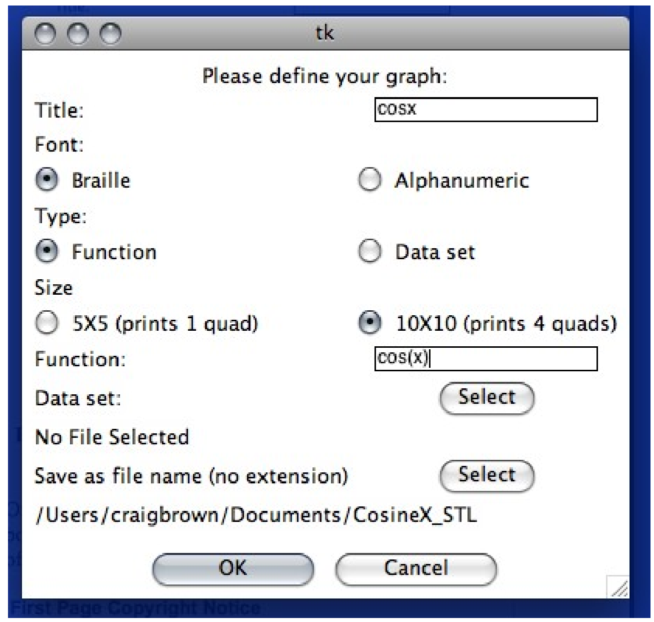

Figure 3. Simple user interface of our software system.

Our python program starts by opening up a simple dialog box (see Figure 3) which enables users to define the details of their graph, including the graph title, font (braille or alphanumeric), whether to plot a function or a data set, size (full graph or first quadrant), data set file of type .csv (Comma Separated Values), and a final destination and name with which to save you .stl file. We will examine the choice between fonts in our next section, but for now, lets examine a few of the other items more closely. All titles are required to consist of a maximum or four characters, containing only numbers and/or letters. These titles are meant mostly as a means for distinguishing multiple prints from one another, rather than being actual full titles. The reason for this is that the printing platform of the Makerbot limits the size to which we can print (roughly 3.5 x 3.5 inches), which also limits the space available for labeling the graph. We are additionally constrained to printing full graphs to four separate quadrants, as trying to print to a single plastic sheet would mean dramatically scaling the size of our grid elements down to a point where they would be difficult to interpret. Thus, we chose to label each quadrant of the grid with the appropriate quadrant number, in addition to labeling it with a unique title.

As a research decision, we decided to limit users to 5x5 quadrants, as this provided a reasonable size for interpretation, as well as a useful standard for rapid prototyping.

USER-CENTERED DESIGN PROCESS AND FEEDBACK

In order to assess the quality of our design, we interviewed a small number of blind and low vision participants, questioning them on their general usage habits with assistive technology, as well as their opinions on a number of design prototypes of our tangible graphs.

Demographic Information

Our study consisted of 3 participants. All three were college educated, and currently professionals working in industry. Participant 1 was a female economist and lawyer at a large government agency dealing with securities law, Participant 2 was a male adaptive technology specialist at that same firm, and Participant 3 was a female and a former psychological therapist. Participant 1 started losing her vision at the age of 12, and is now completely blind. She only maintains a partial use of braille. Participant 2 Also started losing their vision after later in life, and is now low vision, despite being considered legally blind. Participant two does not use braille. Participant three is also currently completely blind, and a braille user. A summer of this demographic information is found in Table 1.

Participant	**1**	**2**	**3**
Gender	Female	Male	Female
Age	35	63	60
Age of sight loss	12	38	31
Degree of sight	Blind	Low Vision	Blind
Braille User?	Minimal	No	Minimal
Profession	Economist/Law	Technologist	Therapist
College	Ph.D/JD	Masters	Masters

Table 1. Summary of participant demographic information.

Findings

Participants were asked to give their opinions on a number of different prototypes. We first started with a laser cut prototype of a grid on hardboard, which contained no function, but did contain a number of different textures for distinguishing the different features of the grid itself (see Figure 4). The reception to this was quite unenthusiastic, as the etched textures proved to be very difficult to distinguish when they were first tried with Participant 1. It was nearly impossible for her to distinguish any of the various features without a significant amount of help from a sighted observer. This effectively made all of the different textures used meaning less, and any texture on the etched board was difficult to distinguish. The most sensibility she was able to derive from it were the simple gridlines on the right side of the board. Participant 1 also noted trouble distinguish the texture of the x and y axis, saying that although they were similar, because of orientation and the *direction* in which she had to slide her finger to count the coordinates, it was in fact easier to feel the texture of the x axis than it was to feel the texture of the y axis. Furthermore, in locating a value on the x axis, she would subsequently get lost when she tried to back track and find the y axis. Because of the ill reception of the hardboard grid, it was subsequently left off the table as a realistic design specification and out of our final software distribution.

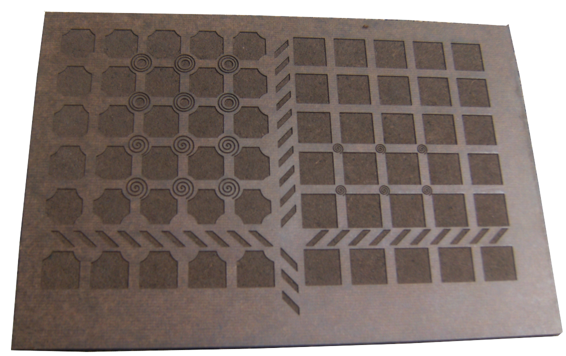

Figure 4. A picture of the laser etched, hardboard grid prototype.

There are a few thing that the failure of the hardboard led us to think about. In particular, we realized that because of the major constraints on the height of the textures, the solution of etching a graph did not really have any major advantages over existing solutions which use braille printers. In fact, etched braille dots are indeed much easier to feel than continuous, etched stretches of hardboard.

Far better results followed from the testing of the 3D printed plastic prototypes. Four particular prototypes were featured. The first was the graph of the basic quadratic function, similar to the cubic function pictured in Figure 1. The difference was that this particular iteration had braille numbers printed onto it. In this sense, it was more similar to the second prototype showed in Figure 5, which shows a data set graph, plotting the US unemployment rate over the last 4.5 years, from January 2007, to June 2011. The third prototype was a graph with two separate data sets plotted at different heights. The last prototype was three pieces; the first two quadrants of the graph of a sine function, as well as a laser cut piece of hardboard, with enough space cut from the middle to hold the two quadrants snugly together.

With minimal instruction (we told her the location of the origin), Participant 1 was able to determine the shape of the first function, correctly identifying it as a parabola. Similar success followed with the graph of the unemployment rate, as Participant 1 was able to follow the shape of the curve with a high degree of detail, correctly following the various rises and dips with her fingers. She was also able to follow minute details, such as finding the vertex of the graph parabolic graph, where its derivative is equal to zero. Participants 2 and 3 had similar success with both plastic graphs, though they were less familiar with mathematics, so they required a little more third party assistance. It is worth noting that all three participants noted that the presence of the grid was a bit of a hinderance, as it was an additional and confusing detail to follow when trying to interpret the function it self. In fact they all suggested that it would be better to either lower the height of the grid lines even further, so that both the braille and the function stood out more than they already did, or to simply eliminate the complete lines, in favor of major tick marks. We were abel to have Participant 3 test one such grid with lowered grid lines, and she reported a significantly better experience with distinguishing the braille, the gris, and the function. Also important is that all of these users already had some familiarity with coordinate spaces, and they both independently suggested that the grid lines might be a useful addition for a younger audience who might be learning mathematics for the first time.

A similarity between Participant 1 and Participant 2 is that they both expressed a concern with the use of braille numbering. Although Participant 1 was completely blind, she explained that with the advent of audible technologies such as screen readers, smaller portions of the blind population were dependent on braille. Participant 2 who was low vision, added the fact that most blind individuals start going blind after the age of 18, and so learning braille is like learning a whole new language, which makes audible solutions are more attractive. Both Participant 1 and participant 2 suggested the use of alphanumeric characters, even as a tactile alternative. These sentiments led us to include the option in our future software designs.

Participant 3 was able to examine prototype three, which contained two line graphs at different heights. She reported being able to follow both the first and second lines from their respective starting points, all the way until she encountered the intersection. At the intersection point, she was able to determine the change in height from line one up to line two, however she struggled to return back down to line one after crossing the intersection, instead following along line two as if it were the continuation of line one. At this point, she suggested increasing the difference in height between one line and the next, or possibly printing one of the lines with a different texture, such as a dotted texture.

Figure 5. A picture of the 3D printed graph of US unemployment from Jan 2007 to June 2011. Braille lettering is shown on the y-axis, however the small size of this particular print caused some of the pieces on both axes to break off.

When asked about the potential for printing full graphs on four quadrants, both Participant 1 and Participant 2 expressed concerns about simplicity. They stressed that whichever mechanism was employed to align these four quadrants, it was imperative that attention be paid to orientation issues (being able to distinguish quadrant 1 from quadrant 2 and so on), and overall simplicity of design. We took this a general warning against including any unnecessary additional parts for assemblage. Our current solution was briefly discussed earlier, where the software will automatically print the number of the quadrant a the top (for quadrants 1 and 2) or bottom (for three and four) of the quadrant, along side the title of the graph. The only additional piece for assembly would be a piece of hardboard wit ht he total size of the 4 quadrants cut out form the middle, so that participants could snugly fit each quadrant together and make a full graph. Participant 3, who actually was able to test our multiple quadrant solution in the fourth prototype, showed very promising comprehension. Again, with minimal assistance, she was able to determine the braille quadrant numbers, and place them into the correct position in the hardboard graph holder. Upon placing them into the hardboard holder, Participant 3 had little trouble navigating both the positive and negative sides of the two quadrant graph.

Some interesting suggestions came from our participants as well. Participant 1 suggested an intelligence feature, which automatically computed important features of the graph. In particular, she suggested that it would be helpful to have the program determine the highest point of unemployment in the unemployment graph, and then print that number next to the point of interest. In this case, Unemployment peaked at 10.2% in 2009, but it was virtually impossible for her to tell what the exact value of the number was. Participant 2 also made an interesting suggestion for the orientation problem, suggesting that you could designate corners by simply clipping the corner opposite the origin. Of course, this still leaves the question of distinguishing the whether or not the clipped corner belongs in, say, the top right position (for quadrant 1) or the bottom left position (for quadrant 3).

CONCLUSIONS AND FUTURE WORK

The results of our study lead us to believe that our software system has moved us towards closing many of the issues raised by previous work in the field of tangible graphs. Despite the many opportunities that have emerged, there are also a number of limitations which must be addressed. We hope to address come of these limitations in future work.

Benefits

Perhaps the greatest benefit from our solution emerges from the its relative ease of use. Previous solutions such as the audible marker solution [5] require users to use complicated setups which are constructed from multiple parts. These parts often have to be assembled and administrated by third party users who are both sighted and knowledgeable of the system. Similarly, haptic devices of themselves are not solutions. Rather, they require users to interface with both computer software and the haptic device any time that they wish to access a graph. Our solution proposes only a one time software interaction any time that a user wishes to access a graph. From then, users only need work with a single piece of plastic for one quadrant graphs. For four quadrant graphs, users interact with the printed quadrants as well as the hardboard graph holder. We see the elimination of electronic devices as potentially less cognitive load than the many pieces of the audible marker solution and the haptic solution.

Another benefit is cost. Consumer level braille printers and haptic devices run in excess of $2000. The Makerbot currently goes for about $1,300, making it among the cheapest solutions on the market.

Also unique to our solution is the ability to extrude different features of our graphs to various levels of height. As mentioned earlier, braille printers have this feature, but they struggle to achieve levels which are easily discernible from one another. In our case, Participant 3 mentioned that even more of a height differential for multi-line graphs might help discern the two lines. This is a request which we can easily process and deliver, given the flexibility of 3D printers.

One might also consider the durability of the hard plastic pieces we printed to be beneficial. Braille embossers typically print to very thin sheets of plastic which an easily wear or bend. Similarly, because we utilize to few physical pieces, one can see how our solution might be easier to keep track of for blind users, who might be less worried about losing pieces and so fourth.

Practical Applications

We believe our solution can offer a meaningful solution to the future instruction of mathematics for individuals with visual impairments. With Makerbot 3D printers currently available as consumer level devices, our technology can readily be seen as useful for both parents who home school their children, and teachers at private and public institutions, who work in the classroom. Aside from merely providing a platform for teachers to plot their own graphs, our solution might also lay the groundwork for translating textbooks across a range of subjects to help make their material more accessible.

Another application of our software might occur in industry. Currently, many professionals who work with mathematical data are constrained in how they can interact with this data when it appears in the news media. Participant 1, who does extensive work with economic data, mentioned that she has trouble accessing graphs which appear in financial news articles. Screen readers such as JAWS, simply do not handle graphs when they are embedded in articles on the web. We believe that our solution can provide a protocol for the authors of these articles to make their work more accessible. As of now, a simple protocol might include requiring authors to provide either the equations which have been graphed, or the excel data sets which were used if no equations were included. Given the simplicity of our system, these two pieces of information would be enough to create tangible derivatives. Such might allow blind and low vision workers and assistive technologists in industry to curate this data for themselves.

Limitations

Perhaps the most substantial limitation of our system resides in the hardware. While the Makerbot provides many benefits, it also comes with a few limitations. Currently, the Makerbot's printing platform only allows us to make prints that are roughly 3.5x3.5 inches in size. We typically look to maximize the size of our prints in this environment, because smaller prints not only result in graphs which are tangibly more difficult to discern, but they often lead to some of the smaller pieces, such as braille lettering, to easily break off. Size is also a problem when titling our prints. Currently, users can only enter titles which are a maximum of 4 alphanumeric characters long. This makes for fairly short title's which cannot capture any real sense of detail. Also stemming from our size limit is the fact that we cannot simply print large four quadrant graphs on one sheet of plastic. This significantly increases the burden on the user, as they have to wait longer for our software to finish processing, keep track of multiple pieces for the same graph, and deal with lining up four disconnected quadrants before they can actually use them. Larger prints might solve many of these physical limitations, however larger prints are currently not possible on the Makerbot's specifications.

Another major limit is speed. Currently, the Makerbot prints one quadrant in roughly an hour. All three of our participants suggested that low wait times would be a significant factor to the usefulness of our solution. In Particular, Participant 1, who works with financial news articles, mentioned that slow print speeds might make our solution to slow to pose any real benefits for her professional purposes. While this problem becomes less of an issue from an instructional standpoint, faster print speeds would greatly increase the usability of our solution for all parties.

Also problematic is the fact that the Makerbot must be self assembled. Again, this mostly stands as an issue to low vision users who would like to have a Makerbot for private use. Some sighted individual would first be needed to complete the assembly of the Makerbot. On a similar note, using the Makerbot itself can be challenging, as it sometimes requires users to first position the print in 3D space so that it correctly sits on the platform. As a result, though our software might function well with traditional screen readers, sighted third parties would still be needed to assemble the Makerbot, install our software, and manipulate the Makerbot for printing.

Future Work

Future work will entail writing additional features to our software. Some features might include adding more customizable features, such as the ability to print either a full grid lines or major tick marks. We also look to include additional functionality which will let users define the range of the both the x- and y-axes. More work might be done with textures, so that graphs with multiple plots will be easier to interpret. Participants also expressed an interest in being notified by some sort of audible cue, which would help them be aware of when our software finishes processing the prints.

Users studies were not conducted to test our software interface for this study. Instead, we only questioned users on the design of our physical prototypes. Future work will include both testing and refinement our our system by having them use our software to try and produce prints of their own.

ACKNOWLEDGMENTS

We thank Jess Martin and Chris Kidd, who worked in the lab during the course of this study, and provided may useful insights and opinions. We also thank the CRA-W and the DREU Program for co-funding our study and making this research possible.

REFERENCES

1. Lederman, S.J., Campbell, J.I. Tangible Graphs for the Blind. *The Human Factors Society 24,*1 (1982), 85-100
2. Ladner, R.E. et al. Automating Tactile Graphics Translation, *Proc. ASSETS '05,* ACM Press (2005), 1-8.
3. Graphic Aid for Mathematics. http://shop.aph.org/webapp/wcs/stores/servlet/Product_Graphic%20Aid%20for%20Mathematics_1-00460-01P_10001_11051
4. Yu, W., Ramloll, R., Brewster, S., Lee, R. Haptic Graphs for Blind Computer Users. In *Proc. First International Workshop on Haptic Human-Computer Interaction 2002*, Springer-Verlag (2001), 1-6.
5. McGookin, D., Robertson, E., Brewster, S. Clutching at Straws: using tangible interaction to provide non-visual access to graphs. In *Proc. 28th International Conference on Human Factors in Computing Sstems 2002*, ACM Press (2010), 1715-1724.