Introduction

Tutors often struggle to communicate mathematical concepts when teaching students remotely. In person, a tutor can easily draw a figure on paper and ask for student input on that same paper. However, current solutions for online tutoring are more opaque, either offering a video connection without shared notes or text editors which don’t permit the diagrams and annotations needed to explain concepts. We designed Impressions to make the tutoring experience more transparent with real-time collaborative notes, annotation, and cursor gestures. When we observed tutor-student interactions after launch, we found that the platform increased transparency and flexibility of expression over existing remote tutoring solutions. Although tutors sometimes underutilized tools and students didn’t always collaborate substantially, overall we found that they had a positive experience with the app.

Prototypes

We developed five prototypes to test different aspects of the remote tutoring experience, from finding tutors to the ease of math communication. We combined results from our final two prototypes, CamCapture and Shared Space, to create Impressions.

Matching Lobby

This prototype tested the hypothesis that tutors are motivated by autonomy and will identify the topics in which they can help students the most. We tested this idea with an anonymous chat lobby (bricolaged using a Discord channel) where both students and tutors can post publicly to each other. Within this channel, tutors were able to see which topics students need help with and can choose to invite a student to a 1-on-1 tutoring session if they are knowledgeable in that particular topic. This idea tested the social dynamics of a community-based tutoring system. Tutors served as natural moderators in order to establish lobby norms.
One takeaway from the matching lobby prototype was a problem of lag time (not real-time). For example, if a student posted that they needed a help on a particular topic, it was unclear how soon and whether that student would be able to receive help from a tutor. The system would need a way to make sure that the student is able to be matched with a tutor in a timely manner, perhaps through incentives such as the tutor getting paid by how fast they respond, or notifications that notify a tutor when a student asks for help. In this prototype, we also found that the audio-based chat provided enough expressiveness to convey mathematical concepts even in the absence of video.

Speech-to-Math

This prototype tested the hypothesis that the biggest roadblocks to remote math explanation are the speed and difficulty of writing complicated equations on a computer. We mocked up a “speech-to-math” interface which converted a user’s spoken equation into text using Google Hangouts and Overleaf. The tutor and student met on a Hangout for a regular tutoring session and shared the Overleaf page. A group member acted as “interpreter” so when either party said an equation after the trigger phrase “begin math”, that equation was written for them in LaTeX in the Overleaf document and visible to all users. Figure 2 shows an example note generated during prototyping.
1 Instructions

To use the speech-to-math system:

- Say "begin math" to start recording
- Speak the equation you would like to write. For example, "sum from x equals one to n of x squared"
- Say "end math" to end recording
- Each equation will be written on a new line in the Latex document

2 Workspace

Prompt:

- Explain what a Markov Decision Process is.
- Given a policy, how do we determine its value?
- How do we use value iteration to determine the optimal policy?

\[ V^{(t)}_s(s) = \sum_{s'} T(s, \pi(s), s')[\text{Reward}(s, \pi(s), s') + \gamma V^{(t-1)}_{\pi_{opt}}(s')] \] (1)

\[ \pi(s) = \text{action at state s} \ t = \text{number of iterations} \]

Figure 2. Speech-to-Math prototype

From testing this interaction in several tutoring sessions, we found that the difficulty of writing math was not the principal barrier to communication. Even when it could be written quickly, LaTeX was insufficiently expressive to capture concepts that tutors wanted to explain by drawing figures or diagrams. To explore how to offer this expressivity over remote connections, we turned to our next prototype.

Tablet Notes

In this prototype, we found that if a student had a tablet, they were able to easily take notes on the tablet in a Khan Academy style explanation. Because the student had a stylus, he was easily able to write down the notes on a OneNote document, share his screen, and explain to the other student the math step by step through a Google Hangout. Since he was using a stylus, he could write as if he were writing on paper and the notes were legible and savable. A similar interaction was simulated using a iPad with the Notability app and screen share functionality.
While tablet notes were effective, many students don’t have devices that support this interaction. One workaround we observed is that tutors will write notes on paper and hold them up to the camera to share. However, these notes are easily forgotten because they are not persistent. We hypothesized that capturing these diagrams and saving them in a space where tutors and students could see and manipulate them would give many of the benefits of tablet notes without the technological cost.

We tested this idea by placing tutors and students on a shared Google Hangout and Google Doc simultaneously. When tutor held up a picture to their camera and said “capture” we took a screenshot of the content and added it to the doc (see Figure 4).
We found that participants appreciated the ability to save their notes but wanted to also modify, add, and point to them on the shared document so they were not just persistent but manipulable. This led us to include a draggable annotation canvas in the final design along with the camera capture feature.

**Shared Space**

In this prototype, we explored different collaborative models of interaction. One model that arose was using a Slack thread model where students and tutors could annotate each others notes. However, we ultimately realized that this rigid structure didn’t allow enough dynamic flexibility that a tutoring session required. Another prototype we attempted was using Google Slides with imported math symbols. However,
this prototype was too brittle and limited in functionality to be useful. Thus, in our design, we ultimately used the shared cursor using Together.js and draggable, handwritten notes to signal where there was confusion since it allowed more flexibility and functionality than either of these prototypes.

![Workspace](image)

**Figure 5. Shared Space Prototype**

**Design**

Our goal with Impressions was to build a more socially transparent, collaborative math online document that allows tutors and students to communicate concepts clearly and efficiently. Using LaTeX text boxes, camera capture with cropping, and downloadable notes, we attempted to establish a “beyond being there” experience that allowed both the tutor and the student to have a neat and organized log of the tutoring session. We focused on features that promoted awareness between the tutor and the student (such as showing each other’s cursors and connecting via audio and chat) while avoiding functionality that would provide excessive transparence or irrelevant information (such as full video or screenshare).

**Implementation**

We built Impressions using React to combine and customize several existing collaboration frameworks. We used Firebase’s Firepad for real-time collaborative text editing. However, because this text editor didn’t support several of the key interactions we desired, including drawing annotations, uploading camera pictures, and math, we built an additional features on top of it using Together.js. Using this system, tutors could invite students to a page via a link where they could see each other’s cursor movements.
and communicate via chat and real-time audio. Integrating the message passing provided by Together.js with react-draggable, we created draggable boxes in which users could write mathematical formulas and draw annotations, which are synced in real time to all participants in the session. We used react-webcam to take camera captures of math written on paper and allow users to crop their images using react-image-crop, then save the cropped image to Firebase. We adapted Firepad's implementation to include a button that allowed users to upload their cropped images from our Firebase. We deployed the final system as a Heroku web app. Source code is publicly available on GitHub.

Deployment

We publicized the launch of the system on Stanford HCI Participants, through several Stanford listservs, and by contacting friends known to be tutors. Because our app aimed to improve existing tutoring sessions, we did not face the cold-start problem that troubles social computing systems relying on network effects. Because the app only required a single tutor and student for a session to be useful, it facilitated and improved existing tutoring interactions. In addition to logging content created in Firebase, we created a Google form asking users about their experience and sat in on three tutoring sessions with the app. We had 83 total visits to the site.

Interaction

In general, we found that tutors used a diverse suite of tools for different problems on the app and appreciated the translucency provided by microphone and pointer communication. Three examples of tutoring sessions we observed are given below, followed by a summary of observations.

Tutoring Session 1:

In this tutoring session, a student explained Bayes’ Theorem to another student. The tutor mostly wrote on the notepad and used the LaTeX math text boxes. The tutor didn’t use the Camera Capture functionality as much to explain the math, although when testing it, she noted that the preview and cropping functionality was very helpful. The tutor noted that the math she was trying to explain was simple enough to easily type out in the notepad and using the LaTeX that she didn’t feel the need to use the Camera Capture component for this tutoring session. She also noted that she liked how the math LaTeX boxes allowed her to also type words with spaces unlike traditional LaTeX. In all, she found it very easy to type the math while explaining the concept in words through a Google Hangout. Impressions allowed both the tutor and the student to save the final notes page and have a record of the conversation.
Figure 6. Tutoring Session 1

**Tutoring Session 2:**

In the second tutoring session we observed, the tutor explained Markov Decision Processes, relying heavily on the annotation functionality to draw diagrams. Although the tutor experimented briefly with math and typing, he fell back on the diagrams as the most flexible tool. He often used his cursor as a pointer to gesture to various parts of the drawing. Although drawing with the mouse was sometimes messy (particularly because there was no erase/delete functionality), the tutor said he preferred it to writing on paper and holding it up to the camera (the CamCapture model).
Tutoring Session 3:

In a third tutoring session that we observed, the tutor was tasked with teaching a student about the idea of summations and converging to the closed-form formula. The nature of the problem made the drawing pad a crucial component in the tutoring process, as the capability to illustrate a triangular array of points to be summed together largely helped with conveying the lesson insight. Feedback for our app was largely positive. Specifically, the easy LaTeX input made writing complex summation formulas simple and straightforward. The ability to drag the math input was a plus as well, since it allowed the formula to serve as annotation for the illustrations. One potential improvement to the app would be to make the drawing pads expandable so that larger illustrations can be made clearer. In addition, different colors for the drawing pad would also have been useful for making distinct lines and figures.
In our prototypes and deployments, we noticed that the Together.js functionality that shared cursor location with the other participant was very important for the tutoring experience and important in creating a socially translucent system. We found that the tutors were able to notice when the student was confused about a certain portion of the lecture based on the student’s cursor location. In several iterations, the student placed a handwritten question mark while the tutor was explaining the material. This presented a “beyond-being-there” experience since the student was able to signal to the tutor his or her confusion without disrupting the flow of the lesson and both the student and tutor knew to return to that portion of the explanation in order to explain the material to the student again.

Another observation was how tutors used the tool suite to adapt their teaching approaches for different problems. For example, when explaining Markov Decision Processes, diagrams were used heavily, while the tutor explaining Bayes’ Rule relied on text and equations.

**Discussion**

In deploying the app, we found that the camera capture portion worked well. One tutor, in particular, noted the usefulness of the cropping functionality to allow people to upload only portions of written math that they wanted to show the other student, without having to upload embarrassing pictures that included their faces or surroundings. This helps achieve social translucence as opposed to complete transparency that might make some users uncomfortable.

Another observation of the current version of our app is that the interactions were largely one-sided -- the tutor wrote and the student often listened. On the one hand, this is a limitation even of in-person tutoring sessions. On the other, we believe it represents
an underutilization of the app’s collaborative potential. One main way that the student interacted was expressing confusion, often through a question mark. In a future iteration of the app, to better facilitate the interaction of the student signaling confusion, we could for example add a special button for students to notify the tutor that they need additional clarification.

In deploying the app to actual tutors, we found that a useful functionality would have been to have the ability to delete and resize the draggable math text boxes and the draggable handwritten notes when they were no longer being used. One limitation of our current version is that these stay as part of the session until the entire window is cleared, making it more difficult to have a clean document if a mistake was accidentally made.

Overall, we found that the diverse suite of tools the platform offered improved communication because it gave tutors the flexibility to choose approaches that fit the problem and their teaching styles. However, this same flexibility also led some features to be underutilized. For example, although the camera capture functionality received positive feedback during prototyping and in discussions with tutors, none of the tutors used it in the sessions we observed.

**Conclusion**

All in all, we found that Impressions successfully facilitated more transparent tutoring sessions. Using our prototypes to guide the development of our important features, we built and deployed a web app that allows students and tutors to take notes, insert math LaTeX text boxes, hand-annotate their notes, and include camera captures. We ran three tutoring sessions with our deployed version that explored different mathematical concepts. Tutors and students used the different functionality to successfully explain their respective concepts. To improve on our existing app, we can expand Impressions to include resizable and deletable annotations, as well as new signals to improve the interaction between the student and tutor when the student has a question. It is our intention that our app become a tool for tutors and students to create lasting impressions on their journey of learning.