SpringErr: A Haptic Feedback System for Breadboards Using Virtual Springs

ABSTRACT
This paper introduces a novel haptic feedback method on breadboards. When inserting a component, a user feels a virtual spring underneath pins, whose stiffness varies to inform the user about different types of electronics errors (e.g. bad connections). We demonstrate the potential of this new haptic output through SpringErr, a haptic prototype. We used our prototype to examine the recognizability of six distinct virtual spring profiles, including *High Stiffness, Medium Stiffness, Low Stiffness, Bump, Pierce, and Disappear*. The results showed that participants could distinguish between them with 94.3% accuracy across the conditions with or without a secondary cognitive task. Through a user-elicitation workshop, we obtained feedback on how users associate spring force profiles with common circuit errors. The outcome of the workshop was used in an informal study, where we gained insights into the effectiveness of the mappings and the usefulness of SpringErr in conveying circuit-related error messages to users. We conclude by discussing the limitations and presenting the lessons we learned from this work.

Author Keywords
Breadboard, haptics, virtual spring

ACM Classification Keywords
H.5.2. [Information interfaces and presentation]: User Interfaces – Haptic I/O

INTRODUCTION

Breadboard is a common tool for iterating circuit design and rapidly prototyping electronic devices, whose functionality has not significantly changed since their creation in the 1970's. However, given the central role of breadboards in modern electronics and the prototyping processes, its user interface provides a potentially rich design space to explore. Existing research has demonstrated the benefits for users by augmenting the breadboard with new sensing capabilities [16, 36, 47, 48], but less effort has been devoted to output [16, 36], especially haptic output in this context.

We propose a method of augmenting the pins of a breadboard with haptic feedback, using a virtual spring with programmable stiffness (i.e. how hard or soft the feedback is). Our haptic system has a virtual spring underneath breadboard pins that can be felt by a user when the legs of a component are inserted into the pins. This type of output can be useful in varying applications, ranging from tutorial guiding, kids education, to error notification for circuit prototyping, to notify a user about varying types of events.

In this paper, we exemplify the usefulness of this new haptic output channel on a breadboard through error notifications associated with a new component before the circuit is powered. For example, a "hard" feeling underneath pins can be used to indicate the occurrence of errors caused by a newly inserted component. Once notified, the user can use a debugging tool (e.g. an oscilloscope) to diagnose the error. Alternatively, different force profiles can be used to indicate different types of errors. For example, a soft feeling underneath can be used to indicate that a newly inserted jumper wire is broken. A hard feeling can be used to indicate more severe issues, such as power issues (e.g., connect VCC of an IC chip to Ground). This new output channel extends the natural haptic feedback a user receives from existing breadboards (e.g., loose pin as an indication of bad seating), and can be a good addition to alternative output channels (e.g., LEDs on the breadboard [16] or computer monitor).

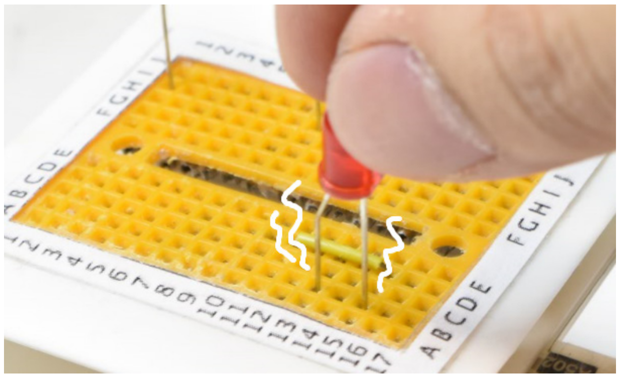


Figure 1. A user inserts a LED into the breadboard. As the virtual spring underneath the pins exhibits some stiffness, he knows that the LED is broken.

When used for error notification, our method is similar to that of syntax checking in IDEs. Our primary benefit is in *reducing errors* before a circuit is powered (e.g., "run" in IDE) much like notifying syntax errors before compiling. The insertion triggering an error notification may or may not be the error source, but the user knows that something went wrong. This is similar to notifying the user about a missing bracket in a nested function call after a ";" is entered without necessarily indicating the location of the error. Notifications based on insertion may not cover all possible errors during the construction of a breadboard circuit (e.g., logical errors),

In submission to CHI 2019

but the occurrence of many common errors – such as defective components or shorted circuits, taking place at insertion (detectable via the technology described in CircuitSense [48]) can be noticed by the user and consequently reduced. This paper focuses on developing and evaluating the haptic output, thus leaving error detection outside the scope of our implementation.

To demonstrate and evaluation the proposed haptic output, we developed a prototype haptic system (called SpringErr) containing 10 × 17 pins (similar to a mini breadboard), with the underside of the pins consisting of a moving piece that is vertically actuated using a gear motor. A pressure sensor is used to sense the strength of a user pressing the moving piece (or virtual spring) using the component legs. The travel distance of the moving piece (or the compression of the spring) is sensed precisely at a sub-millimeter scale, using an inductive proximity sensor. The data from the sensors is used to calculate varying spring behaviors supported by our prototype.

We evaluated user perceptiveness of the proposed haptic system in a controlled experiment with 15 participants, where we determined the recognizability of six different force profiles, including *High Stiffness, Medium Stiffness, Low Stiffness, Bump, Pierce, and Disappear*. These profiles were evaluated using electronic components with legs of different lengths (e.g., *short, medium, or long*), and with (or without) cognitive load to understand how real-world scenarios would affect the perception of force patterns. The results revealed participants could distinguish between the spring profiles with 94.3% accuracy. Through a user elicitation workshop with three new participants, we then associated the haptic feedback with several common circuit errors. A final preliminary user evaluation, requiring participants to construct two breadboard circuits using our prototype, revealed that the error messages could be well received by participants. The result also suggested that participants welcomed the virtual spring feedback as a useful output channel on breadboards.

Our primary contributions of this paper include (1) the notion of virtual spring feedback as a new output channel on breadboards; (2) a proof-of-concept prototype for demonstrating and studying this type of force feedback; (3) the result of a controlled experiment evaluating the recognizability of six spring profiles.

RELATED WORK

There are four relevant areas of related work: circuit prototyping tools, circuit debugging tools, output on breadboards, and programmable stiffness for haptic feedback. We discuss work in each of these respective areas.

Circuit Prototyping Tools

Creating prototypes for electronics and hardware is difficult for users with little or no background in electronics, engineering or programming. As a result, several efforts have been made to make it easier for novice users to create electronic devices by providing easy-to-use tools [8, 14, 21-24, 26, 27, 30, 39-41]. For example, Exemplar [18] and PICL [15] allow users to create sensor-based interactive systems using "programming by demonstration" (i.e. using demos to view actions and modifying them for use). Programmable Bricks [41] allows children to develop electronic hardware using LEGO bricks embedded with computers, sensors, and actuators. While these tools are in general easy to use, they are not designed to be scalable to components that are not platform-specific.

Open-source hardware platforms (e.g., Arduino [1], Phidgets [19], or Microsoft .NET Gadgeteer [44]), allow users (regardless of experience) the freedom to design and create their own electronics projects. Tools have also been developed to make the design of circuits easier for those with limited backgrounds in electronics [7, 9, 10, 28]. Users can now even print their circuits at home on plastic sheets using a modified inkjet printer [23, 26]. Trigger-Action-Circuits [8] leverages the concept of generative design by providing the user with a list of circuit design options based on a high-level description of the user's goal. A user simply needs to construct the circuit on a breadboard by following the instructions generated by the system.

The breadboard, a common tool used by novices and experts alike for prototyping electronic artifacts (in combination with other tools and software), is prone to contributing to errors [12, 32]. These errors can appear in either software or hardware, becoming a major source of frustration especially for novice users [32]. Booth, et al.'s study involving the creation of simple circuit using Arduino [1], suggests *hardware errors* (e.g., miss-wiring, incorrect component, missing components, and bad seating) are common amongst users with varying experience in electronics [12]. Little effort has been made to develop hardware platforms that mitigate hardware errors [45] and as a result, they are still common and hard to debug.

Circuit Debugging Tools

Commercially available tools like Digilent Electronics Explorer [15], augment breadboards with common debugging tools (e.g., oscilloscope, pattern generators, and logic analyzer), but require users to have some background in electronics in order to use them effectively. Research projects like Toastboard [16], Bifrost [31], and Scanalog [43] lower this barrier by providing solutions that are easier to use for novice users. However, an important feature missing in several current hardware debugging environments, is the ability to notify a user when an error occurs. This is common for software debugging tools (e.g., highlighting syntax errors) and is useful for both novices and experts.

The lack of effective error sensing techniques is perhaps one of the major challenges in developing error notification mechanisms on a breadboard. Recent work related to circuit error detection includes Toastboard [16], an intelligent breadboard capable of sensing and visualizing the voltage of a breadboard circuit; CurrentViz [47], a breadboard capable of sensing electric current flowing; and CurrentSense [48], a breadboard that can sense the location and type of an electronic component on a breadboard. These types efforts put into the development of sensing techniques, will soon lead to systems capable of diagnosing hardware errors. What is still missing, however, is a breadboard user interface that is capable of communicating error messages to the user. In this paper, we focused on virtual spring haptic feedback rather than alternatives like vibrotactile, since it naturally extends the intrinsic haptic feedback a user already receives when physically pressing a component into a breadboard.

Output on Breadboard

Using a breadboard for output has been largely overlooked. A majority of existing work has assumed there is a secondary computer monitor to show debugging information to users [16, 31, 43, 47, 48]. However, in reality, there are situations in which a computer monitor may not be available or preferred by the user (e.g. developing a circuit sewn onto a piece of clothing worn by a model during a fashion show). Visible Breadboard [36] uses LEDs to visualize voltage information on each pin. While promising, its limited number of holes (36) and non-standard pitch (3.2cm) make it difficult to work with standard components. Toastboard [16] places an array of LEDs on each side of a breadboard to indicate power, ground, or voltage information. To the best of our knowledge, audio and haptic output on a breadboard has not yet been carefully investigated.

Programmable Stiffness for Haptic Feedback

Within existing the research space for virtual spring feedback, a majority of the work has focused on studying the human perception of spring stiffness [13, 35, 37, 38] using large desktop haptic devices (e.g., [3]). Paggetti, et al. [37]'s study on stiffness perception suggested Weber fraction values between 0.123 and 0.166 for a reference stiffness value of 200 N/m. Programmable stiffness as haptic feedback for interactive systems has been implemented using motors, such as those found in commercial haptic devices [3, 42], mouse-like devices developed for interactive tables [42], or handheld devices [11, 20]. Aside from motors, programmable stiffness has also been implemented on mobiles and wearable devices using programmable gel [33], magnetorheological fluid [25], shape memory alloy [34, 46] and particle jamming [5, 17].

In this work, we did not aim to implement a high fidelity virtual spring. Instead, our goal was to demonstrate the effectiveness of using a spring approach as a means of providing haptic feedback for breadboards.

SPRINGERR PROTOTYPE

Several options exist for providing haptic feedback on a breadboard (e.g., force or vibrotactile). We began exploring this rich space by studying virtual springs as a haptic feedback mechanism, as it is easy and intuitive for users [20].

Hardware

The prototype has a mini breadboard layout with a 10 × 17 grid of pins on a 3D printed case. Inside of the case is the mechanical structure of the virtual spring, composed of a vertical moving piece and a gear motor (Figure 2-3). The moving piece is placed at the bottom of the pins, and is actuated vertically using the gear motor (Micro Metal Gearmotor HP 6V from Pololu) via a rack and pinion mechanism, with a maximum moving distance of 10 mm. We used cylinder sliders at the three corners of the moving piece to provide support and stabilize its motion. The gear motor has a gear reduction ratio of 250. Under a working voltage of 6V, the gear motor's stall torque is 67 kg/m and rotate at a maximum speed of 720° per second, fast enough to react to user input. The motor is controlled using a DRV8835 Dual Motor Driver from Pololu, which is connected to an Arduino Uno board, communicating to a Lenovo X1 laptop, running software written in Python. The device is 60mm × 60mm wide and 46mm high.

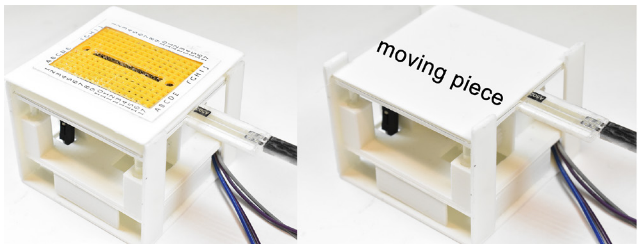


Figure 2. Left: SpringErr prototype. Right: moving piece and pressure sensor with pins removed.

The top of the moving piece was covered by a pressure sensor (FlexiForce A502 from Tekscan) measuring the amount of pressure a user puts on the component leg against it. We cover the sensor with a plastic sheet to protect its surface. The pressure sensor was also connected to the Arduino Uno, providing a sensing resolution of roughly 1g. Sensor readings are relatively consistent across the entire surface of the sensor. The travel distance of the moving piece was measured using a high-precision inductive proximity sensor from Texas Instruments (LDC1614). We used the default PCB sensor coils from the TI LDC1614 Evaluation Module. The inductive sensor works with metallic objects, thus we placed a piece of copper tape on the back of the moving piece facing the sensor. This allowed us to sense the motion of the moving piece with a sub-millimeter resolution. Our tests found that sensor data does not follow a linear relationship with the distance of the moving piece. Therefore, we conducted an experiment to find a mapping between them. The result is shown in Figure 3.

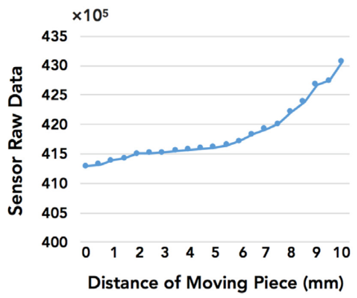


Figure 3. Mapping between LDC1614's raw data and distance from the moving piece

Our mapping resolution is 0.5 mm. The change in distance that is smaller than 0.5 mm is estimated using linear interpretation. The entire system processes at a 200 Hz update rate, which is sufficient to render the proposed haptic feedback.

We mounted 2 × 17 rows of metal clips on the rear side of the pins to hold the legs of an inserted component in place (Figure 4 left). The clips were modified from the those found in regular breadboards, by cutting the clips short and drilling five holes underneath for the component legs to pass through. The clips also connected the five pins in the same row, making the prototype function like a regular breadboard. The clips and 3D printed case measure approximately 1 mm thick. There is also approximately a 0.5 mm gap between the tips of the clips and the moving piece. As such, the maximum distance the moving piece can be compressed by a component (or *maximum compression distance*) equals the maximum depth a component legs can be inserted minus 1.5 mm.

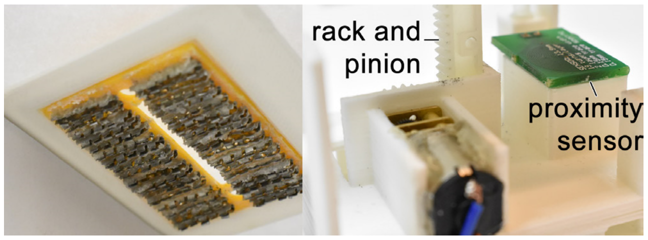
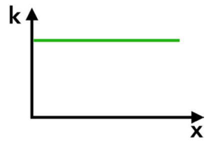


Figure 4. Left: metal clips on the rear side of the pins. Right: mechanical structure, gear motor, and proximity sensor

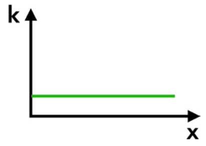
The virtual spring behaviors are implemented using a Mass-Spring-Damper (MSD) model [4], which can be solved numerically using the approached described in [6]. In our implementation, the moving piece *m* measured 10g and the damping coefficient *b* was carefully tuned to be 65 Ns/m through an experiment with 3 participants, in which we also determined the values of the parameters (e.g., *K, Turning points*) of the force profiles described below.

FORCE PROFILES

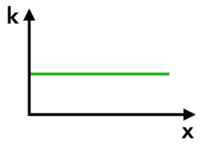
We designed and implemented six force profiles to demonstrate the capability of virtual spring feedback on a breadboard. Three of the profiles use a constant *K* value (e.g., *High/Medium/Low Stiffness*) and the remaining involved a dynamic *K* adjusted based on user input (e.g., *Bump*, *Pierce*, and *Disappear*). The profiles resemble the experience of interacting real-world objects, such as springs or buttons, to minimize learning from the user. When inserting a component, the user feels the stiffness of the spring and associates it with a certain event or message. Different information can potentially be encoded into the level or dynamics of stiffness. For example, a stiff spring could represent one piece of information, and a soft spring representing another. Once the spring was released, the moving piece moved back to its highest position waiting for the next insertion.



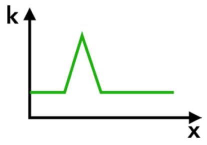
High Stiffness. This profile employs a constant *K* value, set to be high enough to simulate a spring that is difficult to compress. In our implementation, the *K* was set to be 10000 N/m.



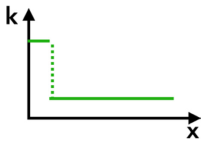
Low Stiffness. It is the softest spring among the three static profiles. Without much effort, a user can compress the spring. In our implementation, the *K* was set to be 100 N/m.



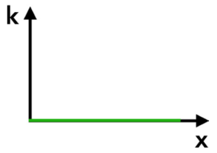
Medium Stiffness. This profile also employs a constant *K* value, but the stiffness of the spring falls between *High Stiffness* and *Low Stiffness*. We used a *K* of 500 N/m in our implementation, which creates a spring force that requires some effort for the user to insert the component.



Bump. This profile uses a dynamic *K*, whose value changes based on the position of the moving piece. When the user compresses the virtual spring, the moving piece stops at the bump, which requires a stronger force from the user to overcome. Aside from the information encoded into this profile, it is also possible to vary the number or density of the bumps to encode more information. Additionally, the 'force' of the bump as well as the 'width' of the bump can be modified to convey even more information to a user. In our implementation, the *K* value was set to 2700 N/m when the moving piece reaches 35% of a component's maximum compression distance, and reduced to 250 N/m when the moving piece is pushed to 50% of the maximum compression distance



Pierce. This profile simulates the feeling of penetrating a piece of 'cloth' using a needle. When being pushed down, the moving piece stops at a certain position with a high *K* value, requiring a much stronger force from the user to push the moving piece down for a short distance, after which, the resistant force becomes significantly lower to give the user an illusion of piercing. Similar to Bump, information can also be encoded into the number or density of the virtual cloth layer. In our implementation, the virtual cloth was placed at the original position of the moving piece. The *K* was set to 4000 N/m when the user touched the moving piece and reduced to 220 N/m after the moving piece is pushed to 10% of the component's maximum compression distance.



Disappear. This profile simulates the feeling of the virtual spring disappearing upon touching the component legs. In our implementation, the height of the moving piece dropped to its lowest position in 160 ms the moment a force acted upon the pressure sensor, surpassing a small predefined threshold (e.g., 0.2N).

USER STUDY – RECOGNIZABILITY

We conducted a user study to explore whether virtual spring force feedback could be an effective, recognizable output channel for breadboards. We were interested in examining how well participants could perceive and distinguish our force profiles. We noted that components with short legs (e.g., ICs) have shorter maximum compression distances than those with long legs (e.g., resistors), and this could affect how well a user can sense the spring profiles. Thus we were also interested in measuring how recognizable our spring profiles are when using components with different leg lengths.

Participants

15 participants (4 females) between the ages of 19 and 28 participated in the study. All participants were right-handed. 10 had previous experience using a breadboard.

Apparatus

The apparatus we used included the SpringErr prototype and nine commonly used breadboard components, categorized by the length of their legs (Figure 5). Within each category, different components had legs of different lengths, but the difference was minimal. Amongst several of the common breadboard components that were considered for the study, we prioritized those that were different in their number of legs, to ensure that our study results could also be generalizable to components with different numbers of legs.

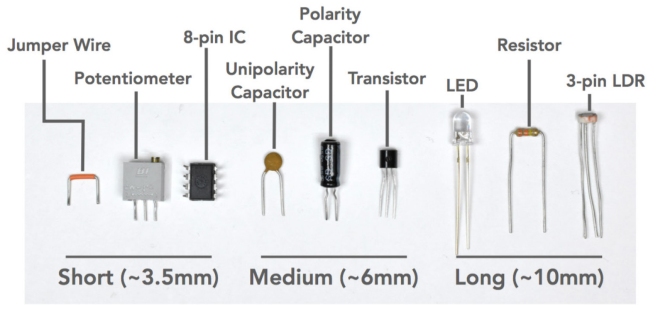


Figure 5. The nine components used in the study.

During the study, a 21.5-inch computer monitor was placed in front of participants to display the experimental user interface (Figure 6). Participants wore noise cancelling headphones to block the noise generated by the motor yet also enabled them to listen to the cognitive task described by the experimental software. The moving pieces were also hidden from participants to ensure that study results were not affected by visual indicators. The experimental software was written in Python and was run on a Lenovo ThinkPad laptop.

Task

The study required participants to insert a component into the SpringErr prototype and report what spring profile was felt. Half of the trials required participants to perform a secondary task to induce cognitive load. This secondary task was introduced to divert participants' attention away from the profile identification task, simulating common scenarios such as interacting with the breadboard while having a discussion with a colleague. When the cognitive task was presented, participants listened to a random list of words, pronounced one-by-one with 0.5s interval to mimic normal speech speed. They were asked to report after a trial the word or the previous word while a spring profile was being presented. This was to ensure that participants performed the two tasks in parallel. The choice of what words to report was randomly determined by the experimental software. We chose to use random words instead of sentences to eliminate the possibility of participants guessing the testing word based on the context of a sentence. All of our participants spoke Chinese as their first language. Therefore, the tested words were chosen from the top 200 most common Chinese words composed of one to two characters [29]. When the secondary task was not used, participants only indicated which spring profile they felt.

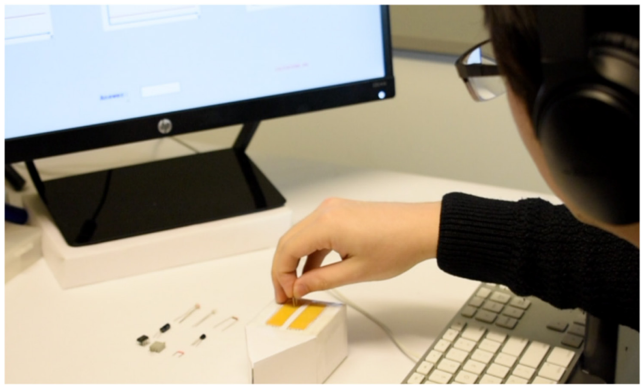


Figure 6. Study setup

Two conditions needed to be satisfied before the audio was stopped: (1) a profile was rendered completely. In the *Bump* or *Pierce* conditions, the profiles were rendered completely after presenting the bump or pierce. In the *High/Medium/Low Stiffness* conditions, the profiles were rendered completely after participants pressed the moving piece to its lowest position. In the *Disappear* condition, the profile was rendered completely after the moving piece dropped to its lowest position; or (2) participants released the spring before a profile was rendered completely. For example, if a participant released the spring in the *High Stiffness* condition without pushing the moving piece to its lowest position, the audio stopped. Similarly, releasing the spring before reaching the bump also stopped the audio. This resulted in participants missing the profile.

Procedure

Prior to the study, participants were shown the SpringErr prototype and informed about the goal of the study. Each participant was allowed several practice trials to familiarize themselves with the spring profiles. They were then asked to perform the task using their right hand.

To start a trial, participants pressed the space bar on a keyboard using the same hand grabbing the component. In the cognitive task condition, pressing the space bar also started the audio. Once the component was inserted, the audio stopped after completing the last word. Participants pressed the space bar again using the same hand to end the trial when they were ready to respond. They verbally reported to the experimenter which spring profile was presented. They also responded to the cognitive task question in the condition where the secondary task was presented. The percentage of correct responses to the cognitive task was shown on a monitor and participants were asked to maintain an accuracy above 90%. All participants successfully maintained a 90% accuracy threshold. Participants were allowed to try a profile as many times as they needed until they felt confident about reporting which one it was. To begin the next trial, participants pressed the space bar again. Breaks were encouraged during the study.

Upon completion of the study, participants filled out a post-experiment questionnaire where they indicated subjective ratings for the recognizability of the spring profiles (1: very hard to recognize, 7 very easy to recognize). The experiment lasted approximately 60 minutes.

Experimental Design

The experiment employed a 2 × 3 × 6 within-subject factorial design. Independent variables were Secondary Task (*Cognitive Load* and *No Load*), Leg Length (*Short, Medium,* and *Long*), Spring Profile (*High Stiffness, Medium Stiffness, Low Stiffness, Bump, Pierce,* and *Disappear*). During each trial, participants performed tasks in one of the *Secondary Task* × *Leg Length* × *Spring Profile* combinations. Within each *Leg Length* condition, each of the three components was used once. The *Leg Length* was counter-balanced among participants. The *Secondary Task* and *Spring Profile* were presented in a random order. The experimental design was thus *2 Secondary Task* × *3 Component Leg* × *6 Spring Profile* × *3 components* × *15 Participants* = 1620 trials.

Dependent measures included profile recognition accuracy and response time. The profile recognition accuracy was defined as the number of correctly identified spring profiles. The response time was defined as the time elapsed between when the component touches the moving piece (e.g., pressure value exceeding a threshold) and the end of a trial.

Results

We analyzed the data using repeated measures ANOVA and Bonferroni corrections for pair-wise comparisons. For violations to sphericity, we used a Greenhouse-Geisser adjustment for degrees of freedom.

Profile Recognition Accuracy

All participants took only 1 attempt to identify the profile in each trial. The average accuracy across all conditions was 94.32% (s.e. = 0.58%). ANOVA yielded a significant effect of *Leg Length* ($F_{1.27,\ 17.79} = 42.48$, $p < .05$), *Secondary Task* ($F_{1,14} = 24.05$, $p < .05$), and *Spring Profile* ($F_{5,70} = 3.81$, $p < .05$). No significant interaction effect was found between them (all $p > 0.05$). Figure 7 (right) shows the recognition accuracy by spring profiles.

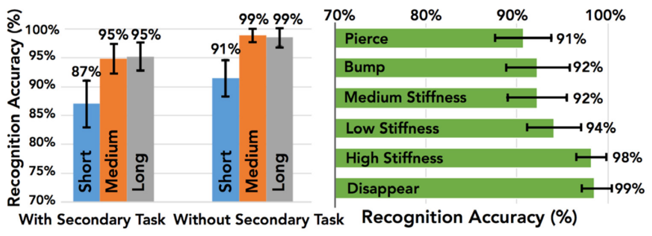
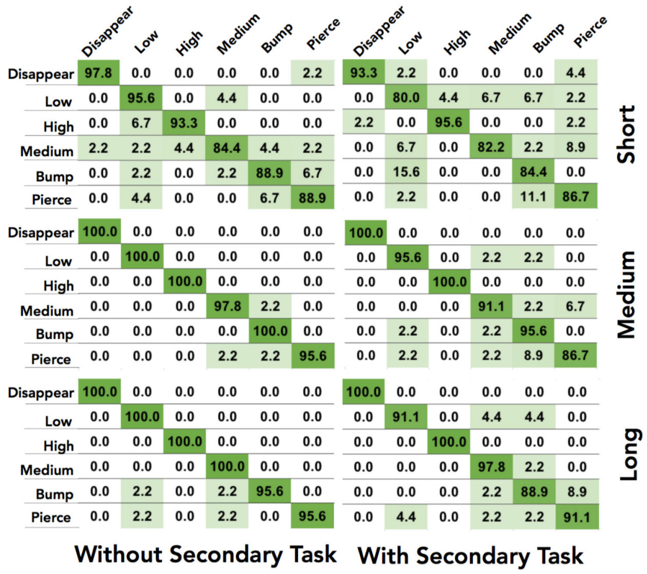


Figure 7. Profile Recognition Accuracy. Left: Secondary Task and Leg length. Right: Force profiles. (Error bars show +/- 2 standard error in all the figures).

Recognition accuracy was significantly higher without a Secondary Task (M = 96.3%, s.e. = 0.66% vs. M = 92.35%, s.e. = 0.94%, $p < .05$). Additionally, post-hoc comparisons revealed no significant difference in recognition accuracy when the component legs were *Long* and *Medium* (M = 96.85%, s.e. = 0.73% vs. M = 96.85%, s.e. = 0.73%, $p = 1$), but recognition accuracy significantly decreased when component legs were *Short* (M = 89.26%, s.e. = 1.33%, both $p < .05$) (Figure 7 left).

Confusion matrices of the recognition accuracy of different spring profiles under the *Leg Length* and *Secondary Task* conditions (Figure 8) revealed that *Low Stiffness* and *Medium Stiffness* were easier to be confused with the other profiles when component legs were *Short* and/or with the *Secondary Task.* This indicates that these profiles require longer maximum compression distance for participants to sense them accurately, especially when user attention is divided by a secondary task. *Bump* and *Pierce* were also easy to be mixed. especially with a secondary task as they both have a dynamic *K,* requiring more attention.



		Without Secondary Task						With Secondary Task					
		Disappear	Low	High	Medium	Bump	Pierce	Disappear	Low	High	Medium	Bump	Pierce
Short	Disappear	97.8	0.0	0.0	0.0	0.0	2.2	93.3	2.2	0.0	0.0	0.0	4.4
	Low	0.0	95.6	0.0	4.4	0.0	0.0	0.0	80.0	4.4	6.7	6.7	2.2
	High	0.0	6.7	93.3	0.0	0.0	0.0	2.2	0.0	95.6	0.0	0.0	2.2
	Medium	2.2	2.2	4.4	84.4	4.4	2.2	0.0	6.7	0.0	82.2	2.2	8.9
	Bump	0.0	2.2	0.0	2.2	88.9	6.7	0.0	15.6	0.0	0.0	84.4	0.0
	Pierce	0.0	4.4	0.0	0.0	6.7	88.9	0.0	2.2	0.0	0.0	11.1	86.7
Medium	Disappear	100.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0
	Low	0.0	100.0	0.0	0.0	0.0	0.0	0.0	95.6	0.0	2.2	2.2	0.0
	High	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0
	Medium	0.0	0.0	0.0	97.8	2.2	0.0	0.0	0.0	0.0	91.1	2.2	6.7
	Bump	0.0	0.0	0.0	0.0	100.0	0.0	0.0	2.2	0.0	2.2	95.6	0.0
	Pierce	0.0	0.0	0.0	2.2	2.2	95.6	0.0	2.2	0.0	2.2	8.9	86.7
Long	Disappear	100.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0
	Low	0.0	100.0	0.0	0.0	0.0	0.0	0.0	91.1	0.0	4.4	4.4	0.0
	High	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0
	Medium	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	97.8	2.2	0.0
	Bump	0.0	2.2	0.0	2.2	95.6	0.0	0.0	0.0	0.0	2.2	88.9	8.9
	Pierce	0.0	2.2	0.0	2.2	0.0	95.6	0.0	4.4	0.0	2.2	2.2	91.1

Figure 8. The confusion matrices of the Profile Recognition Accuracy (%), by Secondary Task and Pin Length.

Response Time

The average response time was 1980 ms (s.e. = 48.5ms) across all conditions. ANOVA yielded a significant effect of *Leg Length* ($F_{2,28} = 9.72$, $p < .05$) and *Spring Profile* ($F_{3.05,42.72}$ =

8.85, $p < .05$). There was no significant effect of *Secondary Task* ($F_{1,14} = 0.26$, $p = 0.62$), nor any significant interaction between the independent variables (all $p > 0.05$).

Post-hoc comparisons revealed that response time was significantly longer using the components with short legs (2371 ms; s.e. = 105.4 ms) than those with medium (1681 ms; s.e. = 51.3ms) No significant difference was found between the other pairs (all $p > 0.05$). This finding remained true for the conditions with or without a *Secondary Task*. Longer in response time with short legs suggests that a short maximum compression distance had negative effects on both recognition accuracy and speed.

Among all the tested profiles, the response time for *Disappear* (1434ms; s.e. = 61.5ms) was the shortest (all $p < 0.05$). This was to be excepted as *Disappear* was distinguishable earlier in each trial. No significant difference was found between all the other spring profiles (all $p > 0.05$) (Figure 9 left)

Finally, response time with and without the secondary task was 1997ms (s.e. = 76.9ms) and 1963ms (s.e. = 59.2ms) with no significant difference between them, suggesting that while the secondary task affected the recognition accuracy, it did not significantly affect participants' speed in response.

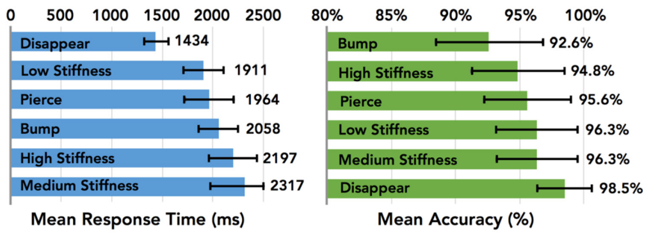


Figure 9. Left: The average response time (in ms). Right: The secondary task accuracy for each spring profile.

Secondary Task Accuracy

The overall accuracy of the *Secondary Task* was 95.7% (s.e. = 0.68%). There was no significant effect of *Leg Length* ($F_{2,28} = 0.25$, $p = 0.78$) or *Spring Profile* ($F_{5,70} = 1.29$, $p = 0.28$), nor did we find any interaction effect between them ($F_{10,140} = 1.58$, $p = 0.12$). The accuracies of secondary tasks by spring profiles are shown in Figure 9 right.

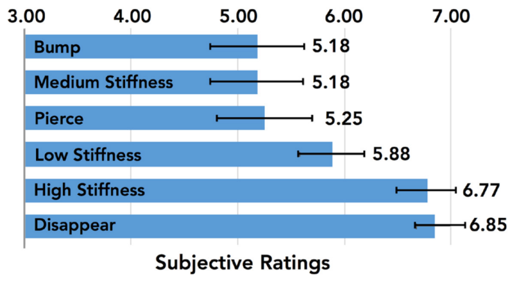


Figure 10. Participant responses to the ease of recognizability of the spring profiles.

Subjective Ratings

Overall, the average ratings for all spring profiles was 5.85 (s.e. = 0.17) with 7 being very easy to recognize. This suggests that all or most participants agreed that the proposed spring profiles were easy to perceive. ANOVA yielded a significant effect of *Spring Profile* on the subjective ratings ($F_{5,70} = 21.38$, $p < .05$). In particular, *Disappear* (M = 6.84, s.e. = 0.09) and *High Stiffness* (M = 6.77, s.e. = 0.14) were rated highest in terms of recognizability (Figure 10). Participants found them significantly easier to recognize than the remaining group of spring profiles (all $p < .05$). Post-hoc tests found no significant difference between these two groups (all $p > .05$). The subjective ratings are consistent with the quantitative results. This again confirms the effectiveness of spring force feedback as a new output channel for breadboard. Participants were welcoming of this new output technique as well. They described it as "*helpful" (P6, P8, P15), "cool" (P3, P12), "awesome" (P13)* and *"amazing" (P2).* They told us that they could imagine this type of haptic feedback also being used when learning electronics and circuits *(P3, P7, P8, P12). "I see it can be useful in classroom to teach circuits and errors"*, said by *P3*.

Summary

This study showed that most force profiles could be easily distinguished using electronic components of different leg lengths, even under conditions where a users' mental workload was high. Performing a secondary task had a negative impact on participants' ability to accurately perceive the spring profiles but the recognizability remained reasonably high. Across all conditions, including leg length and mental workload, *Disappear* and *High Stiffness* were easily perceived, making them excellent candidates for conveying critical messages.

Components with short legs had a negative impact on the recognizability and response speed, especially when the users' attention was divided. This impact is particularly pronounced with *Pierce, Bump, Low Stiffness*, and *Medium Stiffness*, although distinguishing between them was not difficult when full attention was paid. With sensing technologies capable of detecting breadboard components (e.g., [48]), extra feedback channels such as visual or audio can be used to further attract user's attention upon detecting a component with short legs. The issues associated with short legs can also be solved with improved hardware (e.g., thinner clip structure to enlarge the compression distance) and software implementation (e.g., force profiles better designed for short-leg components).

Our results demonstrated that the average accuracy of the secondary task was around 95.7%, suggesting that using the proposed haptic feedback may not significantly impact the performance of a second, parallel task. It is also promising to find that a mental workload did not significantly impact response time. This again confirms that the proposed haptic feedback does not divide attention to the degree that it imposes a significant cognitive load on users. Amongst all the spring profiles, *Disappear* had a significantly shorter response time, making it a good candidate for conveying highly repetitive information (e.g., nothing to report) to assure the efficiency of the overall circuit construction task.

Finally, this study shows that different spring profiles offer unique properties that are suitable for different event messages, which we discuss in the next section.

SPRINGERR EVENT MESSAGES

In this section, we demonstrate several event messages suitable for SpringErr applications. We drew upon common breadboard errors discussed in [16] and chose those that are detectable based on the information of (1) component; (2) pin locations; and (3) resistance, capacitance, and inductance between two points, all retrievable using the technology described in CircuitSense [48]. Sensing such errors is also fast enough that it should not introduce a noticeable latency in providing haptic feedback. We did not implement CircuitSense in this work because our focus is on the haptic output but integrating it into our device is technically feasible by connecting the sensing chip to the metal clips of our prototype.

The error messages are meant to serve as notifications without providing detailed information about the error or suggestions on how to resolve them. Such notifications can sometimes be sufficient for a user to diagnose and correct the error. In cases that more information is needed to understand the error, the user is required to use an oscilloscope or computer. Regardless, a user is notified as soon as an error occurs, which we believe can be helpful to avoid circuit errors.

We did not intend to provide an exhaustive list of possible messages or to validate that such errors can be reliably detected. Instead, we aimed to use them to showcase the potential of using haptic feedback on a breadboard. We then used these error messages in an informal user study to gather insights into the usefulness of the proposed haptic feedback system on a breadboard.

Messages

No error detected. This event represents the stage where no error is detected after a new component is added to the breadboard circuit.

Power management issues. This category of errors is related to problems that may cause power issues. Examples include bad power or ground connections, or an improper amount of voltage or current. A good indication for these types of errors are: (1) a new component is connected in parallel with a component(s) requiring different voltage; (2) a new component is connected in series with a component(s) requiring different current; or (3) the VCC of an IC chip is connected to the ground or vice versa.

Improperly connected components. Examples in this category of errors include shorted circuits or improper component polarity. A new component with very low resistance (e.g., jumper wire) placed between two nodes of a circuit is an indication of shorted circuit. An inconsistency between the polarity of the new component and that of the ones already existing on the breadboard, is an indication of a polarity issue. The polarity of a component (e.g., diode) can be determined by measuring the resistance of the component, where a low resistance indicates forward bias and a high resistance indicates reverse bias.

Bad connections. Examples in this category include issues related to bad wires or broken connections between the current component and the circuit. A new component or its connection to the circuit measuring a high resistance is an indication of bad connection.

Defective components. Components can be broken, and thus it becomes hard to distinguish via visual inspection. Unusual readings of component data (e.g., capacitance or resistance) is a good indication of defective components. For diodes, a measure of high resistance in both forward and reverse bias is an indication of defectiveness. When a bad connection is caused by a defective component, we only notify the user about the bad connection itself and not about the component.

Aside from these common errors, if a target circuit is known by the system (e.g., generated by a design tool [8]), errors like incorrect pin usage or inconsistency with the sample circuit can be detected, and a user can be notified.

User Elicitation Workshop

We conducted a small workshop to elicit initial feedback from potential SpringErr users on how they associate the spring force profiles with these error messages.

We recruited three graduate students to participate in this workshop. All participants had some background in electronics and were familiar with constructing breadboard circuits. We first introduced the SpringErr prototype to the participants and allowed them to experiment with it and try all profiles for several minutes. We then showed them the five messages and asked them to come up with a mapping between the messages and profiles. Note that we had one additional profile compared to the number of error messages. As our aim was to gather early feedback and not have an extensive mapping of profiles to several types of errors, one profile was excluded in the final mapping. Participants were first asked to come up with their own mappings, and then worked in a group to discuss their reasonings until they reached an agreement for each mapping.

Spring Profile	**Message**
Disappear	No error detected
High Stiffness	Power management
Medium Stiffness	Defective component
Low Stiffness	Bad connections
Bump	Improperly connected components

Table 1. User elicited mappings between the spring profiles and error messages.

Table 1 shows the mappings. All participants agreed that the *Disappear* profile should be associated with *no error detected,* as it is the least obtrusive profile that introduces minimal disturbance to the user. For the error messages, their mapping was based on the level of stiffness and severity of the errors. In other words, the harder it is for the user to insert a component into the breadboard, the more severe the error should be. For example, participants associated *High Stiffness* with *power management issues* because the issues related to power management were considered the most severe amongst all the others, as they may damage components or risk the health of users. This is consistent with our findings in Study 1. *Medium Stiffness* was associated with *defective components* because the issues were less severe, but also hard to debug (e.g., visually), thus requiring attention from the user when they occur. *Low Stiffness* was associated with *bad connections* since connection errors do not cause damage to components and can be relatively easier to diagnose via visual inspection. Finally, *improperly connected components* was considered as severe as *power management* issues since such errors may also cause damage to components. Participants rated *Bump* over *Pierce* on this message. They proposed to use *Pierce* in situations where the system is not certain about a severe error, in which case, feeling the momentary blockage of the pierce profile indicated caution, but not prohibition to our user group.

INFORMAL EVALUATION

We conducted a preliminary user evaluation to gain some initial insights into the usefulness of our system in conveying error messages when people are constructing real breadboard circuits.

Participants

Six participants (one female) between the ages of 19 and 23 participated in the study. Three participants were electronics engineers and the rest had no electronics background.

Task and Procedure

Participants were given 15 minutes to understand the purpose of the study, feel the spring profiles, and learn the messages associated with each profile.

They were then asked to construct two sample circuits chosen from the Arduino Starter Kit [2]: the first was Love-o-Meter and second one was Mood Cue. Love-o-Meter measures the temperature of a user's hand and translates it into LED lighting pattern. The circuit involves a SpringErr, a temperature sensor, three LEDs each with a 220 Ω resistor, and a few jump wires. Mood Cue allows a user to control the rotation of a servo motor using a potentiometer. The circuit involves the SpringErr, a potentiometer, a servomotor, two capacitors, and a few jump wires. During the study, the two circuits were presented to the participants in a random order.

To construct the circuits, participants followed our step by step instructions shown on a monitor. For each step, participants inserted a component into a desired location in the SpringErr and felt the corresponding haptic feedback. To control the occurrences of error messages among the participants, we hardcoded the messages in our software. For example, *Medium Stiffness* was rendered to indicate a *defective component* when participants inserted the third resistor in the Love-o-Meter circuit. Bump was rendered to indicate *improperly connected components* when participants inserted the second LED in the same example. For the steps that were scripted as *no error detected*, the *Disappear* profile was rendered. Participants experienced all event messages in both circuits. In each step, they reported which profile they felt, as well as the event message associated with the profile. They did not need to correct the errors.

Upon completing the circuits, we interviewed participants and asked for their opinions on the usefulness of SpringErr, their agreements on the mappings, and message memorability.

Result

All participants were able to identify profiles as well as the messages associated with the profiles with 100% accuracy. Participants found SpringErr effective for delivering error notifications as well as reducing potential hardware errors. With respect to the mappings, participants found them easy and intuitive to learn. Their comments included *"easy to recall" (P1)*, *"easy to memorize" (P4)*, and *"mappings make sense" (P5).* They felt it was reasonable to use *Disappear* to represent *no error detected*. A participant commented that *"this profile did not resist my action, which is consistent with my expectation of notifying a correct action" (P6)*. Participants also agreed that *High Stiffness* is a reasonable feedback for *power management issues*. One participant said that *"it seemed like the breadboard tried to stop me from inserting the component, so I understand it must be something serious, and it is not that difficult to recall that it is something related to power management" (P1)*. They also found it reasonable to use *Bump* to notify *improperly connected components* since *"this feedback is very different from the others and strong enough to feel that something needs my attention" (P5)*. Participants agreed on the mapping of stiffness to severity and found such mappings easy for them to learn and memorize the associated messages. Finally, none of the participants found it a burden to acquire the haptic feedback.

In addition to the tested spring profiles, participants also suggested that other haptic feedback types, (e.g., vibrotactile) could also be added to the breadboard to broaden the information transfer bandwidth. One suggested using SpringErr to deliver SMS notifications so that important messages will not be missed when their focus is on prototyping. This is quite interesting and suggests a broader area of new applications can be enabled using the breadboard as an output device.

LIMITATIONS AND FUTURE WORK

We discuss the limitations of our work and suggest future research for exploring output capabilities on breadboards.

Information transfer bandwidth. In the current iteration of SpringErr, only notifications are conveyed about a single error that is created when the last component is added to the circuit. However, it is possible to encode multiple types of errors into a force profile (e.g. aside from bad connections, *Low Stiffness* can also indicate defective components).

Future research will extend our current work by investigating techniques that can notify users about errors that already exist in the circuit. The information transfer bandwidth of SpringErr is relatively low, which prevents it from conveying detailed information beyond notifications. Expanding the bandwidth of SpringErr is also an interesting area to explore. Other haptic techniques (e.g., vibrotactile) can be integrated into the breadboard to improve the bandwidth or even overcome the limitations of our technique, such as components with short legs. Perhaps more importantly, additional research needs to be carried out to understand the role of output on breadboards and its best uses, such that proper output technologies can be developed in the future to better facilitate rapid prototyping.

Device implementation. The current prototype is bulky. We expect that the size and thickness of SpringErr can be reduced with additional engineering efforts. The structure of the device can be redesigned into a more compact form factor. It is also possible to use a smaller, yet just as powerful motor for the haptic rendering. We are planning to integrate all the electronics into a self-contained form factor. This will enable new opportunities for us to understand its usability in new environments and conditions. The number of moving pieces should be increased (ideally one per pin) to minimize the impact on the stableness of the components already exist on the breadboard from the insertion of a new component.

Study. Our study focused on simple spring profiles to demonstrate the promise of SpringErr. The results should be interpreted with reservation when generalizing them to different implementations. Future research will go beyond the discriminability of the virtual spring profiles by understanding deeper questions regarding the discoverability and perceivability of the virtual spring feedback. Just noticeable difference (JND) experiments could be conducted to understand the minimal amount, and varying levels, of virtual spring feedback that a user is able to detect. Acquiring virtual spring feedback follows the insertion of a component naturally. This action was not reported as a burden by our participants. It is, however, still possible that a user may miss a notification if they do not insert the component deep enough to feel the moving piece. Future research will examine how often these notifications may be missed and investigate techniques to avoid such instances. With the improvements on our hardware, we plan to deploy the device to the maker community in our city, especially for those in wearable fashion who often find it common to prototype wearable circuits outside of a typical lab environment, such as on the body of a wearer. This will allow us to identify the challenges different user groups face and understand the issues with the current device.

Sensing errors. We focused on haptic feedback with the assumption that research in sensing techniques will eventually enable the sensing of hardware errors that can be delivered to the user in real-time by new output technologies like SpringErr. With new sensors and computing devices, one of the major challenges becomes the device form factor. We expect that future breadboards might not be as thin as the ones we are familiar with but will be more powerful.

Applications. While this paper focuses on error notifications, we foresee that the proposed haptic output channel can be useful for many other new applications on a breadboard. Future research will be focused on extending this haptic output channel to guide beginners through step-by-step instructions of circuit tutorials. We are also interested in exploring its potential in education applications to improve kids' learning experiences and make learning electronics and circuits more engaging than the current practice.

CONCLUSION

Overall, our studies demonstrated the potential for improving user experience when prototyping with SpringErr and its haptic feedback techniques. We developed six haptic feedback force profiles for our prototype and conducted a user study to evaluate our user's ability to distinguish between these force profiles. This study revealed that most force profiles could be easily distinguished using electronic components of different leg lengths, even under conditions where the mental workload of a user was high. We then mapped each of our 6 force profiles to common errors that users experience when prototyping with a traditional breadboard and evaluated the effectiveness of SpringErr at communicating them through haptic feedback. Our evaluation revealed that our system was easy to understand and helpful for notifying users about common hardware errors while constructing circuits with SpringErr. We believe that further development of smart breadboard technologies such as SpringErr will enable users to prototype hardware and software inventions easily and effectively, regardless of their familiarity with electrical engineering and software design.

REFERENCE

1. Arduino. http://arduino.cc.
2. Arduino Starter Kit. https://store.arduino.cc/usa/arduino-starter-kit. 2018
3. Omega 3 Haptic Device from Force Dimension. http://www.forcedimension.com/products/omega-3/overview. 2018
4. José Francisco Gómez Aguilar, Juan Rosales García, Jesus Bernal Alvarado and Manuel Guía. 2012. Mathematical modelling of the mass-spring-damper system-A fractional calculus approach. *Acta Universitaria*, 22 (5). 5-11.
5. Ahmed Al Maimani and Anne Roudaut. Frozen Suit: Toward a Changeable Stiffness Suit and its Application for Haptic Games.
6. Roger Alexander. 1990. Solving ordinary differential equations I: Nonstiff problems (E. Hairer, SP Norsett, and G. Wanner). *SIAM Review*, 32 (3). 485.
7. Altium Designer 17 Overview. http://www.altium.com/altium-designer/. 2017
8. Fraser Anderson, Tovi Grossman and George Fitzmaurice. 2017. Trigger-Action-Circuits: Leveraging Generative Design to Enable Novices to

Design and Build Circuitry. in *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, ACM, 331-342.
9. Autodesk Circuits. https://circuits.io/. 2017
10. EAGLE PCB Design and Schematic Software. 2017
11. Akash Badshah, Sidhant Gupta, Gabe Cohn, Nicolas Villar, Steve Hodges and Shwetak N Patel. 2011. Interactive generator: a self-powered haptic feedback device. in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 2051-2054.
12. Tracey Booth, Simone Stumpf, Jon Bird and Sara Jones. 2016. Crossed wires: Investigating the problems of end-user developers in a physical computing task. in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ACM, 3485-3497.
13. Damien Couroussé, Gunnar Jansson, Jean-Loup Florens and Annie Luciani. 2006. Visual and Haptic perception of object elasticity in a virtual squeezing event. in *EuroHaptics 2006 conference*, 47.
14. Kayla DesPortes, Aditya Anupam, Neeti Pathak and Betsy DiSalvo. 2016. BitBlox: a redesign of the breadboard. in *Proceedings of the The 15th International Conference on Interaction Design and Children*, ACM, 255-261.
15. Digilent Electronics Explorer https://store.digilentinc.com/electronics-explorer-all-in-one-usb-oscilloscope-multimeter-workstation/. 2017
16. Daniel Drew, Julie L Newcomb, William McGrath, Filip Maksimovic, David Mellis and Björn Hartmann. 2016. The Toastboard: Ubiquitous Instrumentation and Automated Checking of Breadboarded Circuits. in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, ACM, 677-686.
17. Sean Follmer, Daniel Leithinger, Alex Olwal, Nadia Cheng and Hiroshi Ishii. 2012. Jamming user interfaces: programmable particle stiffness and sensing for malleable and shape-changing devices. in *Proceedings of the 25th annual ACM symposium on User interface software and technology*, ACM, 519-528.
18. Adam Fourney and Michael Terry. 2012. PICL: portable in-circuit learner. in *Proceedings of the 25th annual ACM symposium on User interface software and technology*, ACM, 569-578.
19. Saul Greenberg and Chester Fitchett. 2001. Phidgets: easy development of physical interfaces through physical widgets. in *Proceedings of the 14th annual ACM symposium on User interface software and technology*, ACM, 209-218.
20. Sidhant Gupta, Tim Campbell, Jeffrey R Hightower and Shwetak N Patel. 2010. SqueezeBlock: using virtual springs in mobile devices for eyes-free interaction. in *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, ACM, 101-104.
21. Björn Hartmann, Leith Abdulla, Manas Mittal and Scott R Klemmer. 2007. Authoring sensor-based interactions by demonstration with direct manipulation and pattern recognition. in *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM, 145-154.
22. Björn Hartmann, Scott R Klemmer, Michael Bernstein, Leith Abdulla, Brandon Burr, Avi Robinson-Mosher and Jennifer Gee. 2006. Reflective physical prototyping through integrated design, test, and analysis. in *Proceedings of the 19th annual ACM symposium on User interface software and technology*, ACM, 299-308.
23. Steve Hodges, Nicolas Villar, Nicholas Chen, Tushar Chugh, Jie Qi, Diana Nowacka and Yoshihiro Kawahara. 2014. Circuit stickers: peel-and-stick construction of interactive electronic prototypes. in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 1743-1746.
24. Steven Houben, Connie Golsteijn, Sarah Gallacher, Rose Johnson, Saskia Bakker, Nicolai Marquardt, Licia Capra and Yvonne Rogers. 2016. Physikit: Data engagement through physical ambient visualizations in the home. in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ACM, 1608-1619.
25. Yvonne Jansen, Thorsten Karrer and Jan Borchers. 2010. MudPad: tactile feedback and haptic texture overlay for touch surfaces. in *ACM International Conference on Interactive Tabletops and Surfaces*, ACM, 11-14.
26. Yoshihiro Kawahara, Steve Hodges, Benjamin S. Cook, Cheng Zhang and Gregory D. Abowd. 2013. Instant inkjet circuits: lab-based inkjet printing to support rapid prototyping of UbiComp devices *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, ACM, Zurich, Switzerland, 363-372.
27. Majeed Kazemitabaar, Jason McPeak, Alexander Jiao, Liang He, Thomas Outing and Jon E Froehlich. 2017. Makerwear: A tangible approach to interactive wearable creation for children. in *Proceedings of the 2017 chi conference on human factors in computing systems*, ACM, 133-145.
28. André Knörig, Reto Wettach and Jonathan Cohen. 2009. Fritzing: a tool for advancing electronic prototyping for designers. in *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*, ACM, 351-358.
29. Xingjian Li and TK Wang. 2008. Lexicon of Common Words in Contemporary Chinese, The Commercial Press.
30. Joanne Lo, Cesar Torres, Isabel Yang, Jasper O'Leary, Danny Kaufman, Wilmot Li, Mira Dontcheva and Eric Paulos. 2016. Aesthetic Electronics: Designing,

Sketching, and Fabricating Circuits Through Digital Exploration. in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, ACM, 665-676.

31. Will McGrath, Daniel Drew, Jeremy Warner, Majeed Kazemitabaar, Mitchell Karchemsky, David Mellis and Björn Hartmann. 2017. Bifröst: Visualizing and Checking Behavior of Embedded Systems across Hardware and Software. in *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, ACM, 299-310.
32. David A Mellis, Leah Buechley, Mitchel Resnick and Björn Hartmann. 2016. Engaging amateurs in the design, fabrication, and assembly of electronic devices. in *Proceedings of the 2016 ACM Conference on Designing Interactive Systems*, ACM, 1270-1281.
33. Viktor Miruchna, Robert Walter, David Lindlbauer, Maren Lehmann, Regine Von Klitzing and Jörg Müller. 2015. Geltouch: Localized tactile feedback through thin, programmable gel. in *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, ACM, 3-10.
34. Yusuke Nakagawa, Akiya Kamimura and Yoichiro Kawaguchi. 2012. MimicTile: a variable stiffness deformable user interface for mobile devices. in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 745-748.
35. Ilana Nisky, Assaf Pressman, Carla M Pugh, Ferdinando A Mussa-Ivaldi and Amir Karniel. 2011. Perception and action in teleoperated needle insertion. *IEEE transactions on haptics*, 4 (3). 155-166.
36. Yoichi Ochiai. 2014. Visible Breadboard: System for Dynamic, Programmable, and Tangible Circuit Prototyping with Visible Electricity. in *International Conference on Virtual, Augmented and Mixed Reality*, Springer, 73-84.
37. Giulia Paggetti, Burak Cizmeci, Cem Dillioglugil and Eckehard Steinbach. 2014. On the discrimination of stiffness during pressing and pinching of virtual springs. in *Haptic, Audio and Visual Environments and Games (HAVE), 2014 IEEE International Symposium on*, IEEE, 94-99.
38. Domenico Prattichizzo, Claudio Pacchierotti and Giulio Rosati. 2011. Tactile feedback as a sensory subtraction technique in haptics for needle insertion. *Computing Research Repository (CoRR).*
39. Raf Ramakers, Fraser Anderson, Tovi Grossman and George Fitzmaurice. 2016. Retrofab: A design tool for retrofitting physical interfaces using actuators, sensors and 3d printing. in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ACM, 409-419.
40. Raf Ramakers, Kashyap Todi and Kris Luyten. 2015. PaperPulse: an integrated approach for embedding electronics in paper designs. in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, ACM, 2457-2466.
41. Mitchel Resnick, Fred Martin, Randy Sargent and Brian Silverman. 1996. Programmable bricks: Toys to think with. *IBM Systems journal*, 35 (3.4). 443-452.
42. Alejandro Jarillo Silva, Omar A Domínguez Ramirez, Vicente Parra Vega and Jesus P Ordaz Oliver. 2009. Phantom omni haptic device: Kinematic and manipulability. in *Electronics, Robotics and Automotive Mechanics Conference, 2009. CERMA'09.*, IEEE, 193-198.
43. Evan Strasnick, Maneesh Agrawala and Sean Follmer. 2017. Scanalog: Interactive Design and Debugging of Analog Circuits with Programmable Hardware. in *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, ACM, 321-330.
44. Nicolas Villar, James Scott, Steve Hodges, Kerry Hammil and Colin Miller. 2012. . NET gadgeteer: a platform for custom devices. in *International Conference on Pervasive Computing*, Springer, 216-233.
45. Chiuan Wang, Hsuan-Ming Yeh, Bryan Wang, Te-Yen Wu, Hsin-Ruey Tsai, Rong-Hao Liang, Yi-Ping Hung and Mike Y Chen. 2016. CircuitStack: supporting rapid prototyping and evolution of electronic circuits. in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, ACM, 687-695.
46. Huihui Wang, Dimosthenis Kaleas, Roger Ruuspakka and Robert Tartz. 2012. Haptics using a smart material for eyes free interaction in mobile devices. in *Haptics Symposium (HAPTICS), 2012 IEEE*, IEEE, 523-526.
47. Te-Yen Wu, Hao-Ping Shen, Yu-Chian Wu, Yu-An Chen, Pin-Sung Ku, Ming-Wei Hsu, Jun-You Liu, Yu-Chih Lin and Mike Y Chen. 2017. CurrentViz: Sensing and Visualizing Electric Current Flows of Breadboarded Circuits. in *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, ACM, 343-349.
48. Te-Yen Wu, Bryan Wang, Jiun-Yu Lee, Hao-Ping Shen, Yu-Chian Wu, Yu-An Chen, Pin-sung Ku, Ming-Wei Hsu, Yu-Chih Lin and Mike Y Chen. 2017. CircuitSense: Automatic Sensing of Physical Circuits and Generation of Virtual Circuits to Support Software Tools. in *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, ACM, 311-319.