ConductAR: An Augmented Reality Based Tool for Iterative Design of Conductive Ink Circuits

Koya Narumi
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku,
Tokyo, Japan
narumi@akg.t.u-tokyo.ac.jp

Steve Hodges
Microsoft Research
21 Station Road,
Cambridge CB1 2FB, UK
Steve.Hodges@microsoft.com

Yoshihiro Kawahara
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku,
Tokyo, Japan
kawahara@akg.t.u-tokyo.ac.jp

ABSTRACT

Recent advances in materials science have resulted in a range of commercially viable and easy-to-use conductive inks which novices, hobbyists, educators, students and researchers are now using to design and build interactive circuits quickly. Despite the ease with which practitioners can construct working circuits, one of the major limitations of designing circuits on-the-fly is the difficulty of detecting and understanding errors in prototype circuits. As well as short- and open-circuits, which often prevent a circuit from working at all, more subtle issues like high resistance traces can result in poor performance. Many users can't readily work out how to successfully modify their circuits, and they often don't have the tools or expertise to measure the relevant circuit parameters. In this paper we present ConductAR, a tool which can recognize and analyze hand-drawn, printed and hybrid conductive ink patterns. An on-screen augmented reality style interaction helps users to understand and enhance circuit operation. A key element of ConductAR is its ability to calculate the resistance of a circuit using a camera attached to an off-the-shelf PC or tablet. Our sparse coding technique is fast enough to support rapid iterative prototyping on real circuits using a conductive ink marker and/or eraser as shown in Figure 1. The system thereby enhances the feasibility of circuit prototyping with conductive ink.

Author Keywords
Conductive Ink; Rapid Prototyping; Makers; Education

ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

UbiComp' 15, September 7-11, 2015, Osaka, Japan.
© 2015 ACM. ISBN 978-1-4503-3574-4/15/09…$15.00.
http://dx.doi.org/10.1145/2750858.2804267

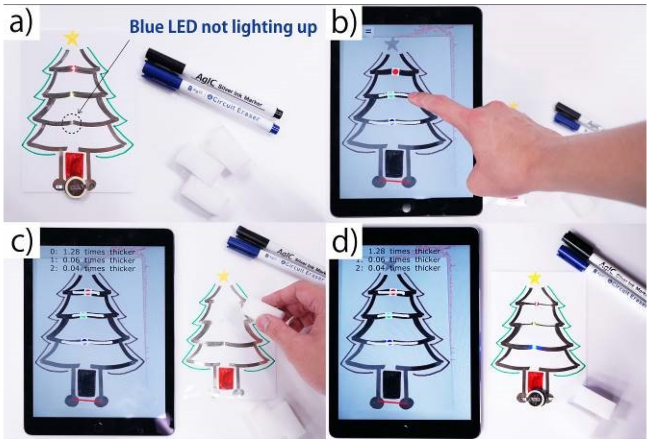


Figure 1: Using ConductAR. (a) Initial circuit created with conductive ink. (b) ConductAR recognizes traces and allows the user to select elements which can be modified. (c) Suggested modifications are presented to the user. (d) After modification with a conductive ink marker and eraser, the the real circuit works successfully.

INTRODUCTION

The use of conductive inks and pastes has become popular both in the research community and also with makers, educators and students. This approach to circuit prototyping presents a 'middle ground' between the two more conventional circuit prototyping methods of bread boarding and printed circuit board (PCB) production. It is lower cost and lower effort than producing a PCB, but more robust, compact and repeatable than using bread boards. Like bread boarding, it supports manual and iterative circuit creation by means of a hand-held conductive ink marker [9]. It is accessible and versatile, allowing a range of users to sketch, annotate, cut, fold [10] and even bind their circuits into books [15].

Despite the many advantages of conductive ink for circuit prototyping, there are some drawbacks. Two of the most serious are the variability of hand-drawn traces and the relatively high resistance compared with the equivalent copper wires used in breadboarding. For example, with silver nano-particle ink a typical sheet resistance is 0.29

ohm/sq.[1] [9], which means that a trace of 1 mm x 100 mm has resistance of 29 ohm. This is not negligible in many applications and may therefore cause unexpected results. To give a practical example, getting multiple LEDs in a circuit to light with equal brightness is much more difficult task than it appears to be because the threshold voltage required varies depending on the color and model of the LED. Furthermore, in some situations, parts of a conductive ink circuit simply won't work due to high trace resistance.

Another drawback with conductive ink based circuits is the inability to 'undo' circuit elements after they have been drawn. Despite the relatively low cost of the materials concerned, it is frustrating to make a new circuit from scratch when a small change is needed. In our experience, even knowledgeable electronic designers make mistakes and need to iterate their designs, and therefore benefit greatly from prototyping tools like breadboards which readily support alterations. This is even more important for students, where trial-and-error is an inevitable and important part of the learning process.

Based on our experience running workshops using conductive ink pens, makers and designers often seek a more spontaneous and artistic way of creating circuits. As shown in Figure 2 many users augment their conductive ink circuits with drawings made with color markers or a printer with color inks. Furthermore, they often try to make their circuits a kind of artwork, combining aspects of the functional circuit with an artistic design – both of which are created with conductive ink. These complex conductive patterns, which differ from the largely linear traces and wires traditionally associated with electronic circuits, are more difficult to debug.

In this paper we present ConductAR, a system which can recognize printed and hand-drawn conductive ink traces using a PC or a tablet with camera. It can analyze the circuit in question by calculating the resistance of each element and detecting the issues associated with creating conductive ink circuits. Circuit analysis typically takes less than a minute and the results are presented right away, supporting an iterative prototyping style. Feedback generated by ConductAR can then be directly applied to the circuit using a conductive ink marker and/or eraser.

In the remainder of this paper, we start by summarizing related work in the area of conductive ink prototyping. Next we present our design goals for ConductAR, and explain the underlying algorithm which allows us to achieve these. Then we experiment with and evaluate the algorithm, and illustrate a specific example application to show its feasibility. Finally, we conclude this paper with our plans for future work.

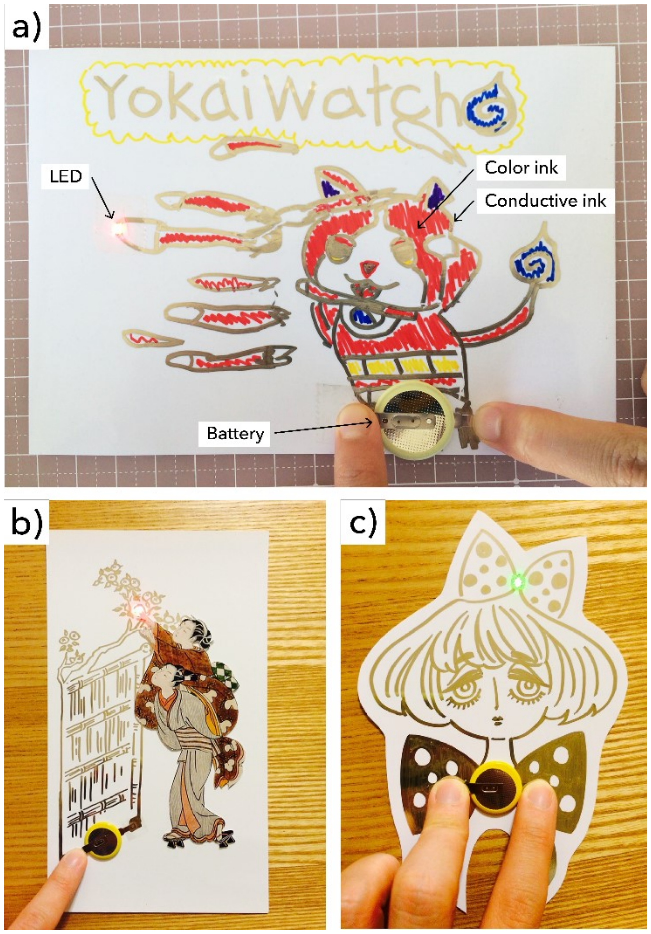


Figure 2: Example conductive ink circuits made in a workshop. Hand-drawn circuits typically combine a functional circuit with artistic elements, which often introduces electrical errors. When this happens the complexity of the conductive pattern makes it hard to analyze and repair the circuit.

RELATED WORK

Traditional electronic circuit prototyping tools

Breadboards are popular for circuit prototyping since they allow circuits to be created and modified very quickly and easily. However, they are typically quite bulky, fragile and utilitarian [6, 10]. For these reasons, PCBs provide a useful alternative. They allow much more freedom with respect to the positioning of components and result in robust and compact circuits. However, PCB design, production and assembly requires time and expertise, and the resulting circuit is hard to modify [6, 10].

Conductive tape, paste and ink for prototyping

As an alternative to PCB production, it is possible to create simple circuits using conductive foil tape [15]. The resulting circuits are robust, but they are limited in resolution and can be fiddly to construct.

An easier approach involves applying a conductive ink or a paste; several of these suitable for electronic circuit

[1] ohms/square is the unit used to represent sheet resistance. It is loosely thought of as "ohms per aspect ratio."

prototyping have been developed in recent years. Carbon-based conductive paste such as Bare Paint [2] can be applied with a brush, but unfortunately does not support narrow traces and small structures. It has a relatively high sheet resistance of 50 ohm/sq., making it unsuitable for many applications. Circuit Scribe [3] is a silver micro-particle paste which, like Bare Paint, can be used with a ballpoint pen to draw circuits onto many different substrates including regular paper. Although Circuit Scribe is a convenient way to prototype circuits, the size of the silver particles is on the order of a few microns which means it cannot be inkjet-printed.

We have previously proposed the use of self-sintering silver nanoparticle ink for circuit prototyping [1, 9, 10]. Unlike the silver-micro-particle and carbon-based inks described above, this approach cannot be used with arbitrary substrates. Instead, it performs best on specially prepared surfaces such as inkjet photo paper, where a thin layer of porous material underneath a catalytic top surface causes the nanoparticle ink to self-sinter immediately at room temperature. But it has the advantage that ink may be applied to the substrate either by hand, for example using a felt pen loaded with nanoparticle ink, or with an inkjet printer. The resulting traces are robust, flexible and have relatively low resistance at around 0.29 ohm/sq.

Adding functionality to conductive ink circuits

The approach of drawing or printing conductive ink onto a paper substrate allows a wide range of circuits to be realized. Examples include an interactive energy harvesting tool proposed by Karagozler *et al.* [7], a sticker which extends the area of a touch sensor depending on a user's preference, see Kato *et al.* [8], and a touch sensor which can be cut to suit the size and topology requirements of an application, presented by Olberding *et al.* [14].

There are also projects which show how conductive printed circuits can be combined with traditional electronic components to support a greater range of interactive scenarios. Kawahara *et al.* described how conductive epoxy and electrically conductive anisotropic transfer tape (or more simply 'Z-tape') can be used to simply 'stick' surface mount components to a conductive ink substrate [9]. Hodges *et al.* presented electronic 'stickers' which make this process more accessible to a variety of users [6] and Ramakers *et al.* have recently extended this design and fabrication approach to allow designers without technical skills to make functional interactive artifacts [16].

Conductak is a recently-developed electrically conductive sticky tack [4] which gives users another option for attaching electronic components to a conductive ink based circuit. Conductak more readily supports iterative development than conductive epoxy or Z-tape approaches. However, circuits based on Conductak can be fragile and relatively bulky.

A final area of active research relevant to this work is the combination of 3D printing with conductive circuit prototyping [17, 18]. This approach holds the promise of allowing a prototype 3D structure to be created with traces already embedded inside it, simultaneously allowing control over both form and the required electrical connectivty.

Modifying a conductive ink circuit

AgIC Inc. have commercialized a conductive ink Circuit Eraser [5] designed to erase silver nano-partible ink traces which were either created with an inkjet printer or drawn by hand. However, this marker is not ideally suited to amending wider patterns because the erasable marker tip makes the surface of the paper dirty during erasure.

We have recently proposed another method of erasing conductive silver ink traces using a melamine sponge soaked in ethanol [13]. This approach minimizes damage to the surface of the paper substrate and allows users to erase and rewrite a number of times in the same place.

Motviation for ConductAR

The circuit erasers described in the previous section allow users to prototype iteratively. Once an issue with any given circuit design has been highlighted and a solution determiend, it is straightforward to both add and remove conductive traces as required. However, detecting such issues and working out how to overcome them is not always easy, especially for non-expert users.

An obvious improvement would be to provide some kind of computer-augmented support for iterative manual circuit prototyping. Despite a number of tools such as [16] which support the development of inkjet printed circuits, we are not aware of research to support manually created circuits.

Inspired by FreeD from Zoran *et al.* [19], we seek to close this gap. FreeD is a hand-held mechanical sculpting tool which is guided and monitored by a computer and harmonizes both predesigned CAD data and user's free will. In the next section, we summarize our design goals for ConductAR which can recognize and analyze the parameters of a conductive ink circuit and aid the user in refining it.

DESIGNING AND IMPLEMENTING CONDUCTAR

Early design choices

Our goals for ConductAR were seven-fold. It needs to:

1. **Analyse arbitrary two-dimensional patterns** of conductive ink incorporated into existing circuits.
2. **Run on commodity computing devices** such as smartphones, tablets and PCs with connected cameras.
3. **Support an iterative prototyping cycle** of (re)draw – test – analyse, which allows us to do trial-and-error based development with conductive ink.

4. **Be extensible** by allowing new functions to be readily incorporated, such as future support for new circuit elements or different power sources.
5. **Work with various conductive inks** using the same basic algorithm by incorporating key ink characteristics.
6. **Be simple to** use, allowing a variety of non-experts such as designers and students to use it.
7. **Provide an accurate analysis** as a measurement device, circumventing the need for convential dedicated test equipment such as multimeters.

Modelling the circuit

There are two main challenges when implementing the analysis tool: Firstly, estimating the resistance of each part of the hand-drawn circuit pattern based on an image of the design is hard. However, without this it is not possible to model the circuit. The second challenge is reducing the computational complexity of the calculation process so that it can be performed quickly on a mobile device such as a smartphone or tablet.

In this work we evaluated and compared two basic methods for calculating resistance:

Graphication method

We call our first method graphication, because it translates the captured circuit image into a graph. As shown in Figure 3 left, we must recognize the extent of the conductive ink patterns in the input image and replace them with an idealized graph Figure 3 right. Each arc in the graph includes a resistor whose value is based on the calculated resistance of that section of the circuit. Circuit elements in the original circuit are also included in the graph. Nodes are detected by processing the input image to find any point where the conductive pattern branches. This enables us to analytically solve for arbitrary node voltages and branch currents.

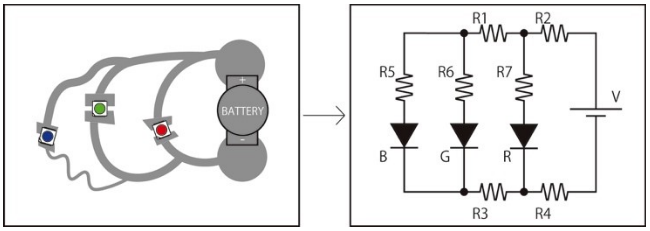


Figure 3: Graphication method. We can convert a conductive ink circuit like the one shown on the left into a graph as shown on the right. The resistors R1 to R7 represent the parasitic resistances intrinsic to each branch between nodes. We can analytically solve all the parameters, i.e. resistance, voltage and current, from this graph.

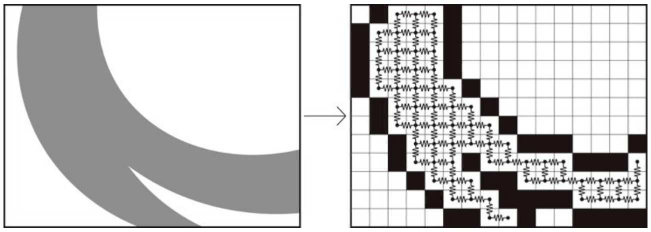


Figure 4: Finite element method. Any given circuit is divided into a group of pixels. The conductive pixels are considered to be connected to their 4-neighbours with a unit constant resistance inherent to the sheet resistance of the conductive ink used.

Table 1: Comparison of two methods for resistance calculation. Overall, graphication is more accurate at modelling a circuit – but it relies on (a) resistance values which are more easily calculated using FEM and (b) accurately detecting the position of nodes.

	Graphication	FEM
Accuracy	Analytic	Approximation
Resistance value	Undecided	Constant
Flexibility	Bad	Good

Finite element method

The second approach we tried is a finite element method (FEM). As shown in Figure 4, this method divides the input image into pixels, and regards each of these as resistance between adjacent 4-neighbours. This is an approximation of course, since in the real conductive ink circuit pixels are effectively infinitesimally small and electrons move freely in all directions.

Comparison between two methods

We show a simple comparison between the two analysis methods, graphication and FEM, in Table 1. Graphication would determine parameters very accurately if we could reliabley determine the position of nodes and the conductive paths between them. However, it is difficult to do this accurately while maintaining the requirement to run on commodity hardware. For instance, Noris *et al.* [12] uses a Quad-Core 2.66 GHz Mac Pro with 4 GB RAM to precisely determine the position of nodes and branches from a raster image of a hand-written line drawing at the size of 1024^2 pixels, and this took at least 24s to complete. Furthermore, certain designs like the example presented in Figure 5 are problematic because it is hard to tell where the nodes and branches are through image recognition. Even if we succeed in recognition, the parameters we get are limited to those between pairs of nodes rather than arbitrary points.

FEM, on the other hand, is easy to implement since the unit resistance is constant for any particular consistently-applied ink. Moreover FEM can deal with any form of circuit and calculate the resistance between arbitrary points. What is more, with this method it is possible to tradeoff accuracy and the speed of calculation simply by changing the chosen ‘pixel’ resolution, so it can be optimized to suit the task and device hardware at hand. As part of this work we successfully ported our FEM algorighm to an iPad Air 2, an off-the-shelf tablet, simly by adjusting the pixel size. Of course the calculation accuracy of FEM is inferior to graphication when the pixel size is relatively large.

On the basis of our analysis of the two methods presented above, we decided to adopt FEM for ConductAR.

Mathematical explanation of FEM

We present a mathematical explanation of FEM here. Firstly, we divide the pattern into pixels and model them as they have a constant value of resistance with 4-neighbours as mentioned in the previous subsection and shown in Figure 4. Next, we select a start and end point for calculating resistance, then we calculate the voltage at every pixel when a certain current flows between the two points, by means of the nodal analysis method.

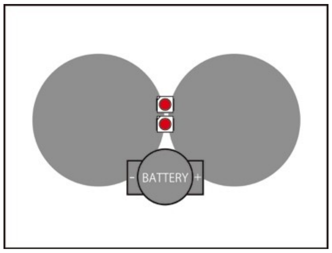


Figure 5: An example circuit which cannot be recognized by graphication. This is a parallel circuit composed of two LEDs and one battery. However, it is difficult to isolate the correct of of nodes to distinguish the branches to each LED or the resistance between nodes.

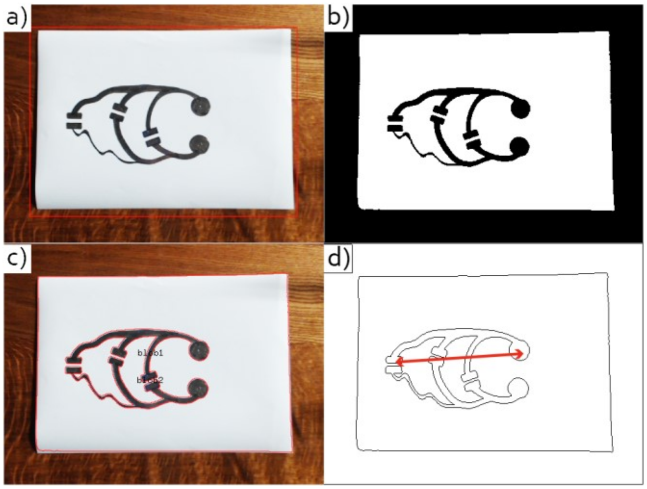


Figure 6: The sequence of calculating resistance: (a) A photogrpahic image of the circuit. (b) Binarized image. (c) Extracted contours are overlaid on the original image. (d) Two points are selected and the resistance between them calculated.

Now let $n \in \mathbb{R}$ be the total number of pixels, $\boldsymbol{i} \in \mathbb{R}^n$ is the current vector applied to every pixel, $\boldsymbol{v} \in \mathbb{R}^n$ is the voltage vector at every pixel, and $\boldsymbol{G} \in \mathbb{R}^{n \times n}$ is the nodal matrix. Then

$$\underbrace{\begin{bmatrix} g_{11} & -g_{12} & \cdots & -g_{1n} \\ -g_{21} & g_{22} & \cdots & -g_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -g_{n1} & -g_{n2} & \cdots & g_{nn} \end{bmatrix}}_{G} \underbrace{\begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}}_{v} = \underbrace{\begin{bmatrix} i_1 \\ i_2 \\ \vdots \\ i_n \end{bmatrix}}_{i}$$

where $\boldsymbol{i}$ is the sparse vector which has at most twice as many nonzero elements as the number of current sources, and g_{ij} is

$$g_{ij} = \begin{cases} \text{Sum of adjacent conductance} & (i = j) \\ \text{Conductance from pixel } i \text{ to pixel } j & (i \neq j) \end{cases}$$

$$= \begin{cases} g, 2g, 3g, 4g & (i = j) \\ 0, g & (i \neq j) \end{cases}$$

where g is a unit conductance attached to where one conductive pixel is adjacent to another one. g_{ij} is g, $2g$, $3g$ or $4g$ when $i = j$, and 0 or g when $i \neq j$ because we adopt the 4-neighbour model in this paper. This approximation results in $\boldsymbol{G}$ having at most 5 non-zero elements in every row.

$\boldsymbol{v}$ is a variable and estimated by solving the inverse matrix problem

$$\boldsymbol{v} = \boldsymbol{G}^{-1}\boldsymbol{i}$$

Having estimated $\boldsymbol{v}$, the resistance from pixel $\boldsymbol{i}$ to pixel $\boldsymbol{j}$ can be calculated as

$$r_{\mathrm{ij}} = \frac{v_{\mathrm{i}} - v_{\mathrm{j}}}{i_{\mathrm{j}}}$$

Ordinarily, this calculation needs $O(n^2)$ memory for matrix $\boldsymbol{G}$. However, by taking advantage of the sparse nature of matrix $\boldsymbol{G}$ which has at most 5 non-zero elements, including a diagonal element and 4-neighbours in every row, we reduce this to only $O(n)$. Furthermore, $\boldsymbol{i}$ is also an absolutely sparse vector, so we can also reduce the time complexity, using an algorithm which leverages this sparsity. In this paper we use SparseLU from the C++ library Eigen to solve the inverse matrix problem.

The complete algorithm

Figure 6 shows the sequence of steps involved in a resistance calculation:

a) Capture an image of the paper with conductive patterns using a camera attached to a PC or tablet.

b) Binarize the image.

c) Extract contours, detect the closed curve, and consider the inside of the curve as a conductive pattern. N.B. in case one conductive pattern is perfectly encompassed with other patterns, we consider the odd number of contours as conductive patterns, and even number of contours as paper surfaces, counting from the paper edge contour to inside of it.

d) Ask the user to indicate where the terminals of the battery will be connected to calculate the absolute resistance between them and thereby simulate the relevant currents.

EXPERIMENTING WITH CONDUCTAR

In this section, we experiment with and evaluate the basic function to calculate resistance between two arbitrary points, in terms of its accuracy and speed. We conducted these experiments using silver nanoparticle ink NBSIJ-MU01 and coated paper NB-RC-3GR120 from Mitsubishi Paper Mill [11]. Recognition was done using a Dual-Core 2.4 GHz Mac Book Pro with 8 GB RAM and a Logicool C920 web camera.

Evaluation on the accuracy

Firstly, we printed the silver ink pattern shown in Figure 7 left as described in [9] and then captured and processed an image as described above. We chose two lattice points at random and then calculated the resistance between these points. After that, we measured the true resistance using an Agilent 34410A multimeter and compared the result with the value calculated by ConductAR. The result of 30 runs of this process is shown in Figure 7 right. From this result, the error rate

$$\mathrm{ER} = \mathrm{abs}\left(\frac{\mathit{calclated\ value} - \mathit{measured\ value}}{\mathit{measured\ value}}\right)$$

was calculated to be less than 7% as a result of t-test (with a significance level $p<0.05$). We belive this error is caused by the slightly uneven drying of the conductive ink, the approximation of 4-neighbours and an imperfect camera image. The influence of resistance variation is discussed later.

Evaluation on the speed

Next, we evaluated the resistance calculation time by changing the size of the silver ink pattern. Figure 8 left shows the pattern we used for this evaluation; the dimensions of the uppermost trace are 5 mm x 150 mm and those underneath are 2, 5, 10 and 20 times as large respectively. Figure 8 right plots the calculation time against the number of pixels. As expected, the time to generate the matrix was $O(n^2)$, whilst the time to solve it was much less than the typical time complexity of the inverse matrix problem $O(n^3)$, due to the calculation taking advantage of the sparsity of matrix $\boldsymbol{G}$ and vector $\boldsymbol{i}$.

CONDUCTAR IN PRACTICE

The basic algorithm we mentioned above is simple but useful for 'debugging' various circuit issues. In order to show its feasibility, we implemented a function which automatically suggests the ideal line width to adjust the brightness of several LEDs wired in a parallel circuit. In our experience running workshops with makers and students this is one of the most frequently asked questions.

Lighting multiple LEDs in a single circuit with equal brightness is a much more difficult task than it appears to be. This is in part because the threshold voltage of each LED is different depending on its color and exact specification. The current passing through each LED, which controls its brightness, is determined by a combination of the threshold voltage and the resistance between the LED and the power source.

Modeling LEDs

To make the problem tractable, we applied a number of constraints. We limited the LEDs to three specific parts: one red, one green and one blue (part numbers LTST-C150CKT, LTST-C150KGKT and LTST-C150TBKT respectively). This reduces the range of LED characteristics we need to account for. Also, because an LED is a non-linear circuit element, we adopted the two following approximations:

1) The LED is treated as a voltage source when the applied voltage is bigger than its threshold voltage.

2) The LED's brightness is proportional to the current passing through it.

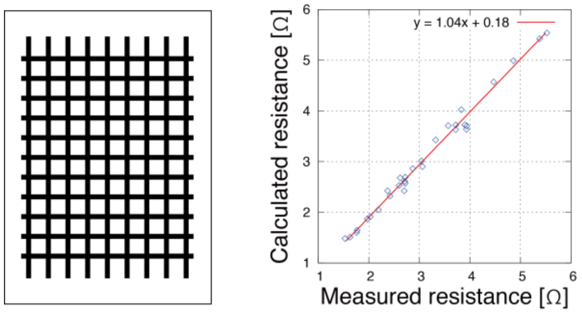


Figure 7: The A4 sized printed pattern to evaluate accuracy of the proposed method (left) and calculated resistance against true values measured by multimeter Agilent 24410A (right).

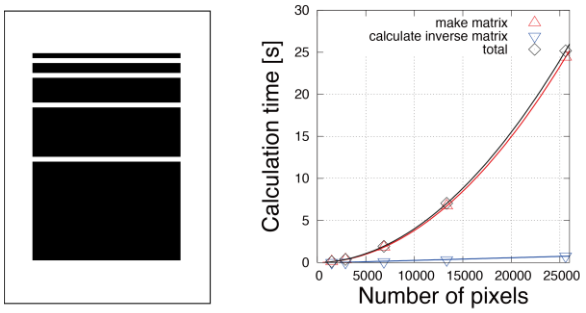


Figure 8: The A4 sized printed pattern to evaluate speed of the proposed method (left) and the calculation speed against the number of pixels (right).

The justification for these approximations are based on Figure 9 and Figure 10 which are taken from the datasheets for the three supported LEDs. Figure 9 shows the LEDs' current-voltage property. It shows that the current is tiny when the voltage is lower than the threshold voltage, whilst the current increases dramatically when applying voltages over the threshold. This led us to adopt approximation (1). From Figure 10, which shows the LEDs' brightness-current property, it's clear that the brightness is roughly proportional to the current, especially when the current is under 30 mA. We adopted approximation (2) for this reason.

Algorithm to average the LEDs' brightness

The approximation we adopted in this paper simplifies the problem "to average the brightness of LEDs" into "to apply a voltage exceeding the threshold voltage to each LED, and supply each with the same amount of current."

Firstly, we need to decide on the target current for all the LEDs. As shown in Figure 11, the branches including the LEDs are translated into current sources which have the same value, assuming to let the current applied to all the LEDs to be I_{LED}. Secondly, let the voltage of a coin cell battery to be V_{BATTERY}, and the internal resistance of a coin cell to be R_{BATTERY}, then the battery can be equivalently transformed into the current source using Norton's theorem, which in turn can be used in the nodal equation, and its values are

$$\begin{cases} I = \dfrac{V_{BATTERY}}{R_{BATTERY}} \\ R = R_{BATTERY} \end{cases}$$

Using this equation we can evaluate the voltage and resistance at every point of the circuit using the same means as shown above.

Returning to our test circuit from Figure 11, we can now determine the voltage between branches including the LED. To set the voltage applied across red LED to V_1, we need:

$$V_1 \geq V_{THREASHOLD-R}$$

where $V_{\mathrm{THRESHOLD\text{-}R}}$ is the threshold voltage of a red LED. If this condition is not met, either I_{LED} will be too large or V_{BATTERY} too low to solve in this situation, so we have to change the parameter and solve it again. In this paper, we gradually make the I_{LED} smaller in order to solve the nodal equation. If the voltages between the branches meet the conditions above, the resistance of the branch we are focusing on is evaluated as and expected to be

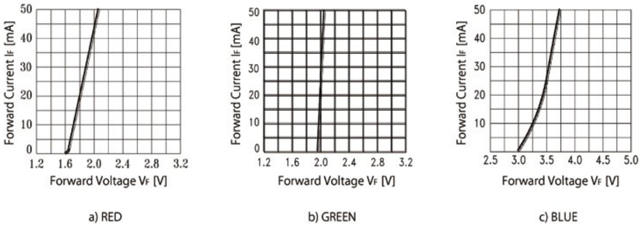
$$R_1 = \frac{(V_1 - V_{THRESHOLD-R})}{I_{LED}}$$

When we adjust the line width of the branch as the resistance becomes R_1, the current applied to the red LED becomes I_{LED}. The width of other lines including LEDs can also be adjusted in the same way.

As the resistance of the each branch can easily be estimated based on the basic algorithm mentioned in the previous section, then the software calculates the ratio of current to ideal resistance. This is shown in Figure 1. The user can change the line width according to this ratio using a conductive ink marker and a conductive ink eraser.

The result of the LED averaging experiment

Figure 12 illustrates the process of using ConductAR to determine how to adjust trace width to equalize the brightness of several LEDs. Firstly, the type of each LED and the battery are placed using the ConductAR user interface. Secondly, the user indicates which line(s) may be changed in width. Finally, ConductAR provides written feedback indicating how much thicker or thinner each line needs to be. This allows users to specify which lines they are happy to alter vs those which they do not want to change.

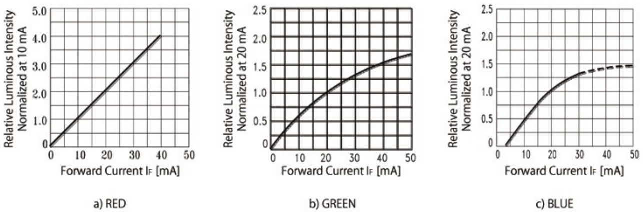


a) RED

b) GREEN

c) BLUE

Figure 9: Current-voltage property of the three supported LEDs.



a) RED

b) GREEN

c) BLUE

Figure 10: Brightness-current property of the three supported LEDs.

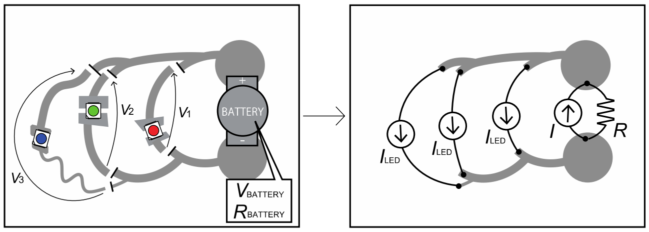
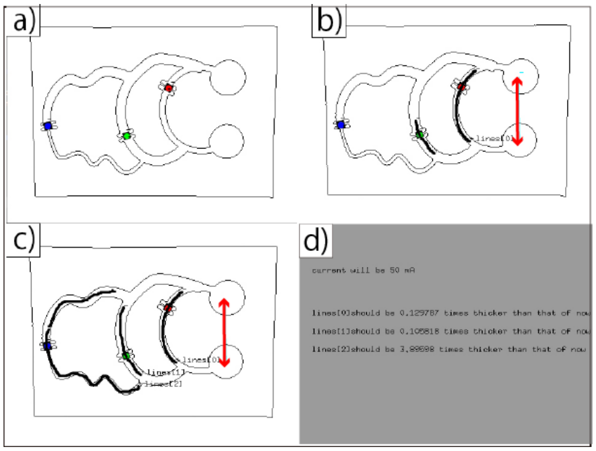


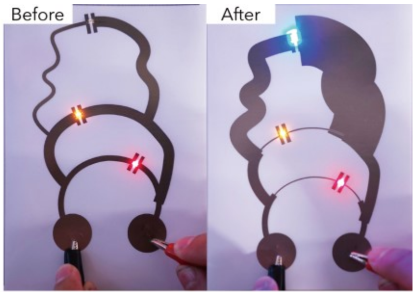
Figure 11: Applying the same current to all LEDs. The battery is modelled as a current source in order to use the basic algorithm of calculating the resistance by the nodal equation.



```
current will be 50 mA


lines[0]should be 0.129787 times thicker than that of now
lines[1]should be 0.105818 times thicker than that of now
lines[2]should be 3.89698 times thicker than that of now
```

Figure 12: Using ConductAR to calculate how to adjust trace widths in a hand-drawn circuit to equalize the brightness of three LEDs. (a) Having taken a photo and binarized the image as shown in Figure 6**, the user indicates the position and colour of the LEDs using the ConductAR user interface. (b) The user then drags a red arrow to indicate the position of the battery. (c) Individual traces which the user is happy to modify are identified. (d) ConductAR provides text-based feedback indicating how much thicker or thinner each trace should be.**



	Before		After	
	Voltage	Current	Voltage	Current
Red	2.11 V	73.59 mA	1.95 V	35.68 mA
Green	2.05 V	117.8 mA	2.14 V	42.21 mA
Blue	2.36 V	1.402 μA	3.32 V	27.14 mA

Figure 13: The photographs and table of the circuit before and after applying the function. Before) the blue LED is not lit, since the voltage and current is too low, 2.36 V and 1.402 µm respectivly. After) the blue LED is lit and the current goes through each branch on the same order.

The photographs in Figure 13 show the circuit before and after using ConductAR. In this experiment, we used A6 sized paper. The circuit was modified and then re-printed according to the ratio of line width we got from the program. By using printed conductive ink patterns instead of hand-drawing, we avoided any error which might be introduced by hand-drawing. We set V_{BATERY} to 6 V and R_{BATTERY} to 6 ohm to model two serially stacked CR-2032 coin cell batteries, since the voltage of a single 3.0 V battery is insufficient to light the blue LED whose $V_{\mathrm{THRESHOLD\text{-}B}}$ is 3.4 V.

The table in Figure 13 compares the voltage across and current passing through each LED before and after the circuit was modified. Before the correction, the voltage applied across the blue LED is less than its threshold of 3.4 V, so very little current flows. After modification the voltage applied across all the LEDs is over their respective thresholds. The current flowing through each LED was closer to the target value 50 mA.

We consider the cause of error between the target current value and the actual current value here. According to Figure 9, LEDs' current-voltage property resembles to the voltage source especially when LED is green and the error rate was 16%, whilst red LED and blue LED are a little further from the property of the voltage source, resulting in the bigger error rate, 46% with blue LED. However there is a big manufacturing variation in device characteristics even among the same kind of LED, so we consider it is enough to adjust the current in the same order as the target value.

ConductAR parameters

Table 2 shows the parameters used for this evaluation of ConductAR. We used silver nanoparticle ink NBSIJ-MU01 and a corresponding unit conductance in the nodal matrix of g = 2.25 or 2.4 S (for A4 and A6 paper respectively). We can apply our algorithm easily to the other conductive inks and pastes, just replacing the value of g into the others which are optimized for the ink we use.

Table 2: The parameters used in this paper.

Camera width	480 pixel
Camera height	360 pixel
Unit conductance g (A4 sized)	2.25 S
Unit conductance g (A6 sized)	2.4 S
Voltage for 1 button battery V_{BATTERY}	3.0 V
Inner resistance for 1 button battery R_{BATTERY}	3 Ω
Red threshold $V_{\mathrm{THRESHOLD-R}}$	1.5 V
Green threshold $V_{\mathrm{THRESHOLD-G}}$	1.7 V
Blue threshold $V_{\mathrm{THRESHOLD-B}}$	3.4 V

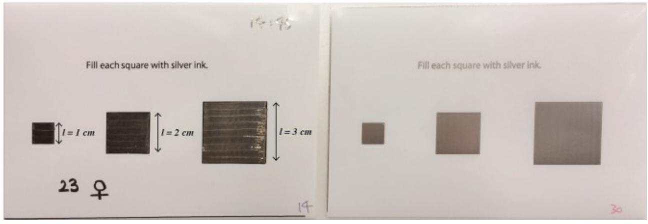
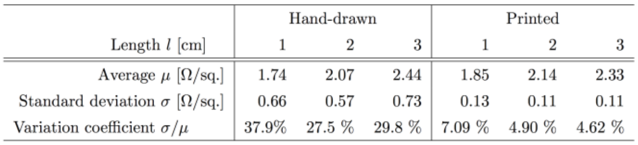

Fill each square with silver ink.
l = 1 cm
l = 2 cm
l = 3 cm
23 ♀
14
Fill each square with silver ink.
30

Figure 14: A hand-drawn sample (left) and the equivalent printed pattern (right). The uneven surface of the hand-drawn pattern is clearly noticeable when compared with the smooth surface of the printed pattern.

Table 3: Variation of sheet resistance between hand-drawn samples and printed samples.



	Hand-drawn			Printed		
Length l [cm]	1	2	3	1	2	3
Average μ [Ω/sq.]	1.74	2.07	2.44	1.85	2.14	2.33
Standard deviation σ [Ω/sq.]	0.66	0.57	0.73	0.13	0.11	0.11
Variation coefficient σ/μ	37.9%	27.5 %	29.8 %	7.09 %	4.90 %	4.62 %

DISCUSSION

In this section, we discuss the feasibility of our methodology for both hand-drawn traces and printed patterns. Compared with the printed patterns used in the previous section, two additional factors influence the resistance of hand-drawn traces.

The first and most significant factor is overdrawing. When users draw a trace with a marker, in our experience they frequently paint the same area more than once to ensure complete coverage. Overdrawing makes the conductive pattern thicker leading to a decrease of sheet resistance [9].

The second factor is variation of drawing pressure. The pressure imposed on the marker changes according to drawing style, resulting in different amounts of conductive material on the substrate, in turn changing sheet resistance.

In order to determine how much these factors affect the sheet resistance of the conductive ink circuit, we ran a study with a variety of users, some familiar with conductive ink markers and others not (age from 20 to 27 years, n=30). We asked the subjects to fill rectangles of size 1 cm x 1 cm, 2 cm x 2 cm, and 3 cm x 3 cm as shown in Figure 14 left. Equivalent printed patterns with identical sizes were used as a control (see Figure 14 right). The resistance was measured 24 hours after each pattern was drawn or printed.

Table 3 shows the variation of resistance between hand-drawn and printed patterns. It shows a much bigger variation among hand-drawn patterns. Printed patterns, on the other hand, have a variation coefficient between 4.6% and 7.1%, consistent with the 7% error rate measured earlier in this paper.

These results suggest us that the calculation of resistance in hand-drawn circuits will make an error on the order of tens and is not as accurate as we calculate in printed circuits. However, as for some applications which take advantage of the relative resistance ratio, not absolute one like equally lighting up LEDs, we don't worry about this high error rate. Figure 1 also demonstrates the feasibility of our method, showing that conductAR is applicable to hand-drawn circuit if we want to some function associated with the relative sheet resistance.

CONCLUSION AND FUTURE WORK

In this paper, we have motivated, described and evaluated ConductAR, a tool to recognize, analyze and visualize parameters of conductive ink circuits. This tool achieved seven goals we mentioned above.

1. It can deal with arbitrary two-dimensional patterns.
2. It works not only on PCs but also on tablets, which are more readily available in many cases.
3. It can analyze hand-drawn circuits, thereby supporting a highly iterative design style.
4. We imagine that additional components can be incorporated in a similar way to the LED support already implemented.
5. This algorithm is applicable to various kinds of conductive inks and pastes with different sheet resistances.
6. Estimation of sheet resistance can be done with minimum effort.
7. The error rate of resistance estimation was under 7% ($p<0.05$).

We believe ConductAR helps novice users with little knowledge or few tools to analyze the circuit, and enhances the feasibility of the iterative design with conductive ink circuits.

There are several areas of future work. The first one is implementation of other diagnosis and assist tools based on the basic resistance estimation algorithm introduced in this paper. For example, we expect that the ability to detect and highlihght short circuits will be very useful. An ability to calculate circuit inductance and capacitance, thereby supporting AC circuit analysis, is planned future work.

A second area for improvement is the provision of feedback. In the implementation presented here, the resistance estimation and suggestions for modification are simply presented as written text. Our intuition is that visual feedback would be easier for users. We imagine an interface where the calculated current is visually overlaid on the circuit. It would also be useful to record a digitized version of the hand-drawn circuit patterns at the time when we recognize the circuit. This would enable users to rapidly replicate their hand-drawn designs with an inkjet printer.

Finally, although we have reported the resistance estimation error was 7%, this number can differ depending on the type of conductive ink and paste used. In the case of silver-nano particle ink, conductivity is relatively stable compared to Circuit Scribe and Bare Paint. However the resistance does change if the pattern is overprinted. In addition, extremely dry or moist environmental conditions may also affect resistance. To overcome this problem, a simple caliblation process may be required.

ACKNOWLEDGMENTS

We would like to thank Myamo, who gave us two great pieces of artwork for this paper, and the anonymous reviewers who gave us a lot of useful technical suggestions and comments.

This research was supported by JST, PRESTO.

REFERENCES

1. AgIc, Inc. 2015. http://agic.cc/.
2. Bare Conductive. 2015. http://www.bareconductive.com/.
3. Circuit Scribe. 2015. https://123d.circuits.io/shop/circuitscribe.
4. Conductak: Stick Circuits Anywhere. 2015. https://www.kickstarter.com/projects/1474433053/conductak-stick-circuits-anywhere.
5. Erasable Circuit Marker: Draw and Edit Circuits Smoothly. 2015. https://www.kickstarter.com/projects/1597902824/erasable-circuit-marker-draw-and-edit-circuits-smo.
6. Hodges, S., Villar, N. and Chen, N. *et al.* Circuit stickers. *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*, (2014).
7. Karagozler, M., Poupyrev, I., Fedder, G. and Suzuki, Y. Paper generators. *Proceedings of the 26th annual ACM symposium on User interface software and technology - UIST '13*, (2013).
8. Kato, K. and Miyashita, H. Extension sticker. *Proceedings of the 33rd annual ACM conference on Human factors in computing systems - CHI '15*, (2015).
9. Kawahara, Y., Hodges, S., Cook, B., Zhang, C. and Abowd, G. Instant inkjet circuits. *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing - UbiComp '13*, (2013).
10. Kawahara, Y., Hodges, S., Gong, N., Olberding, S. and Steimle, J. Building Functional Prototypes Using Conductive Inkjet Printing. *IEEE Pervasive Computing. 13*, 3 (2014), 30-38.
11. Mitsubishi NanoBenefit 3G Series. 2015. http://www.mitsubishiimaging.com/digital-imaging-diamond-jet-NANOINK.html.
12. Noris, G., Hornung, A., Sumner, R., Simmons, M. and Gross, M. Topology-driven vectorization of clean line drawings. ACM Trans. Graph. 32, 1 (2013), 1-11.
13. Narumi, K., Shi, X., Hodges, S., Kawahara, Y., Shimizu, S. and Asami, T. Circuit Eraser: A Tool for Iterative Design with Conductive Ink. *Extended Abstracts: 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI'15*, (2015).
14. Olberding, S., Gong, N., Tiab, J., Paradiso, J. and Steimle, J. A cuttable multi-touch sensor. *Proceedings of the 26th annual ACM symposium on User interface software and technology - UIST '13*, (2013).
15. Qi, J. and Buechley, L. Sketching in Circuits. *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*, (2014).
16. Ramakers, R., Todi, K. and Luyten, K. paperPulse. *Proceedings of the 33rd annual ACM conference on Human factors in computing systems - CHI '15*, (2015).
17. Sarik, J., Butler, A., Villar, N., Scott, J. and Hodges, S. Combining 3D printing and printable electronics. *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction - TEI '12*, (2012).
18. Voxel8: 3D Electronics Printing. 2015. http://www.voxel8.co/.
19. Zoran, A. and Paradiso, J. FreeD. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*, (2013).