Abstract: Printable electronic circuits have received a big adoption from a variety of users such as researchers, hobbyists, designers, and children. The designers want to use electronic circuits along with graphic design yet focus on the creativity and aesthetics of the design. However, current technology requires them to take care of the discouraging electrical behaviors of the circuits. Taking the task of lighting up a bunch of light-emitted diodes (LEDs) as an example, it sounds simple, but is posing significant challenges for inexperience users. Given the non-negligible resistance of conductive ink, it is not straightforward to generate a pattern that lights up the LEDs evenly. Furthermore, a large number of LEDs make it difficult and error-prone to wire them efficiently. It is possible to try existing auto-routers in computer aided design tools to automatically route these LEDs. However, being optimized to make circuits with highly conductive materials such as copper and gold, these auto-routers ignore the intrinsic resistance of the conductive ink. In this paper, we propose an LED auto-router which computationally generates a conductive pattern to balance brightness of multiple LEDs without the need of additional resistors. Our routing algorithm is based on the traveling salesman problem to find the shortest cross-less path through the LEDs, and thus minimize the ink consumption. It then adjusts resistances of the conductive patterns to regulate the current which flows through each LED.

Key Words: conductive ink, auto-router, traveling salesman problem, LED, uniform brightness.

1. Introduction

Sintering-free conductive ink [1], thanks to its easy accessibility, is being used by many groups of researchers, designers and hobbyists to design and rapidly fabricate interactive applications. Conductive ink filled pens [2]–[4] are low-cost and convenient solutions for hand-drawing of working electronic circuits. For more complicated conductive patterns, the conductive ink can be put into an inkjet printer to quickly print the conductive patterns. Researchers take advantage of this to rapidly prototype human-computer interaction related applications such as Extension Sticker [5], Inkantatory paper [6], and touch interfaces [7],[8]. Besides, designers also use conductive inkjet ink to create interactive lighting applications with light-emitted diodes (LEDs) as in [9]. Problems arise here when the number of LEDs increases. Wiring multiple LEDs is complicated and tedious. Furthermore, intrinsic resistance of the conductive inkjet ink makes the resistance of the printed pattern non-negligible, thus, causes an uneven lighting up of the LEDs as shown in Fig. 1 (b).

An example of art-works with LEDs is “Pu Gong Ying Tu (Dandelion Painting)” [10] (Fig. 2). Here, a lot of LEDs were placed behind a painting to bring in interactivity with the subject of the picture. It was manually designed with copper tape and a large number of LEDs. As shown in Fig. 2 (b), the copper tape-based electronic circuit behind the picture is complicated and would take a lot of time to wire it manually. Using conductive ink to print the printable digital design of the circuit will obviously simplify the labor of making and replicating this work. Some conductive ink existed at the time of the project, but the author of this work would not have done it using conductive ink due to the complication of designing a cross-less wiring pattern. Moreover, using conductive ink instead of copper would require a thorough consideration to overcome the troubles caused by the resistance of the conductive ink.

In this paper, we propose the implementation of an auto-
wire, the dimension of these metal patterns are normally put aside as long as the circuit is routable. However, this is not applicable in case of using conductive inkjet ink because of the non-negligible intrinsic resistance in the conductive ink. Ignorance of this resistance leads to generation of conductive patterns that cannot light up multiple LEDs evenly as shown in Fig. 1 (b).

2.2 Tree Based Routing Algorithm

In addition to A* search based algorithms, routing problems can be addressed with tree-based algorithms. Zero-skew clock routing is a typical problem in designing high-performance synchronous VLSI systems. In these systems, it is crucial to assure that the arrival times of a clock signal from a clock source to synchronous elements are the same. This can be analogized to our problem of LEDs brightness balancing in terms of providing the same current to all the LEDs in the circuit. Among different approaches to the zero-skew clock distribution problem such as in [22]–[24], a clock-trees-based algorithm [25] is widely used. However, a tree-based approach will not be a good solution to our LEDs brightness problem. The reason is that a tree-based algorithm will require multiple layers for routing. Although there are ways to fabricate double-sided circuits with conductive ink such as in [26], [27], they complicate the fabrication process significantly and make it difficult to optimize the ink consumption. Routing and balancing brightness of the LEDs in a single-layer circuit are more desirable.

2.3 Design Assistant in Electronic Circuits Prototyping

Since the emergence of consumer grade conductive inks, there are many tools to assist designing with these new materials. Circuit Stickers [28], Ellustrate [29], PaperPulse [30], and Midas [31] each proposed a development tool based on A* search and Lee’s algorithm [14] to support/automate the wiring of electronic components with conductive inks/pastes. However, these tools neither are fully automated nor consider the resistance of the conductive ink, which, as we mentioned above, severely affects the performance of the printed electronic circuits.

To deal with troubles caused by the resistance of the conductive ink, ConductAR [32] used computer vision techniques to estimate the resistance of hand-drawn or printed conductive ink patterns. Thus, it suggests an optimized trace width to each path of the circuits. However, it is an ad-hoc tool that cannot generate 2D paths to connect multiple LEDs on demand.

3. Problem Formulation and Solution

In its normal working zone, the brightness of an LED is roughly proportional to the current that flows through it. Therefore, in order to achieve an even lighting up of multiple LEDs, the circuit needs to assure that all the LEDs get the same amount of current. This can be guaranteed by chaining all LEDs serially and regulating the current by a ballast resistor. However, this approach requires a high voltage of the power source as the number of LEDs increases. In this case, a more reasonable approach is to put all the LEDs in parallel to the power source and regulate the current through each LED with ballast resistors. In a parallel schematic, these ballast resistors play a very important role in regulating and protecting the LEDs. Without them, due to the LEDs’ vicious loop of heating - increasing...
forward current, all the LEDs will be sequentially over-current and burn-out. As mentioned above, the existing auto-routers assume that resistance of the metal pattern is negligible and additional ballast resistors will be added to regulate the LEDs. Given the non-negligible intrinsic resistance of conductive ink, the first assumption does not hold. The second assumption about additional ballast resistors is discouraging. Adding rigid ballast resistors leads to the needs of calculating value of each resistor as well as the labor of attaching all of them to the circuit. Hence, we lose the advantage of using conductive inkjet printing technology. Instead, by means of computational design, the conductive inkjet ink can be used to act as a printable ballast resistor for each LED. We can take advantage from the apparent disadvantage of the conductive inkjet ink (i.e., high intrinsic resistance) and turn it into our favor.

A naive solution to averaging brightness of the LEDs could be rectilinearly connecting the power source to each LED as shown in Fig. 3. Apparently, this approach does not minimize the amount of ink consumed as well as the routing space. Furthermore, it does not guarantee brightness balance when there are multiple LEDs on a same level branch.

### 3.1 Problem 1 - Routing

**Problem statement:** Given a group of $n$ LEDs $\{L_i | i = 1, 2, \ldots, n\}$, and the two poles of a power source $\{L_0, L_{n+1}\}$ respectively denoting anode and cathode as shown in Fig. 4(a), find a route that connects the LEDs in parallel to the power source and minimizes the ink consumption.

**Solution:** For the goal of minimizing the amount of conductive ink consumed, we will find the shortest path to connect all the LEDs in parallel to the power source. Although this can be achieved by using the $A^*$ search algorithms, they are optimized to search for the shortest path between two points on a grid. $A^*$ search algorithms do not search for the shortest path through multiple points (i.e. LEDs). Building the minimum Steiner tree of all LEDs and the power source is the best solution to minimize the total length of connections [33]. Nevertheless, it is not feasible to route the two separated anode net and cathode net on a single layer circuit. To deal with the routing problem, we propose to solve the Euclidean-metric traveling salesman problem corresponding to the set of LEDs and power source poles. Node $L_0$ and $L_{n+1}$ act as the fixed starting and ending points of the TSP path. The TSP path will go from $L_0$, through all LEDs $L_i$ to $L_{n+1}$. Because searching for the TSP path is a NP-complete problem [34], it is more practical to approximate the shortest path using heuristic algorithms such as Christofides’ algorithm [35], the simulated annealing algorithm [36], and the genetic algorithms (GAs) [37]. We implement the TSP solver using the genetic algorithm because it is flexible in the evaluation of the best route.

In a GA based TSP solver, an *individual* is a candidate for the shortest path through all the LEDs. It will be encoded as
the array shown in Fig. 5. In this encoding, the starting and ending points are always fixed at the head and tail of the array. Only the order of the elements in range of first to nth \((L_{0,1,2,...,n})\) are alterable. For every generation of the GA, we generate a population of \(p = 50\) individuals by evolving from the population of the previous generation. Evolution through generations involves running crossover, mutate, and selection operations on the current population. To evaluate each individual, we define a fitness function as the reciprocal of the total length of the corresponding tour [38]. At each generation, we extract the fittest individual (i.e., the one with the highest fitness) for seeding the next generation. We stop the TSP solver when there is not better individual for converge\(_{\text{threshold}} = 10n\) consecutive generations, or when the evolution reaches \(\text{MAX\_GENERATION} = 100n\) generations. Choosing of these thresholds is done through empirical experiments. Detail parameters of the GA TSP solver are listed in Table 1. After stopping the GA evolution, we use a 2-opt algorithm [41] to eliminate all the crossing edges, if any, in the path generated by the GA to make it the shortest path as shown in Fig. 4 (b). Finally, after rotating each LED to an appropriate angle as described in Section 4.2, our algorithm connects all the LEDs along the found TSP path as shown in Fig. 4 (c).

3.2 Problem 2 - Brightness Balancing
The goal of lighting up all the LEDs evenly is achievable when each LED receives the same amount of current. This is fulfilled by adjusting the resistance from the power source to each LED. Because we put all the LEDs in parallel to the power source, the topology as in Fig. 4 (c) has an equivalent circuit as shown in Fig. 6.

In case that the electronic circuit consists of the same color LEDs, all the LEDs have the same magnitude of forward current \(I_F\) and forward voltage \(V_F\). The brightness balancing problem can be stated as follows:

**Problem statement:** Given a set of \(n\) LEDs that are connected to a \(V_F\) power source as shown in Fig. 6, find all the resistances \(R_{0,1,2,...,2n-1}\) that regulate the current through each LED to the common forward current \(I_F\).

**Solution:** The equivalent circuit in Fig. 6 is just a bunch of LEDs that are put in parallel to the power source. The values of the resistors are retrieved by using the nodal analysis on the equivalent circuit. In order to balance the currents through the single color LEDs (i.e., \(I_i = I_F, i = 1, 2, \ldots, n\)), the resistances must satisfy the following two equations:

\[
\begin{align*}
(n-i)R_i &= iR_{n-i+1},  \\
R_0 + R_{2n-1} &= \frac{V_F - V_{F1} - I_F \sum_{i=1}^{n-1} (n-i)R_i}{nI_F}.  
\end{align*}
\]

Because there are many sets of resistances \(R_i\), we assume that \(R_i = R_{2n-1-i}\) where \(i = 0, 1, \ldots, n - 1\). Once the user chooses the input voltage \(V_0\), our algorithm will calculate the values of all the resistances \(R_{0,1,2,...,2n-1}\).

**Conductive patterns as printable resistors:** Once the values of all the resistors are derived from the procedure above, we can print these resistors with the conductive ink. For an \(l \times w\) conductive strip, the resistance between two ends is \(R = R_{1,1} \frac{l}{w}\), where \(R_{1,1}\) is the sheet resistance of the conductive inkjet ink. With \(R\) as derived above, \(R_i\) fixed to each type of the conductive inkjet ink, and \(l\) as the distance between two consecutive LEDs on the TSP path, we can easily calculate the width \(w\) of each conductive segments. Figure 1 (a) and Fig. 4 (d) show an output of our routing algorithm.

3.3 Improved Routing
Our routing algorithm consists of two steps as follows:

- **Step 1:** Find TSP path from the anode of the power source, through all LEDs, to the cathode of the power source. Then connect them so that the LEDs are put in parallel to the power source.
- **Step 2:** Solve the equivalent circuit to get the resistance of all the resistors. Then print these resistors using conductive inkjet ink.

However, taking advantage of GA, we can improve the routing algorithm by combining these two steps into one. Instead of searching for the TSP route, we search for the path that minimizes ink consumption. In order to implement this, we replace the distance based fitness function of our GA implementation with a printed area based fitness function. The printed area based fitness function is defined as reciprocal of the printed area. At each generation of GA, we do as Step 2 above and then calculate the printed area. Smaller printed area routes will be selected to the next generation.

3.4 Multiple Color LEDs
In Section 3.2, we solved the problem of brightness balancing for single color LEDs. That is, all the LEDs in a circuit are the same color which results in the same requirement of forward voltage and forward current. It is possible to solve the problem in a circuit which have different color LEDs. In that case, depending on its color, each LED has different requirements of forward voltage and forward current. The problem can be stated as follows:

**Problem:** Let forward voltage and forward current through LED \(L_i, i = 1, 2, \ldots, n\) be \(V_i\) and \(I_i\), voltage of the power source be \(V_0\). Knowing the order of LEDs which should be connected in parallel to the power source, generate conductive inkjet patterns so that the voltage drop on each LED \(L_i\) is \(V_i\).

The circuit of multiple color LEDs put in parallel to the power source has an equivalent circuit as in Fig. 7. With known \(V_0, V_i,\) and \(I_i\), the resistors are solvable using the nodal analysis. Let \(I_{ab} = \sum_{i=a}^{b} I_i\) all the LEDs \(L_i\) will get its required amount of current when the resistors satisfy the following equations:
\[ \begin{align*}
I_{i+1,R} &= I_{i,R_{n-1}} + V_i - V_{i+1}, \quad (3) \\
I_{i,R_0} + \sum_{i=1}^{n-1} I_{i+1,R} + I_{1,R_{2n-1}} &= V_0 - V_n. \quad (4)
\end{align*} \]

Among many sets of resistances that satisfy the equation system above, we can choose the one that minimizes the area of the conductive pattern. We assign larger resistances to conductive segments which have longer length, and thus overall reduce the width of the pattern.

4. Implementation

We implement the aforementioned algorithm as a plugin of Adobe Illustrator\(^2\).

4.1 Adobe Illustrator Extension

Integrating our auto-router into a commonly used design software as Adobe Illustrator keeps the user from the struggle of learning a completely new tool (Fig. 8). Electronic components like LEDs and the power source footprint are arranged into the toolbox of Adobe Illustrator. Our auto-routing script will route all the LEDs and the power source and then generate a ready-to-print conductive pattern to be printed with conductive inkjet ink. Finally, the user will attach real LEDs to the print and drive them with a real power source.

4.2 Prevent Polar Cross

During discussion of routing, to simplify the implementation of the TSP solver, we assume that the LEDs and power source poles are points on a 2D plane. However, we need to put the dimension of each component into consideration when we route them together. From Fig. 6, we see that the circuit comprises two branches which are the anode branch made up of \(R_{i=0,1,\ldots,n-1}\) and the cathode branch made up of \(R_{i=n,n+1,\ldots,2n-1}\). In order to prevent these two polar branches from crossing each other, we rotate each LED around its center to an appropriate angle. Considering the case of three LEDs as shown in Fig. 9 (a), the directional axis of an LED is defined as the vector from the center point of its anode to the center point of its cathode. We define TSP angle \(\theta\) of an LED as the angle formed by the two edges of the TSP route at the corresponding LED. For each LED, we align its directional axis to the bisector line of the TSP angle \(\theta\) as in Fig. 9 (b), and then, if necessary, we reverse the directional axis of the LED to eliminate the twist (if any) of the anode and cathode TSP edge (Fig. 9 (c)). This equals to switching position of anode and cathode terminal of the corresponding LED.

4.3 Eliminate the Excessively Narrow Conductive Trace

The width of a conductive segment in our TSP routing depends on the required resistance and the length of the conductive segment. Besides, it is also influenced by the characteristic of the conductive inkjet ink and the printing technology. The minimum tolerable width \(W_{\text{threshold}}\) is defined as the narrowest conductive trace that is printable. Any trace with the width \(w < W_{\text{threshold}}\) is either highly resistant or not conductive at all. This usually happens when the required resistance \(R\) is high whilst the distance \(l\) between two ends of the conductive segment is small. In order to solve this, we need to eliminate the excessively narrow conductive traces by converting them into electrically equivalent meander lines. The electrically equivalent meander line need to have the same end-to-end distance as

\[ W_{\text{threshold}} \]

\[ l \]
the original narrow trace, wider width than the original width of the narrow trace, and the same end-to-end resistance as the original narrow trace (Fig. 10). An example of this conversion is shown in Fig. 11.

5. Experiments

Settings for all the experiments in this section are listed in Table 2.

| Substrate          | NB-RC-3GR120 [42] |
|--------------------|-------------------|
| Red LED            | LTST-C150CKT [43] |
| Blue LED           | LTST-C150TBKT [44] |
| Conductive Ink     | NBSIJ-MU01 [45]   |
| Inkjet Printer     | Epson PX-S160T    |
| Sheet Resistance $R_s$ | 0.13 Ω/□ (ohm-per-square) |

Table 2 General setting for the experiments.

5.1 Brightness Balance Evaluation

To evaluate the performance of our algorithm in balancing brightness of multiple LEDs, we placed 22 red LEDs to form the text “AB”, and then connected them using printed conductive inkjet ink as shown in Fig. 1. We use our algorithm and the auto-router in Eagle respectively to generate the conductive pattern in Fig. 1 (a) and Fig. 1 (b). In the case of using Eagle, trace width and clearance are set to 1 mm. The circuits are driven by a 6 V regulated DC power supply (TEXIO PA36-3B [46]). We use a digital tester (Hozan DT-124 [47]) to measure the voltage drop on each LED.

As evidently shown in Fig. 1, our TSP router outperformed the auto-router in Eagle in terms of brightness balancing. A closer look at the voltage drops on each LED (Fig. 12 and Fig. 13) shows that our TSP router give a much better distribution of the voltage drop on each LED. All of them are stable and larger than the minimum required voltage (i.e., above the dashed line). Due to the instability in printing resistor using conductive ink, especially when printing narrow traces, which are located in the two ends of the circuit due to the constraint in Eq. (1), the voltage drop on the first and last LEDs are slightly higher than the others. This difference, along with the non-linearity of the I-V characteristic of the LED, makes the currents flow through these two LEDs higher than the other LEDs. However, this deviation in current is acceptable as the brightness difference it causes is ignorable for human eyes (as shown in Fig. 1 (a)). In contrast, the auto-router in Eagle showed difficulty in providing stable voltage drops to each LED. Most of the LEDs, in this case, have the voltage drop less than the minimum required voltage, which resulted in the uneven lighting of the LEDs.

The experiment with multiple color LEDs (8 red LEDs and 14 blue LEDs) as in Fig. 14 shows that our routing algorithm works well in case of circuits with different color LEDs. Voltage drop on the same color LEDs are stable and above minimum voltage requirement for each color. However, due to mixing the different color of LEDs, the input voltage needs to be higher compared to the case of single color LEDs.

5.2 TSP Routing Time Performance

Performance of our TSP router is also evaluated in terms of running time. With the experimental environment setup as in Table 3, we ran it with different number of LEDs and then measure the routing time in each case.

The experiment result in Fig. 15 shows that our TSP routing algorithm can route up to 60 LEDs in a convenient time frame using a commodity laptop. However, it starts to struggle when the number of LEDs increases more. This can be improved partially by introducing heuristic fine-tuning or dynamic pro-
In addition to running time evaluation, we also evaluate the approximation ratio of our GA TSP solver with difference numbers of LEDs. The approximation ratio here is defined as the ratio between the total length of the route found by our GA TSP router and the total length of the optimal route which is calculated by Concorde TSP Solver [48]. As shown in Fig. 15, we can achieve reasonable approximation of the TSP route with the highest ratio in the case of 100 LEDs is about 1.6. That means in this scenario, the route found by our GA TSP solver is only 1.6 times longer than the optimal one.

5.3 Ink Consumption

Assuming that conductive inkjet printing will always print the conductive ink with the same thickness, we evaluate the ink consumption by measuring the area of the conductive pattern. We do not acknowledge of any tool that helps to route and balance brightness of multiple LEDs. Therefore, we conducted the experiment by comparing printing area in case of different numbers of LEDs and different of input voltages. We distributed multiple LEDs uniformly on an A4-sized artboard. Input voltage is set to 20 V and 30 V for each case. As shown in Fig. 16, increasing number of the LEDs requirement more conductive ink to route all of the LEDs. However, it is possible to reduce the ink consumption by increasing the input voltage.

6. Applications

6.1 Electronic Artwork with Multiple LEDs

With our plugin, users can automatically route a large number of LEDs and print the circuit quickly with conductive inkjet printing (Fig. 17). It is also possible to mix multiple types of LEDs in a single artwork by stacking altogether transparent PET films with different types of LEDs routed on each layer.

6.2 Characters and Emojis with LEDs and Conductive Ink

Characters and emojis help to convey information efficiently. They are often used in designing interactive applications with LEDs. Using our plugin, instead of manually connecting each LED, users can easily generate conductive patterns of all characters in the alphabet table (Fig. 18 (a)) or emojis such as the smiling or frowning face (Fig. 18 (b) and 18 (c)).

6.3 Quickly Made Greeting Card

Although our algorithm is originally implemented in Adobe Illustrator, it is possible to port the algorithm to other platforms which are more familiar to beginners. Figure 19 shows a proposal of a smartphone app that lets users generate LED-based greeting cards by taking a photo, placing LED and route with our TSP router. The generated conductive pattern is ready to be printed and assembled.
7. Discussion and Future Work

Our TSP router helps to generate conductive patterns that wires and regulates the current through each LED to assure that they light up evenly. Besides, we can expand this routing algorithm to other 2-terminal components such as sensors, actuators, and paper speakers. We can also port it to other popular CAD platforms such as Fritzing and KiCAD.

However, our TSP router is still having several issues that should be investigated more in future:

- **Routing time:** Our auto-router can route up to 60 LEDs in a short time. However, in the case that an user would like to light up 1,000 LEDs or more to fill a whole wall or ceiling, a faster algorithm to solve the TSP is necessary.

- **GA parameters tuning:** A genetic algorithm is a heuristic algorithm so it is critical to choose the parameters properly. Although we find that the default parameters (resulted from empirical experiments) in Table 1 is working well in most of the experimental cases, tuning the GA parameters will help to improve the performance of the GA TSP solver. In the future, we want to develop our router so that it can dynamically change the parameters based on the number of LEDs and other user-defined constraints.

- **Power source placement:** Our current algorithm requires users to place the power pads before routing. This is simple in most of the cases. However, when the number of LEDs increases or the artwork is more complicated, it might be challenging for users to decide position of the power pads properly. In addition, the current algorithm allows only one power source for the whole circuit which leads to high voltage requirement when the number of LEDs increases. In the future, our auto-router will suggest the number of power sources and the position of each power pad so that the routed circuit meets user-defined constraints (such as time, ink consumption, required voltage of the power source, and aesthetic).

- **Scalability evaluation:** To assess the scalability of our auto-router, we want to investigate the performance of the router in terms of the number of LEDs, the voltage of the power source, the sheet resistance of the conductive ink, total space that all LEDs occupy, and LEDs distribution.

- **Overlapping:** In our current implementation, we eliminate the crossing edge before brightness balancing using a 2-opt algorithm while temporarily ignore the width of the conductive trace. This approach assure that the TSP route do not have any crossing edge. However, in extreme cases where the routing area is strictly limited by the wiring space or the density of the LEDs distribution, there is a possibility that an anode branch might overlap with a cathode branch causing circuit shorting. This can be prevented by increasing the voltage of the power source. In the future, it is possible to search for a non-overlapping pattern of the TSP routed circuit by either checking for overlapping during TSP search or adjusting the width of the overlapping segments while still assure that the Eqs. (1) and (2) are satisfied.

- **Power consumption optimization:** The conductive pattern area, resistances, and power source voltage have a tight relationship. Based on this relationship, we may find the adequate trade-off between resistance and power source voltage required to drive multiple LEDs.

8. Conclusion

In this paper, we addressed the hassle of wiring and balancing brightness of multiple LEDs (single and multiple colors). We proposed an auto-router based on traveling salesman problem and resistance adjustment of the conductive pattern as a designing tool for creating interactive applications with LEDs.
and conductive inkjet ink. We took advantage of the apparent disadvantage of the conductive ink (i.e., high resistance) to eliminate rigid ballast resistors which are inherently necessary to regulate the LEDs. Besides the deep integration into Adobe Illustrator, our extension provides:

- a tool for arranging and designing with LEDs
- an auto-router based on the traveling salesman problem to find the shortest cross-less path to connect multiple LEDs on a single-layer sheet of paper
- a mechanism of adjusting the width of the conductive patterns to control the resistance to each LED, hence allowing to control the brightness of each LED without the need of additional resistors.

In this research, we aim to support inexperienced users in working with electronic circuits, especially working with conductive inkjet ink and multiple LEDs. Our plugin allows users to focus on creative designing instead of struggling with complicated electronic circuits.

Acknowledgments
This work was supported by JST ERATO Grant Number JP-MJER1501, Japan.

References
[1] Y. Kawahara, S. Hodges, B.S. Cook, C. Zhang, and G.D. Abowd: Instant inkjet circuits: Lab-based inkjet printing to support rapid prototyping of ubicom devices, Proc. 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp ’13), pp. 363–372, 2013.
[2] Mitsubishi Paper Mills Ltd.: Circuit Marker, https://www.mpm.co.jp/electronic/eng/silver-nano/index.html, 2017.
[3] BareConductive: BarePaint, www.bareconductive.com, 2017.
[4] ElectronInks: CircuitScribe, http://www.electroninks.com, 2017.
[5] K. Kato and H. Miyashita: Extension sticker: A method for transferring external touch input using a striped pattern sticker, Proc. Adjunct Publication of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST ’14 Adjunct), pp. 59–60, 2014.
[6] T. Tsuji, N. Koizumi, and T. Naemura: Inkantatory paper: Dynamically color-changing prints with multiple functional inks, Proc. Adjunct Publication of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST ’14 Adjunct), pp. 39–40, 2014.
[7] N.-W. Gong, J. Steimle, S. Olberding, S. Hodges, N.E. Gillian, Y. Kawahara, and J.A. Paradiso: PrintSense: A versatile sensing technique to support multimodal flexible surface interaction, Proc. SIGCHI Conference on Human Factors in Computing Systems (CHI ’14), pp. 1407–1410, 2014.
[8] C. Karatas and M. Gruteser: Printing multi-key touch interfaces, Proc. 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp ’15), pp. 169–179, 2015.
[9] K. Okamura: Shine, paper! Shine! Exhibition exploring silver nanoparticle ink and the future of paper, http://www.takeo.co.jp/en/news/detail/001552.html, 2016.
[10] J. Qi: Pu Gong Ying Tu (Dandelion Painting), http://technologie.com/pu-gong-ying-tu-dandelion-painting/, 2012.
[11] T.D. Ta, F. Okuya, and Y. Kawahara: Lighttrace: Auto-router for designing LED based applications with conductive inkjet printing, Proc. 1st Annual ACM Symposium on Computational Fabrication, Article No. 3, 2017.
[12] E.W. Dijkstra: A note on two problems in connexion with graphs, Numerische Mathematik, Vol. 1, No. 1, pp. 269–271, 1959.
[13] F.O. Hadlock: A shortest path algorithm for grid graphs, Networks, Vol. 7, No. 4, pp. 323–334, 1977.
[14] C.Y. Lee: An algorithm for path connections and its applications, IRE Transactions on Electronic Computers, Vol. EC–10, No. 3, pp. 346–365, 1961.
[15] K. Mikami and K. Tabuchi: A computer program for optimal routing of printed circuit conductors, Proc. IFIP Congress 1968, pp. 1475–1478, 1968.
[16] D.W. Hightower: A solution to line-routing problems on the continuous plane, Proc. 6th Annual Design Automation Conference (DAC ’89), pp. 1–24, 1999.
[17] N. Abboud, M. Grotschel, and T. Koch: Mathematical methods for physical layout of printed circuit boards: An overview, OR Spectrum, Vol. 30, No. 3, pp. 453–468, 2007.
[18] U. Lauther: A data structure for gridless routing, Proc. 17th Design Automation Conference (DAC ’80), pp. 603–609, 1980.
[19] A.C. Finch, K.J. Mackenzie, G.J. Balsdon, and G. Symonds: A method for gridless routing of printed circuit boards, Proc. 22nd ACM/IEEE Design Automation Conference (DAC ’85), pp. 509–515, 1985.
[20] W.L. Schiele, T. Kruger, K.M. Just, and F.H. Kirsch: A gridless router for industrial design rules, Proc. 27th ACM/IEEE Design Automation Conference (DAC ’90), 1990.
[21] Autodesk: EAGLE PCB Design Software, http://www.autodesk.com/products/eagle/overview, 2017.
[22] P. Ramanathan, K.G. Shin, and K.G. Shin: A clock distribution scheme for nonsymmetric VLSI circuits, Proc. 1989 IEEE International Conference on Computer-Aided Design. Digest of Technical Papers, pp. 398–401, 1989.
[23] T.-H. Chao, Y.-C. Hsu, J.-M. Ho, and A.B. Kahng: Zero skew clock routing with minimum wirelength, IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, Vol. 39, No. 11, pp. 799–814, 1992.
[24] R.-S. Tsay: An exact zero-skew clock routing algorithm, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 12, No. 2, pp. 242–249, 1993.
[25] N. Menezes, A. Balivada, S. Pullela, and L. T. Pillage: Skew reduction in clock trees using wire width optimization, Proc. 1993 IEEE Custom Integrated Circuits Conference (CICC ’93), pp. 961–964, 1993.
[26] T. Ta, M. Fukumoto, K. Narumi, S. Shino, Y. Kawahara, and T. Asami: Interconnection and double layer for flexible electronic circuit with instant inkjet circuits, Proc. 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp ’15), pp. 181–190, 2015.
[27] H.A. Andersson, A. Manuilskiy, S. Haller, M. Hummelgärd, J. Ciden, C. Hummelgärd, H. Olin, and H.-E. Nilsson: Assembling surface mounted components on ink-jet printed double sided paper circuit board, Nanotechnology, Vol. 25, No. 9, pp. 094002, 2014.
[28] S. Hodges, N. Villar, N. Chen, T. Chugh, J. Qi, D. Nowacka, and Y. Kawahara: Circuit stickers: Peel-and-stick construction of interactive electronic prototypes, Proc. SIGCHI Conference on Human Factors in Computing Systems (CHI ’14), pp. 1743–1746, 2014.
[29] J. Lo, C. Torres, I. Yang, J. O’Leary, D. Kaufman, W. Li, M. Dontcheva, and E. Paulos: Aesthetic electronics: Designing, sketching, and fabricating circuits through digital exploration, Proc. 29th Annual Symposium on User Interface Software and Technology, pp. 665–676, 2016.
[30] R. Ramakers, K. Todi, and K. Luyten: PaperPulse: An integrated approach for embedding electronics in paper designs, Proc. 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI ’15), pp. 2457–2466, 2015.
[31] V. Savage, X. Zhang, and B. Hartmann: Midas: Fabricating custom capacitive touch sensors to prototype interactive objects, Proc. 25th Annual ACM Symposium on User Interface Software and Technology (UIST ’12), pp. 579–587, 2012.

[32] K. Narumi, S. Hodges, and Y. Kawahara: ConductAR: An augmented reality based tool for iterative design of conductive ink circuits, Proc. 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp ’15), pp. 791–800, 2015.

[33] M.W. Bern and R.L. Graham: The shortest-network problem, Scientific American, Vol. 260, No. 1, pp. 84–89, 1989.

[34] C.H. Papadimitriou: The Euclidean travelling salesman problem is NP-complete, Theoretical Computer Science, Vol. 4, No. 3, pp. 237–244, 1977.

[35] N. Christofides: Worst-case analysis of a new heuristic for the travelling salesman problem, Management Sciences Research Report, No. 388, 1976.

[36] X. Geng, Z. Chen, W. Yang, D. Shi, and K. Zhao: Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search, Applied Soft Computing Journal, Vol. 11, No. 4, pp. 3680–3689, 2011.

[37] J. Grefenstette, R. Gopal, B. Rosmaita, and D. Van Gucht: Genetic algorithms for the traveling salesman problem, Proc. first International Conference on Genetic Algorithms and their Applications, pp. 160–168, 1985.

[38] T.K. Paul: Traveling salesman problem by genetic algorithm, http://www.iba.t.u-tokyo.ac.jp/english/Online/TSP/Method.html, 2005.

[39] M. Obitko: Introduction to genetic algorithms, http://www.obitko.com/tutorials/genetic-algorithms/crossover-mutation.php, 1998.

[40] D.E. Goldberg and K. Deb: A Comparative Analysis of Selection Schemes Used in Genetic Algorithms, Foundations of Genetic Algorithms, pp. 69–93, 1991.

[41] J.-Y. Potvin: Genetic algorithms for the traveling salesman problem, Annals of Operations Research, Vol. 63, No. 3, pp. 337–370, 1996.

[42] Mitsubishi Paper Mills Ltd.: Resin coated paper NB-RC-3GR120, http://www.k-mpm.com/agnanoen/agnano_media.html, 2013.

[43] Lite-On Technology, Inc: SMD LED product datasheet LTST-C150CKT, http://optoelectronics.liteon.com/upload/download/DS-22-98-0002/S_110_LTST-C150CKT.pdf, No. DS-22-98-002, 2006.

[44] Lite-On Technology, Inc: SMD LED product datasheet LTST-C150TBKT, http://optoelectronics.liteon.com/upload/download/DS22-2000-026/LTST-C150TBKT(0630).pdf, No. LTST-C150TBKT, 2010.

[45] Mitsubishi Paper Mills Ltd.: Silver nano-particle ink NBSIJ-MU01, http://www.k-mpm.com/agnanoen/agnano_ink.html, 2013.

[46] TEXIO TECHNOLOGY CORPORATION: TEXIO regulated DC power supply PA36-3B, http://www.texio.co.jp/en/product/detail/5, 2017.

[47] HOZAN TOOL IND.CO., LTD.: Hozan DT-124, http://www.hozan.co.jp/E/catalog/Measuring/DT124.html, 2017.

[48] D. Applegate, R.E. Bixby, V. Chvátal, and W.J. Cook: Concorde TSP solver, http://www.math.uwaterloo.ca/tsp/concorde/index.html, 2017.

Tung D. Ta
He is a Ph.D. student in the Department of Information and Communication Engineering, the University of Tokyo. His research interests are in the areas of digital fabrication, flexible electronic circuits, and soft-bodied robot. He received his M.E. in 2016 from the Department of Information and Communication Engineering, the University of Tokyo, and B.E. from the Department of Software Engineering, FPT University in 2012. He is a student member of IEICE, IPSJ, and ACM.

Fuminori Okuya
He is a Ph.D. student in the Graduate School of Information Science and Technology, the University of Tokyo, Japan. His research interests are in control engineering, compressed sensing, data mining, and soft-robot. He received the bachelor of engineering degree in 2015, and master of engineering degree in 2017 from the University of Tokyo.

Yoshihiro Kawahara (Member)
He is an Associate Professor in the department of Information and Communication Engineering, the University of Tokyo. His research interests are in the areas of computer networks and ubiquitous and mobile computing. He is currently interested in developing energetically autonomous information communication devices. He received his Ph.D. in Information Communication Engineering in 2005, M.E. in 2002, and B.E. in 2000. He joined the faculty in 2005. He is a member of IEICE, IPSJ, ACM, and IEEE. He is a committee member of IEEE MTT TC-24 (RFID Technologies). He was a visiting assistant professor at Georgia Institute of Technology and MIT Media Lab.

SICE JCMSI, Vol. 11, No. 4, July 2018