Personalized Food Printing for Portrait Images

Haiming Zhao [a], Jufeng Wang [a], Xiaoyu Ren [a], Jingyuan Li [b], Yong-Liang Yang [c], Xiaogang Jin [a,*]

[a] *State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310058, China*
[b] *Shiyintech Corp., Hangzhou 310058, China*
[c] *Department of Computer Science, University of Bath, Claverton Down, Bath BA2 7AY, UK*

ARTICLE INFO

Article history:

food printing, image abstraction, path optimization

ABSTRACT

The recent development of 3D printing techniques enables novel applications in customized food fabrication. Based on a tailor-made 3D food printer, we present a novel personalized food printing framework driven by portrait images. Unlike common 3D printers equipped with materials such as ABS, Nylon and SLA, our printer utilizes edible materials such as maltose, chocolate syrup, jam to print customized patterns. Our framework automatically converts an arbitrary input image into an optimized printable path to facilitate food printing, while preserving the prominent features of the image. This is achieved based on two key stages. First, we apply image abstraction techniques to extract salient image features. Robust face detection and sketch synthesis are optionally involved to enhance face features for portrait images. Second, we present a novel path optimization algorithm to generate printing path for efficient and feature-preserving food printing. We demonstrate the efficiency and efficacy our framework using a variety of images and also a comparison with non-optimized results.

1. Introduction

Sugar painting is a traditional Chinese folk art that uses hot, liquid sugar to create two dimensional figures (see Figure 1(a)). Although this form of art is well known [1], it requires a skilled artisan to manually create delicate figures, which restricts its popularity. The recent advances of 3D printing techniques allow easy access of physical fabrication for novices, providing a good opportunity to popularize sugar painting in the wider food industry.

3D printing enables efficient realization of physical objects from digital models, which fascinates both professionals and amateurs, and facilitates customized design and fast prototyping. In the recent years, 3D printing industry has developed novel applications in the area of food science and technology. These applications leverage the power of 3D printing to customize food with complex geometry [2]. The development of both hardware and software allows novice users to fabricate edible items with customized patterns.

3D food printing is an emerging research area with great potential for food industry (see Figure 2). Researchers start from producing food with traditional additive manufacturing. Multi-material food printing technique is further introduced to create complex geometries [2]. The problem of maintaining precise 3D shape after cooking process is also considered [3].

In this work, inspired by sugar painting, we aim at using 3D printing techniques for personalized food fabrication. Our goal is to extract personalized information from digital images, including but not limited to the most popular medium - portrait photos, and then use a 3D printer to automatically fabricate the extracted information, as shown in Figure 1(b-c). Compared with traditional additive manufacturing suitable for printing solids, our personalized food printing problem subjects to further constraints as follows. (i) The material used in our work, such as maltose and chocolate syrup, is viscoelastic, which requires larger radius of the print nozzle. As a result, food printers produce edible objects with lower resolution and less accuracy in contrast to FDM and SLA printers. (ii) The image abstraction process is required due to the lower resolution of food printer. Current image abstraction methods are capable of generating visually plausible results. However, we have to bridge the gap between the virtual abstractions generated by graphics algorithm and the physical printing paths for 3D printer. (iii) In 3D food

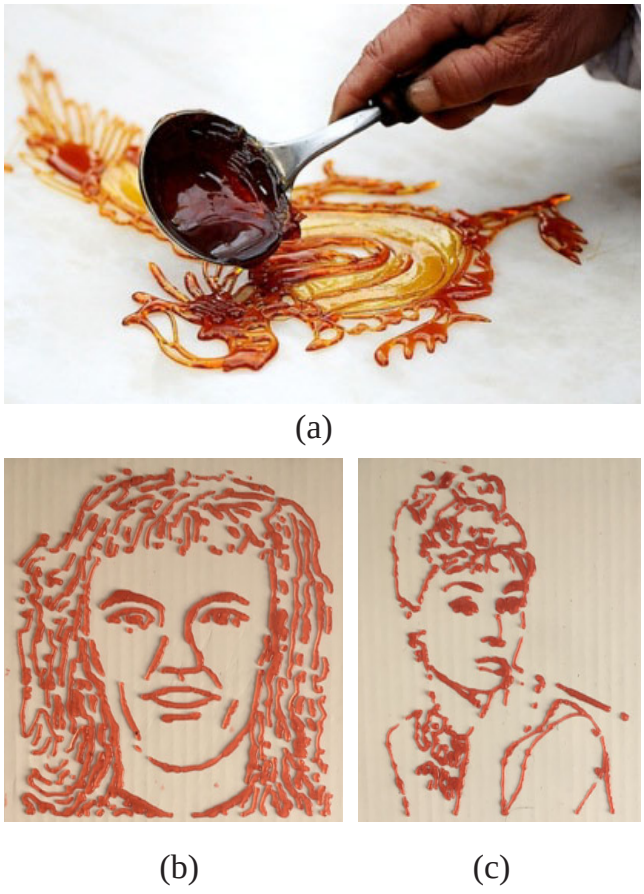

Fig. 1. Inspired by sugar painting, we present a customized food printing framework driven by portrait images. (a) An artisan manually creates a pattern using hot sugar. (b) and (c) The customized food printing results automatically generated by our framework.

printing process, frequent extrusion and retraction of the edible material is likely to cause unnecessary or insufficient printing, resulting in printing path with non-uniform width. Besides, these operations are time-consuming. Therefore, our approach aims to filter out redundant details without losing the characteristics of the input image. In order to improve the printing quality and speed up the printing process, less retractions and moving distance is preferred in our printing path optimization algorithm.

Our work is also inspired by ingenious frameworks proposed in recent publications. Tool path continuity enhancement methods have been widely applied to additive manufacturing. Some researchers focus on the tool path geometry to avoid sharp turns and ensure continuous path [4, 5]. While previous algorithms are applicable to traditional 3D printing problem, the new challenges of 3D food printing remain unsolved.

Face image processing is well studied over the years. Stylized cartoon faces are widely used to simplify portrait images and generate appealing sketches. An example-based sketching method [6] is capable of generating cartoon-like portraits. A data-driven framework is further developed to improve the stylization [7], making the results natural and attractive. Although these approaches are effective, the face in the portrait image is limited to front view and the detailed features such as nevus, wrinkles and cheekbones can be easily eliminated.

We present a novel framework to fabricate personalized edible patterns by taking into account the fabrication constraints in 3D food printing. A data-driven method is applied to abstract the input image. A new printing path optimization method is presented to generate printable path for 3D food printing. The printing quality is improved by adjusting image abstraction result according to the size of print nozzle. A Depth First Search (DFS) based method is used to optimize the printing pattern and speed up the printing process. We present a novel approach to fabricate personalized edible patterns by combining 3D printing, food customization, and computer graphics. We demonstrate our framework on a variety of images including portrait photos, vector arts, artistic drawings. The experiments show that our framework can well preserve image features while saving printing time.

2. Related Work

2.1. Computer graphics for 3D printing

3D printing draws significant research interests of graphics researchers recently [8]. The physical properties of 3D printed objects have been studied with the help of computer-aided design algorithms to meet different fabrication requirements. Structural analysis helps to improve the strength of printed objects. Zhou *et al*. [9] involve physical simulation process to analyse the worst load distribution. Stress relief methods are also adopted to improve structural soundness of the printed objects by geometry refinement such as thickening and hollowing [10]. Other physical properties such as static equilibrium and dynamic motion also attract researchers' attention. Prévost *et al*. [11] introduce an effective method to fabricate static models which are able to stand by themselves. Bächer *et al*. [12] take a step further to print models that are able to spin. Zhao *et al*. [13] present a novel framework to produce personalized roly-poly toys. 3D printing with multiple materials is also investigated in recent years. Chen *et al*. [14] use a reducer-tuner model to translate user-defined model specifications into material-specific representations. Sitthi-Amorn *et al*. [15] present an impressive platform for multi-material 3D printing. Unlike all previous work, we focus on 3D food printing which is yet to be studied in computer graphics community.

2.2. Food fabrication using 3D printing

As 3D printing matures in industry, novel applications emerge in the area of food science and technology [2]. There are plenty of potential uses of 3D food printing, such as customized food design, small scale food production and personalized nutrition. Researchers have provided critical insights into engineering solution for 3D food printing. Considering the consistency, viscosity, and solidifying properties of food materials, some are natively printable (chocolate, cheese, cake frosting) since they can be extruded smoothly from a syringe. Several solutions are presented to retain their shape consistency throughout the cooking process by additives and recipe control [16, 3]. Current food manufacturing techniques are suitable for massive food production. As for customized food design, it often

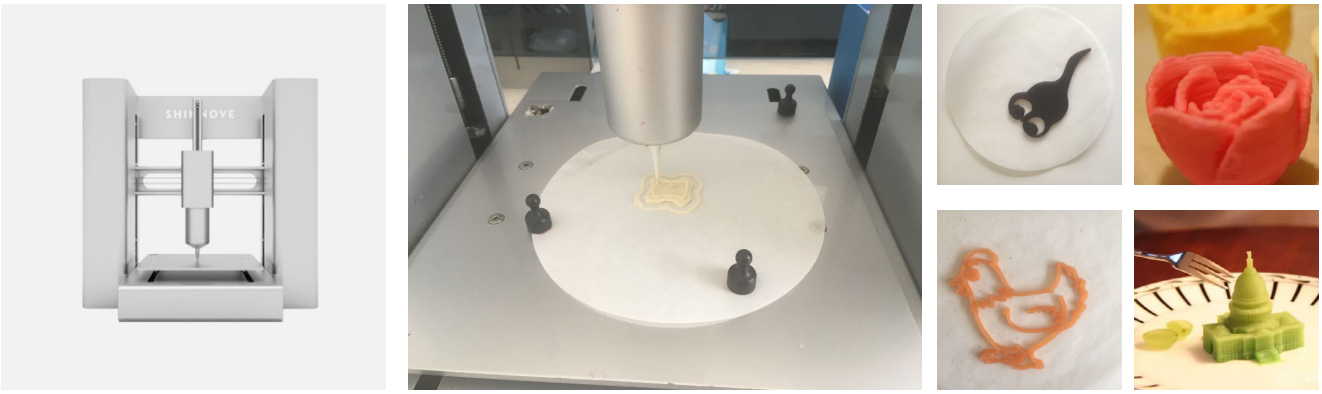

Fig. 2. The 3D food printer used in our framework (Left). The print nozzle extrudes edible material during the printing process (Middle). Examples of printed edible objects (Right).

requires craftsmen with professional skills and the fabrication process is labor-intensive and time-consuming. 3D printing is promising for small scale food production and customized food design. Using 3D printing to fabricate food with customized nutrition is discussed in [17]. This solution is able to take personal health data (height, age, health condition, etc.) to produce food with sufficient caloric. One unsolved problem in food printing is how to minimize the difference between designed shapes (visualized in computer) and printed objects (fabricated by 3D food printer). The melting and concreting process is not so accurate as using traditional material, resulting in wider printing path and lower resolution result. Also, to keep the uniform width of the printing path, frequent on-off switching of material extrusion should be avoided. Moreover, the overlap of neighboring paths should be considered in the printing pattern generation to avoid cluttered results. Our framework addresses the above issues using a specialized path optimization algorithm, leading to plausible results for personalized food printing.

2.3. Image abstraction

Image abstraction helps to simplify the visual details and emphasize the salient content. DeCarlo *et al*. [18] present a stylization and abstraction method based on computer vision techniques. The highlighted visual elements are determined by a model of human perception and users' eye movements. Kang *et al*. [19] introduce an easy-to-use algorithm to automatically generate a coherent and stylistic line drawing of a photograph. This work is later extended to video abstraction [20]. Fu *et al*. [21] propose a system which is able to derive a plausible stroke order from an input image.

The human vision system can be easily trained to distinguish and recognize faces from ancient times. There are numerous publications concerning face beautification [22], recognition and detection [23, 24]. Chen *et al.* [6] present an example-based sketching algorithm to abstract portrait into stylized cartoon faces. It is effective but the results depend on the training set. Wang *et al.* [25] propose a data-driven sketch synthesis framework, which filters out redundant information and preserves prominent features suitable for 3D printing. Berger *et al.* [26] combine line drawing abstraction and artistic styles to sketch portraits. Zhang *et al.* [7] provide a data-driven framework for generating natural and attractive cartoon-like portrait. But the framework is limited to face from front view and does not preserve detailed features like nevus, wrinkles and cheekbones.

2.4. Path optimization for 3D printing and graphics

3D printing allows rapid prototyping in computer-aided manufacturing. Several works based on enhanced Genetic Algorithm [27] and Artificial Neural Network (ANN) [28] are proposed to optimize printing path for efficient 3D printing. Jin *et al*. [29] present a novel method to optimize the zigzag filling path. They also investigate how to avoid sharp turns in a printing path to accelerate printing [4]. Wojcik *et al*. [30] introduce a MZZ-GA framework to solve path planning problem which uses Genetic Algorithm (GA) to improve path generated by Modified Zig Zag (MZZ). Path optimization algorithms are also employed for graphics applications. Pedersen and Singh [31] propose a novel maze synthesis method, which generates a continuous path with no intersections. Wong and Takahashi [32] extract sharp edges from an input image and use semi-Eulerization method to seek the path that traverses all edges. Zhao *et al*. [5] adopt connected Fermat spirals as a tool-path pattern for 3D printing. The optimized path is a single continuous curve, enabling efficient and quality fabrication. While prior works demonstrate effectiveness in different scenarios, they are too restrictive to be applied for 3D food printing as in our case.

3. Algorithm

3.1. Overview

As shown in Figure 3, our framework consists of four components: input image processing, image abstraction, path optimization, and 3D printing. The FDoG filter in [19] is applied for image abstraction. By taking into account the properties of the food printer, a novel path optimization algorithm is presented to efficiently fabricate the abstraction results with high quality. We also employ further techniques to improve the printing results for portrait images. Sketch synthesis [25] provides an alternative to filter out the redundant information of the input image and preserve prominent features. Face detection and face landmark labeling [24] is used to enhance the face region for better feature preservation.

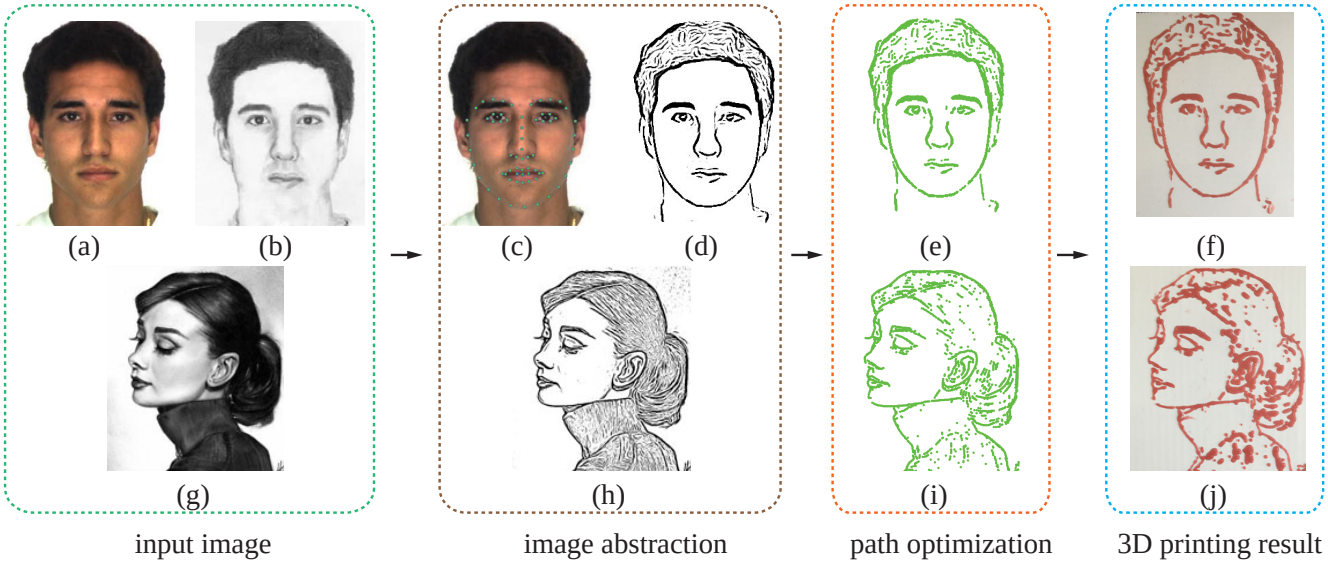

Fig. 3. The overview of our framework. Given an input image (g), our framework applies image abstraction (h) and path optimization (i), to generate personalized pattern that can be directly fabricated (j). Sketch synthesis (a-b) and face enhancement (c-d) can be further employed to improve the printing quality of portrait images.

3.2. 3D food printer

The 3D food printer used in our experiment is SHINNOVE-S2 [33] (see Figure 2). The size of the printer is 420 $mm \times$ 380 $mm \times 400$ mm and the print volume is 150 $mm \times 150$ $mm \times$ 100 mm. SHINNOVE-S2 is specially designed for printing edible material, such as chocolate and jam. It is based on FDM printing technology and is able to extrude edible material from the printing nozzle. The extrusion of the material is controlled by a specially designed motor and it is also controlled by G-code. A practical printing process is demonstrated in the accompanying video.

As a matter of fact, normal inkjet based printing would naturally be more applicable to 2D raster images, even powder bed fusion based approaches (powdered sugar for example) would be a better choice. Considering the high cost of previous method, we adopt current solution. The price of SHINNOVE-S2 is about \$ 1000 and each printed object costs less than \$ 0.5.

3.3. Path optimization algorithm

The input image processing and image abstraction are performed at pixel level. Restricted to the printer's nozzle width, 3D food printers fabricate edible objects with lower resolution. If the FDoG output is directly used as the input for 3D food printer, the printed object varies too much from the line abstraction result, as shown in Figure 10 (b-c). The result is often overfilled due to the overlap of neighboring paths extruded by the printer. Therefore, one straightforward idea to avoid overlap is to attenuate the printing path using morphological operators. The erosion operation in OpenCV [34] is able to erode the FDoG output to 1 pixel in width. However, the printing result is not satisfying due to artifacts caused by overlap between neighboring paths (Figure 8(d) and Figure 9(c)). Thus we present a novel method to optimize the printing path, which involves 3 steps:

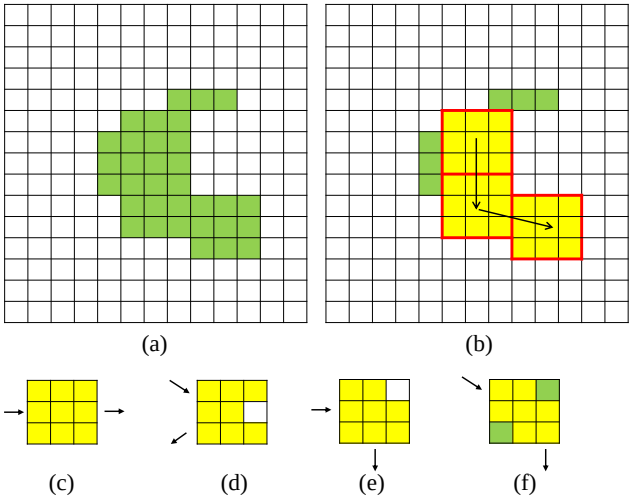

Fig. 4. (a) The pixels highlighted in green indicate the FDoG output. (b) A set of 3×3 patches (in yellow) is generated to cover the FDoG output. Patch boundaries are highlighted in red. We use the pixel at the center to represent individual patch. The printing path needs to travel all the patch centers. The black arrows illustrate the printing path. (c-f) The pixels highlighted in yellow, white and green illustrate filled, empty and half-filled region by the printing material respectively.

1. Section 3.3.1: We use n $s \times s$ square patches (denoted as $\mathcal{P} = \{P_i\}$) measured by pixels to cover the output from FDoG filter. There is no overlap between neighboring patches.

2. Section 3.3.2: The center T_i (denoted as $\mathcal{T} = \{T_i\}$) of P_i is used to denote individual printing patch. We use a DFS-based method to generate printing path from all printing patches, which travels through all the patch centers.

3. Section 3.3.3: According to the parameters of the 3D printer, the corresponding G-code is generated.

The advantage of the above method is that it not only simplifies the path optimization process, but also ensures the quality of the printing result. In order to avoid overlap, the printing path

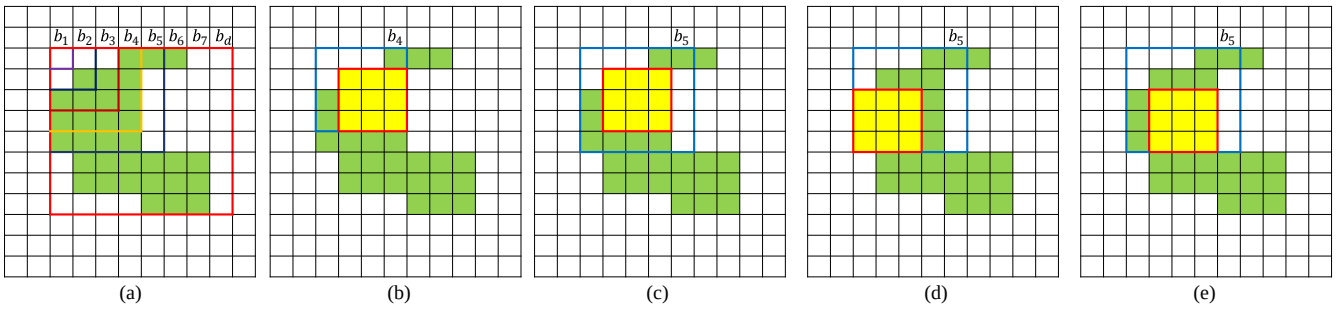

Fig. 5. Solution to Equation 1.

and the moving direction of the nozzle should be optimized at the same time, which is well known to be NP-complete [35]. Instead, we introduce a proper patch size to approximate the coverage of printer's material extrusion, and optimize a patch layout (a layout with a set of non-overlapping patches) to cover the abstracted image features. This reduces the complexity of the optimization and makes the problem tractable.

3.3.1. Patch size determination

In the following, we will first discuss how to optimize the patch layout based on an intuitive illustration, then detail the patch size selection process.

As shown in Figure 4(a), we use 3×3 square patches to cover the result from FDoG filter. While the starting and ending location of the printing path for each patch can change (see Figure 4(c-f)), most of the pixels in the patch need to be covered by printing material. Our patch coverage plan is able to avoid the overlap of neighboring printing paths, but few pixels are not covered with printing material as intended. As shown in Figure 4(d-e), the pixels in white are empty. And in Figure 4(f), the pixels in green are only half-filled.

The optimization problem aims at generating a path that travels through T_i's. Let the FDoG output shown in Figure 4 be F (in green) and the region covered by 3×3 patches be V (in yellow). We set that all patches are within the area of F. The goal is to cover F as much as possible by V:

$$\min_{\forall P_i \in F} (F - V). \tag{1}$$

where $\mathcal{P} = \{P_i\}$ presents the set of all patches. The above square packing problem is well known to be NP-hard [36]. In practice, we solve it based on a divide-and-conquer approach where F is hierarchically subdivided into subregions. We solve for the patch layout within each subregion using dynamic programming [37]. The whole image region is subdivided into several connected graphs. In each graph dynamic programming is used to solve for Equation 1.

As shown in Figure 5, the FDoG result is covered by 3×3 square patches. The square $i \times i$ is denoted as b_i in Figure 5(a) and the patch layout solution for b_i is denoted as $pl(b_i)$. It's obvious that $pl(b_i)$ is only related to the layout plan for b_{i-1}, b_{i-2} and b_{i-3}. Take $pl(b_4)$ for example. There is no layout plan for $pl(b_1)$ and $pl(b_2)$ because b_1 and b_2 are smaller than 3×3 square patches. The layout plan for $pl(b_3)$ is not available either because the top-left corner is empty. However, $pl(b_4)$ is obtained based on $pl(b_1)$ as shown in Figure 5(b) by filling the space between b_4 - b_1. The same procedure is operated on $pl(b_2)$ and $pl(b_3)$ by filling the space between b_4 - b_2 and b_4 - b_3 respectively. However no available patch layout solution is obtained. When it comes to $pl(b_5)$, there are multiple layout plans. $pl(b_5^j)$ is used to denote the j-th possible layout plan for $pl(b_5)$. $pl(b_5^1)$ as shown in Figure 5(c) is obtained from $pl(b^4)$. $pl(b_5^2)$ (as shown in Figure 5(d)) and $pl(b_5^3)$ (as shown in Figure 5(e)) are calculated from $pl(b_2)$. $pl(b_5)$ is chosen from $pl(b_5^1)$, $pl(b_5^2)$ and $pl(b_5^3)$, which minimizes Equation 1:

$$pl(b_i) = \arg\min_{pl(b_i^j)} (F - V). \tag{2}$$

This routine continues until $pl(b_d)$ is obtained, which is a typical dynamic programming problem [37]. Since the patch is a square, it is straightforward to set $b_1, ..., b_d$ as squares. Therefore, b_d is the smallest square which covers the FDoG result.

To determine suitable patch size in the image space, we need to scale the nozzle size according to the image area with respect to the printing bed area. The nozzle used in the experiments is 0.8 mm wide (denoted by w), the maximum printing size is 120 mm×120 mm in our experiment (denoted by a) and the maximum dimension of the input image (denoted as p) is the larger between width and height. Then the corresponding width of the nozzle in pixels (denoted by s) can be computed from:

$$\begin{aligned} s &= \frac{w}{a} p, \\ p &= \max(height, width). \end{aligned} \tag{3}$$

The patch size used in our approach is related to s. In Figure 10, the image size is 500 pixels in width and 600 pixels in height. s is 4 pixels according to Equation 3. In practice, if larger s is used, then patch overlap can be avoided but it oversimplifies result. Considering the viscosity of the material, the printing path should be thinner than the width of the nozzle. Besides, smaller patch size favors more details. By considering the above factors, we use 3×3 patch size (see Figure 10) in all our experiments.

3.3.2. DFS-based path optimization method

Our path optimization problem is to find a shortest path for the print head to travel through $\mathcal{T}$, where T_i is set as vertex. It is similar to travelling salesman problem (TSP) [35] and Chinese

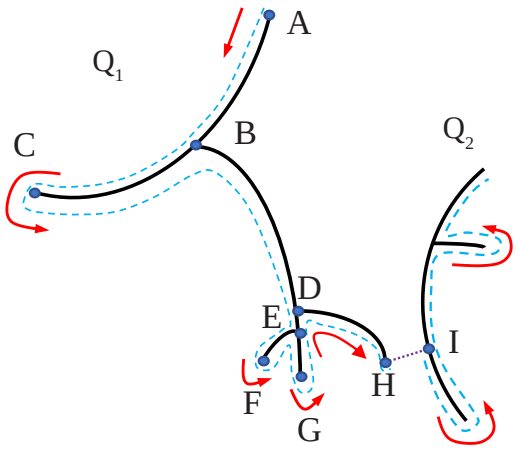

Fig. 6. Depth First Search (DFS) based method. The vertex with smaller depth is visited first.

postman problem (CPP) [38]. Richard M. Karp proves that that the Hamiltonian cycle problem is NP-complete [39], which implies that TSP is NP-hard. A few variants of the CPP have been studied and are shown to be NP-complete [40]. To make the solution feasible, we apply method based on Depth First Search (DFS) to optimize the path.

Our DFS-based method aims to reduce the moving distance of the print head to speed up the printing process. We assume that two patches are adjacent if the distance between the centers of these patches is less than $2s$. $\mathcal{P}$ is divided into a number of connected graphs (represented as $Q = \{Q_1, ..., Q_i, ..., Q_n\}$). The moving distance consists of two types of paths. Let the moving distance within each Q_i be D_{IQ_i} and the sum be $D_{IQ} = \sum_i D_{IQ_i}$. Let the moving distance between connected graphs be D_{OQ}. We would like to minimize the overall moving distance of the printer:

$$\min(D_{IQ} + D_{OQ}). \tag{4}$$

An example illustration of the algorithm is in Figure 6. There are two connected graphs Q_1 and Q_2. D_{IQ} is minimized using DFS-based method. During DFS process, the vertex with smaller depth is first visited to avoid back tracking with long distance. When the printing path travels through vertex B, edge BC is visited before BD. D_{OQ} is optimized using greedy algorithm. The printing path within Q_1 starts at vertex A and ends at vertex H. Starting from H, we search for the closest vertex Q which has not been visited (vertex I), and set I as the starting vertex to traverse next graph Q_2.

3.3.3. G-code generation

After the printing path is optimized, the corresponding G-code is generated for 3D food printer. As the printing result is a two dimensional figure, only one layer is used. The print head needs to be lifted by a certain height (h_{lift}) when the extrusion of the material stops. After that, the print head travels through non-print area and come down when arriving at next starting point of the printing path. This procedure is used to prevent the scratch of the printed figure on printing plane. Since the vertical movement of the print head also costs time, larger h_{lift} leads to longer printing time.

If h_{lift} is too small, the print head could scratch the extruded material along the way as shown in Figure 10(c) (especially the area around two ears and the neck). We use $h_{lift} = 1.9$ mm in all our experiments.

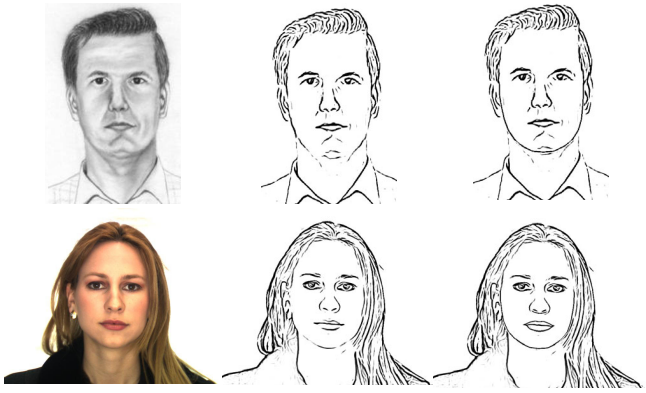

Fig. 7. The result of face enhancement. The first row is the result of enhancing cheek region only. The second row enhances nose and mouth region as well. The left column shows the input image. The middle column shows the FDoG output. The right column shows the result with face enhancement.

3.4. Face detection and landmark labeling

Since FDoG filter [19] is based on shape and color filtering of the input image, the abstraction results for portrait images are likely to lose face features. As shown in Figure 7, the outline of face region is not well preserved after performing FDoG filter. To handle this problem, we use OpenFace [24] to detect face landmarks and enhance the profile of cheek, mouth and nose.

3.5. Face sketch synthesis

FDoG filter is able to obtain feature-preserving results. The filtered result may not be an appealing abstraction, especially for portrait images. Therefore, sketch synthesis [25] can be further applied to generate feature sketches of the input portrait. The synthesized sketch better convey what the artists see as the most distinctive features of a human face. As shown in Figure 8, (k) is the result from synthesized sketch and (d) is from FDoG filter directly. The sketch (e) looks different from the input photo (a) because of the stylistic exaggeration of facial features. It also filters out the redundant details and emphasizes face features. (e) also contains sketches in hair and eye region, which helps FDoG filter to abstract prominent features (i).

4. Results

We test our framework on a variety of input images. The printing material is strawberry jam, other printing materials such as chocolate syrup can be used as well. The same printing speed is used for all the examples. The maximum dimension of the input image is resized to 600 pixels.

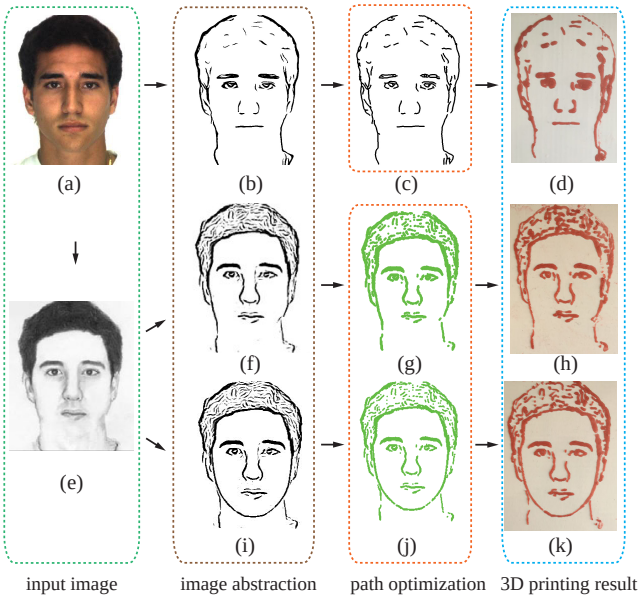

Fig. 8. Comparison of different methods. An erosion-based method is used as shown in (a-d). (a-e-f-g-h) shows results without using face enhancement. (a-e-i-j-k) is the full pipeline of our approach.

4.1. Examples for different printing methods

We use an erosion-based method as an alternative. If we apply basic image processing based on OpenCV and common G-code generation, the result is shown in Figure 8(a-d). For the input image (a), the FDoG output is shown in (b). As mentioned in Section 3.3.1, the FDoG output needs to be refined by taking into account printer properties such as the width of the nozzle. The existing erosion method provided by OpenCV is able to thin the FDoG output (as shown in (c)), and the printing result is presented in (d). This routine (a-d) is denoted as erosion-based method. It is easy to see that the difference between (c) and (d) is significant. The reason is that erosion method is not able to avoid overlap between neighboring paths, leading to obvious artifacts in the resultant fabrication. We should note that Figure 8(d) and (k) are not sufficient enough to show that our approach produce better results. Figure 9(c) and (e) are using the same FDoG results as input.

We can also see that sketch synthesis is effective to improve printing result. Our result (k) is able to preserve the prominent features of the input image (a). By only applying FDoG filter, the abstraction of hair is too sparse, while the eye region is too dense. For better feature preservation, we also perform sketch synthesis (e). And face enhancement can further help to elevate face features before printing. (f-h) are the results without face enhancement. The cheek region is not well preserved. In contrast, face enhancement better preserves cheek features.

4.2. Examples for different patch sizes

The patch size for path optimization plays an important role in our approach. As shown in Figure 10, large patch size (5×5 in Figure 10(j)) results in over-simplified result. Small patch size (1×1 in (f)) is not a good choice either. Actually if the width of the nozzle ($s = 4$) is larger than the patch size, it can easily cause overlap between neighboring paths and overfilled printing results. This effect also happens with the patch size 2×2 as shown in Figure 10(g).

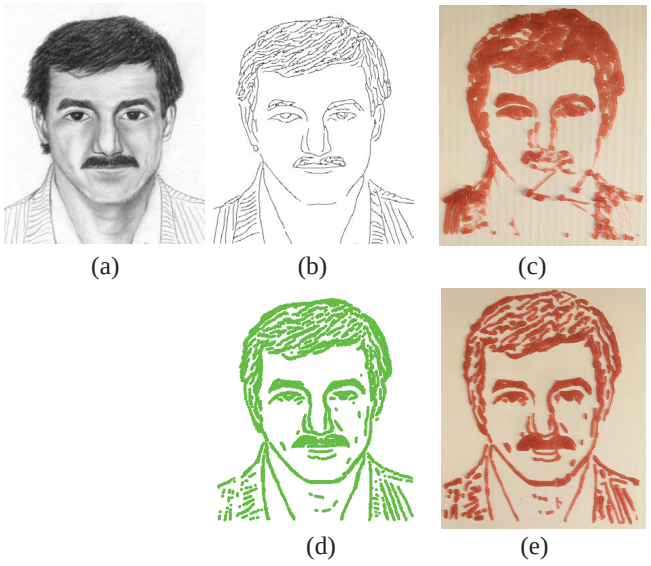

Fig. 9. Comparison between our approach and erosion-based method. (a) The input image. (b) Erosion result. (c) Food printing result. The simulation of printing path for (a) using our method is shown in (d). (e) Our printing result using 3D food printer.

Based on the discussion in Section 3.3.1, $s = 4$ is the 'ideal' width for the nozzle. However, the practical result (see Figure 10(i)) is too sparse. This is because the FDoG output can be oversimplified by large patch size. Given that the printing path is likely to be smaller than the width of the nozzle, the patch size used in our experiment is 3×3. Besides, slight overlap of the neighboring paths leads to smoother printing result (see Figure 10(h)).

4.3. Comparison of printing time

The printing time is related to several factors, such as the length of the printing path, the up-and-down process of the print head, and the speed of print head movement. The experiments below are under the same printing speed using the same 3D food printer.

Directly printing FDoG output pixel by pixel is time-consuming. As shown in Figure 10, (c) is the printing result from (b) using DFS method to generate the path. The printing time for (c) is 1920 seconds, while our result (h) costs 585 seconds. By using proper patch size and DFS-based path optimization method, our method simplifies the printing path of the print head, which reduces the printing time significantly.

Our method is also faster than erosion-based method (Table 1). The path of the erosion-based method is also generated using DFS-based algorithm. Although the printing result of erosion-based method(Figure 8(d) and Figure 9(b)) seems more concise than our approach (Figure 8(k) and Figure 9(e)), our method takes less time to print due to the patch-based path optimization process.

Besides, there are obvious artifacts in erosion-based result, as shown in Figure 9(c). Due to the overlap of neighboring paths, the stacked strawberry jam is higher than h_{lift} in some

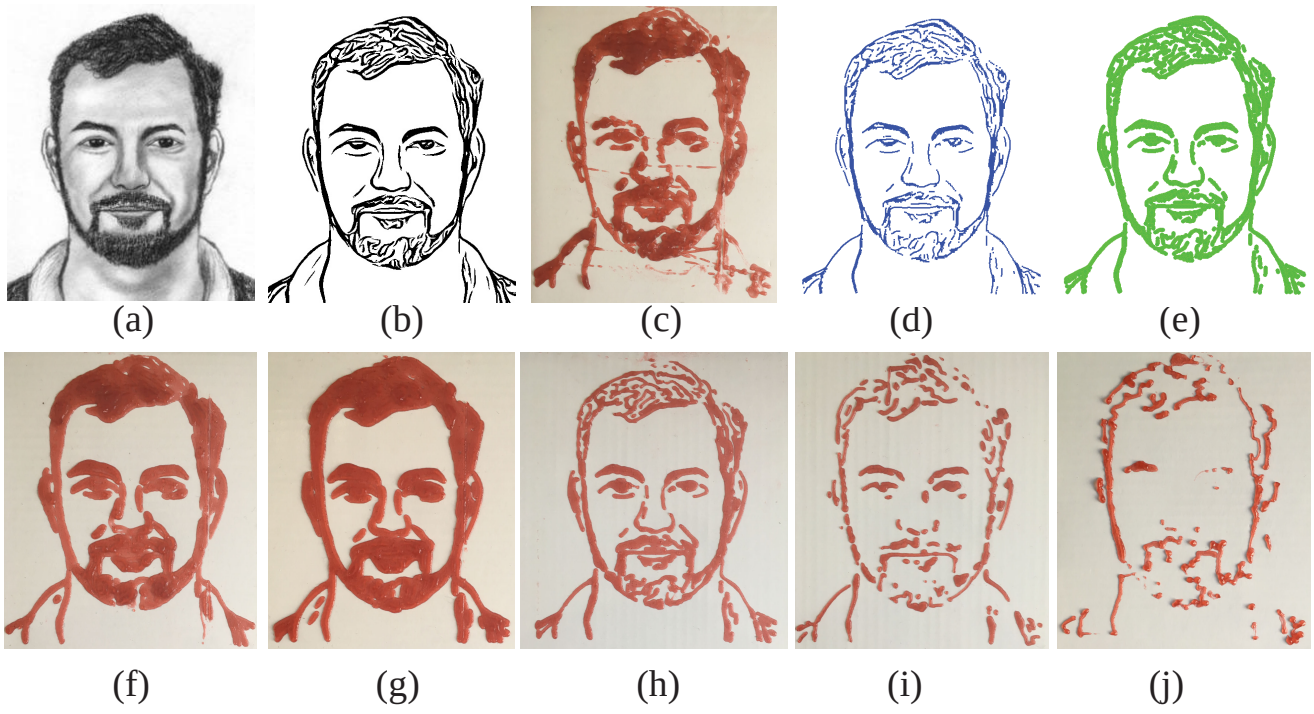

Fig. 10. Comparison of results using different patch sizes. (a) The input image. (b) The FDoG output. (c) Result by printing (b) pixel by pixel using DFS method to generate the printing path. Using our method, the patch centers of (h) are shown in (d). (e) is the simulated printing result of (d) and (h) is the printing result. (f-j) are the results using patches with different sizes, 1×1, 2×2, 3×3, 4×4 and 5×5 respectively.

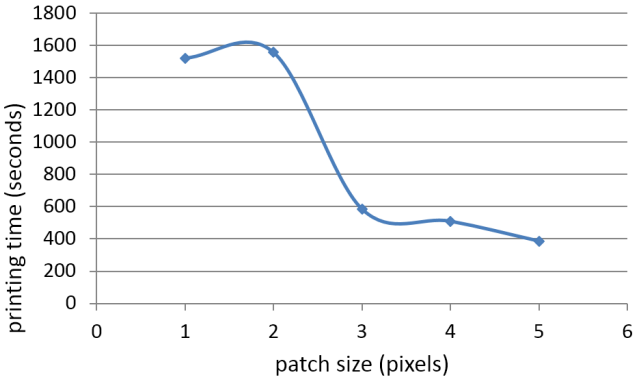

Fig. 11. Printing time using different patch sizes.

areas. It occludes the path of print head when moving through these areas, leading to erroneous results (in cheek and shoulder region).

The printing time is also related to patch size. The printing path becomes shorter as the patch size increases, so as the printing time. The printing time for Figure 10 (f-j) is represented in Figure 11. In the examples using 1×1 and 2×2 patches, the printing paths are densely distributed. Thus the printing time is mainly affected by vertical movements of the printer. This results in less time with patch size 1. When patch size increases from 2 to 5, the printing time becomes less.

4.4. More results

Figure 15 shows more printing results. It is easy to see that our printing method preserves the hair style and the features of the portrait images.

	erosion-based method	Ours
Figure 8	820 (d)	626 (k)
Figure 9	1039 (c)	737 (e)

Table 1. Printing time (in seconds) comparison between erosion-based method and ours. The letter following printing time indicates the corresponding example.

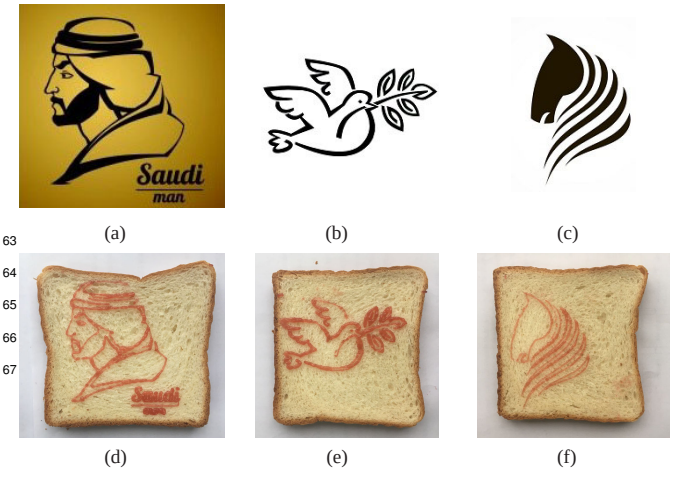

Fig. 12. Results printed on sliced bread.

Our method is able to print two dimensional figures on other edible items. As shown in Figure 12, the strawberry jam is printed on bread slices. The dimension of the bread is around 100 mm×100 mm and the printable area is 85 mm×85 mm. There are numerous tiny holes on the bread. When the bread is placed on the printing plane, the edge of the bread is higher

than the center, making it difficulty to stick printing material to the bread. As a result, the amount of the extrusion is increased (1.53× compared with results generated in Figure 10).

Based on the pipeline shown in Figure 3, our framework can be directly applied to other images as well. Some animal images are printed on bread in Figure 12(e-f).

4.5. Performance

We have implemented our algorithm on a desktop PC with an Intel Xeon(R) E3-1230 CPU (3.30GHZ) and 8G memory. Given a typical 500×600 pixel image as shown in Figure 10, it takes 0.687 seconds to generate the printable G-code of patch size 3×3 (h) from the image abstraction result (b). The smaller patch size will result in longer computational time. It takes 1.634 seconds to generate printing path for (f) using 1×1 patches. For Figure 8(i) with less details, it takes 0.564 seconds to generate the printing path for 3×3 patches.

5. Limitations

The sketch synthesis method from [25] has limitations. Only portraits facing front are preferable. For those images as shown in Figure 13(a,d), this method does not work very well. Our framework can also process such images using the routine in Figure 3(g-j). However, the printing results may not be so appealing compared with the input images. For input image in Figure 13(d), the FDoG output contains too many tiny details especially in the eye region (see Figure 13(e)), which results in artifacts in the printing result (see Figure 13(f)) due to the low resolution constraints of the 3D printer.

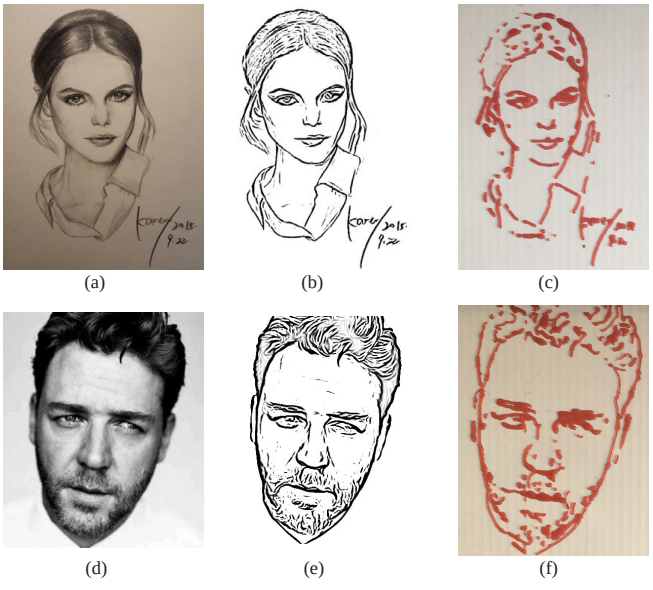

Fig. 13. Limitations. The frequent on-off switches of material extrusion leads to nonuniform width in hair regions. The printing results heavily rely on the FDoG output.

For FDoG filter, other than causing artifacts by redundant details, the printing quality can also be affected due to frequent switches of material extrusion. The viscosity of the edible material can easily cause less or over extrusion when the material switch is invoked. Thus the printing path can be either too thin or or too wide. Figure 13(b) shows that the hair region consists of many short strokes. This is because the material extrusion is switched on and off very frequently. It can also lead to nonunified width of the printing path as shown in Figure 13(c).

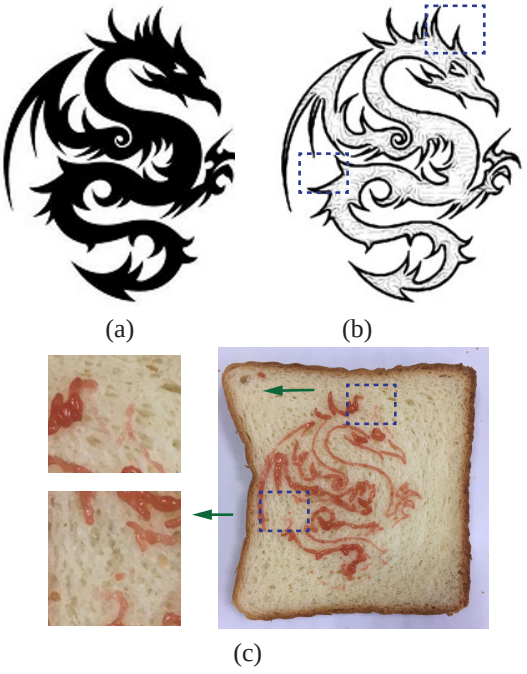

Fig. 14. Limitations. The print bed effects the printing quality.

The shape of the print bed also effects the printing quality. Sliced bread, for example, is not an ideal print bed. The edge of the bread is higher than the center. Therefore, the print bed is not horizontal. Also, as the sliced bread can absorb print material, the printing path is hard to preserve the desirable features. Numerous tiny holes on the bread can aggravate the situation, resulting in discontinuous paths as shown in Figure 14(c).

6. Conclusions and Future Work

We have proposed an effective framework that automatically fabricates customized food from an input image. Image abstraction is firstly applied to extract prominent image features. An optimized patch size is also determined to facilitate the path optimization process. Then a novel DFS-based optimization is presented to optimize the printing path. Finally, the corresponding G-code is generated for the real fabrication. Our framework ensures that the printing result preserves the abstracted features from the input image while saving printing time. For portrait images, the facial features can be better preserved by sketch synthesis and face enhancement, aiming at filtering out redundant information while keeping prominent features. The effectiveness of our framework is validated by experiments and comparisons on a variety of images using a tailor-made 3D food printer.

In the future, our framework can be further refined by improving the relevant image processing tools. We would like to explore better sketch synthesis technique that is robust to varying poses and lighting conditions, to improve the quality of the

Fig. 15. More results. The top row shows input portrait images. We should note that there is no sketch synthesis process involved in these examples.The bottom row shows corresponding results generated by our framework.

printed figures. Besides, the performance of image abstraction largely affects the printing result in our approach. The most difficult parts for portrait images are hair, mustache and wrinkles. More effective image abstraction techniques can be further investigated to better extract features from these parts.

As shown in Fig. 2, the food printer is capable to fabricate 3D models with multiple layers. For portraits, printing a bas-relief result requires the 3D face reconstruction process based on a single image. The reconstruction method introduced by Zhu *et al*. [41] is able to produce 3D face model. We can extend current approach to print 3D models by optimizing the printing pattern and path layer by layer. Nevertheless, there are several challenges. The reconstructed model determines the quality of the printing result. The spatial relation between layers should be considered, such as the lower layer is supposed to support the upper part. In addition, printing multiple layers is time-consuming. Inner carving [11] may be helpful to reduce printing material usage and speed up printing process. The 3D-aware printing path generation algorithm introduced by Lensgraf and Mettu [42] can be adopted to save printing time, which optimizes the printing routine based on the local feature independence of the printing object.

References

[1] Wikipedia, . Sugar painting — wikipedia, the free encyclopedia. 2016. URL: https://en.wikipedia.org/w/index.php?title=Sugar_painting&oldid=717653990; [Online; accessed 2-February-2017].

[2] Lipton, JI, Cutler, M, Nigl, F, Cohen, D, Lipson, H. Additive manufacturing for the food industry. Trends in Food Science & Technology 2015;43(1):114 – 123. doi:http://dx.doi.org/10.1016/j.tifs.2015.02.004.

[3] Southerland, D, Walters, P, Huson, D. Edible 3D Printing. In: NIP & Digital Fabrication Conference. 2011, p. 819–822(4).

[4] an Jin, Y, He, Y, zhong Fu, J, feng Gan, W, wei Lin, Z. Optimization of tool-path generation for material extrusion-based additive manufacturing technology. Additive Manufacturing 2014;1C4:32 – 47. doi:http://dx.doi.org/10.1016/j.addma.2014.08.004.

[5] Zhao, H, Gu, F, Huang, QX, Garcia, J, Chen, Y, Tu, C, et al. Connected Fermat Spirals for Layered Fabrication. ACM Trans Graph 2016;35(4):100:1–100:10. doi:10.1145/2897824.2925958.

[6] Chen, H, Liu, Z, Rose, C, Xu, Y, Shum, HY, Salesin, D. Example-based Composite Sketching of Human Portraits. In: Proceedings of the 3rd International Symposium on Non-photorealistic Animation and Rendering. 2004, p. 95–153. doi:10.1145/987657.987673.

[7] Zhang, Y, Dong, W, Deussen, O, Huang, F, Li, K, Hu, BG. Data-driven Face Cartoon Stylization. In: SIGGRAPH Asia 2014 Technical Briefs. 2014, p. 14:1–14:4. doi:10.1145/2669024.2669028.

[8] Liu, L, Shamir, A, Wang, C, Whitening, E. 3d printing oriented design: Geometry and optimization. In: SIGGRAPH Asia 2014 Courses. 2014,.

[9] Zhou, Q, Panetta, J, Zorin, D. Worst-case Structural Analysis. ACM Trans Graph 2013;32(4):137:1–137:12. doi:10.1145/2461912.2461967.

[10] Stava, O, Vanek, J, Benes, B, Carr, N, Měch, R. Stress Relief: Improving Structural Strength of 3D Printable Objects. ACM Trans Graph 2012;31(4):48:1–48:11. doi:10.1145/2185520.2185544.

[11] Prévost, R, Whiting, E, Lefebvre, S, Sorkine-Hornung, O. Make It Stand: Balancing Shapes for 3D Fabrication. ACM Trans Graph 2013;32(4):81:1–81:10. doi:10.1145/2461912.2461957.

[12] Bächer, M, Whiting, E, Bickel, B, Sorkine-Hornung, O. Spin-it: Optimizing Moment of Inertia for Spinnable Objects. ACM Trans Graph 2014;33(4):96:1–96:10. doi:10.1145/2601097.2601157.

[13] Zhao, H, Hong, C, Lin, J, Jin, X, Xu, W. "Make it swing: Fabricating personalized roly-poly toys ". Computer Aided Geometric Design 2016;43:226 – 236. doi:http://dx.doi.org/10.1016/j.cagd.2016.02.001.

[14] Chen, D, Levin, DIW, Didyk, P, Sitthi-Amorn, P, Matusik, W. Spec2Fab: A Reducer-tuner Model for Translating Specifications to 3D Prints. ACM Trans Graph 2013;32(4):135:1–135:10. doi:10.1145/2461912.2461994.

[15] Sitthi-Amorn, P, Ramos, JE, Wangy, Y, Kwan, J, Lan, J, Wang, W, et al. Multifab: A machine vision assisted platform for multi-material 3d printing. ACM Trans Graph 2015;34(4):129:1–129:11. doi:10.1145/2766962.

[16] Lipton, J, Arnold, D, Nigl, F, Lopez, N, Dan, C, Norn, N, et al. Multi-material food printing with complex internal structure suitable for conventional post-processing. In: 21st solid freeform fabrication symposium. 2010, p. 809–815.

[17] Wegrzyn, TF, Golding, M, Archer, RH. Food Layered Manufacture:

A new process for constructing solid foods. Trends in Food Science & Technology 2012;27(2):66 – 72. doi:http://dx.doi.org/10.1016/j.tifs.2012.04.006.

[18] DeCarlo, D, Santella, A. Stylization and Abstraction of Photographs. ACM Trans Graph 2002;21(3):769–776. doi:10.1145/566654.566650.

[19] Kang, H, Lee, S, Chui, CK. Coherent Line Drawing. In: Proceedings of the 5th International Symposium on Non-photorealistic Animation and Rendering. 2007, p. 43–50. doi:10.1145/1274871.1274878.

[20] Kyprianidis, JE, Kang, H. Image and Video Abstraction by Coherence-Enhancing Filtering. Computer Graphics Forum 2011;30(2):593–602. doi:10.1111/j.1467-8659.2011.01882.x.

[21] Fu, H, Zhou, S, Liu, L, Mitra, NJ. Animated construction of line drawings. ACM Trans Graph 2011;30(6):133:1–133:10. doi:10.1145/2070781.2024167.

[22] Liao, Q, Jin, X, Zeng, W. Enhancing the Symmetry and Proportion of 3D Face Geometry. IEEE Transactions on Visualization and Computer Graphics 2012;18(10):1704–1716. doi:10.1109/TVCG.2012.26.

[23] Baltrusaitis, T, Robinson, P, Morency, LP. Constrained Local Neural Fields for Robust Facial Landmark Detection in the Wild. 2013 IEEE International Conference on Computer Vision Workshops 2013;:354–361doi:doi.ieeecomputersociety.org/10.1109/ICCVW.2013.54.

[24] Baltrusaitis, T, Robinson, P, Morency, LP. OpenFace: An open source facial behavior analysis toolkit. 2016 IEEE Winter Conference on Applications of Computer Vision (WACV) 2016;:1–10doi:doi.ieeecomputersociety.org/10.1109/WACV.2016.7477553.

[25] Wang, X, Tang, X. Face Photo-Sketch Synthesis and Recognition. IEEE Trans Pattern Anal Mach Intell 2009;31(11):1955–1967. doi:10.1109/TPAMI.2008.222.

[26] Berger, I, Shamir, A, Mahler, M, Carter, E, Hodgins, J. Style and Abstraction in Portrait Sketching. ACM Trans Graph 2013;32(4):55:1–55:12. doi:10.1145/2461912.2461964.

[27] Wah, PK, Murty, KG, Joneja, A, Chiu, LC. Tool path optimization in layered manufacturing. IIE Transactions 2002;34(4):335–347. doi:10.1023/A:1012839601085.

[28] Balic, J, Korosec, M. Intelligent tool path generation for milling of free surfaces using neural networks. International Journal of Machine Tools and Manufacture 2002;42(10):1171–1179.

[29] Jin, G, Li, W, Gao, L. An Adaptive Process Planning Approach of Rapid Prototyping and Manufacturing. Robot Comput-Integr Manuf 2013;29(1):23–38. doi:10.1016/j.rcim.2012.07.001.

[30] Wojcik, M, Koszalka, L, Pozniak-Koszalka, I, Kasprzak, A. Mzz-ga algorithm for solving path optimization in 3d printing. In: Proceedings to International Conference on Systems; vol. 15. 2015,.

[31] Pedersen, H, Singh, K. Organic Labyrinths and Mazes. In: Proceedings of the 4th International Symposium on Non-photorealistic Animation and Rendering. 2006, p. 79–86. doi:10.1145/1124728.1124742.

[32] Wong, FJ, Takahashi, S. A Graph-based Approach to Continuous Line Illustrations with Variable Levels of Detail. Computer Graphics Forum 2011;30(7):1931–1939. doi:10.1111/j.1467-8659.2011.02040.x.

[33] Corp., S. Shinnove-s2. 2017. URL: http://www.shiyintech.com/shiyintech/products/14303587_0_0_1.html.

[34] OpenCV, . Eroding and dilating. 2017. URL: https://docs.opencv.org/3.2.0/db/df6/tutorial_erosion_dilatation.html.

[35] Wikipedia, . Travelling salesman problem — wikipedia, the free encyclopedia. 2017. URL: https://en.wikipedia.org/w/index.php?title=Travelling_salesman_problem&oldid=764709411; [Online; accessed 13-February-2017].

[36] Leung, YT, Tam, TW, Wong, CS, Young, GH, Chin, FYL. Packing squares into a square. Journal of Parallel & Distributed Computing 1990;10(3):271–275.

[37] Wikipedia, . Dynamic programming — wikipedia, the free encyclopedia. 2017. URL: https://en.wikipedia.org/w/index.php?title=Dynamic_programming&oldid=775335246; [Online; accessed 24-April-2017].

[38] Wikipedia, . Route inspection problem — wikipedia, the free encyclopedia. 2016. URL: https://en.wikipedia.org/w/index.php?title=Route_inspection_problem&oldid=747462033; [Online; accessed 2-November-2016].

[39] Karp, RM. Reducibility among Combinatorial Problems. Springer US; 1972.

[40] Crescenzi, P, Kann, V. A compendium of NP optimization problems. 2017. URL: http://www.nada.kth.se/~viggo/problemlist/compendium.html.

[41] Zhu, X, Lei, Z, Yan, J, Yi, D, Li, SZ. High-fidelity pose and expression normalization for face recognition in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015, p. 787–796.

[42] Lensgraf, S, Mettu, RR. Beyond layers: A 3d-aware toolpath algorithm for fused filament fabrication. In: IEEE International Conference on Robotics and Automation 2016. 2016, p. 3625–3631. doi:10.1109/ICRA.2016.7487546.