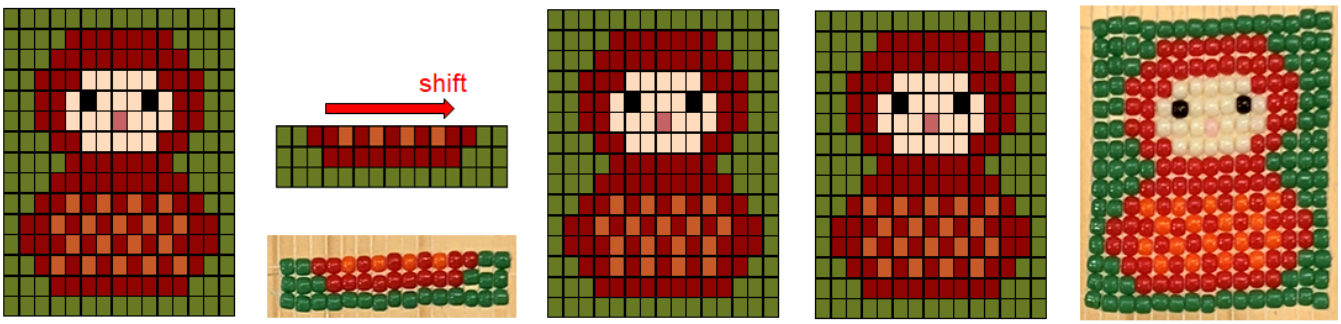
DOI: 10.1111/cgf.14694
Pacific Graphics 2022
N. Umetani, E. Vouga, and C. Wojtan
(Guest Editors)

Volume 41 (2022), Number 7

Pixel Art Adaptation for Handicraft Fabrication

Yuki Igarashi[1] and Takeo Igarashi[2]

[1]Ochanomizu University, Japan
[2]The University of Tokyo, Japan



(a) Original pixel art (b) Error (c) Without adaptation (d) With adaptation (e) Fabricated

Figure 1: *System overview. The inputs include the original pixel art (a) and the fabrication error (b). Continuing fabrication without adaptation breaks the lattice pattern (c). The system adapts the remaining pixel art minimizing distortion between the original and adapted pixel art (d). In this way, the user can recover from the error and continue fabrication (e).*

Abstract

Knitting and weaving patterns can be visually represented as pixel art. With hand knitting and weaving, human error (shifting, duplicating, or skipping pixels) can occur during manual fabrication. It is too costly to change already-fabricated pixels, so experts often adapt pixels that have not yet been fabricated to make the errors less visible. This paper proposes an automatic adaptation process to minimize visual artifacts. The system presents multiple adaptation possibilities to the user, who can choose the proposed adaptation or untie and re-fabricate their work. In typical handicraft fabrication, the design is complete before the start of fabrication and remains fixed during fabrication. Our system keeps updating the design during fabrication to tolerate human errors in the process. We implemented the proposed algorithm in a system that visualizes the knitting pattern, cross-stitching and bead weaving processes.

CCS Concepts

• ***Computing methodologies*** → ***Computer graphics; Modeling and simulation;***

1. Introduction

Handicrafts have broad appeal as hobbies. Some handicraft designs can be represented as pixel art, including knitting, cross-stitching, and bead weaving embroidery patterns. For these types of handicrafts, it is necessary to fabricate each pixel based on the design drawing. However, even if the user is careful, errors often occur. Pixels may be accidentally shifted, duplicated, or skipped. It takes considerable effort to untie and re-fabricate handicraft patterns.

We surveyed 24 people who had experience of knitting pixel art patterns, and 23 of them said that they had made mistakes. When asked what they did when they made a mistake, all of them answered, "Untied it." Some said, "If it's a small mistake, I leave it as it is" or "I untie and fix that part." One person who answered that she had never made a mistake said, "I used counting tool."

In addition, untying due to mistakes in bead weaving is difficult because these works are "tied up" in a stepwise process. If a person skilled in a given handicraft makes a small error, they may continue working by adapting the design around the error. However, this is difficult for beginners.

In this paper, we propose an algorithm that adapts the design when the user makes an error (Figure 1). The system presents the

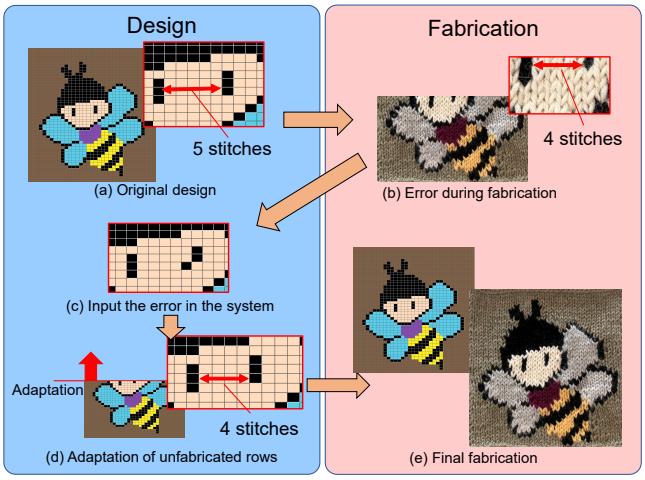


Figure 2: *Our system adapts the design during fabrication to tolerate errors.*

adapted design as pixel art immediately so that the user can decide whether to use it or untie and re-fabricate.

Computer-based handicraft [MI07, IIM12] and garment design support systems [UKIG11, BSK*16] have been proposed. In these systems, the design is complete before the start of fabrication and remains fixed during fabrication. Our system keeps updating the design during fabrication to tolerate human errors as in [ZSP13] (Figure 2).

This paper contributes 1) a handicraft fabrication workflow that adaptively changes the design during manual fabrication to tolerate errors, 2) adaptation algorithms using pixel art, and 3) verification of the method using knitting, cross-stitching, and bead weaving patterns. We describe related work in Section 2, the proposed algorithm in Section 3, and a prototype system that implements the proposed algorithm in Section 4. We describe the results of application to knitting, cross-stitching, and bead weaving patterns in Section 5. Finally, we summarize the paper and describe future work in Section 6.

2. Related Work

Several computer applications have been used to support the design of handicrafts. For example, Plushie facilitates design creation using 3D modeling and physics simulation in parallel, allowing children to design stuffed animals [MI07]. Beady helps users create original polyhedron bead designs with a uniform edge length by providing an interface based on sketch gestures and physical simulation [IIM12]. Beady also included a step-by-step construction guide for easy fabrication based on 3D graphics. However, although Beady facilitates the fabrication process after the design is finished, it does not allow users to return to the design during fabrication.

Igarashi et al. [IIS08] proposed an algorithm that uses a 3D surface model as input and converts it into a knitting stitch model. Yuksel et al. [YKJM12] presented an interactive modeling tool for knitting that grants the user precise control over the placement and shape of each stitch through immediate feedback. The final model is produced by a physical simulation that relaxes the local yarn shape, while preserving the overall shape of the cloth model to ensure predictable results. Fully automatic algorithms have also been proposed [WGF*18, WSY19], and research on knitting machine outputs has also been published [NWYM19, WTY*21].

Some research has also been conducted on production during design. Peng et al. [PWMG16] presented a 3D modeling approach (On-The-Fly Print) that allows the user to design 3D models digitally while a low-fidelity physical wireframe model is printed in parallel. RoMA is an interactive fabrication system designed to facilitate the integration of hands-on design with fast incremental printing [PBW*18]. Zoran and Paradiso [ZP13] developed FreeD, a handheld digital milling device. FreeD lets the user reinterpret the predesigned model by making dynamic creative decisions.

Han et al. [HWH*18] presented a novel unsupervised learning method for pixelization. They modeled the pixelization and depixelization processes in the same network in a bi-directional manner, with the input serving as training data. Kopf et al. [KSP13] presented a novel content-adaptive image-downscaling method that optimizes the shape and locations of down-sampling kernels to better align them with local image features. These methods are designed to make the initial design. Adaptation is not discussed.

Seam carving [AS07, RSA08] and image warping [GSCO06, WGCO07] have been used to resize inhomogeneous images. Seam carving greedily removes or inserts 1D seams that pass through the less important regions in the image. Wang et al. [WTSL08] presented a "scale-and-stretch" warping method that allows images to be resized with arbitrary aspect ratios, while preserving visually important features. To support the design and fabrication of handicrafts, we propose an algorithm to adapt the unfinished part of handicraft patterns in real-time without changing the size of the image.

3. Algorithm

The algorithm is designed based on the following assumptions. Fabrication starts from the bottom and proceeds upwards. The user identifies an error immediately after completing a row with an error and runs the algorithm to adapt the remaining rows. An error is limited to the last row. Multiple pixels can be changed in the last row. An error changes pixel color to another color used in the same row. The algorithm is designed for the most typical error of shifting a pixel one to the left or right.

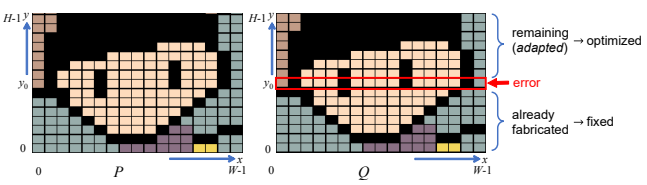


Figure 3: *Fabricated and remaining rows.*

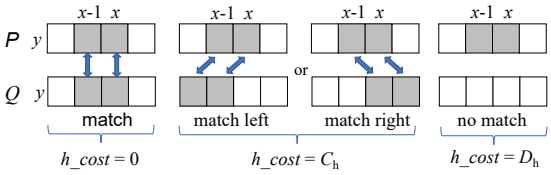


Figure 4: *Computation of horizontally cost h_cost*(P,Q,x,y).

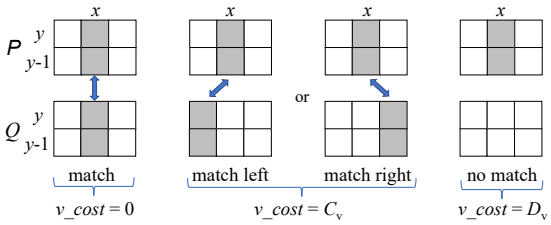


Figure 5: *Computation of vertically cost v_cost*(P,Q,x,y).

Figure 1 shows an overview of the algorithm. The input is the original pixel art, $P[x][y], x \in [0,\ldots,W-1], y \in [0,\ldots,H-1]$ (Figure 1(a)), and the pixel row with the error, $Q[x][y_0], x \in [0,\ldots,W-1]$ (Figure 1(b)). W is the width and H is the height of the input pixel art. The origin of the coordinates is in the lower left of the image. This algorithm does not adapt pixels that have already been fabricated ($Q[x][y] = P[x][y], x \in [0,\ldots,W-1], y \in [0,\ldots,y_0-1]$) (Figure 3). The output is the adapted design for the remaining part ($Q[x][y], x \in [0,\ldots,W-1], y \in [y_0+1,\ldots,H-1]$).

We obtain the adapted pixel art Q for the given original pixel art P by solving the following optimization problem.

$$Q = \underset{Q}{arg\,min}\, cost(P,Q) \tag{1}$$

$$\begin{aligned} cost(P,Q) = w_h \sum_y \sum_x (h\_cost(P,Q,x,y) + h\_cost(Q,P,x,y)) \\ + w_v \sum_y \sum_x (v\_cost(P,Q,x,y) + v\_cost(Q,P,x,y)) \end{aligned} \tag{2}$$

where $h\_cost(P,Q,x,y)$ and $v\_cost(P,Q,x,y)$ represent the cost associated with two horizontally and vertically adjacent pair pixels in P, w_h and w_v represent the weights, respectively.

If the same pair is in Q at the same location, the cost is zero. If the same pair is in Q but within one pixel displacement, then the cost is small C_h, C_v. If the same pair does not appear within one pixel displacement, then the cost is large D_h, D_v. We use $C_h = C_v = 1$, $D_h = D_v = 10$. See Fig. 4, 5 for the description of *h_cost* and *v_cost*.

We solve the optimization problem (eq. 1) from bottom to top row in a greedy way, and from left to right column in each row by dynamic programming. So, the computation cost is O(HWC^2), where C is the number of colors used in the original pixel art. In pixel art for handicrafts, the number of colors is about 10 at most.

Algorithm 1 Proposed algorithm

```
for(int y=0; y<H; y++) // greedy
  float cost[W][C] = 0;
  int prev[W][C] = Pointer for backtrace. Previous color
  for(int x=0; x<W; x++) // DP
    for(int c=0; c<C; c++) // color of Q[x][y]
      min_cost = float.MAX_VALUE;
      min'_c = 0;
      for(int c'=0; c<C; c'++) // color of Q[x-1][y]
        Q[x-2][y] = prev[x-1][c'];
        Q[x-1][y] = c';
        Q[x][y] = c;

        cost = cost[x][c']
          + w_h(h_cost(P,Q,x,y) + h_cost(Q,P,x-1,y))
          + w_v(v_cost(P,Q,x,y) + v_cost(Q,P,x-1,y));

        if (cost(c') < min_cost)
          min_cost = cost;
          min'_c = c';
      cost[x][c] = min_cost;
      prev[x][c] = min'_c;

// back trace. finalize Q.
c' = arg min cost[W-1][c];
      c
for(int x=W-1; x>=0; x- -)
  Q[x][y] = c';
  c' = prev[x][c'];
```

See Algorithm 1 for the detailed code. *h_cost* and *v_cost* return a higher cost if x or $x-1$ points to -1.

4. Prototype system

We created a prototype system that implements the proposed algorithm in real-time on a Windows PC (Core i7, 8MB RAM). In the prototype system, the pixels do not have to be square. The user can specify the vertical and horizontal size of a pixel. This system has the ability to convert images to pixel art. It has an automatic conversion and a pixel-by-pixel editing interface. The system also provides pixel art visualization results for knitting, cross-stitching, and bead weaving as shown in Fig. 6.

In our observation, the best weights of the cost function differ depending on the input image. Our system thus performs multiple adaptations by changing the weights and presents the results to the user as in suggestive interface [IH01] as shown in Figure 7. We use $(w_h, w_v) = (1,0),(0,1),(1,1),(2,1),(1,2)$. The reasons for selecting these parameters are: horizontal only, vertical only, both, horizontal more emphasis, and vertical more emphasis. The system then eliminates duplicated adapted pixels and presents the result to the user. The system first shows the original and erroneous pixel art. Then, the system arranges the adapted pixels in descending order of duplication. By clicking on a thumbnail image, the system shows the corresponding pixel art and the changed pixel chart. The adaptations are performed in real-time.

5. Results

Figure 8 shows the results of applying the proposed algorithm to various pixel arts. In the Daruma's tummy example, the first suggestion still suggests the original pattern after the error (Fig. 8(1c)). The next suggested pattern looks more natural with the pattern switched (Fig. 8(1d)). The last suggestion is also a good example of a different pattern (Fig. 8(1e)). The bee example is an example of wrong spacing between the eyes (Fig. 8(2b)). Instead of continuing to fabricate the pattern as it is (Fig. 8(2c)), an inward pattern has been suggested (Fig. 8(2d)). The alien is an example of mistaken foot (Fig. 8(3b)). If the user makes the top part of the mistake using the first pattern, it will look like they designed the toe (Fig. 8(3c)). The pattern suggested by the system is a design that changed the feet to straight (Fig. 8(3d)). The pine tree design is an example of the wrong parting of the branches (Fig. 8(4b)). The user can make it as is, but an example redesigned by the system is more attractive. The last example is a strawberry with multiple errors of white seeds (Fig. 8(5b)). The adapted pattern by the system looks the same as those that a human would make by themselves (Fig. 8(5d)).

Showing the pattern shown in Fig. 8, we asked 15 people (the ages of 20-25) which of the system's presentations they would choose. The results for the question of whether or not they were good at pixel art are shown in Fig. 9. After showing them the error step, we randomly presented them with the without-adaptation image and with-adaptation image, and asked them which one they would choose. They were not informed which image was the adapted. The results are shown in Table 1. There were no cases where the user did not select with-adaptation image. When asked if it would be useful for the system to present the information, most people answered that it would be useful (Fig. 10). We received the following comments on the merits of using the system to present adapted pixel arts. "I can start creating right away", "If I make a mistake, I can correct it right away.", "Easy!", "If I just think about it by myself, my thinking tends to be biased, but if the system suggests it, I may be able to make a better design."

(a) Pixel art (b) System visualization (c) Fabricated

Figure 6: *Pixel art visualization results and fabricated results.(top) knitting, (middle) cross-stitching, and (bottom) bead weaving.*

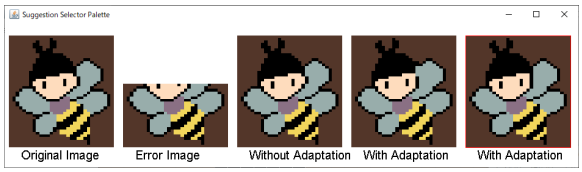


Figure 7: *Adaptation by changing the weights in the algorithm and presenting the result to the user.*

Table 1: *A study of which pixel art users prefer.*

Name	Without adaptation	With adaptation
Daruma	0 (0.0 %)	14 (93.3 %) , 1 (6.7 %)
Bee	0 (0.0 %)	15 (100.0 %)
Alien	9 (60.0%)	6 (40.0 %)
Pine tree	10 (66.7 %)	5 (33.3 %)
Strawberry	0 (0.0%)	15 (100.0 %)

6. Conclusion and Future work

We proposed an algorithm for adapting handicraft designs to accommodate errors made by users during fabrication. We also demonstrated a prototype system implementing the proposed algorithm. The system allows users to continue fabrication without unraveling their work to fix errors during knitting, cross-stitching, and bead weaving.

Our approach has some limitations. Fig. 11 shows some failing examples. It fails to shift diagonal strips (top). It does not work for more than one pixel shift (middle). The algorithm cannot propose adaptations based on structural information such as symmetry and repeating patterns (bottom).

Knitting and bead weaving are completed in a stepwise manner from the bottom row upward. The proposed adaptation method can also be applied to cross-stitch, but the user does not have to fabricate cross-stitch from bottom to top. In the future, we plan to consider an algorithm that adapts the remaining areas in any direction. In the current system, the user has to input a color error using GUI, but we are considering automatic input such as acquiring it with a camera.

Acknowledgments

We are grateful to Prof. Noriko Arai for her helpful advice. We thank to Moeko Nakajima for her pixel design. We wish to express our appreciation to the survey respondents and test users for their cooperation. We appreciate the valuable feedback by the anonymous reviewers. This work was supported in part by JST, PRESTO Grant Number JPMJPR16D1 and CREST Grant Number JPMJCR17A1, Japan.

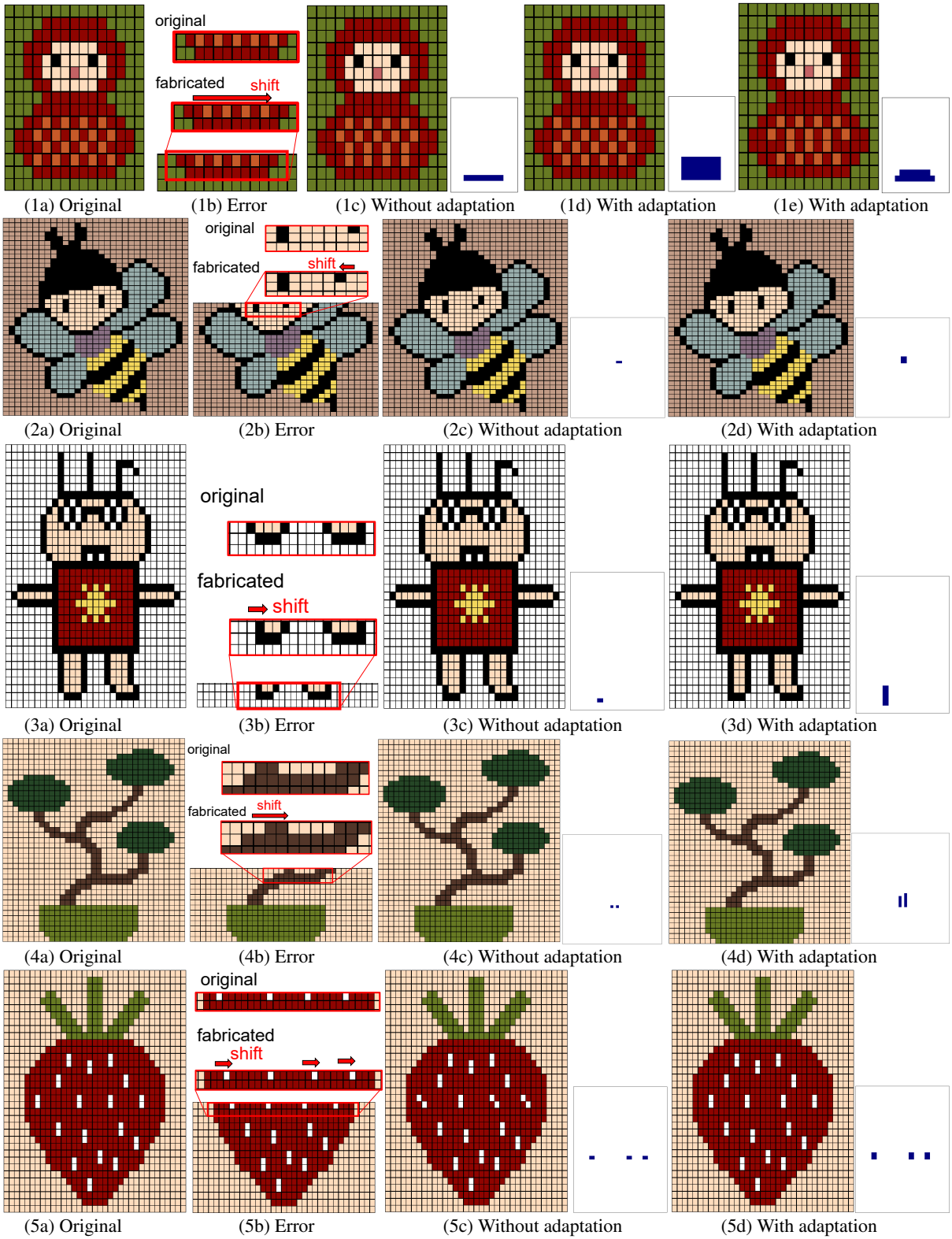


Figure 8: *Results obtained using the proposed algorithm.*

References

[AS07] AVIDAN S., SHAMIR A.: Seam carving for content-aware image resizing. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007) 26*, 3 (2007), 10–es. doi:10.1145/1276377.1276390. 2

[BSK*16] BARTLE A., SHEFFER A., KIM V. G., KAUFMAN D. M., VINING N., BERTHOUZOZ F.: Physics-driven pattern adjustment for direct 3d garment editing. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2016) 35*, 4 (2016). doi:10.1145/2897824.2925896. 2

[GSCO06] GAL R., SORKINE O., COHEN-OR D.: Feature-aware texturing. In *Proceedings of the Eurographics Symposium on Rendering Techniques* (01 2006), vol. 11, pp. 297–303. doi:10.2312/EGWR/EGSR06/297-303. 2

[HWH*18] HAN C., WEN Q., HE S., ZHU Q., TAN Y., HAN G., WONG T.-T.: Deep unsupervised pixelization. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2018) 37*, 6 (2018), 243:1–243:11. doi:10.1145/3272127.3275082. 2

[IH01] IGARASHI T., HUGHES J. F.: A suggestive interface for 3d drawing. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2001), UIST '01, Association for Computing Machinery, p. 173–181. doi:10.1145/502348.502379. 3

[IIM12] IGARASHI Y., IGARASHI T., MITANI J.: Beady: Interactive beadwork design and construction. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2012 31*, 4 (2012). doi:10.1145/2185520.2185545. 2

[IIS08] IGARASHI Y., IGARASHI T., SUZUKI H.: Knitting a 3d model. *Computer Graphics Forum 27*, 7 (2008), 1737–1743. doi:10.1111/j.1467-8659.2008.01318.x. 2

[KSP13] KOPF J., SHAMIR A., PEERS P.: Content-adaptive image downscaling. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2013) 32*, 6 (2013). doi:10.1145/2508363.2508370. 2

[MI07] MORI Y., IGARASHI T.: Plushie: An interactive design system for plush toys. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007) 26*, 3 (2007), 45–es. doi:10.1145/1276377.1276433. 2

[NWYM19] NARAYANAN* V., WU* K., YUKSEL C., MCCANN J.: Visual knitting machine programming. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2019) 38*, 4 (2019). (*Joint First Authors). doi:10.1145/3306346.3322995. 2

[PBW*18] PENG H., BRIGGS J., WANG C.-Y., GUO K., KIDER J., MUELLER S., BAUDISCH P., GUIMBRETIÈRE F.: Roma: Interactive fabrication with augmented reality and a robotic 3d printer. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (2018), CHI '18, p. 1–12. doi:10.1145/3173574.3174153. 2

[PWMG16] PENG H., WU R., MARSCHNER S., GUIMBRETIÈRE F.: On-the-fly print: Incremental printing while modelling. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (2016), CHI '16, p. 887–896. doi:10.1145/2858036.2858106. 2

[RSA08] RUBINSTEIN M., SHAMIR A., AVIDAN S.: Improved seam carving for video retargeting. *ACM Transactions on Graphicss (Proceedings of SIGGRAPH 2008) 27*, 3 (2008), 1–9. doi:10.1145/1360612.1360615. 2

[UKIG11] UMETANI N., KAUFMAN D., IGARASHI T., GRINSPUN E.: Sensitive couture for interactive garment editing and modeling. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2011) 30*, 4 (Aug 2011). doi:10.1145/2010324.1964985. 2

[WGCO07] WOLF L., GUTTMANN M., COHEN-OR D.: Non-homogeneous content-driven video-retargeting. In *2007 IEEE 11th International Conference on Computer Vision* (2007), pp. 1–6. doi:10.1109/ICCV.2007.4409010. 2

[WGF*18] WU K., GAO X., FERGUSON Z., PANOZZO D., YUKSEL C.: Stitch meshing. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2018) 37*, 4 (2018). doi:10.1145/3197517.3201360. 2

[WSY19] WU K., SWAN H., YUKSEL C.: Knittable stitch meshes. *ACM Transactions on Graphics 38*, 1 (2019). doi:10.1145/3292481. 2

[WTSL08] WANG Y.-S., TAI C.-L., SORKINE O., LEE T.-Y.: Optimized scale-and-stretch for image resizing. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2008) 27*, 5 (2008). doi:10.1145/1409060.1409071. 2

[WTY*21] WU K., TARINI M., YUKSEL C., MCCANN J., GAO X.: Wearable 3d machine knitting. *IEEE Transactions on Visualization and Computer Graphics* (2021). doi:10.1109/TVCG.2021.3056101. 2

[YKJM12] YUKSEL C., KALDOR J. M., JAMES D. L., MARSCHNER S.: Stitch meshes for modeling knitted clothing with yarn-level detail. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2012) 31*, 3 (2012). doi:10.1145/2185520.2185533. 2

[ZP13] ZORAN A., PARADISO J. A.: Freed: A freehand digital sculpting tool. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2013), CHI '13, p. 2613–2616. doi:10.1145/2470654.2481361. 2

[ZSP13] ZORAN A., SHILKROT R., PARADISO J.: Human-computer interaction for hybrid carving. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2013), UIST '13, Association for Computing Machinery, p. 433–440. doi:10.1145/2501988.2502023. 2

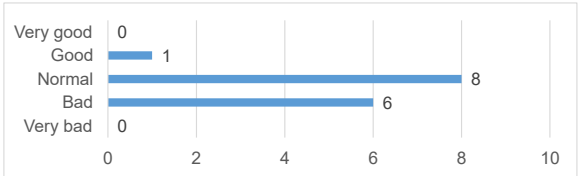


Figure 9: *Whether you are good at pixel art or not.*

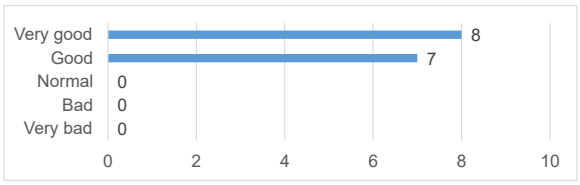


Figure 10: *Whether it is useful for the system to present.*

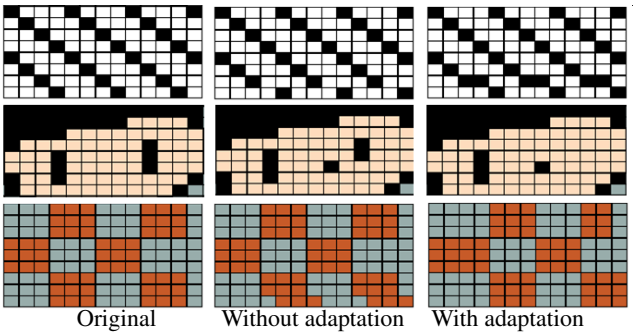


Figure 11: *Failure examples of algorithms.*