3D Dice Mosaics: A Multidirectional Dithering System

Hanan Tanasra[1], Gershon Elber[2], Yoav Sterman[3]

[1]Architecture, Technion, Haifa, Israel; hanantn@campus.technion.ac.il
[2]Computer Science, Technion, Haifa, Israel; gershon@cs.technion.ac.il
[3]Architecture, Technion, Haifa, Israel; sterman.yoav@technion.ac.il

Abstract

Dithering of images is a widely used technique for reducing the number of colors in an image to a small subset while maintaining as much information as possible. Dithering is typically applied on a single image although a few results could also be found on the simultaneous dithering of several images. In this paper, we describe a method for the simultaneous dithering of two or three gray-scale images, using the different faces of the die. This work presents (1) a dithering method which calculates the orientation for placing each die, (2) a method for generating a parametric 3D model of a jig to position the cubes, and (3) a positioning plan for the assembly process. As part of our results, we present two assembled dice mosaics, each with 4,096 dice, one that simultaneously dithers two images and the other simultaneously dithers three images.

Introduction

Dice mosaics are artwork produced by positioning dice in a 2D array. Each die (one dice cube) has one face that points outward and represents one pixel. The number of dots on the faces creates different effective grey-levels, which are mapped to a target image. Thousands of dice may be required to form a low-resolution image. However, since dice are created in an industrialized process, large quantities may be purchased at a minimal cost. In this work, we aim to simultaneously dither two or three images by exploiting two or three faces of each die. Figure 1 shows how the dice will be positioned. On the left side, the dice will be placed on the edge, while on the right side, they will be positioned on the corner.

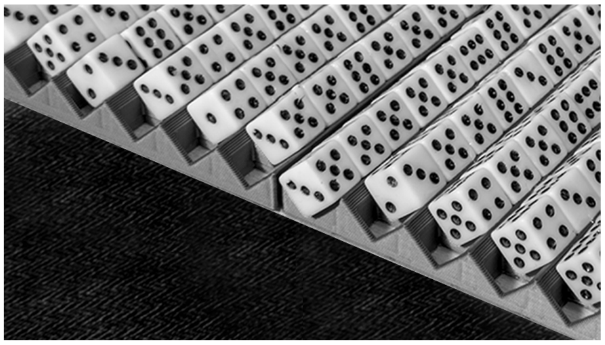
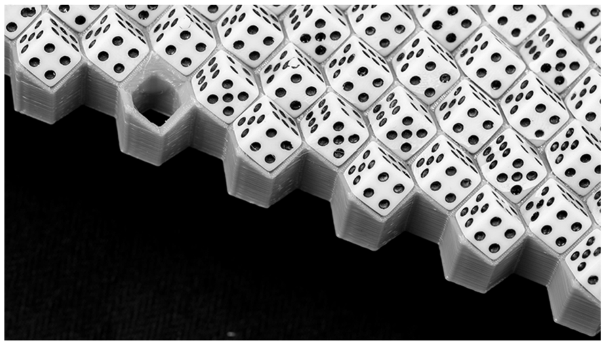

Figure 1: *(on the left) The dithering of two views, (on the right) The dithering of three views.*

Since the arrangement of the dots in the dice is fixed, the orientation of the dice in one view affects other views as well. To calculate the error in each view and between views, we have used a multi-image variation of the Floyd-Steinberg dithering algorithm [10]. This method is used to transform a high-resolution image to a lower-resolution version with less color depth, that computes the grey-level errors between each input pixel grey-level and the grey-level of the selected face of the cube. Then, those errors are diffused to neighboring pixels in each image, for all input images simultaneously. In addition to the dithering algorithm, we created a parametric 3D model of a jig for positioning all the dice. The model is scalable to fit the entered image size selected by the user and then 3D printed using a low-cost desktop

3D printer. Since the bed size of 3D printers is fixed, smaller jig parts may be glued together to form bigger ones in order to fabricate larger mosaics. We have also developed a workflow for the assembly of the dice mosaics. The algorithm outputs a paper assembly plan that is placed underneath the jig. Holes in the jig reveal numbers that assist in the dice positioning, in the different orientations. We created two mosaics to illustrate our algorithms and workflow. The first with two images (dice positioned on an edge), and the second with three (dice positioned on a corner). Each mosaic has the size of a 64 x 64 pixel image and is composed of 4,096 dice.

The rest of this work is organized as follows: we start by discussing previous work, only to continue with the core algorithms we have used here. Then, we present some results, and finally conclude with some discussion and potential future work.

Prior Work

Ken Knowlton, a pioneering computer graphics artist, created a variety of mosaics using different techniques. Among his famous works is a portrait of Albert Einstein made up of 999 dice arranged in a mosaic, demonstrating Knowlton's innovative use of unconventional materials to create digital art [13]. Lately, there has been an interest in generating dice mosaics by artists and hobbyists. Several open-source projects and web-based tools are available for translating images to assembly plans for creating the dice mosaics [6][7][17]. A company called “Dice Ideas" is offering commissions for custom dice mosaics based on customers' photos, where each mosaic presents one image [6]. Robert Bosch presented the lenticular dice mosaic titled “Monster Mash,” which displayed Frankenstein’s monster on one side and Dracula on the other using only 600 dice. It allows viewers to see the two portraits from two angles, although little was explained about the assembly process [11][2].

Another example of such mosaics is mosaics made of domino tiles. Similar to dice, domino pieces also have a different number of dots that are used to represent different intensities of brightness. Domino mosaics use complete sets of domino pieces to produce an image. The standard double nine dominoes set is comprised of 55 sets. An integer linear programming is often used to calculate these mosaics [1], however, improved and more scalable approaches were recently developed [3][4]. Another example of producing an image using a limited set of tiles are mosaics made from Rubik’s cubes. Rubik’s cubes enable the generation of colored mosaics, using the six colors of the cube. Each face is divided into nine squares that can be scrambled to create a high-resolution image when used with other cubes. Dan et al. created a robot that automatically arranges Rubik’s to construct such mosaics [5]. David Plaxco's paper outlines a set of criteria for categorizing pixelated projections of knots and uses it to define photogenic knot projections [14]. Both mosaics, either using domino or Rubik's cubes, portray a single image. Our goal is to present multiple images by using more than one face of the die, by placing them on an edge or a corner. A multi-image algorithm was previously used to dither and encode images, viewed from orthogonal directions, in a single object. In [12], short line segments are exploited as the basic multi-image dithering primitive, in [9], 3D space curves, and in [8], 3D dithering matrices were employed, towards a realized 3D cloud of points that was laser etched in a glass cube.

Algorithm

We now present the different stages of our dice-based dithering approach for multiple gray-scale images and multiple views. We start with a description of the multi dithering algorithm, only to discuss the assembly process of these physical arrangements, which consist of thousands of dice.

Multi-view Dithering Using Dice

In this work, we dither gray-scale images using dice with each die portraying one pixel in each image. The right side of Figure 2 presents how each face of a die presents a number, as one to six dots, and hence with varying effective grey-levels. The digit 1 effectively introduces the brightest face whereas the digit 6 presents the darkest face. Since the six faces of the dice do not cover the full range of grey-levels, a preliminary normalization step is required to map the image pixel intensity to the effective grey-level range that can be achieved using the dice faces. The arrangement of the digits across the faces of the dice is already decided beforehand. Consequently, this predefined condition places restrictions on the adjacency of the die's faces. When a die is balanced on an edge, as shown in Figure 2 in die (A), two of its adjacent faces are visible to the viewer, and it tiles the plane with effective rectangular shapes. While when balanced on a corner three adjacent faces are visible, as shown in Figure 2 in die (B), and it tiles the plane with hexagonal shapes.

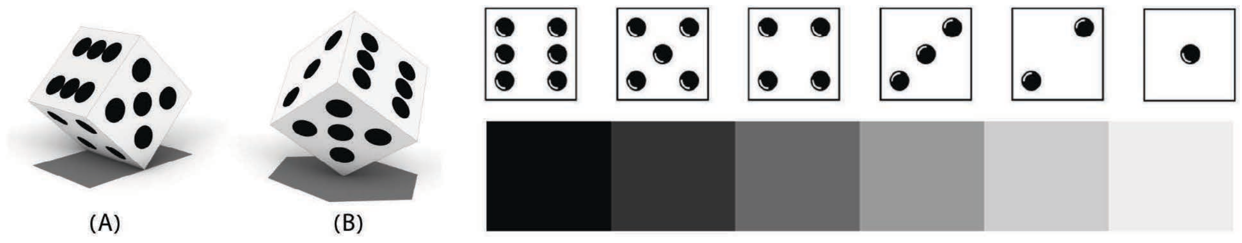

Figure 2: *On the left, we see the two die orientations employed in this work: (A) die cube on its edge with rectangular orthogonal projection, for dithering of two views, and (B) die cube on its corner for dithering of three views, with hexagonal orthogonal projection. On the right, we see the color gradient translation to die faces.*

It is these two or three exposed faces that we seek to select and optimize so two or three images will be simultaneously dithered using dice and seen from the given two or three viewing directions. We examine two or three pixels from different input images at the same time. Based on the two or three detected grey-levels, we select the two or three faces of the die that best match these grey-levels. However, some error is likely to result in the two/three pixels that the two/three selected faces of this die approximate. The error is measured as the difference between the input grey level of the pixel and the grey level that the selected face of the cube presents (using function **DiceFaceGreyLevel**). We compute the error in each pixel, as Err_i for each image, and propagate that error to the (unprocessed yet) neighboring pixels of that image, in a way similar to the way in which the classic Floyd-Steinberg algorithm operates on a single image. As a result, the Floyd-Steinberg algorithm preserves the overall intensities of the input images, employing neighboring pixels to compensate for errors in the current grey-level intensities. Algorithm 1, given below, computes proper orientations of the die to achieve simultaneous dithering of two/three gray-scale square images using dice, with error diffusion (Floyd Steinberg).

Input: n square gray-scale images, $\boldsymbol{I_i}$, i = 1 to n, n = 2 or 3, of the same size (Size, Size);

Output: An orientation layout of dice, one die per pixel in each input image;

Algorithm:

```
for ($y$ := 0; $y$ < Size; $y$++)
    for ($x$ := 0; $x$ < Size; $x$++)
        $p_i$ := $\boldsymbol{I_i}(x, y)$,  $i$ = 1 to $n$;                    // Collect the 2 or 3 relevant pixels.
        Orient($x$, $y$) := BestDiceCubeOrient($p_i$, $i$ = 1 to $n$);    // Find best die orientation.
        for ($i$ := 1; $i$ < $n$; $i$++)                                  // For each image:
            $\mathrm{Err}_i$ := $p_i$ – DiceFaceGreyLevel(Orient($x$, $y$), $i$);  // Compute the local error.
            Diffuse error $\mathrm{Err}_i$ to neighboring pixels as follows:     // Diffuse error to neighboring
                        $I_i(x + 1, y)$ += 7/16 $\mathrm{Err}_i$ ;              // pixels, in all images,
                        $I_i(x - 1, y + 1)$ += 3/16 $\mathrm{Err}_i$ ;          // following the classic
                        $I_i(x, y + 1)$ += 5/16 $\mathrm{Err}_i$ ;              // Floyd Steinberg.
                        $I_i(x + 1, y + 1)$ += 1/16 $\mathrm{Err}_i$ ;

return Orient($x$, $y$),  $\forall x, y$;
```

Algorithm 1 employs two functions, **BestDiceCubeOrient** and **DiceFaceGreyLevel**. Function **DiceFaceGreyLevel** simply returns the effective grey-level of the *i* face of the die in the computed orientation, Orient (*x*, *y*) as is shown in Figure 2. Function **BestDiceCubeOrient** computes the best orientation of a die, given the desired two/three grey-levels of the respective pixels in the two/three input images. A die on an edge (Figure 2 left cube) has 24 different orientations, having 12 positioning edges while each placement of an edge can be flipped 180 degrees. A die on a corner (Figure 2 right cube) also has 24 different orientations, having eight positioning corners while the die can be rotated around that corner 120 or 240 degrees, or 3 different orientations for each corner. In other words, the function **BestDiceCubeOrient** traverses all the presented discrete set of combinatorial possibilities and selects the best simultaneous match for the three images, while (the errors in) all images are weighed equally.

We utilized the renowned artworks Mona Lisa by Leonardo da Vinci and Salvador Dalí take by Philippe Halsman for dithering two images. For dithering three images, we employed the same images as the previous case along with the inclusion of the famous painting Girl with a Pearl Earring by Johannes Vermeer. The dithering output of each case is shown in Figure 3.

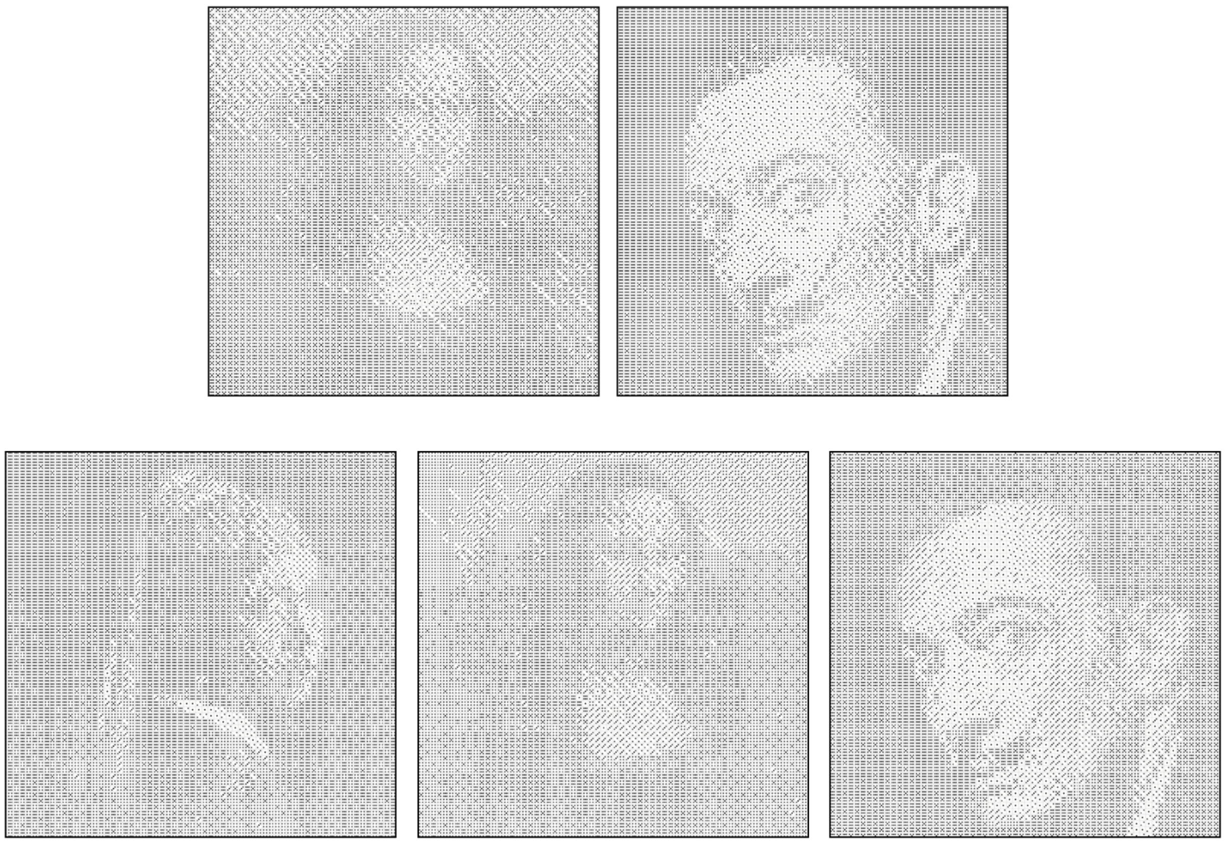

Figure 3: *(On the top) The dithering of two views, (on the bottom) The dithering of three views.*

Parametric Jig

With the orientations of all dice resolved, following Algorithm 1, the remaining challenge is the assembly of such a multi-dice arrangement. Toward this end, special parametric jigs are designed, and 3D printed. The jig is comprised of individual units, each of which corresponds to the tile shape that the dice project, as explained and displayed before in Figure 2. This leads to a non-square jig size, the dithered image will be scaled to fit the jig's shape. The shape of each unit is aligned with the standing angle of the die, and it securely holds its lower part, ensuring accurate and consistent placements. To expand the parametric potential of the jig, two primary parameters were identified: grid extent and dice size. These parameters were incorporated into a Grasshopper and Rhino programs to keep the jig's shape simple and reduce the printing duration. The jig's size is designed to align with the size of the images used, which are all square-shaped. For example, when generating mosaics with 64 x 64-pixel images, the jig size must encompass 64 x 64 units of dice. Since we need to 3D print the jig, changing this parameter helped us generate small jigs to fit the printer bed. The unit size is determined by the size of the dice used, which in our case is currently 8 mm. By integrating these parameters, we were able to create a jig that can be easily modified to accommodate various image sizes and dice dimensions. After the jig is generated, it can be exported as an STL file and sent directly to the printer for production.

Fabrication and Assembly Process

The jigs were designed so the prescribed orientation of each individual cube can be encoded and used for the assembly. Therefore, the jig's design can orient the die while exposing the assembly paper beneath it through the designed holes, as shown in Figure 4. During the assembly, the assembly sheet provides the proper orientation of each die. For example, the number '563' reveals that the top-left face of the dice should be a face with 5-dots, the bottom-middle has 6-dots, and the top-right has 3-dots. Similarly, '56' indicates that the left face is a 5-dots one and the right is a 6-dots.

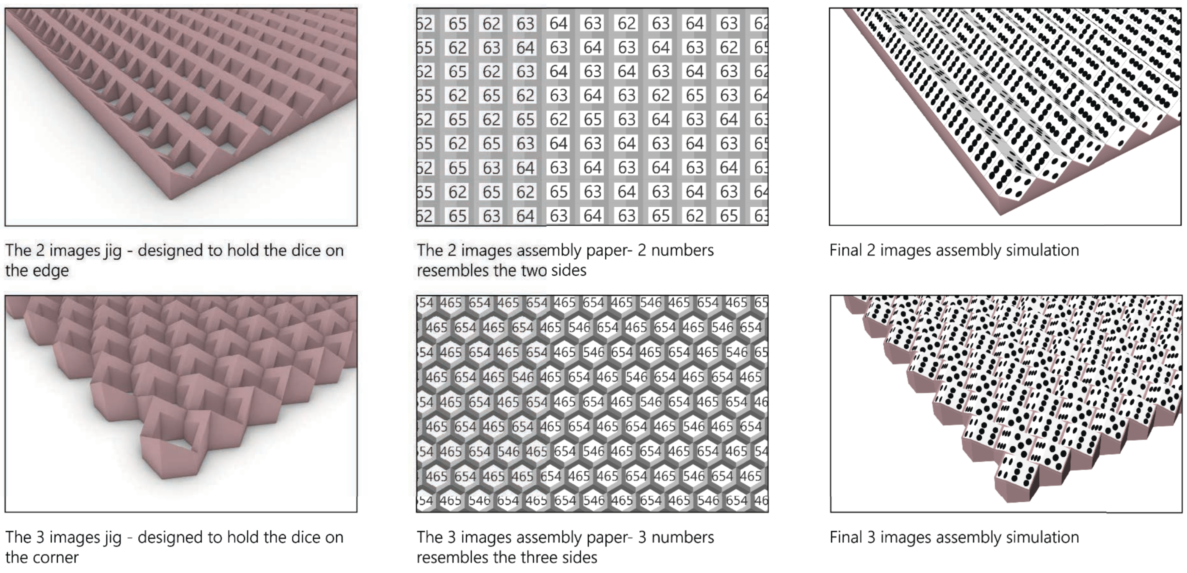
The 2 images jig - designed to hold the dice on the edge
The 2 images assembly paper- 2 numbers resembles the two sides
Final 2 images assembly simulation
The 3 images jig - designed to hold the dice on the corner
The 3 images assembly paper- 3 numbers resembles the three sides
Final 3 images assembly simulation

Figure 4: *The assembly process of two (first row) and three (second row) dithered images. The jigs are designed so the die will be properly oriented. The assembly paper can be seen through the jig holes. Super glue was then used to strengthen and attach the final assembly.*

Results

The size of the two-image dithering mosaic is 514 x725 x10 mm, while the size of the three-image mosaic is 631 x 730 x 10 mm. The first visible difference is that the models are not square in size, as the dithering algorithm generates. Since the die size on an edge vary from when it is on its corner, the jig size for each mosaic changes. Viewers can still see each image perfectly matched to the size of the jig. To start tessellating each mosaic, small jigs were printed and then glued together. The family members gathered to assemble. Each 64 x 64 pixel mosaic required a single assembly sheet and 4,096 dice. It took 12 hours to assemble the three images, which required finding the precise orientation of the three faces. The two images, on the other hand, took only 6 hours to complete. The beauty of this artwork is finding the proper angle to see the dithered picture clearly, see Figure 5. Viewers have to walk around the mosaic to figure it out. The whole process for creating the two mosaics is documented in [15]. The three-images model is presented in [16]. When viewed from a top-down perspective, the two mosaics display a flawless illustration created by combining individual pixels into a cohesive whole as seen in Figure 6.

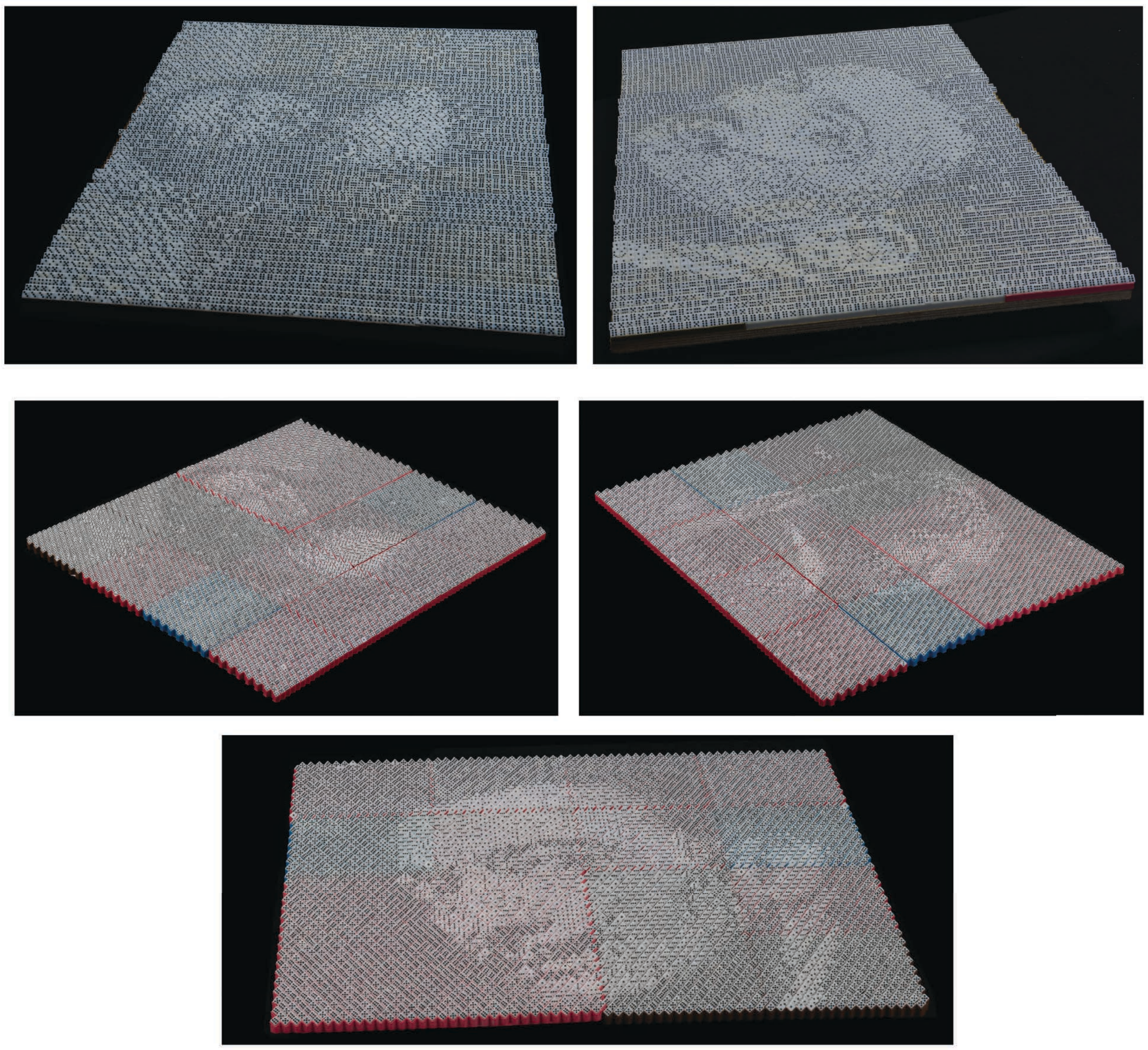

Figure 5: *Upper part, the two angles from the two-images dithering model, the Mona Lisa (left) and Salvador Dalí (right). Lower part, the three angles from the three-images dithering model, the Mona Lisa (left), Salvador Dalí (bottom), the Girl with the Pearl Earring (right).*

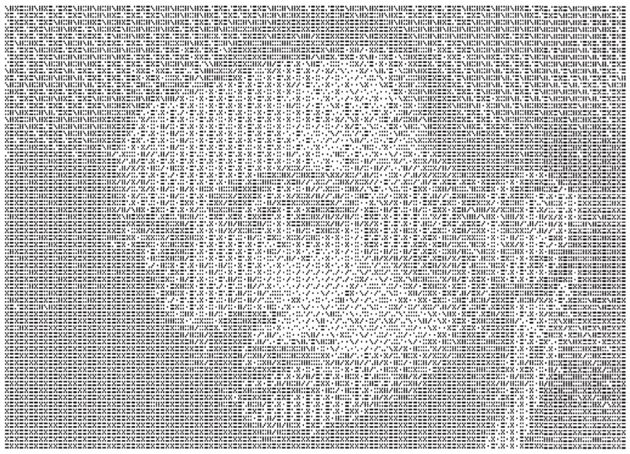
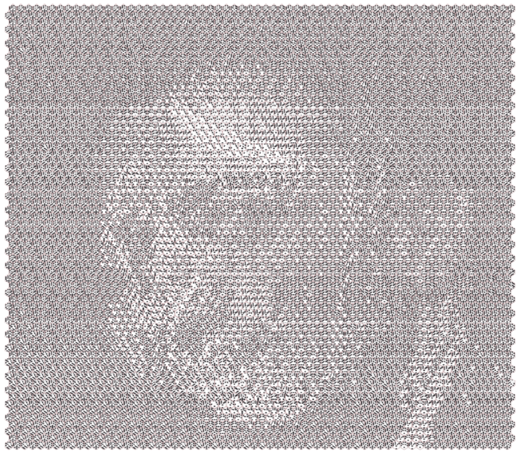

Figure 6: *On the left, the combined top view of the two images is displayed, while on the right side, the top view of the three images merged together is shown.*

Conclusions and Future Work

We introduced a workflow for simultaneously dithering two or three images using dice, with the main input being 2 or 3 square images and the output being a 3D printed jig with assembly paper. Observing the effect of dithering on multiple images revealed that while dithering two images results in an enhanced level of visibility, dithering three images led to a decrease in detail visibility; see Figures 6 and 7. This work can possibly be extended to dithering $N > 3$ gradient grayscale images using different placement abilities such as a transparent jig so the back side of the dice, facing the jig, can also dither some images. Then, Rubik's cubes and similar may also be used to create simultaneous dithering, in color. Furthermore, the jigs can be enhanced in a way that eliminates the need for gluing separate parts together, such as incorporating the connectors within the printed jig shape.

Clearly, the manual assembly process employed here could be automated using a pick-and-place robotic arm. Picking a die on its edge or corner might require some effort, possibly using an air suction end effector of the right shape to hold the die on an (opposite) corner or edge. Additionally, one must be able to identify the appropriate (digit) orientation of the die that is to be picked and positioned.

Acknowledgments

We would like to thank Dvir Biton and Foad Abu Dahood for helping in developing the dithering algorithm. Thanks to Mohammed Badrie and our families for assistance in the assembly of the 3-image and 2-image actual models.

References

[1] R. Bosch. "Constructing Domino Portraits." *Tribute to a Mathemagician*, 2004. pp. 251-256.

[2] R. (Bob) Bosch. (2021, May 20). "Accordion-folded-paper version of my lenticular dice mosaic, Monster Mash." [Tweet]. *Twitter*. https://twitter.com/baabbaash/status/1395105257768824843.

[3] H. Cambazard, J. Horan, E. O'Mahony, and B. O'Sullivan. "A Hybrid Approach to Domino Portrait Generation." In *AAAI*. 2008. pp. 1874-1875.

[4] H. Cambazard, J. Horan, E. OMahony, and B.O'Sullivan. "Domino Portrait Generation: a Fast and Scalable Approach." *Annals of Operations Research*. vol 184, no. 1, 2011, pp 79.

[5] V. Dan., G. Harja. "Art Rendered using Rubik's Cubes." *ARRC*, 256.2, 2020. pp 42–47.

[6] Dice Ideas, 2023, https://www.diceideas.com.

[7] Diceify, 2023, http://www.diceify.art.

[8] G. Elber. “3D-Dithered Ortho-Pictures: 3D Models from Independent 2D Images.” *Bridges Conference Proceedings*, Baltimore, Maryland, USA, Jul. 29 –Aug. 1, 2015, pp 207-214.

[9] G. Elber. “Simultaneous 3D Dithering of Multiple Images by Curves.” *Computer & Graphics Journal*, vol 105, August 2022, pp 146-152. Also Shape Modeling International, online event, June 2022.

[10] R.W. Floyd. “An Adaptive Algorithm for Spatial Gray-scale.” In *Proc*. Soc. Inf. Disp., vol. 17, 1976. pp. 75–77.

[11] G4G Celebration. Gathering4Gardner Apr 2022 [Video]. *YouTube*. https://www.youtube.com/watch?v=GDredFSzPNU&ab_channel=G4GCelebration.

[12] F. Habib, S. Asman, G. Elber. “Dithering by Wires of Orthogonal Images.” *The Hyperseeing magazine*, Fall 2021, pp 41-54. Also Shape Modeling International - FASE, online event, November 2021.

[13] K. Knowlton. Mosaics, 2023, https://www.knowltonmosaics.com/pages/AEdice.htm.

[14] D. Plaxco. “Photogenic Knot Projections on $n \times n \times n$ Rubik’s Cubes.” *Bridges Conference Proceedings*, 2022. pp 331–334.

[15] H.. Tanasra, 3D Dice Mosaics: A Multidirectional Dithering System April 2023 [Video]. *YouTube*. https://www.youtube.com/watch?v=dbNHG7Bqlrs&t=10s&ab_channel=HananTanasra.

[16] Technion - Center for Graphics and Geometric Computing. DitherDiceCube Jan 2023[Video]. *YouTube*. https://www.youtube.com/watch?v=NovFcADLn40&ab_channel=Technion-CenterforGraphicsandGeometricComputing.

[17] K. Weichel. *Dice mosaic generator*, 2013, https://kweic.github.io/diceImage.