Computational Aesthetics in Graphics, Visualization, and Imaging (2011)
D. Cunningham and T. Isenberg (Editors)

Generating Op Art Lines

Tiffany C. Inglis and Craig S. Kaplan

University of Waterloo, Canada

Abstract

A common technique in Op Art is the use of parallel lines to depict simple shapes such as circles and squares. Some artists have attempted to create more complex images using this technique but faced the problem of producing undesirable artifacts such as line breaks and T-junctions within their artworks. To this end, we developed a novel algorithm that takes an arbitrary image and automatically generates the corresponding Op Art composition of this style. For 2-colour images, the algorithm produces artworks without any unwanted artifacts; for images with more colours, the basic algorithm cannot guarantee the removal of all artifacts, but we use a global optimization technique to minimize the number of artifacts. The results have applications in graphics design, data visualization, puzzle creation and line drawings.

1. Introduction

Op Art, short for Optical Art, is a style of abstract art associated with optical illusions that gained popularity in 1965 with an exhibition called *The Responsive Eye*. Typically, Op artists make use of simple geometric shapes and highly contrasting colours to create artworks that "trick the eye", for example by revealing hidden images or inducing a sense of movement. Many well-known Op Art works such as Bridget Riley's *Descending* (1965) and Victor Vasarely's *Zebra* (1944) are done in black and white for maximum contrast, and rely on a series of non-intersecting lines or curves to convey forms and shapes. A canonical example of this style is a piece entitled *Square of Two* (1965) by Reginald H. Neal (see Figure 1, top layer). At first glance, the image shows a 2×2 tiling of concentric squares that seems to scintillate as the eye moves across the image. A closer look reveals that the entire image is constructed out of lines in two orthogonal directions. The interior of each concentric square is filled with parallel lines. By converting each line direction into a colour, an underlying image containing two colours is revealed (see Figure 1, bottom layer). Notice that adjacent regions containing lines of different directions are strategically aligned so that endpoints of lines meet orthogonally. Together, these line bends create the illusion of a boundary without actually drawing it. This is known as an *illusory contour* [PM87]. Most of the region boundaries in *Square of Two* are composed of line bends, although we occasionally encounter line breaks, T-junctions and crosses. In this case, they only occur on vertices of concentric squares, and are arranged symmetrically within the artwork, suggesting the artist had made a conscious decision to avoid these topological artifacts.

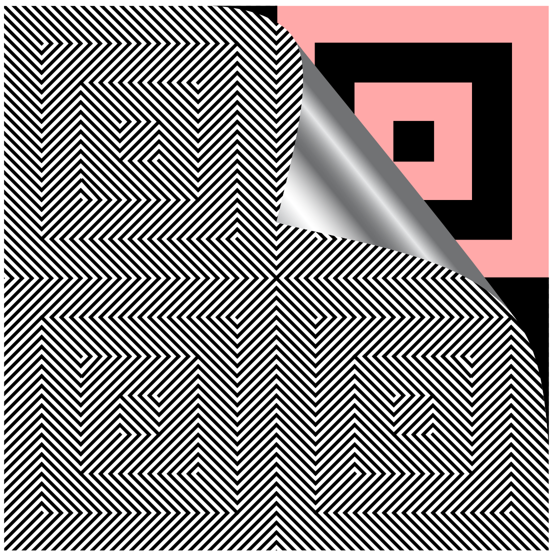


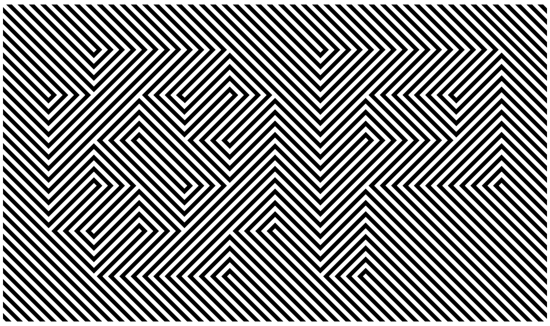
Figure 1: A vector graphic reconstruction of Reginald Neal's *Square of Two* (top layer), the main source of inspiration for our research. The two families of lines depict concentric square regions (bottom layer) through illusory contours created by the line bends.

CAe 2011, Vancouver, British Columbia, Canada, August 5 – 7, 2011.
© 2011 ACM 978-1-4503-0908-0/11/0008 $10.00

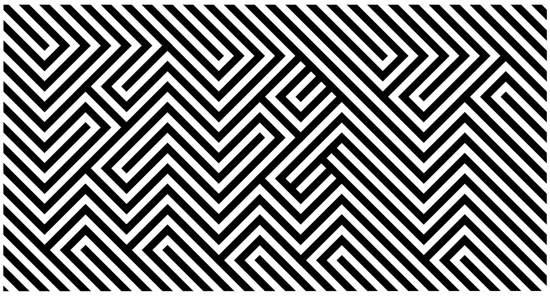
DOI: 10.2312/COMPAESTH/COMPAESTH11/025-032

1.1. Previous work

This style of Op Art—we shall refer to this as Neal-style Op Art—has been explored by other artists. House42 created an Op Art font with a block-like structure called Dioptical (see Figure 2a); Graphic designer Manolo Guerrero designed a similar font called OPTICA Normal (see Figure 2b) in this style. Another example containing more complex shapes is an image of a skull created by Swiss graphic designers Sébastien Vigne and Jolien Notter for the cover of the book *L'expo qui rend fou / H. P. Lovecraft et le livre de raison* (see Figure 3).



(a) Dioptical, a font designed by House42.



(b) OPTICA Normal, a font designed by Manolo Guerrero.

Figure 2: Vector graphic reconstructions of two fonts inspired by Op Art.

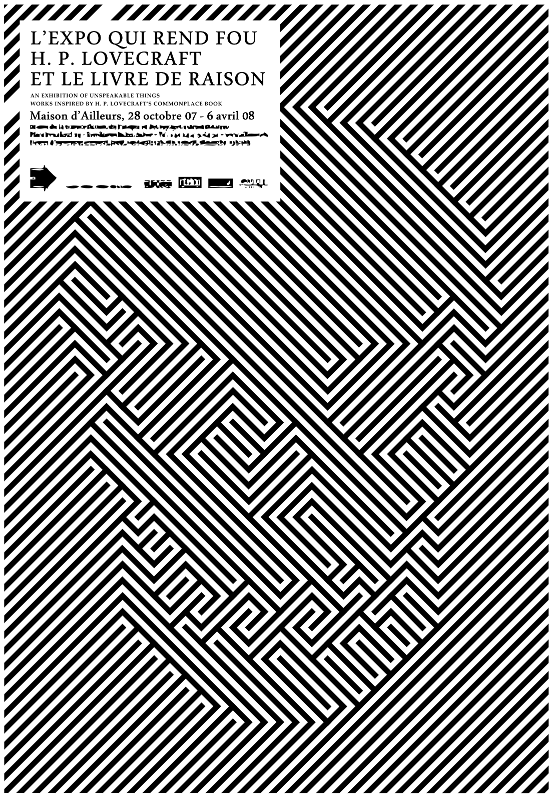


Figure 3: A vector graphic reconstruction of the cover art for *L'expo qui rend fou / H. P. Lovecraft et le livre de raison.*

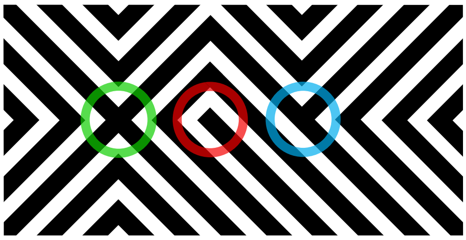


Figure 4: Examples of unwanted artifacts: a line break (circled in red), a T-junction (circled in blue), and a cross (circled in green).

Most of these artworks can be constructed using a straightforward approach: take a 2-colour image, replace each region with parallel lines, then adjust the alignment so that the endpoints of these lines meet orthogonally on the boundaries. This method can produce line breaks, T-junctions and crosses (see Figure 4) along region boundaries. These artifacts are undesirable because they interfere with the illusory contours created by the line bends.

The designers of the Dioptical font used an alternate construction method involving tiling to avoid the artifact problem. They created two square tiles—each filled with parallel lines in one of two orthogonal directions—such that when two tiles meet along an edge, the lines join seamlessly to form an illusory contour without artifacts. Since a square tile is the smallest unit, this method produces a block-like font, and greatly limits the freedom to express curved boundaries.

Whatever construction method is used, there is an inherent trade-off between expressivity of curves and the reduction of artifacts. Our goal is to develop a general method for creating Neal-style Op Art with better curve expression and fewer artifacts than existing artworks. As far as we know, there has been no previous work done in this area, although there has been some recent interest in Op Art in the computer graphics community, such as Dodgson's mathematical analysis of Bridget Riley's works [Dod08].

1.2. Problem statement

Consider a planar map [dBvKOS00] coloured in such a way that neighbouring regions have different colours. This map represents the underlying image for an Op Art composition we want to produce; we ignore the unbounded face of the map since it corresponds to the exterior of the image. The number of colours in this map determines what the output Op Art composition will look like. In this paper, we will use the *k-colour problem* to refer to the problem of constructing a corresponding Op Art composition from a k-colour map. From the Four Colour Theorem, we know that no more than four colours are necessary to colour any planar map. Therefore, it suffices to develop k-colour Op Art algorithms for $k = 2, 3, 4$.

In the two sections that follow, we will describe our algorithms for converting any 2- or 3-colour input map into an Op Art composition. Each algorithm starts by assigning a line direction to each colour then filling in each region with parallel lines corresponding to the region colour. This process leaves behind a large number of line break artifacts along region boundaries, and are dealt with differently for each value of k. For $k = 2$, we offer an artifact-free algorithm that creates Op Art compositions with two orthogonal families of lines. For $k = 3$, there are no artifact-free solutions in general, but we present an optimization-based framework that attempts to minimize artifacts. We propose a 4-colour algorithm similar to the 3-colour algorithm, but since there are further complications, it is left for future work.

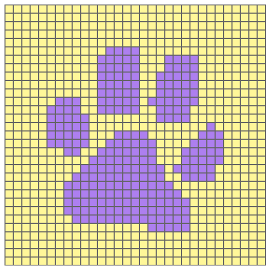
2. The 2-colour algorithm

The 2-colour problem has been tackled in a number of ways by Op artists and graphic designers. The works we have seen so far use diagonal orthogonal lines. For simplicity, we will describe our approach using just horizontal and vertical lines, and later generalize to arbitrary line directions. The following algorithm produces the corresponding Op Art composition from an arbitrary 2-colour input map:

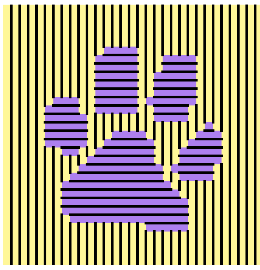
1. Specify the inputs: a 2-colour planar map (see Figure 5a) and the desired line spacing s for the Op Art output.
2. Construct a grid of squares, each of size $s \times s$. Rasterize the map to the grid so that each square gets one colour (see Figure 5b).
3. Assign a line direction—either horizontal or vertical—to each colour. Then for the interior of each region, draw all the grid edges corresponding to the direction of the region colour (see Figure 5c).
4. Draw alternate edges along region boundaries (see Figure 5d).
5. Render all the lines generated above with a thickness of $s/2$ (see Figure 5e) to maximize the overall contrast.

The above algorithm produces a basic Op Art composition containing horizontal and vertical lines. To obtain a more interesting result, we wish to apply some transformation T

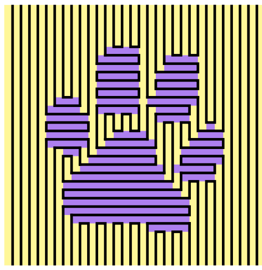
(a) A 2-colour input map.



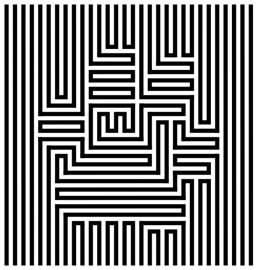
(b) Step 1: Rasterize the map to a grid of squares.



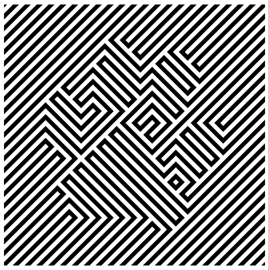
(c) Step 2: Fill region interiors with lines corresponding to the region colours.



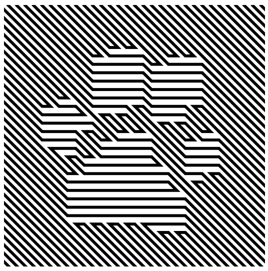
(d) Step 4: Draw alternate edges along region boundaries.



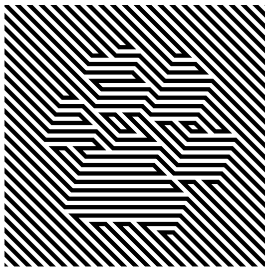
(e) Step 5: Render lines with the appropriate thickness.



(f) Output with rotated lines.



(g) Output with sheared lines that have unequal line spacings.



(h) Output with sheared lines that have equal line spacings.

Figure 5: An example illustrating the 2-colour algorithm for creating Neal-style Op Art, and some transformations that can be applied to the lines.

to the lines while keeping the underlying image fixed. To do this, we apply the inverse transformation T^{-1} to the input map, feed the transformed input into the above algorithm, then apply T to the output. For example, to rotate the lines by θ, we rotate the input by $-\theta$, apply the algorithm to it, then rotate the output back by θ. The image shown in Figure 5f is created with a $\pi/4$ rotation. We can also create an Op Art composition with non-orthogonal lines by applying a shear transformation. Figure 5g shows an example created with a $\pi/4$ horizontal shear containing unequal line spacings in the two directions; this may or may not be desirable, depending on whether or not we wish to convey tone. If we wish to obtain equal line spacing in both directions, an additional scaling factor is required. Figure 5h shows an example created with the transformation T as the composition of a $\pi/4$ horizontal shear and a horizontal scaling of $\sqrt{2}$. The resulting Op Art composition has equal line spacing in both directions. In general, T can be any transformation—even a nonlinear one—as long as it is invertible.

2.1. Proof of correctness

The Op Art composition produced by this algorithm (see Figure 5e) enjoys two desirable properties:

1. It does not contain any artifacts, except possibly line breaks on the image border.
2. The lines are densely packed on the grid (i.e. each grid vertex not on the image border has a degree of at least 1).

These properties are consequences of a more general property that *each grid vertex not on the image border has degree 2*. The easiest way to prove this statement is to enumerate all possible configurations of a grid vertex and show that it is always adjacent to exactly two edges. Clearly, a vertex surrounded by four squares of the same colour has degree 2 due to Step 2 of the algorithm. For a vertex surrounded by squares of different colours, Figure 6 shows the three possible configurations up to symmetry and swapping of colours. Without loss of generality, assign vertical lines to the yellow region and horizontal lines to the purple region. Figure 6 shows how Step 3 and Step 4 of the algorithm applies to each vertex configuration. Each of the three configurations splits into two possible cases at Step 4, because there are two ways to draw alternate boundary edges. In all six cases, the central vertex is adjacent to exactly two edges after Step 4, Q.E.D.

3. The 3-colour algorithm

In the 2-colour algorithm, we rasterize the input map onto a square grid, fill its region interiors with lines, then draw alternate boundary edges to produce an artifact-free Op Art composition. Let us try extending this algorithm to the 3-colour case. First we need to choose three line directions, one for each colour. In the 2-colour algorithm, we used a square grid composed of orthogonal lines to ensure that all the line bends have the same angle. By the same reasoning,

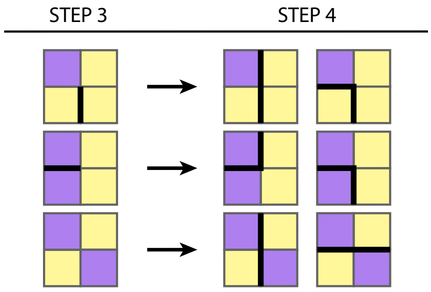
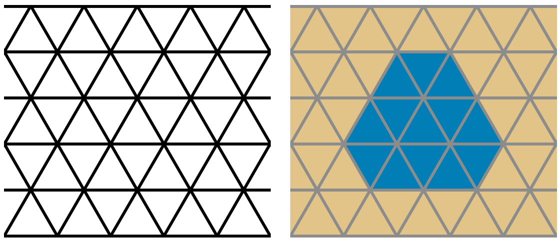


Figure 6: For vertices not on the image border, there are three possible configurations (up to symmetry and colour swap) at Step 3 of the algorithm. After performing Step 4, there are six possible configurations, each with a central vertex of degree 2.



(a) The triangular grid used for creating Op Art from a 3-colour input image.

(b) An image that cannot be made into an Op Art composition without artifacts.

Figure 7: Op Art compositions corresponding to 3-colour images can be created on a triangular grid, but artifacts cannot always be eliminated, as in the case of the 2-colour problem.

the 3-colour algorithm should use an equilateral triangular grid composed of lines separated by angles of $\pi/3$ (see Figure 7a). With this setup, we propose a 3-colour algorithm that rasterizes the input map to a triangular grid, and then proceeds as outlined in the 2-colour algorithm to produce an Op Art composition using three line directions.

From testing this algorithm, we quickly found that, unlike the 2-colour case, it does not eliminate all the artifacts in the resulting Op Art composition. We could try a different pairing of region colours to line directions, but there is no guarantee that we can produce an artifact-free composition. Figure 7b shows an example of a planar map that does not correspond to any artifact-free compositions, regardless of the colour-direction pairing chosen.

3.1. Optimization approach

Since it is impossible to find an artifact-free solution to a general 3-colour problem, we re-pose it as an optimization problem. As before, we rasterize the 3-colour input map on a triangular grid and fill the region interiors according to some colour-direction pairing. It remains then to find the optimal configuration of boundary edges that minimizes the number of artifacts. In each configuration, a boundary edge is either drawn or not, resulting in 2^N possible configurations, where N is the number of boundary edges. Obviously this is too big a configuration space to search through by brute force; we tried solving the associated integer programming problem but the computational cost is not feasible for any reasonably sized input.

Considering the size of the configuration space, we propose an algorithm based on simulated annealing [PFTV92]. We search over a space of possible boundary edge configuration for the optimal configuration associated with the fewest artifacts. A cost function defined in terms of the number of artifacts is used to compare the quality of one configuration to another. Algorithm 1 outlines the steps in the annealing process:

```
v ← InitializeConfiguration()
T ← InitializeTemperature()
while not Terminate() do
    v′ ← GetRandomNeighbour(v)
    if Cost(v′) < Cost(v) then
        v ← v′
        v_opt ← UpdateOptimum()
    else
        p ← AcceptPr(v,v′,T)
        if Random() < p then
            v ← v′
        end
    end
    T ← UpdateTemperature()
end
```

Algorithm 1: Simulated annealing

Let v denote the current configuration. We initialize v as a set of random binary choices for the boundary edges (since each edge is either drawn or not). A neighbour of v is defined as any configuration that differs from v by one edge. Figure 8 shows a configuration surrounded by its six neighbours. Define the cost of v as

$$\texttt{Cost}(v) \;=\; \sum_{i\in\{1,3,4,5,6\}} w_i N_i, \tag{1}$$

where N_i is the number of degree-i vertices along the region boundaries and w_i is the weight or the penalty imposed on each additional degree-i vertex. We experimented with

$$(w_1, w_3, w_4, w_5, w_6) \;=\; (1,1,1,1,1), \tag{2}$$

which penalizes all the artifacts equally, and

$$(w_1, w_3, w_4, w_5, w_6) \;=\; (1,\infty,\infty,\infty,\infty), \tag{3}$$

which allows only line break artifacts. The results obtained using these two cost functions will be compared and analyzed in Section 3.2.

The acceptance probability is defined as

$$\texttt{AcceptPr}(v,v',T) \;=\; \exp\left(\frac{\texttt{Cost}(v)-\texttt{Cost}(v')}{T}\right) \tag{4}$$

where T is the temperature function. To determine T, first we do an initial survey by taking random samples within the configuration space and calculating the cost for each sample configuration. Compute the standard deviation σ of these costs and set the initial temperature $T_0 = \sigma$. Then for subsequent temperatures, we use a simple decrement rule

$$T_{k+1} \;=\; 0.95T_k \tag{5}$$

from an exponential cooling scheme. At each temperature T_k, we perform a number of trials and encounter one of two scenarios: if η new configurations (counting those with both higher and lower costs) have been accepted at $T = T_k$, then decrease the temperature to T_{k+1}; if L trials are reached at $T = T_k$ with fewer than η acceptances, then terminate the search. L should be a function of the problem size, which is the number of boundary edges N. So we define L as

$$L \;=\; \max(5, 0.2N), \tag{6}$$

and set

$$\eta \;=\; 0.6L. \tag{7}$$

With three colours and three line directions, there are a total of six colour-direction pairings. We run the algorithm for each pairing to get six resulting configurations, and take the most optimal configuration as the Op Art output.

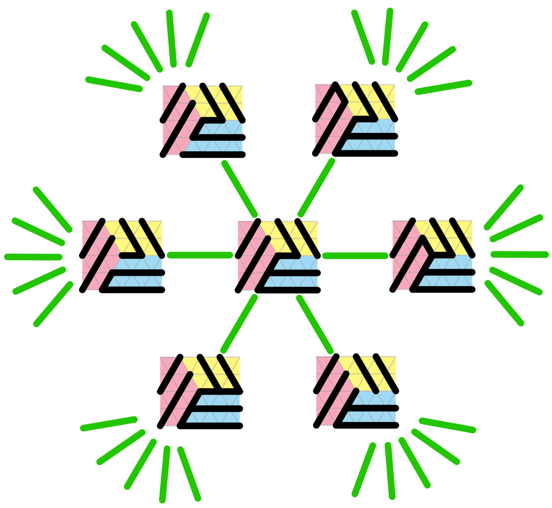
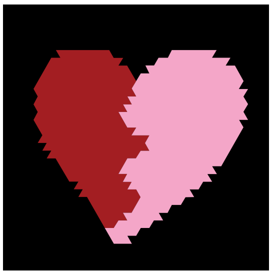


Figure 8: A subset of the configuration space is shown here for a small 3-colour input map. Since there are only six boundary edges (i.e. grid edges between two differently coloured regions), each configuration has exactly six neighbours.

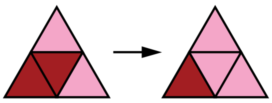
3.2. Suggested improvements

One problem with using a triangular grid is that the rasterized map often contains jagged boundaries with acute angles (see Figure 9b) even if the original input consists of only smooth boundaries (see Figure 9a). We fix this problem by applying a morphological smoothing operation to remove all acute angles that occur on a boundary between exactly two regions. This type of acute angle occurs when a triangle has two of its edges on a region boundary. In other words, it is adjacent to two other triangles with a different colour and one triangle of the same colour, as shown by the left image in Figure 9c. To remove the acute angle, we perform the operation shown in Figure 9c that recolours the central triangle to match the two neighbours with a different colour. This operation removes the acute angle with minimal shifting of the region boundary. We apply this operation to every triangle adjacent to one region boundary to produce the result shown in Figure 9d.

(a) The input map contains smooth region boundaries.



(b) The rasterized map has jagged boundaries that are not representative of the original boundaries.



(c) A smoothing operation is applied to all triangles along region boundaries to remove acute angles.

(d) The rasterized map after the smoothing operation is applied to it.

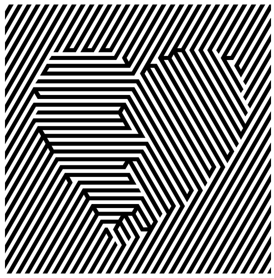
Figure 9: Resampling an image on a triangular grid creates jagged boundaries. This is fixed by applying a smoothing operation to the boundaries.

Earlier, we defined the cost function to be

$$\texttt{Cost}(v) = \sum_{i \in \{1,3,4,5,6\}} w_i N_i, \qquad (8)$$

where N_i is the number of degree-i boundary vertices and w_i is the associated weight. By letting all $w_i = 1$, the algorithm attempts to minimize the total number of artifacts, but does not favour one type of artifacts over another. If we set $w_1 = 1$ and $w_i = \infty$ for $i = 3,4,5,6$, the only artifacts allowed would be the line breaks. Figure 10 shows the difference between the two types of outputs produced. We prefer the composition in Figure 10a because the region boundaries appear more continuous with only line breaks and no other artifacts.

(a) Cost function parameters: $w_1 = 1$ and $w_i = \infty$ for $i = 3,4,5,6$.



(b) Cost function parameters: $w_i = 1$ for $i = 1,3,4,5,6$.

Figure 10: Op Art outputs generated from simulated annealing with two different cost functions. The region boundaries appear more continuous if we allow only line break artifacts.

4. The 4-colour algorithm

For the 4-colour problem, we propose a method similar to the 3-colour algorithm, but instead of an equilateral triangular grid, we use a grid that consists of parallel lines in four different directions separated by angles of $\pi/4$ (see Figure 11). Compared to the 2- and 3-colour grids, this grid is unique in that it contains both degree-2 and degree-8 vertices. In addition, the spacing between diagonal lines is smaller than that of the horizontal and vertical lines, which means the regions corresponding to the diagonal lines will look darker than other regions in the Op Art output. The simulated annealing approach will still work on this grid, but we leave for future work the problem of generating 4-colour Op Art composition with evenly spaced lines in all directions.

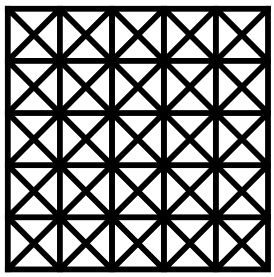


Figure 11: The triangular grid used for the 4-colour problem.

5. Practical issues

All the algorithms presented so far take a k-colour planar map as input, where $k = 2, 3, 4$. In practice, we want to use an arbitrary image with any number of colours as input to create an Op Art composition. Here we outline several ways to transform an image into a k-colour map.

For a grayscale image, thresholding can be used to create a black-and-white image that corresponds to a 2-colour map. A map produced this way usually contains small regions and jagged boundaries, which makes an unsuitable input for the Op Art algorithm. A better technique to process an image is *artistic thresholding* [XK08], which creates a black-and-white image with segments that depict forms better. For a coloured image, posterization is a simple processing technique that guarantees a specified number of colours in the output, although it suffers from many of the same problems encountered with thresholding. In general, we avoid working with photograph-like images because it is difficult to transform them into workable planar maps with large regions and smooth boundaries, while staying faithful to the original images. Instead, images that are more abstract and cartoon-like make much more suitable inputs.

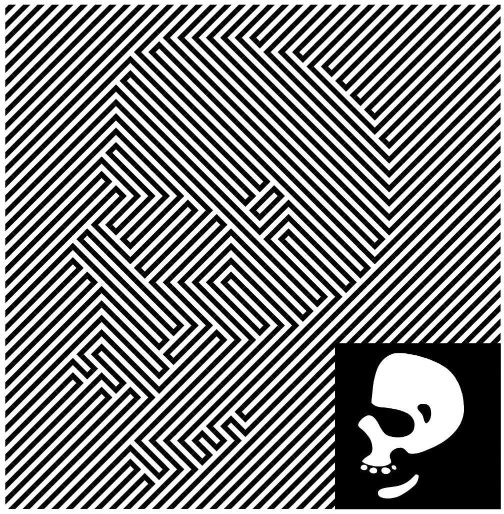
Even among images with similar levels of detail, some translate better into Op Art than others. Through experimentation, we found that images whose regions depict tonal variation generally lead to less recognizable outputs (see Figure 12a), because Op Art compositions do not preserve tonal information. In contrast, images with regions that convey shapes generate more easily recognizable Op Art outputs (see Figure 12a) since Op Art regions encode shape information well. These results are consistent with Xu's findings [Xu09].

6. Results and future work

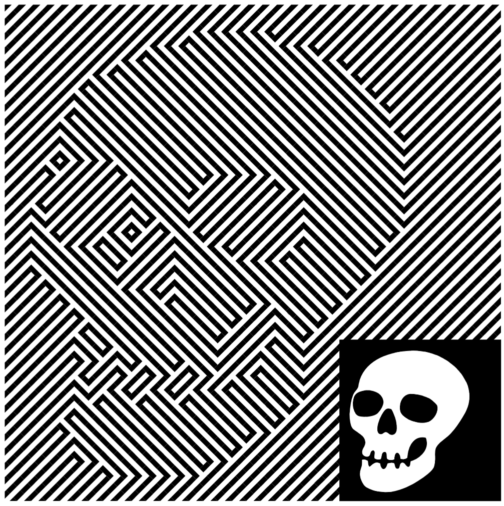
To demonstrate the results produced by our 2- and 3-colour Op Art algorithms, we have chosen the lizard tilings from M. C. Escher's *Metamorphosis II* (see Figure 13) as inspirations for the Op Art compositions in Figure 14a and Figure 14b.

When viewed from a distance, the 2-colour Op Art composition appears more even in tone than the 3-colour one. The latter contains some artifacts along the region boundaries which give the boundaries a slightly broken look. Despite the presence of the artifacts, we can still see the illusory contours that outline the lizards very clearly. In both compositions, the region boundaries are not as smooth as those in the input images. The problem can be easily fixed by increasing the resolution of the underlying grids. Unfortunately, due to the space restriction, we are not able to showcase larger Op Art compositions in this paper.

For future work, we would like to explore ways to improve our 3-colour algorithm. One way would be to add to the set of moves allowed in the configuration space search.



(a) The input image (inset) depicts the tones of a white skull on a black background; this translates into a less recognizable Op Art composition because the tonal variation is not preserved.

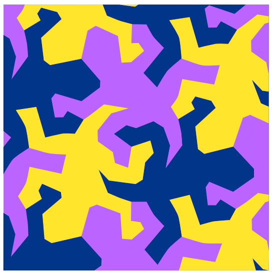


(b) The input image (inset) conveys shapes in the skull image; this translates into a more recognizable Op Art composition because the Op Art regions retain information about shapes.

Figure 12: Op Art compositions created from different types of input images.

Instead of only moving to neighbours that differ by one edge, we could move within a larger neighbourhood and possibly search the space more efficiently. For a given 3-colour input map, we could consider other colourings that are not simply permutations of the given colouring. The search space would be much larger but we can potentially discover configurations with much fewer artifacts. As for the 4-colour problem, we want to develop a 4-colour algorithm that al-

(a) A 2-colour lizard tiling.

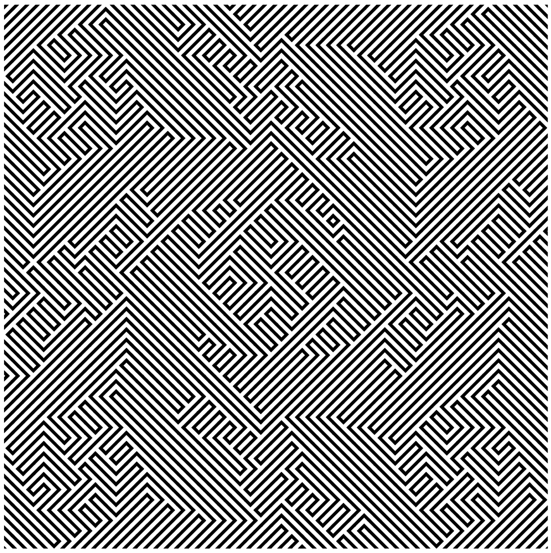


(b) A 3-colour lizard tiling.

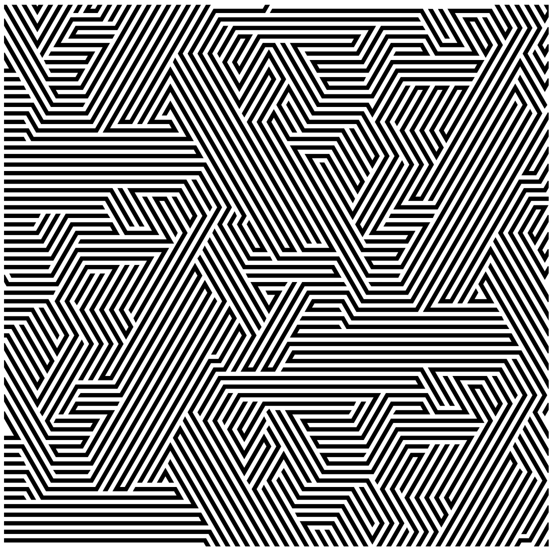
Figure 13: Lizard tilings, inspired by Escher's *Metamorphosis II*, are used to generate the compositions in Figure 14.

lows equal line spacings in all four line directions. In general, we would to experiment with different line spacings, varying line thicknesses, and even curves in order to convey the tones and forms in the underlying image.

Our algorithms have potential applications in a number of areas. In data visualization, if we wish to plot several datasets simultaneously in a non-preferential way (using colours or shades of gray may introduce an implicit hierarchy which could be undesirable for, say, designing political maps) or without clearly defined boundaries, then using our algorithm to depict regions with line patterns would be appropriate. To reduce the headache-inducing Op Art effect, decrease the line thickness so that the black-and-white contrast is weaker. Our algorithm could also be used in recreational computational geometry, such as creating mazes [XK07] or designing logic-based puzzles containing underlying images. In particular, we would like to develop a method for generating Slitherlink puzzles so that a completed puzzle resembles a Neal-style Op Art composition drawn with one single loop, similar to TSP Art [KB05]. Our research also opens up the possibility of incorporating illusory contours in general illustration. Since illusory contours convey shape without disrupting tone, they may be incorporated into hatching algorithms for line drawings.



(a) 2-colour Op Art output.



(b) 3-colour Op Art output.

Figure 14: The Op Art compositions created based on the 2- and 3-colour lizard tilings in Figure 13.

References

[dBvKOS00] DE BERG M., VAN KREVELD M., OVERMARS M., SCHWARZKOPF O.: *Computational Geometry: Algorithms and Applications*, second ed. Springer-Verlag, 2000. 3

[Dod08] DODGSON N. A.: Regularity and randomness in Bridget Riley's early Op Art. In *Proceedings of Eurographics Workshop on Computational Aesthetics in Graphics, Visualization, and Imaging* (2008), pp. 107–114. 2

[KB05] KAPLAN C. S., BOSCH R.: TSP Art. In *Renaissance Banff: Bridges 2005: Mathematical Connections in Art, Music and Science* (2005), pp. 301–308. 8

[PFTV92] PRESS W. H., FLANNERY B. P., TEUKOLSKY S. A., VETTERLING W. T.: Simulated annealing methods. In *Numerical Recipes in C: The Art of Scientific Computing*, second ed. Cambridge University Press, 1992, ch. 10.9, pp. 444–455. 5

[PM87] PETRY S., MEYER G. E.: *The Perception of Illusory Contours*. Springer, 1987. 1

[XK07] XU J., KAPLAN C. S.: Image-guided maze construction. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers* (2007). 8

[XK08] XU J., KAPLAN C. S.: Artistic thresholding. In *NPAR '08 Proceedings of the 6th international symposium on Non-photorealistic animation and rendering* (2008). 7

[Xu09] XU J.: *Wholetoning: synthesizing abstract black-and-white illustrations*. PhD thesis, University of Waterloo, 2009. Section 1.2.2. 7