Constructing Drawings of Impossible Figures with Axonometric Blocks and Pseudo-3D Manipulations

Tiffany C. Inglis
David R. Cheriton School of Computer Science
University of Waterloo
piffany@gmail.com

Abstract

Impossible figures are found in graphical designs, mathematical art, and puzzle games. There are various techniques for constructing these figures both in 2D and 3D, but most involve tricks that are not easily generalizable. We describe a simple framework that uses axonometric blocks for construction and permits pseudo-3D manipulations even though the figure may not have a real 3D counterpart. We use this framework to create impossible figures with complex structures and decorative patterns.

Introduction

Impossible figures [1, 6] are optical illusions involving objects that can be drawn in 2D but are infeasible in 3D (i.e., cannot be physically constructed). Figure 1a shows the impossible cube and the Penrose triangle, both of which have geometric requirements that are impossible to satisfy in 3D, and the blivet, which relies on negative space to create an illusion. Impossible figures are also featured in many of M. C. Escher's lithographs [7], such as the Penrose stairs—an ever-ascending staircase—that is featured in both *Ascending and Descending* and *Waterfall*.

Many of these impossible figures resemble 3D objects that can be constructed out of blocks under parallel projection. It seems natural that one should be able to apply the same block manipulations used to construct real 3D objects to construct these impossible figures. However, since impossible figures have no 3D counterparts, defining a set of pseudo-3D manipulations is non-trivial. We define the set of impossible figures we are interested in, determine the parameters for the parallel projection applied to the basic building block, describe a set of operations for constructing these figures with blocks, resolve ambiguities in edge interpretations, and explore different rendering possibilities.

Infeasibility Conditions

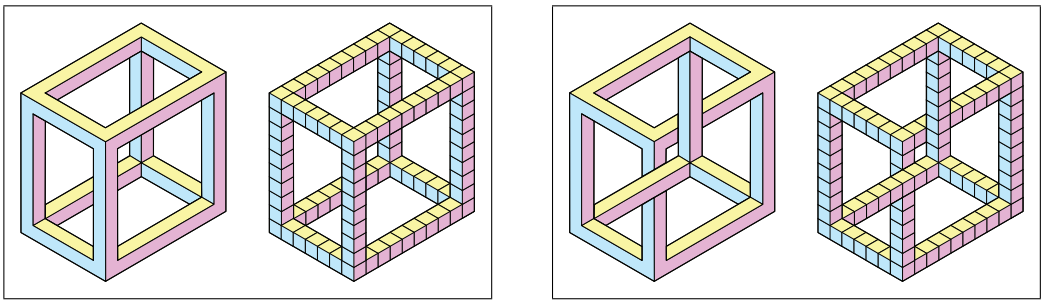
Swedish artist Oscar Reutersvärd is known as "the father of the impossible figure" for inventing many impossible figures, three of which are shown in commemorative stamps in Figure 1b. One of his creations is the Penrose triangle, one of the most basic impossible figures that was independently devised by Roger Penrose. Although it is infeasible

(a) (b)

Figure 1: *(a) Three well-known impossible figures: the impossible cube, the Penrose triangle, and the blivet, and (b) Reutersvärd's impossible figure designs on commemorative stamps.*

(a) (b) (c) (d)

Figure 2: *Penrose triangle constructions in 3D: (a) sculpture by MacKay and Abas, (b) sculpture by Hamaekers, (c) pendant by vgenel, and (d) 3D print by chylld.*



(a) (b)

Figure 3: *Two constructions created using blocks: (a) a regular wireframe cube, and (b) an impossible cube.*

in 3D, there are various clever 3D constructions that look like the Penrose triangle when viewed from a certain angle. Figure 2a shows a sculpture by MacKay and Abas that is disconnected at one vertex. In Figure 2b, a sculpture by Hamaekers uses curved edges to achieve the illusion. Figure 2c shows a pendant designed by vgenel with twisted edges. In Figure 2d, chylld designed a 3D-printed illusion containing partially inverted cubes.

We draw on Reutersvärd's impossible figures as a source of inspiration because many of his designs can be expressed as a set of fundamental building blocks drawn with parallel projection. We will refer to figures that can be constructed using blocks as *constructions*. These constructions include both feasible figures, such as the wireframe cube shown in Figure 3a, and impossible figures, such as the impossible cube shown in Figure 3b. Note that the blivet is not considered a construction at all since it employs a different type of illusion and contains rounded parts. The infeasibility of the impossible cube is related to the way the blocks are arranged. To describe the infeasibility of a construction, we need to first define two properties: (1) *local consistency* and (2) *global consistency*.

Local consistency refers to the non-ambiguity of the pairwise block orderings. This ordering refers to the order in which the block projections are drawn in 2D, and not the order in which the blocks appear to be arranged in 3D. Given two blocks in a construction, they can be ordered in one of four ways: (1) in front, (2) behind, (3) non-interacting, and (4) inconsistent. In Figures 4a and 4b, the darker block is in front of and behind the lighter block, respectively. In Figure 4c, the two blocks are non-interacting since their faces do not overlap. In Figure 4d, the ordering of the two blocks is ambiguous because each block is partially occluding the other. For a construction to be locally consistent, the pairwise ordering of the blocks must be non-ambiguous. Otherwise, the construction is locally inconsistent. Figures 4e and 4f show two locally consistent constructions, and Figure 4g shows a locally inconsistent construction due to several ambiguous block orderings. Kulpa's paper [4] also discusses a classification of impossible figures with various degrees of impossibility.

Global consistency is basically feasibility. A construction is globally consistent if it is possible to assign a 3D position to each block so that the projection gives us the 2D drawing. A necessary condition for global consistency is that the ordering of *all* the blocks (not just pairwise orderings) must be well-defined. In other words, we should be able to apply the painter's algorithm [2] to draw all the blocks sequentially from back to front in the picture plane. The Penrose triangle in Figure 4f, for example, is globally inconsistent because every block is in front of one neighbour and behind another.

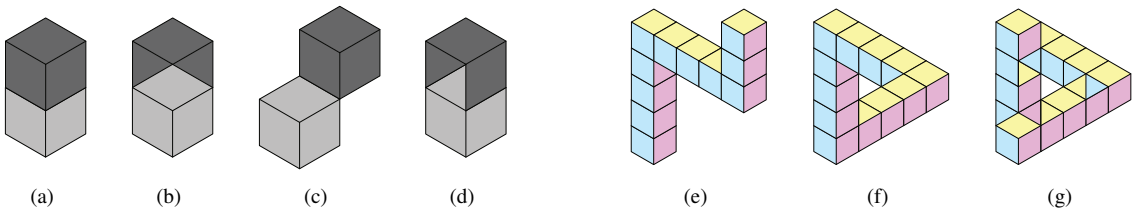


Figure 4: *The darker block is (a) in front of, (b) behind, (c) non-interacting with, and (d) inconsistent with the lighter block. (e) A construction that is both globally and locally consistent, (f) a construction that is globally inconsistent but locally consistent, and (g) a construction that is both globally and locally inconsistent.*

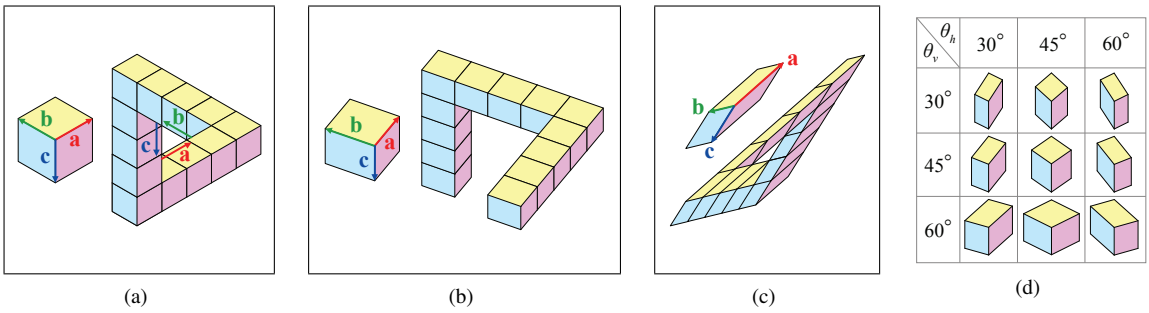


Figure 5: *(a) An isometric block that is suitable for constructing impossible figures, (b) a trimetric block that creates misalignment when constructing a Penrose triangle, and (c) an oblique block that can be used to construct impossible figures but appears skewed. (d) Various axonometric blocks that are suitable for constructing impossible figures.*

We want to construct impossible figures that are locally consistent but globally inconsistent. Both the impossible cube and the Penrose triangle fall in this category. By requiring constructions to be locally consistent, we avoid constructions that are simply a set of faces that are meaningless in 3D. Now that we have defined the limitations of the set of constructions we want to create, the next step is to define the parameters for the building blocks.

Parallel Projection Parameters

We are interested in constructing impossible figures from parallel projected blocks. Axonometric projection is a type of parallel projection where the axis or the plane of the object is not parallel to the projection plane. In particular, this means an axonometric block always has three faces visible, which is useful for communicating depth in a 2D drawing.

Suppose we wish to construct a Penrose triangle with five blocks on each edge. Choosing an arbitrary axonometric block will not work, as the ends do not coincide (see Figure 5b). However, it will work with an isometric projection where all the axes are equally foreshortened, as shown in Figure 5a. Reutersvärd often used the isometric perspective for his impossible figure drawings, as we saw in Figure 1.

Isometric blocks are not the only axonometric blocks we can use for constructing impossible figures. In Figure 5a, we labeled the projected axes of the block as $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ (note that these are 2D vectors). To align the first and last blocks, the condition $\mathbf{a}+\mathbf{b}+\mathbf{c}=\mathbf{0}$ must be satisfied. In addition, we impose a second condition that the block is in fact an axonometric projection of a regular prism, so that it does not look as distorted as the construction shown in Figure 5c.

To satisfy the second condition, suppose we have an $L_1 \times L_2 \times L_3$ rectangular prism. After applying an axonometric projection with a vertical rotation of θ_v and a horizontal rotation of θ_h, we obtain

$$\mathbf{a} = L_1 \begin{pmatrix} -\sin\theta_h \\ \cos\theta_v \cos\theta_h \end{pmatrix}, \quad \mathbf{b} = L_2 \begin{pmatrix} \cos\theta_h \\ \cos\theta_v \sin\theta_h \end{pmatrix}, \quad \mathbf{c} = L_3 \begin{pmatrix} 0 \\ -\sin\theta_v \end{pmatrix}. \tag{1}$$

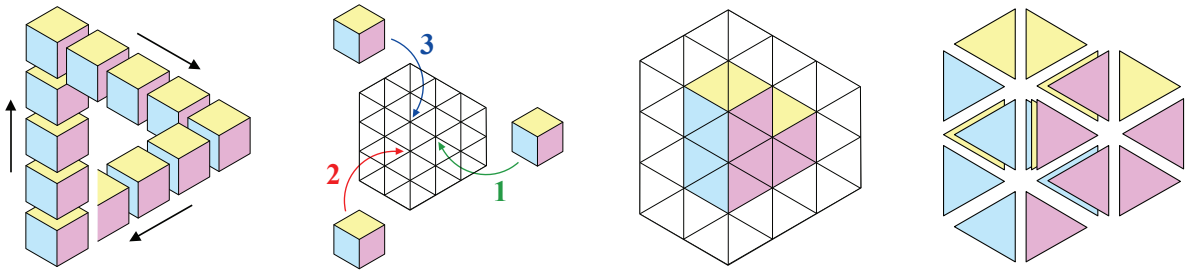


Figure 6: *(a) Constructing a Penrose triangle by stacking blocks in sequential order requires the last block to be cut in half. (b-d) Instead, place blocks on a triangular grid, where each grid space contains up to three layers.*

Assume $\mathbf{c} = (0,-1)^T$. Then the condition $\mathbf{a}+\mathbf{b}+\mathbf{c} = \mathbf{0}$ implies, after dividing both sides by $\cos\theta_v$, that

$$\begin{pmatrix} -\sin\theta_h & \cos\theta_h \\ \cos\theta_h & \sin\theta_h \end{pmatrix} \begin{pmatrix} L_1 \\ L_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 1/\cos\theta_v \end{pmatrix}. \tag{2}$$

Multiplying by the inverse matrix gives us

$$\begin{pmatrix} L_1 \\ L_2 \end{pmatrix} = \frac{1}{-\sin^2\theta_h - \cos^2\theta_h} \begin{pmatrix} \sin\theta_h & -\cos\theta_h \\ -\cos\theta_h & -\sin\theta_h \end{pmatrix} \begin{pmatrix} 0 \\ 1/\cos\theta_v \end{pmatrix} = \begin{pmatrix} \cos\theta_h/\cos\theta_v \\ \sin\theta_h/\cos\theta_v \end{pmatrix}, \tag{3}$$

yielding

$$\mathbf{a} = \sin\theta_h \begin{pmatrix} \cos\theta_h/\cos\theta_v \\ \sin\theta_h \end{pmatrix} \quad \text{and} \quad \mathbf{b} = \cos\theta_h \begin{pmatrix} -\sin\theta_h/\cos\theta_v \\ \cos\theta_h \end{pmatrix}. \tag{4}$$

Now we have an expression for each of the three axonometrically projected axes of the block in terms of the projection parameters θ_v and θ_h, which we can use to adjust the shape of the projected blocks, as shown in Figure 5d. For the rest of the paper, we will use $(\theta_v, \theta_h) = (54.7°, 45°)$, which corresponds to an isometric projection.

Triangular Data Structure

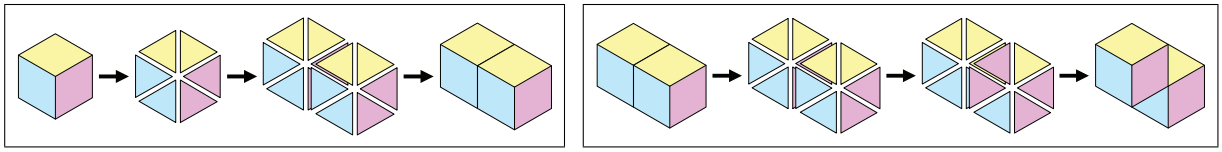
When creating a globally consistent construction, each block can be represented either by its 3D location or by the order in which it is drawn. However, impossible figures are not globally consistent. For example, if we try to draw a Penrose triangle by the stacking blocks in the order as shown in Figure 6a, then the last block must be cut in half to appear as if it lies behind the first cube.

We need an alternative data structure to track the local ordering of the blocks. Observe that any construction created with these blocks can be placed on a triangular grid. For example, if we place the three blocks in Figure 6b on a triangular grid in the order they are labelled, we get the L-shaped construction shown in Figure 6c. Note that each face of a block fits into two triangular grid spaces exactly, and the edges of a block are aligned with the grid axes.

In Figure 6c, only the visible faces are shown, but each triangular grid can actually contain up to three layers. Figure 6d shows the occluded layers, where the block drawn first has all its faces at the bottom, and the block drawn last has all its faces at the top. This layered structure allows us to deal properly with occlusion. For example, if we were to remove the top block, then some of the previously occluded faces of the middle block will be revealed.

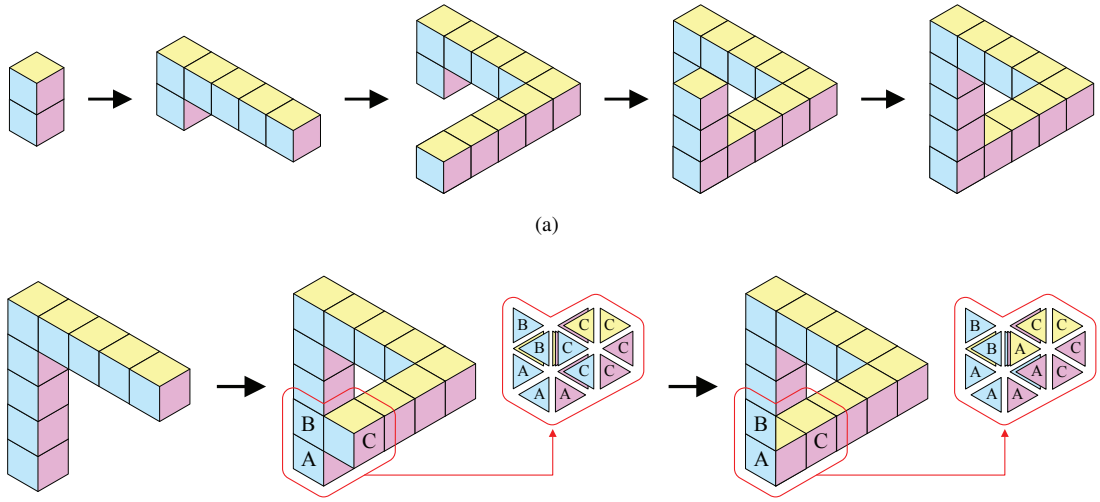
Block Operations

Let us define some terminology for blocks drawn using this triangular data structure. Two blocks are considered *neighbours* if they have overlapping triangular faces. For example, the three blocks in Figure 6d are all pairwise neighbours. If A and B are neighbouring blocks, then exactly two (triangular) faces of A overlap with two of B. Of the overlapped faces, if both faces of A are occluded, then A is behind B. Otherwise, if both faces of B are occluded,



(a) (b)

Figure 7: *Block operations: (a) Place a block in front or behind another, and (b) swap the ordering of two blocks.*



(a)

(b)

Figure 8: *(a) A step-wise construction of a Penrose triangle. (b) Here, swapping A and C creates local inconsistencies.*

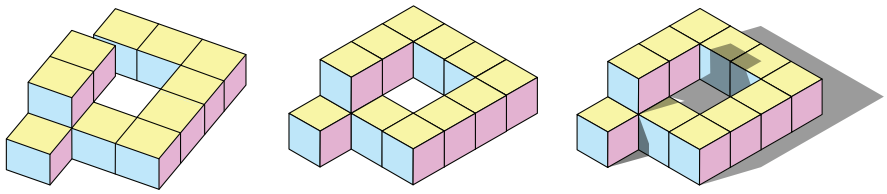
then B is behind A. If A and B each contain one occluded face, then the construction is locally inconsistent. We want to avoid local inconsistency when manipulating the blocks.

We define four pseudo-3D block operations: (1) place block in front, (2) place block behind, (3) remove block, and (4) swap two blocks. To place a block in front (of all other cubes), place its faces above all other faces. To place a block behind, place its faces underneath all other faces. In Figure 7a, the bottom-right block is placed in front of the top-left block. Conversely, the top-left block is placed behind the bottom-right block. To remove a block, simply remove all its faces. Removing a block may reveal some previously occluded faces. To swap two cubes, we swap the order of their overlapping faces. Figure 7b shows two blocks swapped so that the bottom-right cube, originally in front, is now behind the top-left block.

The swapping operation is the key to drawing impossible figures because it changes the local ordering of two blocks. Therefore it is possible to achieve global inconsistency while maintaining local consistency. In Figure 8a, we demonstrate one method of constructing a Penrose triangle. Starting from the middle of the left edge, add blocks in front of the previous block until the last block overlaps the first block, and then swap them.

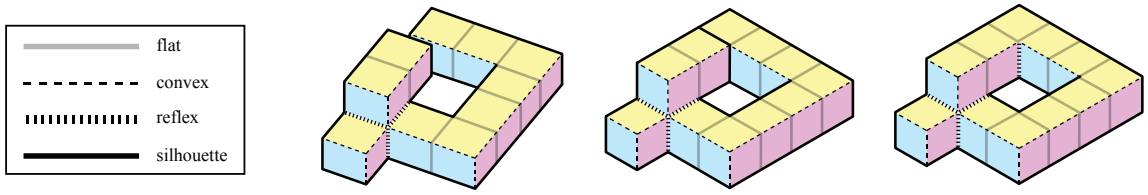
One flaw with the current definition is that the swap operation can introduce local inconsistencies. For example, suppose we construct a Penrose triangle by stacking blocks as before, but starting from a vertex rather than from the middle of an edge (see Figure 8b). Let A, B and C be the first, second and last block drawn, respectively. To create the optical illusion, we swap A and C as before, but now this creates local inconsistencies between the three blocks. The problem is that swapping A and C affects the order of B. Originally, B was in front of A but behind C. When the order of A and C were swapped, it is unclear whether B should remain in front of A or behind C.

To prevent swapping from introducing local inconsistencies, whenever we swap two blocks, we check if the swap would cause any local inconsistencies with the two blocks and their neighbours. If so, we swap back and notify the user that the operation is invalid. It is not always obvious whether two blocks are swappable, but intuitively speaking, two blocks can be swapped if their overlapping regions do not contain faces from other blocks, such as the first and last blocks in Figure 8a.



(a) (b) (c)

Figure 9: *(a, b) A globally feasible construction, when viewed from certain angles, can appear to be an impossible figure. (c) Adding shadows can somewhat alleviate the illusion.*



(a) (b) (c) (d)

Figure 10: *(a, b) For an non-ambiguous interpretation, edges are stylized differently depending on their types. (c, d) With different edge stylizations, the same set of faces can appear to have different geometries.*

Edge Classification

Under parallel projection, objects remain the same size regardless of their distance to the viewer. This property is important for constructing impossible figures since blocks farther away can join nearer ones. However, it also means one construction may have multiple interpretations even though we want to communicate only one specific interpretation.

Suppose we want to convey the structure in Figure 9a. This construction is globally consistent, but when viewed from a different angle (see Figure 9b), it appears to be an impossible figure that is two blocks tall on its left side and only one block tall on its right side [3]. If we want to convey to the viewer that it is *not* an impossible figure, one way to do so is to add shadows, as shown in Figure 9c. But even with shadows, it is not always immediately obvious how the blocks are connected. Moreover, it is not clear what shadows are cast by impossible figures.

A more direct way of showing exactly how the blocks are arranged is to render the edges in different styles depending on the geometry. We distinguish between four different types of edges: flat, convex, reflex, and silhouette (Figure 10a shows the different edge styles). If an edge is between two front faces, then we measure the dihedral angle (i.e., the internal angle between the two faces). We say the edge is *flat* if the angle is 180°, *convex* if 90°, and *reflex* if 270°. Otherwise, if an edge is between one front face and one back face, then it is a *silhouette* edge.

Figure 10b shows the stylized edges applied to non-ambiguous perspective. In Figure 10c, we have the same set of stylized edges applied to the ambiguous perspective to enforce the notion that the blocks do not form a loop. However, if we wish to create an impossible figure out of this construction, then we stylize the edges as shown in Figure 10d so that the blocks appear to form a loop.

To determine the type for an edge, we look at two triangular faces adjacent to it. Figure 11a is a look-up table for which edge types correspond to which pair of adjacent faces (note that the colour of a face corresponds to its orientation, which has been used consistently throughout the paper). Figure 11b shows how to use this table with an example containing three blocks and four different edge types, all labeled accordingly. For the sake of completeness, the look-up table also shows six adjacent face pairs that result in impossible edges. However, we will not encounter this edge type using only the set of operations defined in the previous section.

The adjacent faces are not always enough to determine the edge type exactly. A few adjacent face combinations correspond to two different edge types. Figures 11c and 11d show two different interpretations of the edges between the two blocks. Whenever an edge has two possible interpretations, one of them is always a silhouette edge. Currently our algorithm chooses the non-silhouette option whenever possible, because it correctly draws impossible figures such as the Penrose triangle.

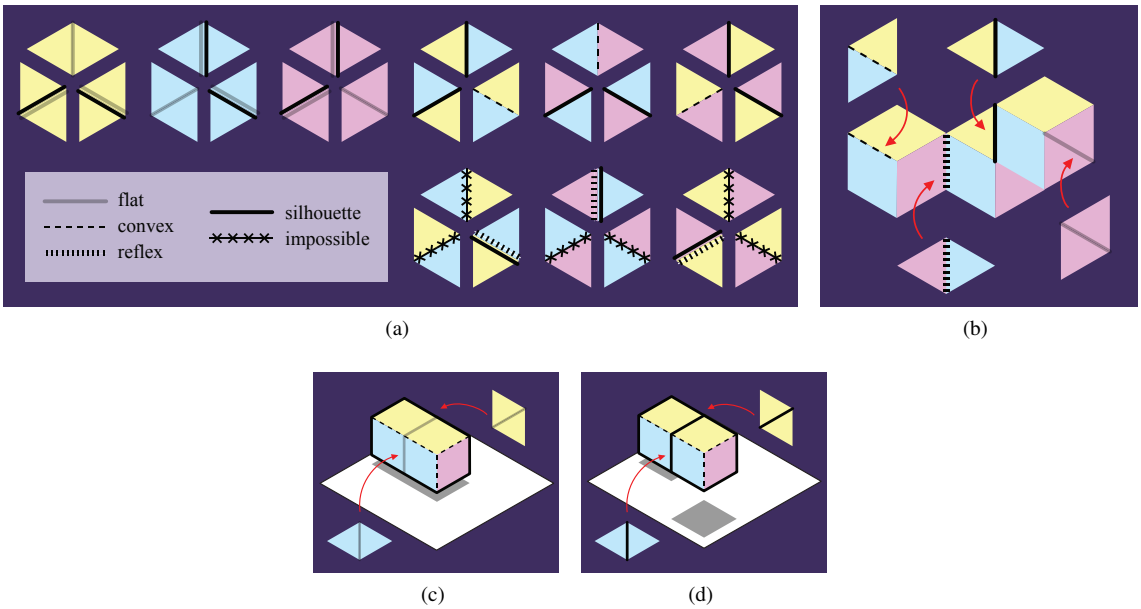


Figure 11: *(a) A look-up table for determining the edge type based on adjacent triangular faces. (b) An example showing how to use the table. (c, d) Some adjacent face combinations correspond to two different edge types.*

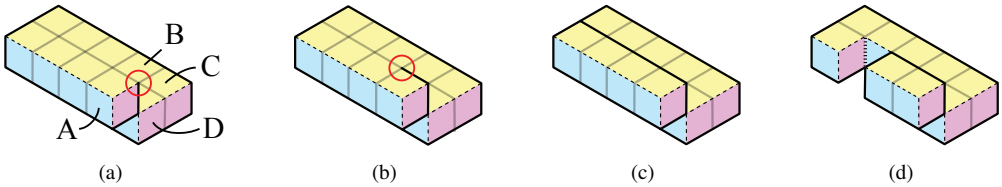


Figure 12: *(a) The current method for edge type detection can create odd-looking vertices. (b) The problem cannot always be fixed locally. (c, d) The solution may affect many edges, and may not be compatible with the block operations.*

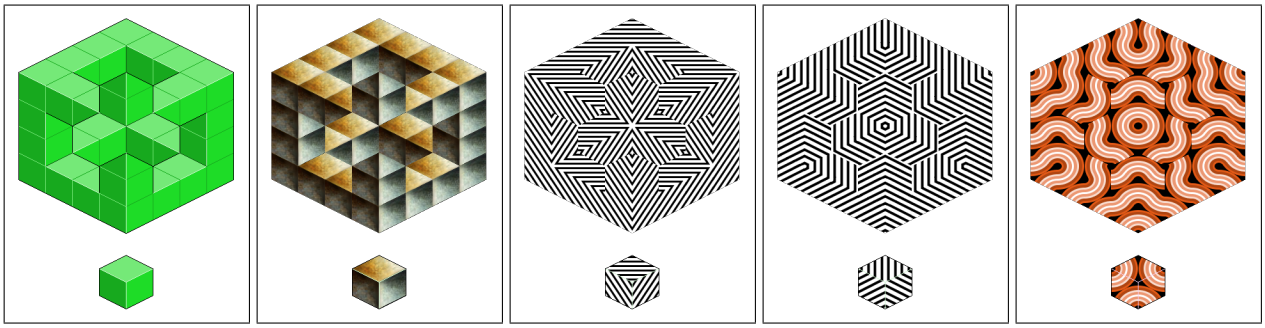
Our algorithm appears to fail in some cases, although this is only an intuitive observation since we have not strictly defined what constitutes failure. Figure 12a shows how our algorithm currently stylizes edges. Notice that the region around the circled vertex looks odd. Let A, B, C and D be the four blocks around the vertex, as labelled. At first glance, A appears to be in front of D because A partially occludes D. However, the flat edges between A and B, B and C, and C and D suggests that all four blocks are co-planar, and therefore D should be in front of A.

One way to fix this disparity is to change an adjacent edge from a flat edge to a silhouette edge, as shown in Figure 12b. However, a similar problem arises around the next vertex: because the four surrounding blocks appear to be co-planar, all four incident edges should be flat. We can continue to fix the problem by converting more edges into silhouette edges, until we get the construction shown in Figure 12c.

Even though the problem appears to be localized around a vertex, this solution potentially affects more than just its incident edges. Now what if a block is removed, as shown in Figure 12d? Should some of the silhouette edges change back to flat edges? Currently we do not have a satisfactory answer, and this problem is left for future work.

Artistic Rendering

In addition to stylizing the edges, we can also stylize the faces for interesting effects. Figure 13a depicts an impossible cube with solid colour faces. We can add some texture to the faces to create a cubist effect, as shown in Figure 13b. Adding straight line patterns to the faces we can create two different Op Art styles, as shown in Figures 13c and 13d. Although the last two stylizations look more abstract, we can still see the block structure because all the convex and



(a) (b) (c) (d) (e)

Figure 13: *(a) Solid colour rendering, (b) cubist rendering, (c, d) Op Art renderings, and (e) bacon rendering.*

reflex edges are depicted as illusory contours created by the line bends, while all the silhouette edges are represented as seams with different patterns on either side. If we remove the line bends by using circular patterns as shown in Figure 13e, then the block structure basically disappears, leaving only an abstract pattern. When designing the pattern on the blocks, we refer to the look-up table in Figure 11a to see how the pattern affects each edge type.

Limitations and Future Work

One limitation of our algorithm is the restriction placed on the swapping operation to avoid introducing local inconsistencies. We want to improve the algorithm's flexibility by making smarter decisions about how to swap blocks instead of just disallowing it. The detection of the edge type is also not perfect since there are ambiguities that are difficult to resolve. As for designing patterns to put on the blocks, currently the designs are created manually to ensure that boundary conditions are satisfied. We would like to create a tool to assist with this tedious process.

For future work, we want to make puzzle games based on these impossible figures. The idea is to let players explore a world filled with impossible figures that they can walk on and alter the structure to achieve some goals. The game play would be similar to that of Echochrome [5], a puzzle game in which players explore 3D structures from different parallel projections. The main difference is that, while Echochrome relies solely on prebuilt 3D structures designed specifically to trigger certain optical illusions, we want to give players the freedom to create their own impossible figures in 2D.

References

[1] Bruno Ernst. *Impossible Worlds: 2 in 1 Adventures with Impossible Objects (Evergreen Series)*. Evergreen, 2006.

[2] J. F. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley, 1990.

[3] B. Gillam. Even a Possible Figure Can Look Impossible! *Perception*, 8(2):229–232, 1979.

[4] Zenon Kulpa. Putting Order in the Impossible. *Perception*, 16(2):201–214, 1987.

[5] S. Owada and J. Fujiki. DynaFusion: a modelling system for interactive impossible objects. In *Proceedings of the 6th International Symposium on Non-photorealistic Animation and Rendering*, pages 65–68. ACM, 2008.

[6] L. S. Penrose and R. Penrose. Impossible Objects: A Special Type of Visual Illusion. *British Journal of Psychology*, 49(1):31–33, Feb. 1958.

[7] Al Seckel. *Masters of Deception: Escher, Dali & the Artists of Optical Illusion*. Sterling Publishing Co., Inc., July 2004.