Bring Anyone To Any Wall

Taiwei Cui[1], Abhijeet Chowdhury[2], Crane He Chen[1]
[1]Khoury College of Computer Science, Silicon Valley
[2]Khoury College of Computer Science, Boston
cui.ta@northeastern.edu, chowdhury.ab@northeastern.edu, h.chen1@northeastern.edu

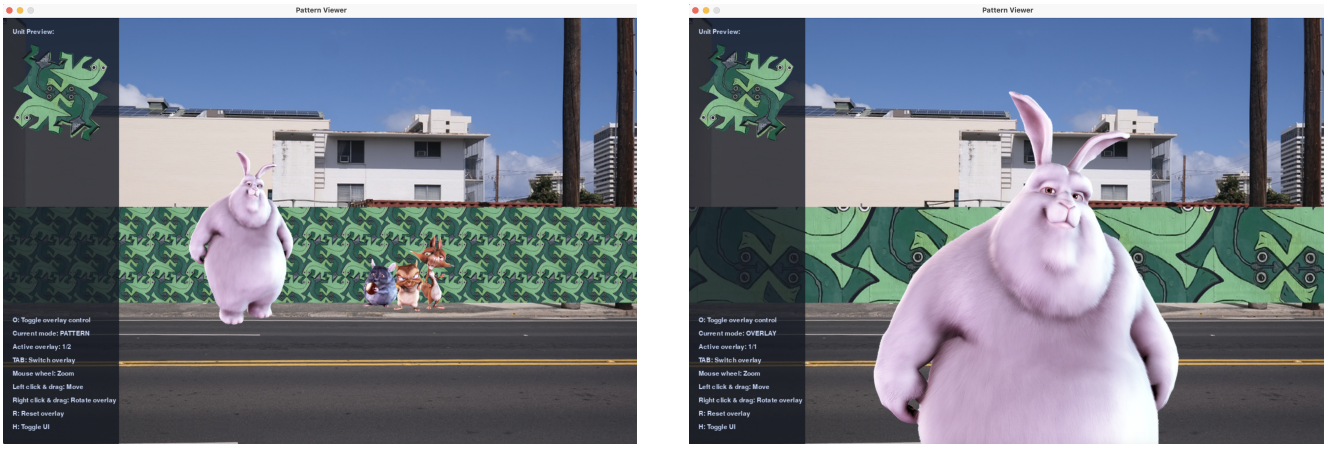


Figure 1. Interactive demonstration of our GUI: users provide a background wall photo, one or more subject images, and a chosen wallpaper-group motif; the system then tessellates the motif and alpha-blends everything into the final tourism-style composition. (a) Multi-subject placement on the tessellated wall; (b) Single-subject placement. The photo of the lizard tessellation mural was taken in 2024 on the corner of McCully and Algaroba streets, Honolulu, Hawaii.

Abstract

We live in exciting times where foundation models such as SegmentAnything V2 and generative shape fill empower digital artists to realize their creative vision like never before. Building on this momentum, we propose a photoshop-like GUI, specializing at bringing anyone to any wall by alpha channel blending. Our pipeline takes as input two photographs (one with a tourist destination containing a wall, another with people) and one illustration (tessellation of the Euclidean plane using one of the 17 wallpaper groups), then outputs an edited artistic tourism photo. Specifically, we employ SegmentAnything V2 to extract clean layers from real photos and generative shape fill to re-create textures for the interlocking shapes crafted by the graphical artist M.C. Escher. Our tool is highly interactive. Text prompting is used to control the texture of the wall. The mouse and keyboard are used to freely zoom, rotate, and reposition both the tessellation and human assets to refine the composition. We contribute a software tool and a data set to the rich history of tiles and decorative art. The GUI, the corresponding source code, and a dataset featuring all the interlocking shapes from the 116 M.C. Escher symmetric pattern will be made freely available.

1. Introduction

Digital artists today benefit enormously from foundation vision models, which have dramatically lowered the barrier to sophisticated image manipulation. In particular, SegmentAnything V2 (SAM) has been shown to extract precise object masks across a wide variety of scenes, enabling downstream editing without manual segmentation[5]. Meanwhile, generative shape-fill methods allow networks to re-texture complex regions guided by both user input and learned priors[6]. Along this trend, Aigerman and Groueix

introduce a fully automated, text-guided pipeline for generating perfectly repeating Escher-style tilings by jointly optimizing mesh geometry and texture[1]. Building on these advances, we designed a GUI that helps artists create playful, tourist-style images.

We propose a Photoshop-like graphical user interface that brings anyone "to any wall." Our pipeline takes (1) a photograph of a tourist destination containing a wall, (2) a portrait photo with people, and (3) a user-selected wallpaper-group tessellation. By leveraging SAM for accurate layer extraction and generative shape fill for recreating textures within Escher-style interlocking shapes, our system enables artists to compose richly detailed tourism images with minimal effort. Text prompts control texture style, while intuitive mouse and keyboard controls allow zooming, rotating, and positioning of both tessellations and human subjects.

Our contributions are threefold:

- Interactive GUI: A user-friendly interface leveraging state-of-the-art segmentation and inpainting models for real-time composition (Fig. 1)
- Pipeline Design: An intuitive workflow combining SAM and generative shape fill with structured tessellation patterns from M.C. Escher's 17 wallpaper groups (Fig. 2)
- Data & Code Release: A curated dataset of all 116 Escher interlocking shapes, alongside our source code and GUI, to spur further research in decorative-pattern-driven image editing.

We believe this tool will empower both amateur and professional artists to explore new forms of digital tourism art, and serve as a foundation for future extensions in pattern-aware image editing.

2. Background: Wallpaper Symmetry Groups

2.1. What Is a Wallpaper Group?

Think of the repeating tiles on your bathroom floor or the intricate mosaics in a historic courtyard behind every such design is a *wallpaper group* [2, 3]. Formally, a wallpaper group describes how to repeat a small motif (anything from a geometric shape to a photo cutout) so it covers the infinite plane without gaps and looks *exactly* the same after simple moves:

- **Slide** it horizontally or vertically (translation),
- **Spin** it around certain points by $360°/n$ (only $n = 2, 3, 4, 6$ work in a seamless tiling),
- **Flip** it across a line (reflection),
- **Flip&slide** along that line (glide reflection).

An algorithmic way of thinking about wallpaper group can be found here Table 1. Amazingly, no matter how ornate the art is, there are only *17* distinct symmetry recipes that satisfy these rules.

2.2. Describing Symmetries with Simple Codes

Each of the 17 groups has a compact label (orbifold notation) that spells out its allowed moves. For example:

- `*442`:
  - '$\star$' indicates one or more mirror lines (like the axes in a classic checkerboard),
  - '4,4,2' means there are two spots you can spin by $90°$ and one by $180°$.
- `632`:
  - no mirrors or glides,
  - three rotation centers of orders 6 ($60°$), 3 ($120°$), and 2 ($180°$)—think of a honeycomb pattern.

(For an applied, graphics-oriented treatment, see Kaplan [4].)

2.2.1. Reading Orbifold Notation

You read the code left to right:

1. A leading '$\star$' if mirror (reflection) lines are present.
2. One or more numbers (`2,3,4,6`) for rotation centers of that order.
3. A '$\times$' for each glide reflection (cross-cap).

So `*442` tells you exactly which flips and spins you'll apply when you tile your motif.

2.3. Why Exactly 17? The Magic Theorem

At its heart lies a neat "counting" trick: every valid flat-plane pattern satisfies

$$\sum_{\text{rotations}} \left(1 - \tfrac{1}{n_i}\right) + \tfrac{\#\text{mirrors}}{2} + \#\text{glides} = 2.$$

Here, each rotation of order n_i contributes $(1 - 1/n_i)$, each mirror line counts as $\frac{1}{2}$, and each glide as 1. For instance, `*442` gives

$$(1 - \tfrac{1}{4}) \times 2 + (1 - \tfrac{1}{2}) + \tfrac{1}{2} = \tfrac{3}{2} + \tfrac{1}{2} + \tfrac{1}{2} = 2,$$

while any attempt at haing a 7 fold rotation would break the sum. Solving this equation under $n_i \leq 6$ and nonnegative integers yields exactly the 17 wallpaper groups.

3. Methodology

Our system implements a novel approach to artistic image manipulation by augmenting modern AI-powered segmentation with classical tiling theory. The methodology can be broken down into four main components: image segmentation(pre-processing), tiling pattern generation, interactive pattern manipulation, and final integration.

3.1. Image Segmentation Pipeline

We leverage the Segment Anything Model V2 (SAM-V2) to decompose input images into three distinct layers:

- **Person layer:** Contains the person(s) of interest

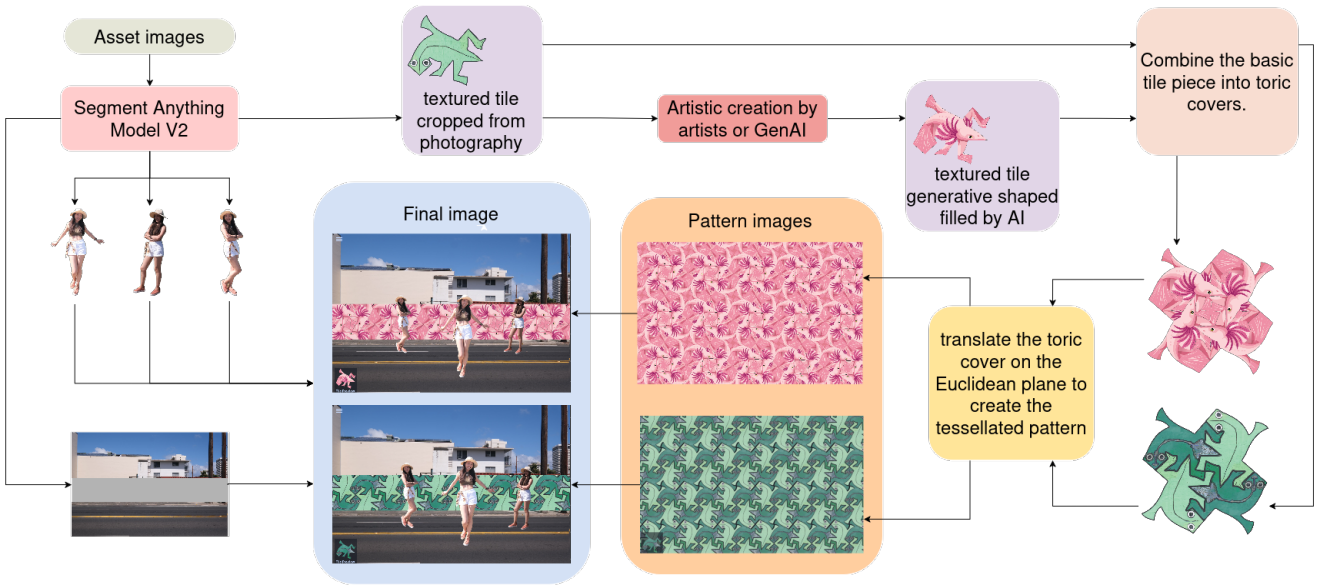


Figure 2. SAM-V2 first segments people of interest and background(right) from the input photo. A motif from any of the 17 wallpaper groups (can be artist- or GenAI-designed), combined into a toric lattice, and tiled to form a full-resolution pattern image (right). The pattern is then composited onto the wall region, yielding the final tourism-style edit (centre).

- **Background layer:** Contains the background
- **Wall layer:** The 2D plane designated for pattern application, we call it a wall

The segmentation process utilizes SAM-V2's powerful instance segmentation capabilities to generate precise masks for each layer, ensuring clean separation between elements.

We mainly modify the Wall layer with our system. We generate a tile pattern and replace the Wall layer with it.

3.2. Wallpaper Group Implementation

Our system implements instances for all 17 wallpaper groups from tiling theory, providing artists with a comprehensive set of symmetric patterns. As is shown in Table 1, each wallpaper group is defined by its fundamental symmetry operations:

- Translations along two independent directions
- Rotations (2-fold, 3-fold, 4-fold, or 6-fold)
- Reflections across axes
- Glide reflections

The implementation uses transformation matrices to represent these symmetry operations, ensuring mathematical accuracy while maintaining artistic flexibility. For each wallpaper group, we compute:

- The fundamental domain (smallest repeating unit)
- Symmetry operation matrices
- Boundary conditions for seamless tiling

3.3. Interactive Pattern Manipulation

We developed an interactive interface that allows artists to:

<table>
<tr><th rowspan="2">Size of smallest rotation</th><th colspan="4">Has reflection?</th></tr>
<tr><th colspan="3">Yes</th><th>No</th></tr>
<tr><td>360° / 6</td><td colspan="3">p6m (*632)</td><td>p6 (632)</td></tr>
<tr><td rowspan="2">360° / 4</td><td colspan="3">Has mirrors at 45°?</td><td rowspan="2">p4 (442)</td></tr>
<tr><td colspan="2">Yes: p4m (*442)</td><td>No: p4g (4*2)</td></tr>
<tr><td rowspan="2">360° / 3</td><td colspan="3">Has rot. centre off mirrors?</td><td rowspan="2">p3 (333)</td></tr>
<tr><td colspan="2">Yes: p31m (3*3)</td><td>No: p3m1 (*333)</td></tr>
<tr><td rowspan="4">360° / 2</td><td colspan="2">Has perpendicular reflections?</td><td colspan="2">Has glide reflection?</td></tr>
<tr><td colspan="2">Yes</td><td>No</td><td rowspan="3">Yes: pgg (22×)</td><td rowspan="3">No: p2 (2222)</td></tr>
<tr><td colspan="2">Has rot. centre off mirrors?</td><td rowspan="2">pmg (22*)</td></tr>
<tr><td>Yes: cmm (2*22)</td><td>No: pmm (*2222)</td></tr>
<tr><td>none</td><td colspan="2">Has glide axis off mirrors?</td><td colspan="2">Has glide reflection?</td></tr>
<tr><td colspan="2">Yes: cm (*×)</td><td>No: pm (**)</td><td>Yes: pg (××)</td><td>No: p1 (o)</td></tr>
</table>

Table 1. Decision chart that classifies all 17 wallpaper–group symmetriesby smallest rotation, reflections, and glides. It lists the corresponding group in both IUC and orbifold notation (e.g. *p6m (*632), p4m (442), pgg (22×), p1 (o)) and serves as a quick reference for selecting the correct wallpaper group during pattern synthesis.

- Scale and translate patterns in real-time
- Scale and translate person(s) of interest in real-time
- Preview base unit before final rendering

3.4. Pattern Generation and Integration

The final composition process involves:

1. Applying the selected wallpaper group transformations to the base tile
2. Generating the complete pattern across the wall layer
3. Integrating the patterned wall with the person and background layers
4. Rendering the final composition with proper depth ordering

Our implementation ensures that pattern generation re-

mains computationally efficient while maintaining visual quality through:

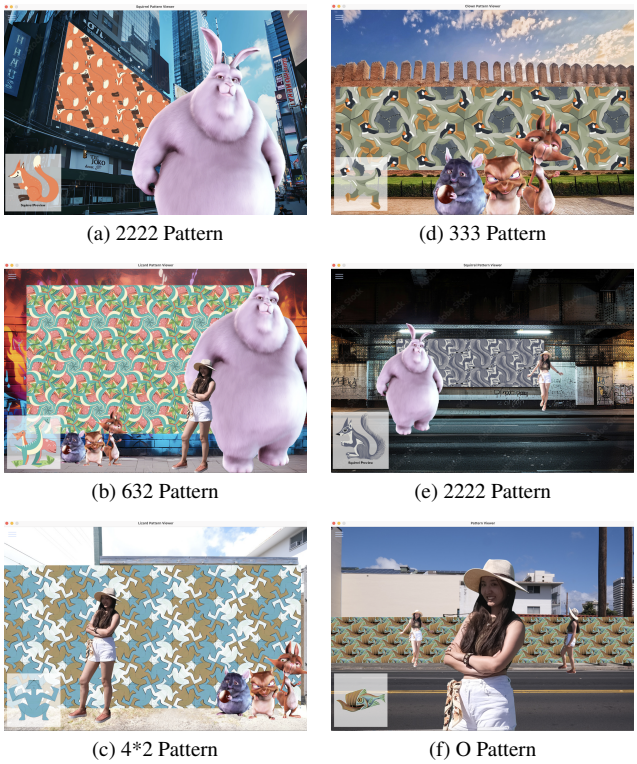
- Cached pattern generation
- Dynamic resolution adjustment based on viewing parameters
- Smooth interpolation for pattern deformations

4. Results & Implementation Details

We built our interactive tool in Python 3.9, using **Pygame2** for real-time rendering and input handling and **Pillow** (PIL) for all raster image operations, loading PNGs, high-quality resampling, alpha compositing, and rotations. At its core sits a single application class, `PatternViewer`, which on startup initializes the display, loads the motif images and background frame, and constructs the "unit tile" by pasting and rotating each motif onto a small off-screen PIL canvas. That unit tile is then repeated across the plane according to the selected wallpaper-group symmetry to form the full pattern image (see Figure 3).

Within the main loop, `PatternViewer` polls mouse and keyboard events to pan or zoom the pattern and to enter overlay editing mode. Whenever the user changes zoom or pan—as illustrated by the different scales in Figure 3—the tessellation is lazily regenerated by tiling the unit image into a larger off-screen PIL canvas and converting it once into a Pygame surface for blitting. Each frame then renders the composite drawing the tiled pattern, the static frame, any movable overlays, and a lightweight UI panel.

To keep performance smooth, we only rebuild the tessellated pattern when its parameters change, avoiding full PIL work on every frame. Overlays are always resampled from the original image using Pillow's LANCZOS filter rather than from an already scaled surface, preventing progressive blur or alpha-channel artifacts. Finally, before entering the main render loop we invoke `gc.collect()` to clear any startup churn and cap the frame rate at 60 FPS using Pygame's clock.



(a) 2222 Pattern (d) 333 Pattern

(b) 632 Pattern (e) 2222 Pattern

(c) 4*2 Pattern (f) O Pattern

Figure 3. Generated artworks using different wallpaper-group symmetries: (a) `2222`: four 2-fold rotation centers at the corners of the fundamental rectangle, producing a diamond-grid tiling; (b) `632`: rotation centers of orders 6,3, and 2 on a hexagonal lattice for a honeycomb pattern; (c) `4*2`: one 4-fold center with mirror lines and a 2-fold center for a reflective square grid. M.C. Escher original not AI generated; (d) `333`: three 3-fold centers per equilateral triangle domain, yielding a triangular tessellation; (e) `2222`: Same as (a); (f) `o`: pure translation group (no rotations or reflections), giving a simple repeat grid. M.C. Escher original not AI generated

5. Conclusion and Future Work

We presented *Bring Anyone to Any Wall*, a Photoshop style GUI that combines SegmentAnything V2 for high quality layer extraction with generative shape fill inpainting to enable seamless real time compositing of people onto Escher style wallpaper group tessellations. Our key contributions include an interactive interface coupling text prompted texture synthesis with mouse and keyboard controls for zooming, rotating, and positioning; a pipeline supporting all 17 wallpaper groups and built atop a dataset of 116 MC Escher tile pieces; and the release of our code and data to foster further work in pattern driven image editing.

Looking ahead, we aim to broaden both the creative and practical scope of the system. On the interactivity side, we will introduce direct manipulation of individual tiles such as pinching, dragging, and continuous rotation of tiles or the entire pattern under lattice constraints via a lightweight solver and GPU feedback. On the geometry front, we plan to handle non axis aligned and curved facades by estimating wall homographies or depth maps and subdividing surfaces into piecewise planar patches, ensuring visually seamless tilings even on slanted, cornered, or irregular architectures.

Acknowledgments

We thank the Blender Foundation for the Peach Open Movie Project. Its assets are released under the Creative Commons Attribution 3.0 license and may be reused with attribution to "Blender Foundation—www.blender.org." We also thank the M.C. Escher Foundation for preserving and sharing Escher's interlocking tile artworks, which inspired our tessellations based on the 17 wallpaper groups.

References

[1] Noam Aigerman and Thibault Groueix. Generative escher meshes. In *ACM SIGGRAPH 2024 Conference Papers*, New York, NY, USA, 2024. Association for Computing Machinery. 2

[2] John Horton Conway, Heidi Burgiel, and Chaim Goodman-Strauss. *The Symmetries of Things*. A K Peters/CRC Press, 2008. 2

[3] Owen Jones. *The Grammar of Ornament*. Day & Son, 1856. Reissued by Princeton University Press, 2006. 2

[4] Craig S. Kaplan. *Introductory Tiling Theory for Computer Graphics*. Morgan & Claypool, 2009. 2

[5] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollar, and Ross Girshick. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4015–4026, 2023. 1

[6] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Free-form image inpainting with gated convolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4470–4479, 2019. 1