TILE: An interactive, multi-touch generative system for designing and placing architectural tiles

Benjamin Harwood[1], Alan Dorin[1⋆], Michael Wybrow[1], and Rowan Opat[2]

[1] Clayton School of Information Technology,
Monash University, Clayton, Victoria 3800, Australia,
[2] Opat Architects, Suite 311.145 Russell Street, Melbourne 3000, Australia

Abstract. In this paper we describe TILE, a novel system for the interactive design and exploration of architectural tiling patterns. Many contemporary architectural designs employ tiled surfaces including walls, floors, windows and roofs. Surprisingly, a significant portion of such design work is arranged manually by architects, which is both tedious and time consuming. The software described here operates on a large multi-touch table and facilitates the rapid creation of tiling regions of user-specified form overlaid on architectural plans. Within these regions arbitrary repetitive tile designs may be generated using the basic bricking patterns. Also, unique and complex tile designs based on the style of artist M.C. Escher may be interactively created and tested on the plan. The software exports the tiling patterns to standard CAD software for incorporation into 3D architectural models. It has been developed in conjunction with a practising architectural firm.

Keywords: architectural design, multi-touch diagram editing, tiling patterns

1 Introduction

Patterns generated with tiles and bricks have a diverse and distinguished history in architecture. Examples include the surfaces of ancient Roman villas dating from around 2000 years ago (Figures 1a and 1b), the surfaces of Medieval Islamic architecture (for example, the Alhambra palace in Granada) and many European Medieval and Renaissance buildings (e.g. Figure 1c). This tradition has continued unabated and is ripe for some degree of automation. Here we present an interactive, multi-touch diagramming tool, TILE,[1] for designing tiles and generating architectural surface patterns.

Contemporary architectural design is a hybrid process, partly analogue and partly digital; partly spontaneous and intuitive, and partly analytic and methodical. An aspect of contemporary architectural design that embodies both

⋆ Corresponding author, alan.dorin@monash.edu
[1] TILE: Tile It Like Escher

(a) (b) (c)

Fig. 1. (a) Floor mosaic, Musei Capitolini, Roma (b) Tiled column from Pompeii (1st C), Il Museo Archeologico Nazionale di Napoli, (c) Striped stonework of the Duomo Siena (13th C), Siena.

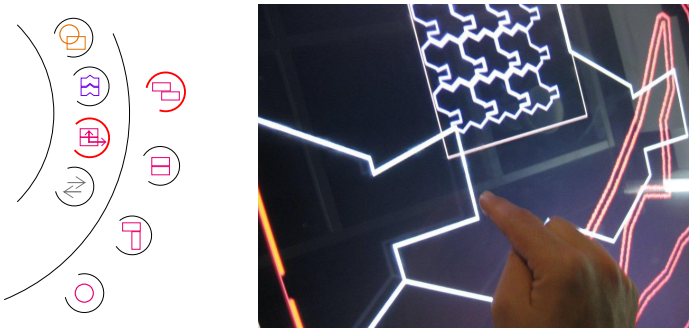
intuitive and methodical activities is surface pattern design. Surface patterns may be realised in brickwork, ceramic or other tiling (e.g. Santa Caterina Market roof, Barcelona based on a design by Toni Comella, EMBT architects, 2005), or even realised in glass and metal panels (e.g. Federation Square, Melbourne, Lab Architecture Studio and Bates Smart architects, 2002).

Even if the choice or design of a pattern is inspired or intuitive, as with any architectural activity, to realise it in a form suitable for construction requires meticulous care. The diagramming software described in this paper facilitates creative tile design and arrangement through interactive editing tools on a multi-touch table/display. This allows architects to work in detail on plans while simultaneously allowing quick and easy evaluation of the overall effect. The software automates tile placement over architectural surfaces that would otherwise require tedious work. Tile designs and the regions to which they are applied remain user-editable throughout the process. Designs can be exported from our software to 2 or 3D CAD software for manipulation and rendering with other components of an architectural design.

Common 2D design software provides little support for the combination of tile design and layout activities. The fact that these steps do not always occur in sequence—often the process of tiling is an iterative one—adds to the difficulty of completing the task manually and provides impetus for automation. The major tasks to be carried out in architectural tiling are:

- Adjustment of individual tile aesthetics: size, shape, texture, colour;
- Adjustment of inter-tile position and orientation;
- Generative application of tiles to a surface ensuring conformity to irregularities at predetermined boundaries and to voids within a region.

The approach in our TILE software is to utilise multi-touch interaction to handle the first two tasks above and for generative software to automate tile placement.



(a) (b)

Fig. 2. (a) TILE main menu (Items listed top-down). Inner circle: region mode, Escher tiling mode, vector tiling mode, export/import. Outer circle (vector tiling mode selected): stretcher-bond, stack-bond, herringbone, packed circles. (b) A user manipulates an Escher tile pattern on the multi-touch panel.

Although there has been some work on multi-touch diagram editing in general (e.g., [2]), there has been little specifically in the context of architectural visual design. Most software is operated with traditional devices such as mice and tablets. However, the introduction of portable multi-touch devices like the *iPad*[2] and laptop and standalone multi-touch trackpads is changing the viability of off-the-shelf multi-touch software. *Sketchbook Pro* is one example.[3]

Software for architectural applications fits into different stages of the design process from inception to final plan. Sketching tools have been recognised as important for conceptual design, allowing fast and flexible prototyping through efficient expression of short-term memory [5]. This has included work on 3D sketch-based modelling systems which interpret a user's strokes as the edges of 3D models [3, 9].

Some work on procedural generation for architectural applications has been conducted. For instance, the design of buildings [6, 7] and cities [8, 10].

Existing software for tile placement such as *Floor Estimate Pro.*[4] and *Precision Tile Pro.*[5] focuses on conventional floor and wall tiling using basic carpet, tile, vinyl, wood and laminate tiles arranged in simple, sometimes restricted predetermined patterns. Whilst these applications are suited to standard residential or occasional industrial use, the scope of our software is less conventional. In addition to standard tiling patterns, we allow creative design of intricate tile shapes utilising a technique employed by the graphic artist M.C. Escher (1898–1972) for tile design. This opens possibilities for tile shape and pattern that are infrequently employed in architecture.

[2] `http://www.apple.com/ipad/`
[3] `http://m.autodesk.com/sketchbook/`
[4] `http://www.floorcoveringsoft.com/`
[5] `http://www.laurelcreeksoftware.com/`

In the next section we present a sample use case. Section 3 provides a detailed discussion of individual aspects of TILE. Section 4 gives sample tiling patterns and a discussion of the software's merits, weaknesses and scope for future work. Our conclusion appears in Section 5.

2 TILE Sample Use Case

Here we outline a sample use case to give an overview of our software's workflow. Details of each step of the process are given in sections 3–4.

User	imports, positions and scales an architectural plan to give a clear view of the regions to be tiled.
User	constructs regions to be tiled and lays them over the plan.
User	selects a tile form and a repetition pattern from the library.
Software	automatically fills selected regions with whole or cut tiles using the selected tile and pattern.
Software	reports the number of whole tiles required.
User	adjusts the pattern's placement relative to region boundaries to satisfy aesthetic criteria or to maximise tile usage or reduce the number of tiles to be cut.
User	unsatisfied with the result, enters a tile design mode to design custom tile forms and layout patterns.
Software	automatically recomputes and redisplays the result.
User	satisfied, exports tiling and region data for use with other CAD systems.

3 TILE System Details and Design

This section describes the system in more detail. We used a PQ Labs 42" multi-touch G3 screen overlay[6] that interfaces with a computer via USB, and a matching display. The overlay and its Java API respond to up to 32 simultaneous contact points and a number of standard gestures including various taps, drags, pinches and rotations. We employed only a few of the recognised gestures (Table 1) and did not extend the gesture recognition software.

3.1 Standard interaction and gestures

TILE is set up for a right-handed user sitting at the bottom of the tabletop screen.[7] The main user-positioned, hierarchical menu controls TILE's modes (Figure 2a). It is usually activated when the user touches the surface with an open palm or the base of a fist. The menu appears as concentric arcs immediately to

[6] `http://multi-touch-screen.com/`

[7] It appears simple to set up the interaction for left-handed operation but we have not tested this. More challenging would be a set up for use by four simultaneous users seated around all sides of the tabletop.

the right of the contact point, ready for operation with the fingers or thumb of the left hand. The user's right hand is more often used for direct interaction with tiles and tiling regions. Menu options are represented along the circumference of the two arcs. The inner arc allows for the selection of different modes, highlighting the current one. The outer arc displays the current sub-mode and alternatives. When a selection is made, the menu can be hidden with a double tap anywhere away from its items. The workspace is then updated to reflect any changes made.

3.2 Region Construction

Complex tiling regions may be constructed within TILE either by instantiating primitive shapes from the library (a triangle, quadrilateral and circle) and translating, rotating and scaling these; or by drawing a closed freehand boundary. All forms may be combined using the Boolean operations union, subtraction, exclusive or, intersection (Figure 3a). Figure 4 illustrates a composite region constructed using these operations. Regions coupled with Boolean operations may be manipulated as single entities or individually. When a composite region is selected, its components are outlined with a dotted stroke to identify sub-regions subject to any edit operations. Inactive regions are outlined with a solid stroke.

Freehand boundaries are created by dragging out an ordered sequence of control points. These are automatically joined by line segments to define a tile clipping region (Figure 3b).

Table 1. Multi-touch gestures and the corresponding TILE actions they trigger.

Gesture	**Context (mode)**	**Software Response**
Single tap	Menu navigation	Select a menu item
Double tap	Region editing	Edit Boolean operators applied to a composite region
	Tile editing	End tile editing
	Main menu	Close the menu and enter the selected mode
	Tile application	Apply the current tiling pattern to a region
Palm tap	Any mode	Activate the main menu
Single down	Region creation	Begin creating a new primitive shape
Single up	Region creation	Finish creating a new primitive shape
Single drag	Region creation	Adjust the size of a new primitive shape
	Region editing	Translate a composite region
	Tile application	Translate a tile pattern relative to containing region
	Vector tile editing	Translate primitive shape; Alter vector length/direction
	Escher tile editing	Position a control point of a cut
Double drag	Region editing	Translate a primitive component of composite region
Pinch/	Region editing	Scale a composite region
un-pinch	Tile editing	Scale a tiling pattern within a region
	Vector tile editing	Scale a primitive shape
Rotate	Region editing	Rotate a composite region
	Tile editing	Rotate a tiling pattern within a region
	Vector tile editing	Rotate a primitive shape

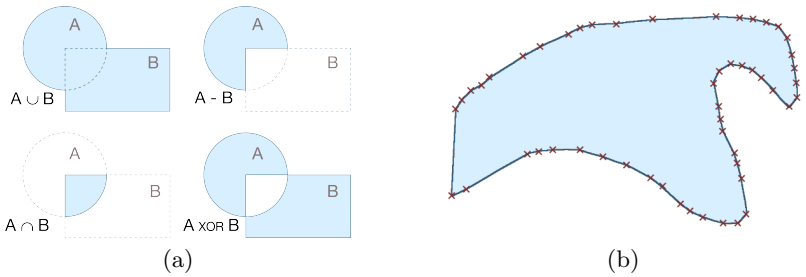


Fig. 3. (a) The basic Constructive Area Geometry operations for region generation are (clockwise from top left) union, subtraction, exclusive or, intersection, (b) A freehand region generated by tracing out a boundary (control points marked).

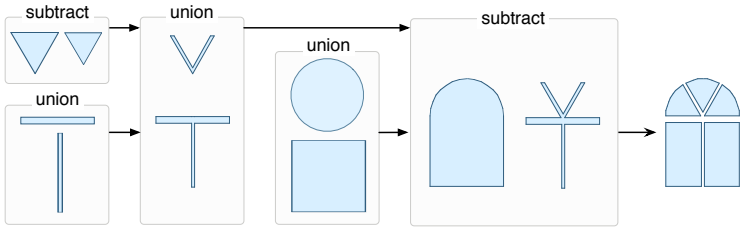


Fig. 4. Boolean tree showing construction of a composite region suitable for filling with a window lattice.

The system stores all regions in a tree data-structure containing shapes and Boolean operations. This allows previous operations to be altered or undone.

3.3 Standard Tiling Patterns

TILE provides a number of common brickwork layouts including stack-bond, herringbone and stretcher-bond (Figure 5a–c). These are presented in a library menu (Figure 2a, outer circle). The user can augment the library with completely original patterns or patterns derived from the library. TILE's two pattern creation modes are described in the following subsections.

3.4 Vector tiling

Vector tiling mode allows specification of the dimensions and relative placement of one or more tiles. The mode presents the user with an area in which to place primitive shapes representing tiles that may be individually positioned, re-sized and rotated. The interface displays two vectors that determine how tiles are repeated and hence the way a pattern is generated (Figure 5a–d). The horizontal vector specifies the offset between each column of tiles and the (more) vertical vector gives the row offset. Each vector has a drag-able control point at its tip. As this is modified a live preview of the tiling pattern is presented.

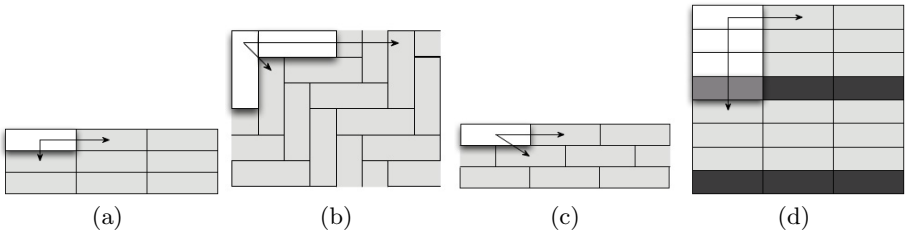


Fig. 5. Images of (a) stack-bond, (b) herringbone, (c) stretcher-bond bricks, (d) a stripe generated with a block of sub-tiles (cf. Figure 1c). These patterns are all generated using vector tiling. Vector axis alignments to generate these patterns from the base arrangements (illustrated in white) are shown.

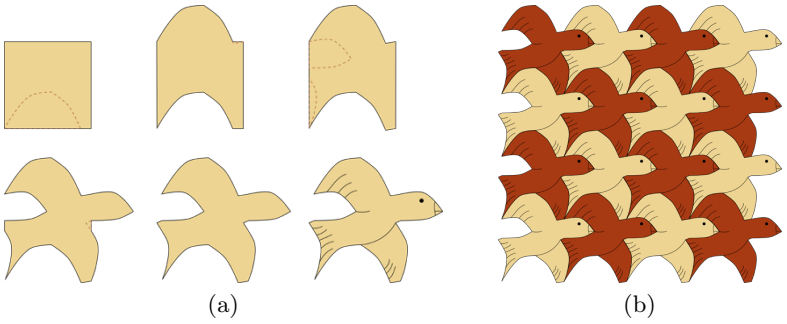


Fig. 6. (a) Six steps required to produce a bird-shaped tile. Dotted lines indicate cuts. Regions marked by cuts are detached from one edge and attached to the opposite edge to enforce valid tile creation, (b) Example of Escher tessellations produced by TILE, where tiles have been textured with other software.

The software constrains the length and orientation of the vectors to stop them becoming unmanageably short or parallel since these cases lead to unrealisable tile overlaps. We do not constrain the vectors to ensure neighbouring tiles abut one other as the user might intend to fill inter-tile space with grout or other supporting material.

Even once a design has been created and applied, the tiles and application pattern remain interactively editable.

3.5 Escher Tiles

TILE's second pattern creation mode employs Escher tiling. M.C. Escher employed a number of tiling methods in his artworks [1]. Here we focus on tessellation with a single tile shape. When using this technique the TILE user's task is to design a tile with concavities on one edge mirrored by convexities on the opposite edge. In this way, when the tile is repeated in stacked columns and rows the plane is properly tessellated (Figures 6 and 7a).

In this mode the user is presented with a square "pattern block" into which they cut regions that are automatically moved to the opposite side of the block, changing its outline. The designer specifies the polygon for the cut region as a series of points, the first and last of which must be located on the same boundary edge of the block (so that there is no ambiguity regarding to which side of the block a region should be moved). To begin a cut, the user taps a point on the shape boundary, then taps one or more points internal to the boundary. Finally they select a cut completion point on the initial edge. The position of each point can be interactively chosen by placing a single finger on the touch panel at the desired location. Adjustments to a point's position are made by dragging the contact finger and lifting it from the surface to lock the point into its place. During this operation, visual feedback indicates limits on locations for placement of valid points. All points are constrained to lie within the tile—the system prevents cut-lines from crossing existing tile boundaries including other cut lines. The user constructs the final Escher tile by applying this action repeatedly (Figure 6a). As with vector tiling, a live pattern preview is shown during Escher tile creation (Figure 2b).

3.6 Input and Output

TILE can save and load its regions and tiling patterns and can import architectural plans in BMP, JPEG and PNG file formats. Plans may be translated and uniformly scaled to serve as a background guide for laying tiling regions.

A completed design can be exported in DXF format[8] for import into CAD software such as *AutoCAD*. This allows it to be slotted into an architectural design workflow.

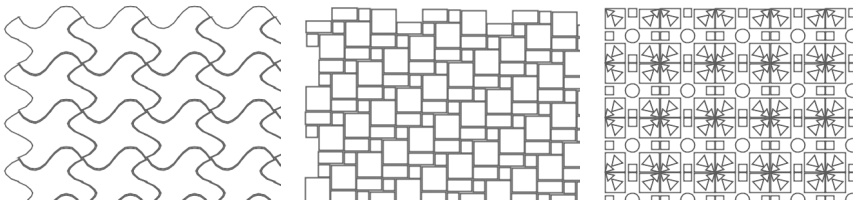
4 Results and Discussion

Research and development of TILE was carried out in conjunction with Opat Architects. Consequently it has been possible to put the software into practice in real-world applications. The software allows for rapid adjustment of tiling parameters and, once regions have been defined—a relatively slow process that could be handled more effectively by automation (a task for future work)—the process of designing unique tiles and placing them into regions is relatively easy to master. The same can be said for selecting from the basic tiling patterns offered by the included library.

The large multi-touch table affords collaboration in a way that a single point device driven interface such as a mouse or graphics tablet cannot. However, since TILE does not currently distinguish different user's touch points, designers must take turns, so as not to interfere with one another's actions.

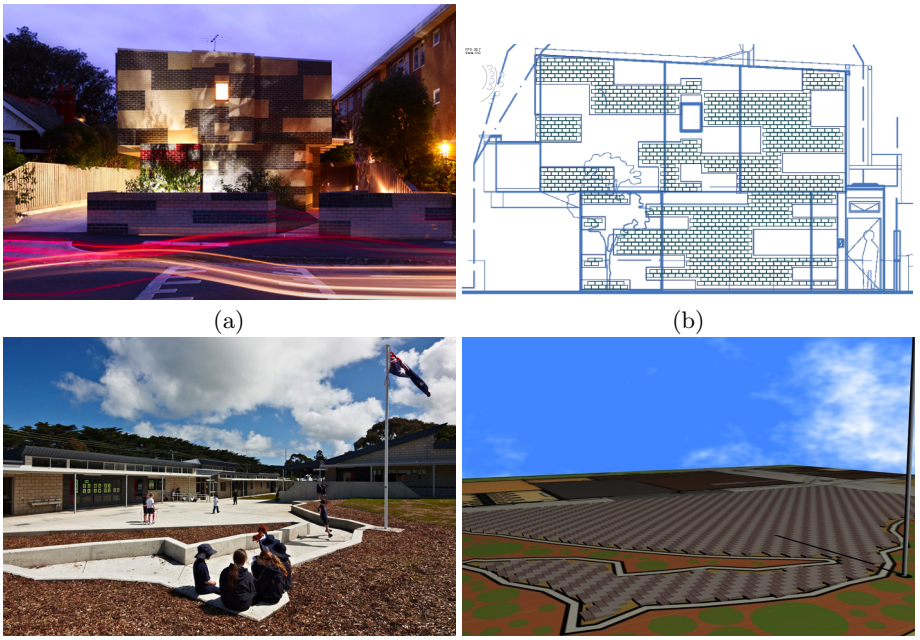
The software excels at tiling large and complex regions with cut or whole tiles (Figures 7–8), making short work of the clipping and pattern generation.

[8] `http://autodesk.com/techpubs/autocad/acad2000/dxf/dxf_format.htm`



(a) (b) (c)

Fig. 7. Sample tiling patterns: (a) a pattern generated using Escher tiling (b) an ornate repeated grid generated with multiple sub-tiles (c) a repeated and offset pattern generated with multiple interlocking sub-tile sizes.



(a) (b)

(c) (d)

Fig. 8. Site photographs and tiling patterns overlaid on architectural plans. (a) Complex brickwork regions on East St. Kilda multi-housing development, Melbourne, Australia. Opat Architects 2011. (b) TILE screen grab of bricked regions. (c) Concreted region, Inverloch Primary School, Victoria, Australia. Opat Architects, 2011. (d) Sample 3D rendering of the irregular region paved with whole bricks.

4.1 Future Work

We have experimented with a few tile design methods apart from those detailed here. We implemented a Reaction-Diffusion system [11] based on the Grey-Scott equations [4] for multi-touch interaction with the aim of employing the concentration of chemicals the system generates to choose aperiodic tile colour, shape

and orientation across arbitrary surfaces and regions. We can achieve results like Figure 1a with such a system. We have also experimented with tiles of varying depth or height specified by a generative process such as the Reaction-Diffusion system just mentioned.

Automatic region creation from imported architectural plans with user fine-tuning would accelerate the tiling process. It would also be beneficial to provide a tool for aligning tiles to an axis that meanders parallel to selected region borders, whilst maintaining the desired pattern generation technique. This would be useful to design brick paths where a pattern's axis is not fixed in global space.

Lastly, we have considered how to morph tile forms across a surface replicating some of M.C. Escher's works in which tiles begin (for instance) shaped like fish and gradually metamorphose into lizards.

5 Conclusions

We have presented new multi-touch interactive software for generating architectural tiling patterns. The software facilitates the overlay of 2D regions to be tiled on an architectural plan, the design and placement of tiles or bricks in standard and custom designs in these regions, and for the export of the final result to standard architectural modelling software. It has been employed successfully to generate tiling patterns for specific architectural projects.

Acknowledgments. The Centre for Research in Intelligent Systems (CRIS), Faculty of Information Technology, Monash University provided the funding to purchase the multi-touch overlay used in this project.

References

1. Escher, M.: The graphic work of M.C. Escher. Gramercy Pub. Co. (1984)
2. Frisch, M., Heydekorn, J., Dachselt, R.: Diagram editing on interactive displays using multi-touch and pen gestures. In: Goel, A., Jamnik, M., Narayanan, N. (eds.) Diagrammatic Representation and Inference, Lecture Notes in Computer Science, vol. 6170, pp. 182–196. Springer Berlin / Heidelberg (2010)
3. Igarashi, T., Matsuoka, S., Tanaka, H.: Teddy: a sketching interface for 3d freeform design. In: ACM SIGGRAPH 2007 courses. SIGGRAPH '07, ACM, New York, NY, USA (2007)
4. Lee, K.J., McCormick, W.D., Ouyang, Q., Swinney, H.L.: Pattern formation by interacting chemical fronts. Science 261, 192–194 (1993)
5. Lipson, H., Shpitalni, M.: Conceptual design and analysis by sketching. Artificial Intelligence for Engineering Design, Analysis and Manufacturing 14, 391–401 (2000)
6. Moloney, J.: Designing Kinetics for Architectural Facades: State Change. Taylor and Francis Ltd. (2011)
7. Müller, P., Wonka, P., Haegler, S., Ulmer, A., Van Gool, L.: Procedural modeling of buildings. ACM Transactions on Graphics 25, 614–623 (2006)

8. Parish, Y.I.H., Müller, P.: Procedural modeling of cities. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques. pp. 301–308. SIGGRAPH '01, ACM, New York, NY, USA (2001)
9. Schmidt, R., Wyvill, B., Sousa, M.C., Jorge, J.A.: Shapeshop: sketch-based solid modeling with blobtrees. In: ACM SIGGRAPH 2007 courses. SIGGRAPH '07, ACM, New York, NY, USA (2007)
10. Soddu, C., Colabella, E.: Recreating the city's identity with a morphogenetic urban design. In: 17th International Conference on Making Cities Livable (1995), `http://www.soddu.it/freiburg.htm`, accessed December 2011
11. Turing, A.M.: The chemical basis of morphogenesis. Philosophical Transactions of the Royal Society 237(641), 37–72 (1952)