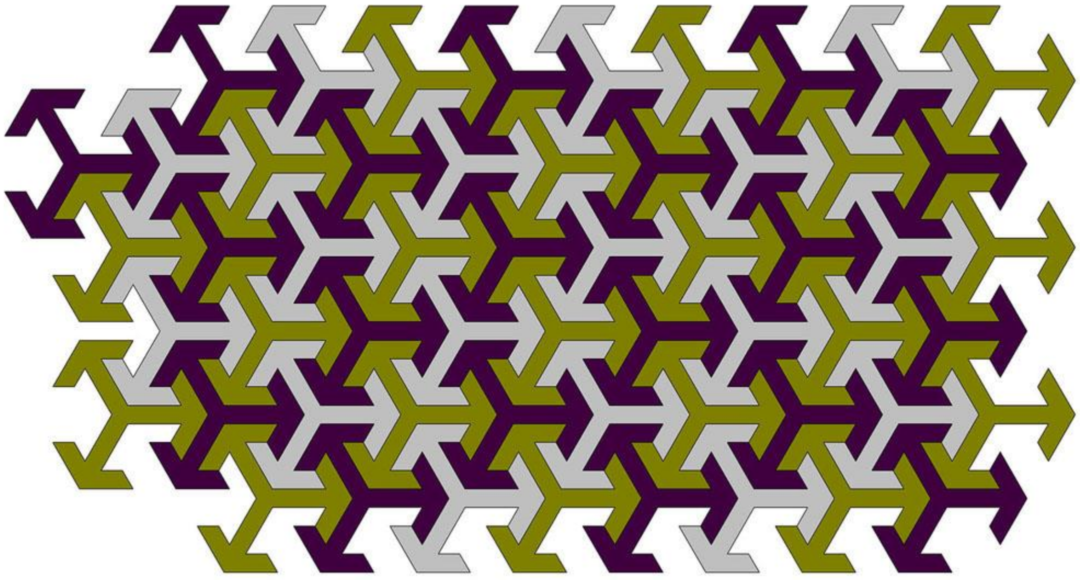
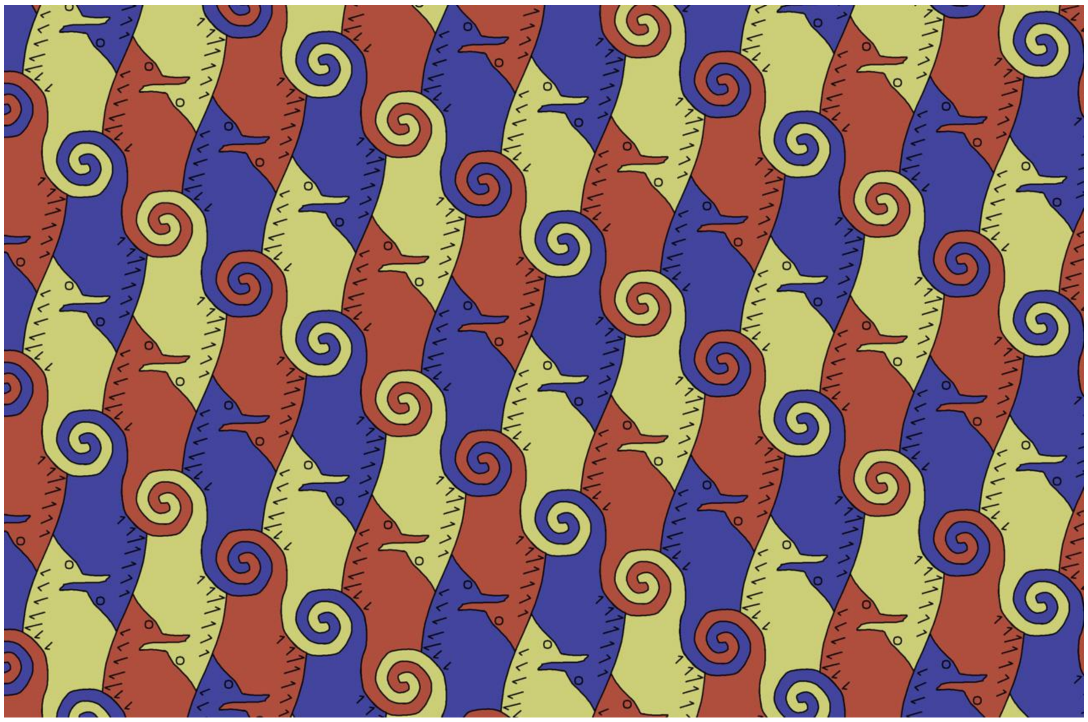
EscherSketch™

A tile creation, tessellation and transformation tool



Tessellation of 3-way arrow tiles created on a hexagonal grid



A recreation of a classic tessellation by M. C. Escher

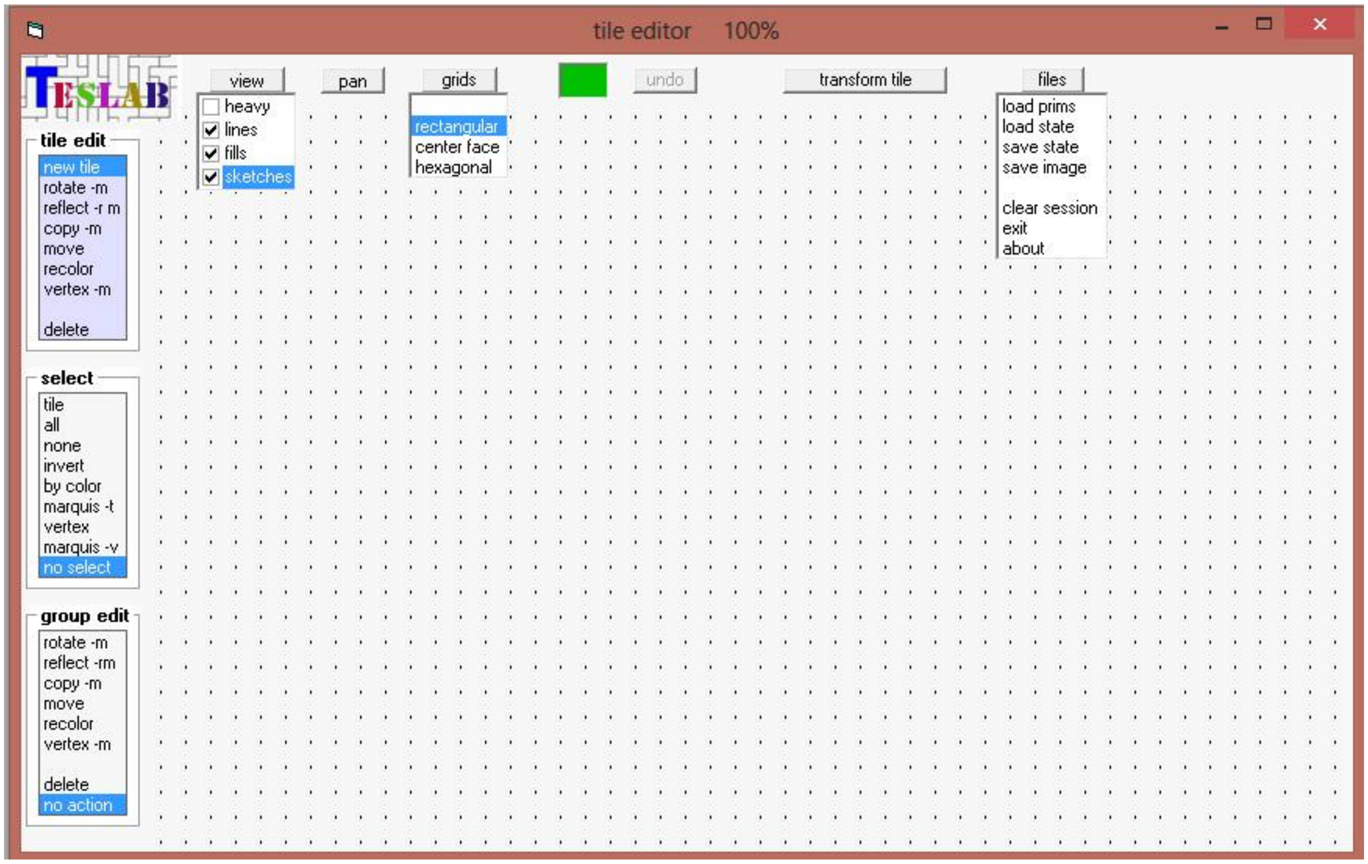
Overview and EscherSketch Program Startup

EscherSketch allows a designer to rapidly create tessellations of tiles such as those that are familiar in traditional Islamic art, mosaics, stained glass and many other forms. It also facilitates the creation of complex tiles that can be assembled to tessellate a plane in the style made famous by the Dutch artist M.C. Escher. There are two windows used to perform these functions: a **tile editor** and a **tile transformation editor**. These will be fully explained in the following sections.

EscherSketch starts in the **tile editor** with a rectangular grid. To create a tile, click points on the grid to create a shape. The shape becomes a tile when it is closed, that is when you return to click on the point where you started. Once closed, the tile will be filled with the color that is indicated by the colored rectangle at the top of your window, in this case green. You may then begin creating a new tile.

The Tile Editor

The tile editing window consists of two parts: the **editing pane** that is studded with grid points, and is the canvas on which you can create and modify designs, and a set of tool lists and tool **command** buttons organized along the top and left edges of the editing window.



View of the tile editing window with all tool lists expanded

The grid

The grid of points is more than just a convenient reference for organizing your design. It also defines *snap points*, that is points to which all tiles and vertices will have their positions adjusted to ensure tight fitting and gapless tessellations. There are several different grids, each appropriate to different types of tiles**.** **EscherSketch** supports one drawing for each grid type. You can move between drawings by selecting a new grid. You can disable snapping temporarily by holding down the spacebar during any editing operation. It is recommended that you use this sparingly, if at all.

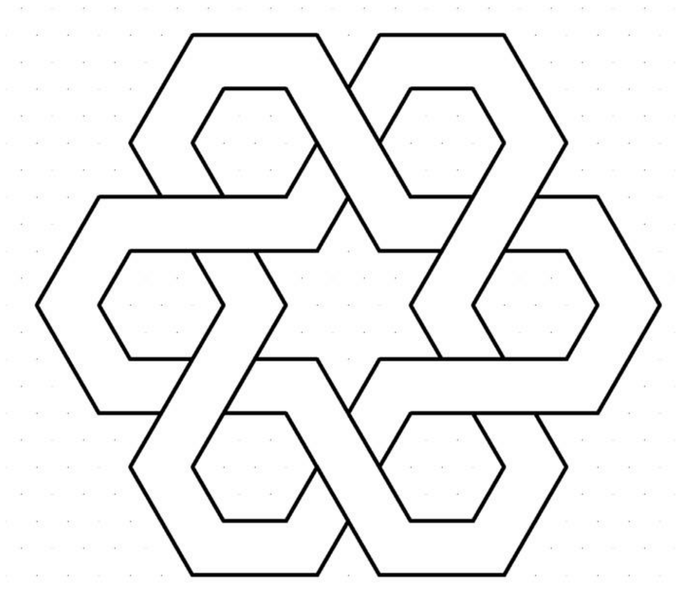
The <u>grids</u> pulldown menu

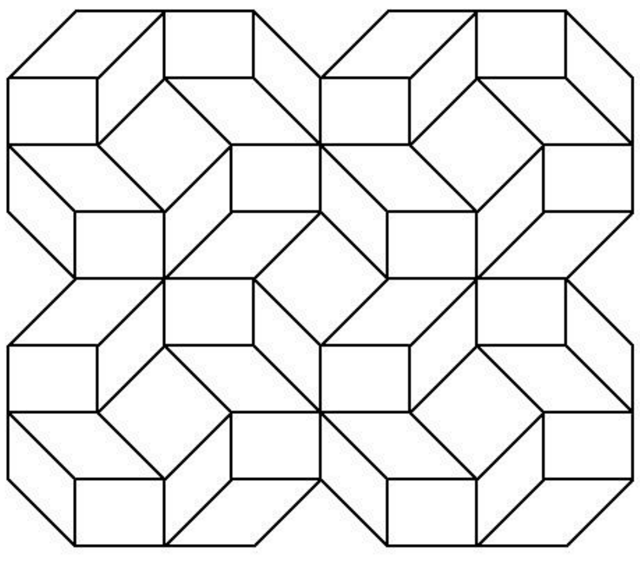
Click on the **<u>grids</u>** command button to reveal the grids options list.

EscherSketch supports multiple grid types, each one appropriate to a different class of tiles. Tiles will only be displayed on the grid on which they were created, thus you can create a different drawing for each of the grid types. The default grid is **rectangular**. This grid is appropriate for tiles with angles that are multiples of 45 degrees such as squares and isosceles right triangles, or any tile featuring 45 degree edges. However it is by no means limited to tiles of these types. Another very useful grid is **hexagonal**. This grid is appropriate for tiles with angles that are multiples of 30 degrees such as perfect hexagons, equilateral triangles, 30-60-right triangles and isosceles quadrilaterals. Another type of grid is **center face** which is set on a 45 degree diagonal and is similar in function to the rectangular grid.

The <u>pan</u> tool

The tessellation plane is larger than the window on which it is displayed. You can navigate around the plane by clicking on the **<u>pan</u>** command button. The mouse pointer will then display a hand that you can use to click and drag your view around in the window. When active, the **<u>pan</u>** button is displayed in red. It can be deactivated by clicking it again, or by selecting any other tool. There is also a set of zoom functions that are active at any time, including while panning. They are the "**>"** key (no shift required) to zoom in, the "**<**" key to zoom out and the "**/**" key which will zoom out all the way to view the entire tessellation plane at a small scale. The tile editor window can be resized at any time, for example by stretching or maximizing it, then use the pan and zoom tools to recenter the image.



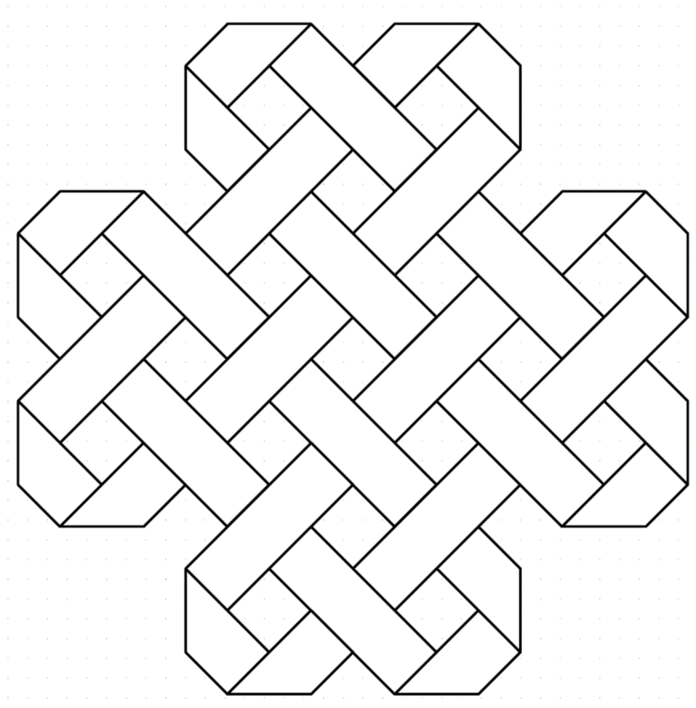


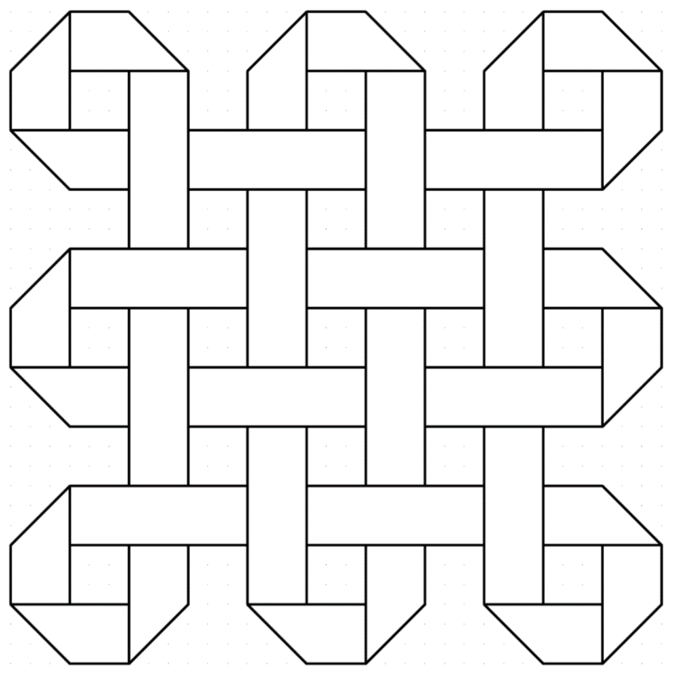
The <u>view</u> pulldown menu

Click on the **<u>view</u>** command button to see a list of options that affect the way tiles are drawn on the screen. By default**,** **EscherSketch** will draw a black border around each tile and fill the tile with the color that was selected when the tile was generated. Unchecking **lines** in the view pulldown will cause tiles to be drawn without the black border and unchecking **fills** will cause all tiles to be filled with white, thus creating a line drawing effect. If both of these options are unchecked tiles will be displayed simply as outlines that are colored with the tile color. Checking the **heavy** option causes the black tile borders to appear extra wide. This is a nice effect, for example when creating stained glass replicas. The final checkbox in the pulldown, **sketches**, is used for complex tiles that will be discussed later, and has no immediate effect when the program is started.

The color chip

The **color chip** is the colorful rectangle right in the middle of your tool bar. It displays the color that will be applied to newly created tiles, and is used as the default color by the **recolor** tools. You can change its color two ways: first, by clicking on it which will allow you to acquire a color from the editing pane by clicking on a tile or by right clicking it which will bring up a color dialog (see **<u>tile</u> recolor** tools on p. 5). The color chip is also useful for copying a color from one tile to another tile (or group of tiles) using the **recolor** tools.





The <u>tile edit</u> tool list

This is the first of three tool lists used to create and modify tessellations. Each tool in this list tools operates on individual tiles. Click any tool in the **<u>tile</u>** list to activate the tile tools. When active, the list will be highlighted with a blue background. **EscherSketch** starts up with the **tile** list active.

The **new tile** tool is the program default, that is, the tool that will be active when no other tool is selected. A cross-hair icon will be displayed as the mouse pointer. Click any point on the grid and it will be highlighted by a red circular marker. This signifies the starting point of a new tile. Click additional points to add new edges to your tile. If you want to back up and reenter an edge simply reclick the last point. There is no limit on the number of edges for any individual tile. When you return to the starting point the shape becomes a tile and is filled with the color displayed in the color chip. A tile can also be closed by right clicking in the editing pane as long as at least three points have already been entered. If less than three points have been entered a right click will abort the new tile (no tile will be created).

The **move** tool allows you to select a tile and drag it to a new location on the grid. The **copy –m** (copy – move) tool is similar to **move** except the selected tile will be copied before you drag it to its new location. If you do not move it to a new location, no copy will be made and you will hear a beep. This is done to avoid inadvertently creating stacked identical tiles which could cause problems down the road.

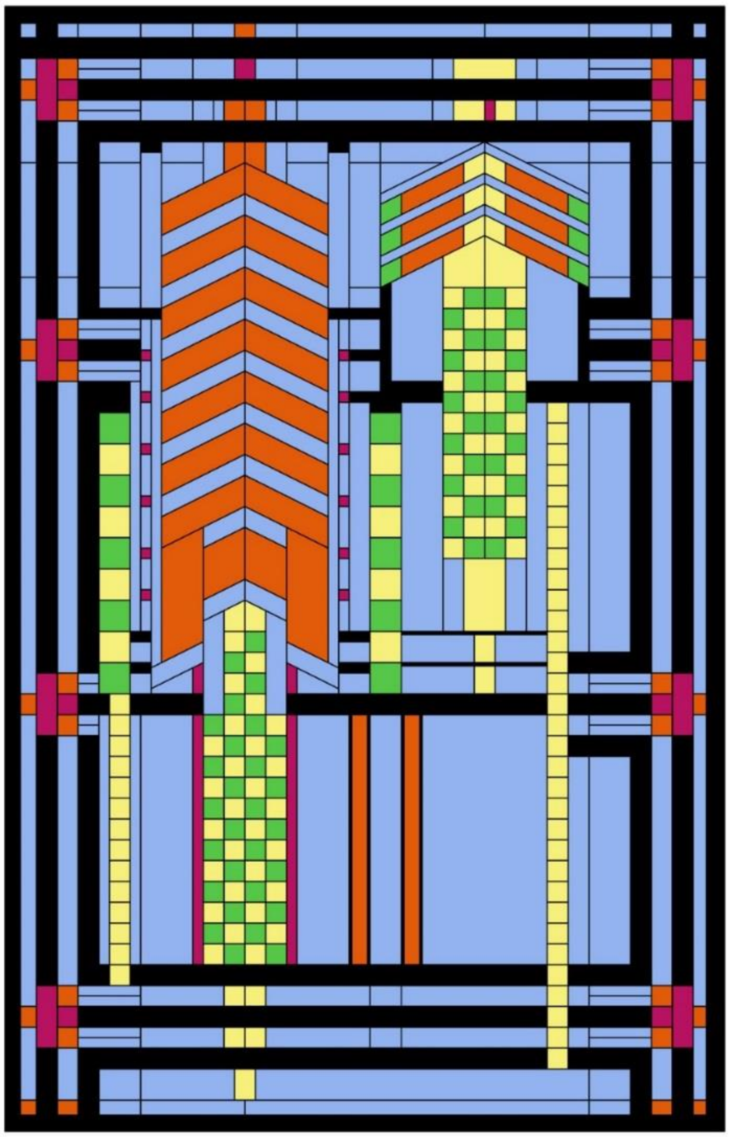
The **rotate –m** (rotate-move) tool is multifunctional in that it allows you to rotate a tile and move it to a new location in one operation. Select the tool then drop your mouse button on any tile to select it. With the mouse button down, drag vertically to rotate the tile about its center. Rotation angles are snapped to increments of 15 degrees to allow you to arrive accurately at any of the most desired angles. Like the snap grid, the snap angle can be overridden by temporarily by holding down the spacebar during rotation (again, this creates a tile that is off grid. Beware!) The final rotation angle is set when the mouse button is released, and the rotated tile is left “floating” for you to move and snap into place.

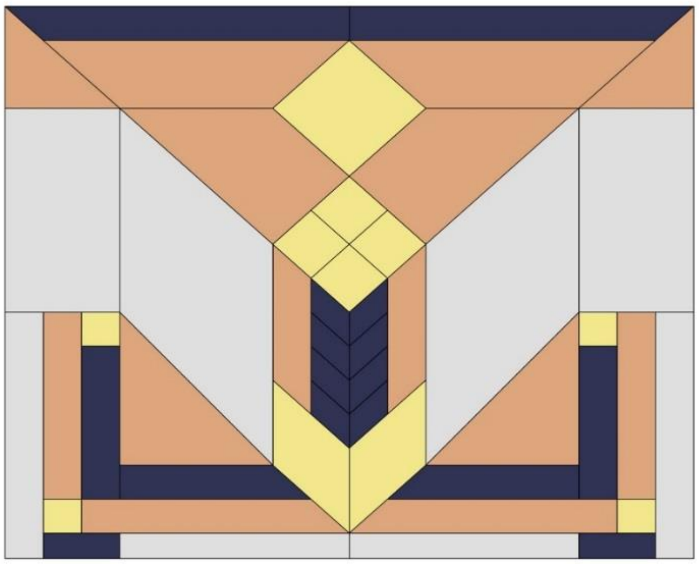
The **reflect –rm** (reflect-rotate-move) tool combines three operations into a single particularly useful tool. It is similar **to rotate –m** except that the tile is reflected (mirrored) about a line through the center of the tile at an angle that is perpendicular to the tile’s initial rotational orientation before rotating it (by holding down the mouse button and dragging it vertically). This combination allows you to reflect a tile about any axis (yes, this really *does* work). Like **rotate –m**, the tile is left floating for you to move and snap into place.

The **vertex –m** tool allows you to select a vertex and drag it to a new location. If the selected vertex is shared by more than one tile (i.e. *stacked*), the entire stack will be moved (thus modifying multiple tiles). This is the one time that the snap grid might be disabled (with the spacebar) with successful results. (Still, beware!)

The **recolor** tool allows you to select a tile and assign it a new color by means of a color selection dialog box. When this dialog appears it will have preselected the color displayed in the **color chip**. You can modify that color by selecting from a set of predetermined color samples, or by creating a new color using the color creation tool on the right half of the dialog box. With the recolor tool active it is simple to copy a color from one tile to another. Simply click on the color chip then click on any point on any tile on the screen to acquire a new color for the chip, then click any tile to apply the acquired color to that tile.

The **delete** tool allows you to select and delete a tile with a single click.





A pair of Frank Lloyd Wright stained glass designs replicated with EscherSketch

The <u>select</u> and <u>group edit</u> tool lists

The tools in these two lists work together to make edits on multiple tiles simultaneously.

Click on any tool in the **<u>select</u>** tool list to activate the list of selection tools. When activated this list will be highlighted with a yellow background. The purpose of these tools is to build a *select set* for group edits. In general, these tools are additive, meaning each use adds to the set of already selected tiles.

<u>Select</u> tile allows you to click on a tile to add it to the select set. Once selected, the tile is filled with a crosshatch pattern to signify it is part of the select set. Using this tool on a tile that is already selected will remove it from your select set (this is the one select tool that is not always additive.)

Groups of tiles can be selected all at once with several tools. **<u>Select</u> by color** allows you to click on a tile, and then it selects all tiles of the same color. Select ***marquis –t*** allows you to draw a bounding box, or *marquis*, in the editing pane, then selects all of the tiles that have a vertex that falls within that box. To draw a marquis, click and drag from one vertex of the box to the opposite (diagonal) corner. When you let up the mouse button the marquis disappears and the tiles will be selected.

<u>Select</u> all selects all tiles. This is useful for moving your entire tessellation to a new location in the plane (for example when some tiles have been partially or completely moved outside the editing pane). **<u>Select</u> none** clears your select set, that is deselects all tiles. **<u>Select</u> invert** deselects all selected tiles and then selects the remaining ones in a single operation.

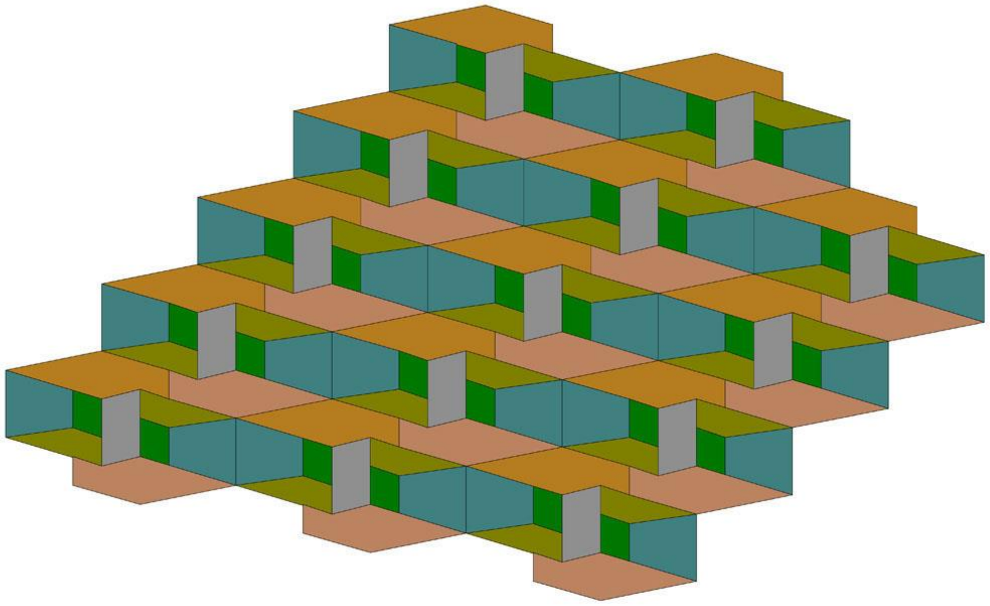
There are two select tools that allow you to select vertices. Select **vertices** allows you to select individual stacks of vertices by clicking on them. This tool also allows you to deselect vertices by clicking ones that are already selected. Select **marquis –v** is similar to **marquis –t** except that it selects vertices, not tiles.

Click on any tool in the **<u>group edit</u>** tool list to activate the list of group editing tools. When activated this list will be highlighted with a yellow background. These tools operate on select sets. For example, the **move** tool allows you to click anywhere in the editing pane then drag the selected tiles to a new location. For every tool in the **<u>tile edit</u>** tool list there is a corresponding tool in the **<u>group edit</u>** tool list, except that it takes its effect on the entire set of tiles you have selected. Like the **<u>tile</u> copy –m** tool, **<u>group</u> copy –m** will require you to move your selection to a new location to complete the copy operation.

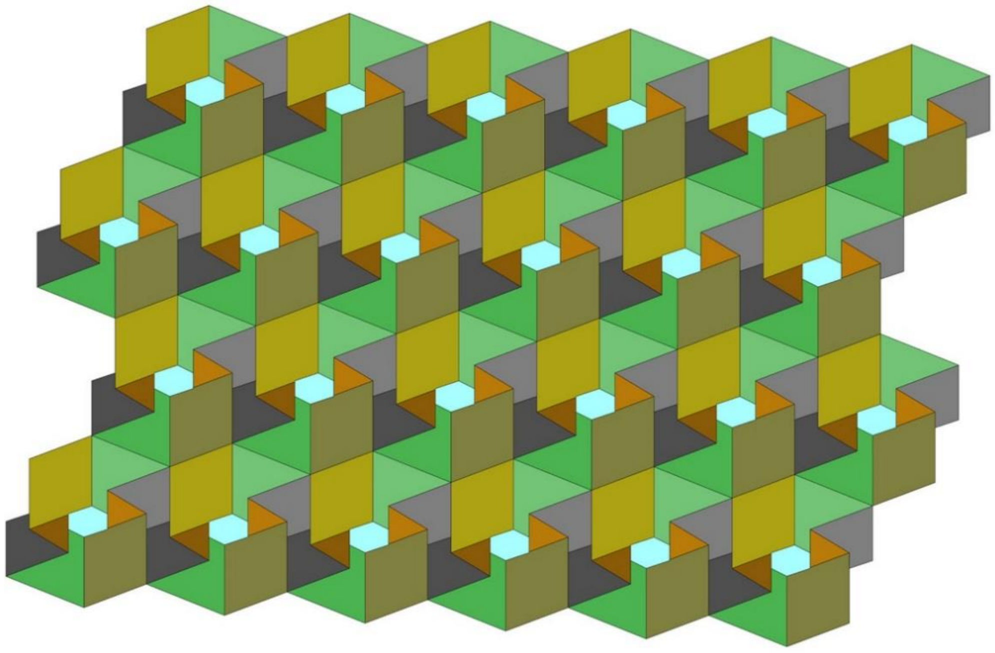
The **undo / redo** command button allows you to undo your last edit command. It does not apply to **<u>select</u>** commands.

General comments on the tool command buttons

Any tool list can be deactivated by clicking any other tool command button. Doing so will also deactivate any tool that was selected in the original list, and return the editing pane to its default tool of **<u>tile</u> new tile**. Only one tool list and one tool can be active at any one time.



An adaptation of a Josef Albers creation presents an optical paradox



Another Albers adaptation depicts tessellating Mobius strips

The <u>files</u> pulldown menu

Click the **<u>files</u>** command button to reveal the **<u>files</u>** pulldown menu.

The **<u>files</u> load prims** command will load a set of readymade tiles (known as *primitives*) with interesting properties. All of these tiles tessellate and display various types of symmetry that are particularly useful as starting points when creating the complex tessellating tiles that are the subject of the second half of this document. Normally this command would be executed at the very start of an editing session. Any of the tiles you do not wish to keep for future use can be eliminated as follows: select the tiles that you want to keep using the **select tiles** tool, then use the **select invert** tool to shift your selection to the tiles you do not want, then use the **group edit delete** tool. After that, just move the remaining tiles to a useful location on the grid and use them just like any other tiles that were created using the **tile edit new tile** tool. Note that there is a rich set of primitives available on both the rectangular and hexagonal grids.

Loading primitives can be done at any time, not just at **EscherSketch** startup. However doing so will overwrite all of your existing tiles so **EscherSketch** will issue a reminder for you to save your work.

The **files save state** command saves your tessellations in a binary file format with a .tes file extension. These files can be restored later using the **load state** command. In general you can save the state of your session at any time, and it is good practice to do so periodically. **EscherSketch** will prompt you for a file name and the save command will be completed. In addition, the first time you run the program **EscherSketch** will give you the option to associate files with the .tes extension to this application. By clicking yes you will enable the ability to start the program by double clicking any .tes file in the windows fie explorer.

The **files load state** command is most often used at the start of an editing session, similar to the **load prims** command. Loading a new program state will overwrite any work you may have in progress, so **EscherSketch** will remind you of this and give you a chance to save your work or cancel the load command altogether. If you agree to proceed you will be prompted for a file name then the load command will be completed.

The **files save image** command creates an image file in one of four supported formats: Bitmap (.bmp) and JPEG (.jpg) formats are handily portable to a large number of other graphics programs that can be used to print your creations, convert them to other image formats and possibly perform a variety of other image modification functions. Metafiles (.wmf) and Enhanced Metafiles (.emf) are vector formats. They are recognized by far fewer external programs, but can provide a link to many versatile CAD programs that can gain access to all manner of devices, everything from cutting and milling machines to embroidering sewing machines. This is most applicable to transformed (complex) tiles that are the subject of the remainder of this document. After selecting this tool **EscherSketch** will prompt you for a file type and name, then provide you with a low resolution screen image of your entire editing pane and instruct you to pull a marquis around the portion to be included in the image. This is done in exactly the same way that we did using the **select marquis -t** and **marquis –v** tools earlier, by clicking and dragging from one diagonal corner to the other. You can redraw the marquis as many times as you need to get the exact framing you desire, then click the enter button on the prompt window. You will then be prompted to select a resolution. Generally, the lowest resolution produces the smallest file size and the fastest response, but also the lowest image quality.

The **clear session** options erases all tiles on all of the grids. You might want to save your work first (**files save state**). If you have unsaved work **EscherSketch** will remind you and give you a chance to save before your session is cleared.

Finally there is an **exit** command to close **EscherSketch**. Upon exiting you will be asked, if appropriate if you want to save your work (in a .tes file, of course). You can also close **EscherSketch** by using the familiar red X button on the top border of the window.

The transform tile / return to transform command button

This button is your gateway to the **tile transformation editor** (or just transform editor) which is described in the remainder of this document. To open the transform editor, click the **transform tile** button and select a tile from the tile editing pane. The **return to transformation** function will be described shortly.

The Tile Transformation Editor

Background

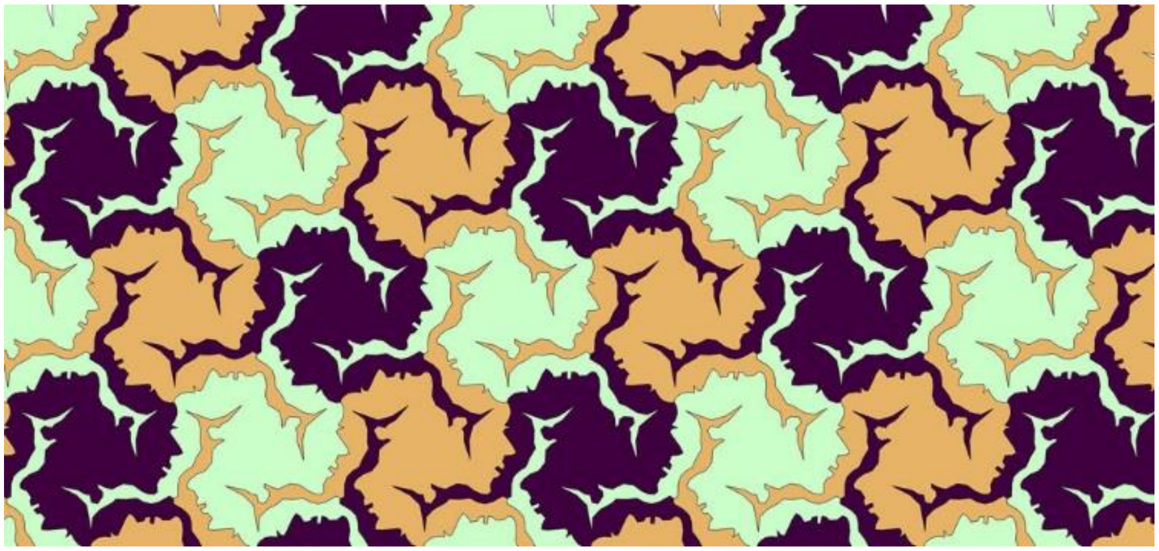
You may have wondered as I did just how M.C. Escher and other artists were able to create those fantastic images of interlocking creatures going this way and that. Actually there is a method to it. At their core, these images are tessellations of just the types that can be created using the **EscherSketch tile editor**. The trick is to generate these seemingly incomprehensible tiles with shapes that remind us of the creatures they become, and that go back together with no gaps or spaces.

The methods used to create tiles like this are rigorous from a mathematical standpoint, and overall not too hard to understand (after all they are visual). They stem from the three familiar symmetries inherent in some polygons, namely *translational, rotational* and *reflective symmetry*, and a fourth one that you may have never considered called *glide reflection symmetry*.

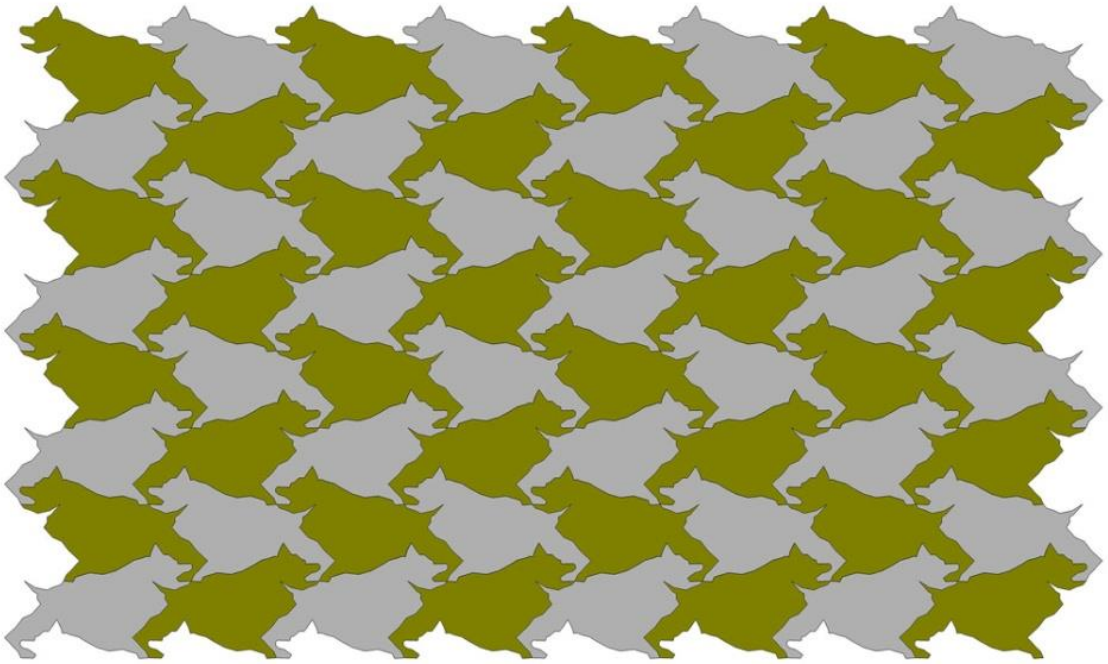
These properties of polygons must have been abundantly familiar to M.C.E. and others who created these captivating images, but the task is still daunting using traditional tools like pencils, tracing paper and compass points. To create these tessellations with extreme precision is even more arduous, and a testament to the patience and skill of those artists.

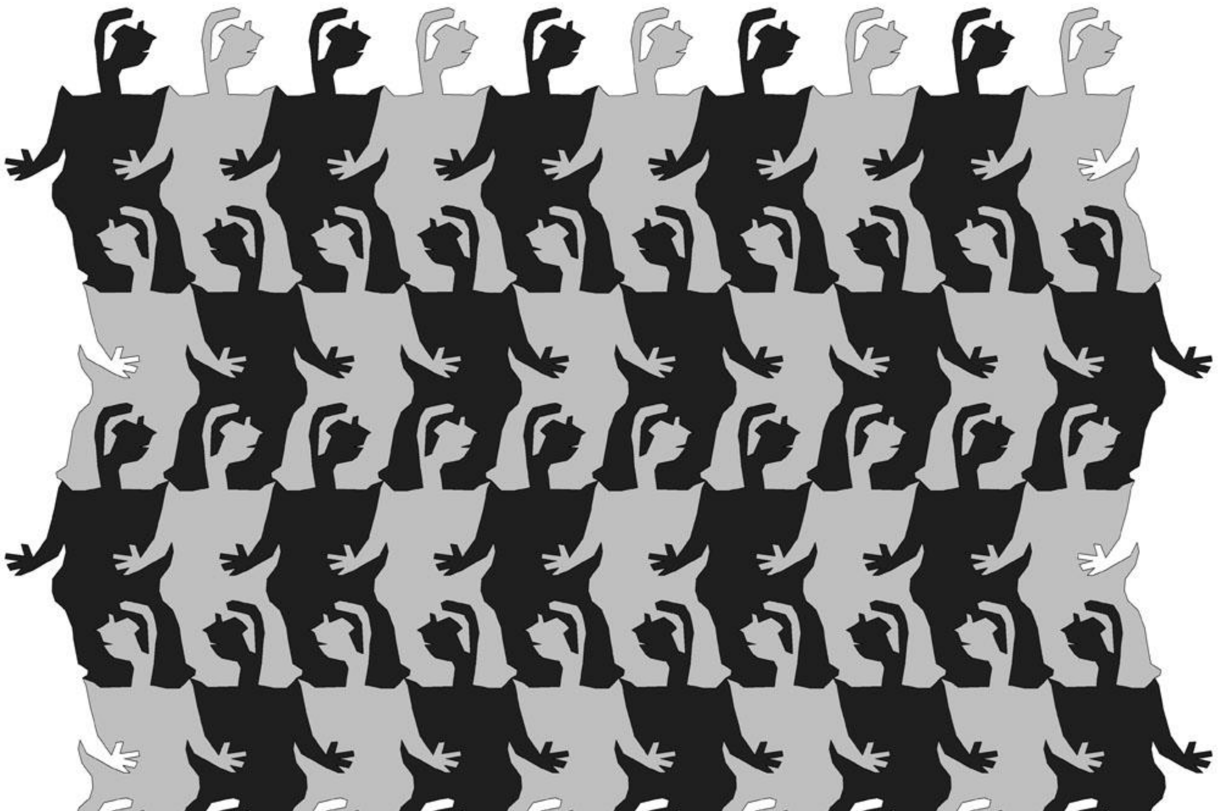
The **EscherSketch tile transformation editor** is designed to assist designers, artists, students and enthusiasts who endeavor to make these creations on their own, but with a level of automated assistance that the masters probably never dreamed of.

I will not attempt here to present the methodology in any detail; I am not even an expert on it. But I will reiterate that it is not that hard to understand. There are plenty of good references on the subject, some listed at the end of this document, and others that are available on line. You can easily find them using your favorite search engine with key phrases like tessellations, Escher, geometric symmetry, tile (or geometric) transformations and many others. With a little research and practice, and the **EscherSketch tile transformation** editor you can become a master of these images in your own right.



"three faces with hair" created from a hexagon using vertex rotation transformations

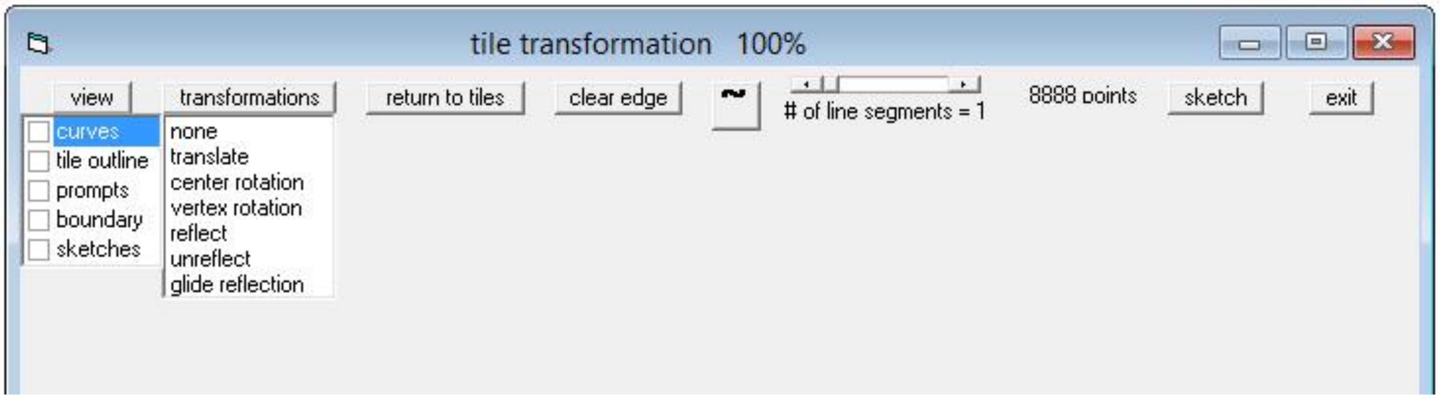

A recreation of an Escher classic using glide reflection and translation transformations


Guys dancing created from a parallelogram using glide reflection transformations

Overview and startup

The process of creating a tessellation of complex tiles begins and ends in the **tile editor** with at least one, and possibly many separate excursions to the **transformation editor** and back. To start off, create a single tile in the tile editor. It must be one of three types: a triangle, a quadrilateral or a "special" hexagon, that being a hexagon with all three pair of opposite edges that are both parallel and congruent (the same length). These are all polygons that are known to tessellate the plane (fill the plane entirely by themselves with no gaps or spaces) and they are universally used as the starting point for complex tiles. Click on the **transform tile** button and select the tile you just made.

This starts the tile **transformation editor** which will appear in a new window that features a black line drawing of a polygon representing the starting tile, and a tool bar at the top of the window. There is an editing pane outlined in grey that contains the tile. The edges of this pane form a bounding box beyond which you cannot move points. You can zoom this pane the same way you did the tile editor (by using "**<**", "**>**" and "**/**" on your keyboard), and you will probably need to do this several times during the transformation process to get the detail you desire. This editor features a prompt message in red somewhere above and to the left of your polygon. At startup it will prompt you to select a transformation. You do this by clicking the **transformations** button in the toolbar along the top of the window and selecting one of the tools from the pulldown.

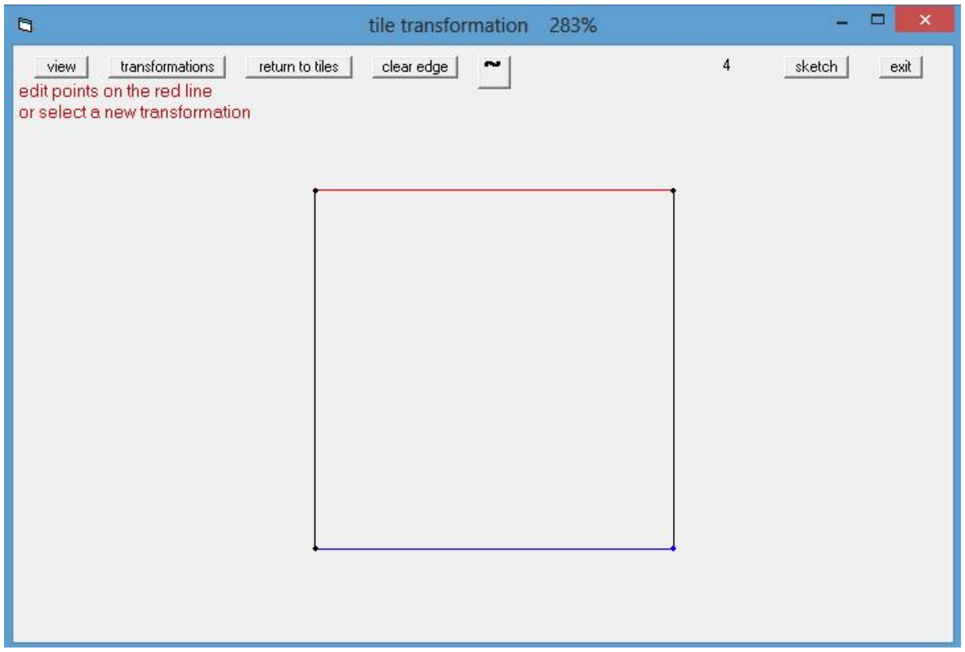


View of the transformation editor window with the tool lists expanded

There is no grid in this editing pane. It is very free form in that way, but very constrained in other ways that help to keep you on track. For example, it will not allow you to move the vertex points of your starting tile. This helps ensure that your transformed tile will snap back together once you return to the **tile editor**. Also, the lines and points that you CAN edit will always be highlighted in red. The ones you can edit will depend on the transformation tool that is currently selected. **EscherSketch** will not let you move points outside the edges of the editing pane, although that pane is quite large. The starting tile will always be present as a dashed green line, although it will initially be hidden beneath the black polygon. Initially you will be prompted to select a transformation.

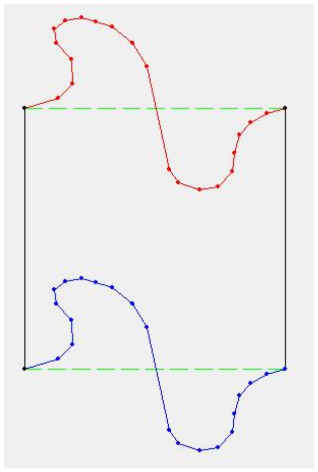
The transformation tools

So to start, click the **transformations** button in the tool bar to expose the tool list, then select one of the **transformation** tools. The **translate** tool is probably the easiest to understand so we will use that as our first example. The editor will prompt you to select a source edge. This particular tool requires that there be a destination edge that is both parallel and congruent to the source edge. Select an appropriate edge. EscherSketch will look for (and hopefully find) the appropriate destination edge. The source edge will now appear in red (thus you can edit it) and the destination edge will appear in blue. If your starting tile was a perfect square created on the rectangular grid, your window should look something like this, perhaps after zooming in:



edit points on the red line
or select a new transformation

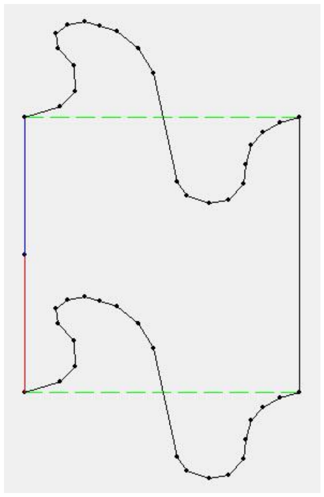
You can now add (by clicking on the red line) and move (by clicking and dragging) points anywhere you like on the source (red) edge. You can also delete points that you already added by right clicking them. Each time you add a point, the editor will create a corresponding point on the destination (blue) line. As you add and move points you will soon notice that the resulting pattern is such that the transformed tile will go back together in the tile editor by simply stacking a copy of itself on top. This will be true no matter how complex the source edge becomes. Your partially completed tile may look something like this:



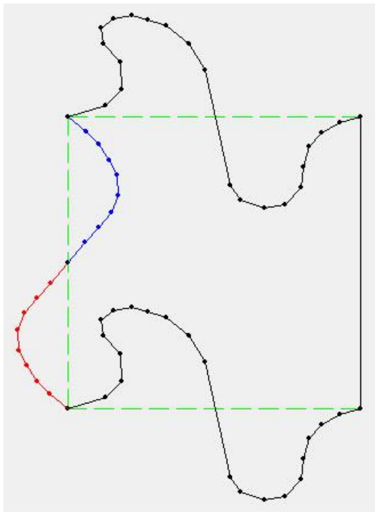
You can leave this transformation any time, perform transformations on other edges, scroll the window or even return to the tile editor and back and subsequently return to this edge to continue your transformation. To return to it, select the translation tool and click any point on the source edge. **EscherSketch** will recognize that there are edits on the destination edge, and because it can't know where those edits came from will ask your permission to proceed. Because you know that the destination edge was generated by the translation tool you can just agree to proceed.

Now let's try a different transformation on a different edge, say the left edge. On this edge we will apply a **center rotation** transformation. Center rotations can be performed on any edge because the source and destination are on the same edge. Select the **center rotation** tool from the transformation menu and the editor will prompt you to identify an edge.

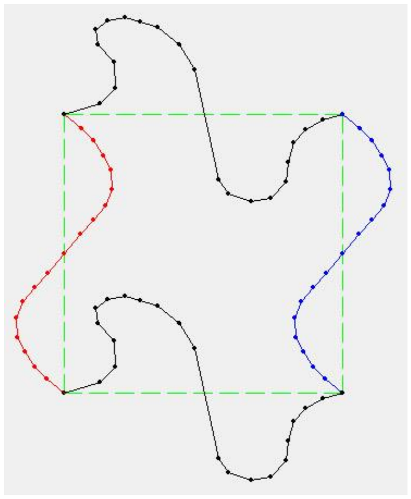
Once you have done that your tile will look like this:



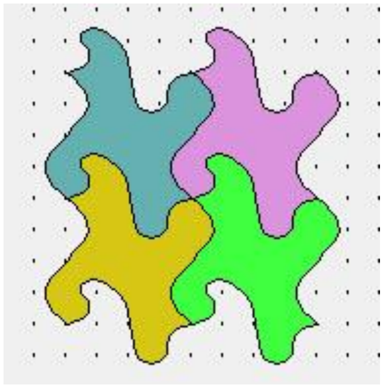
The selected edge will be divided into a source half and a destination half with a center point marked in the middle (the center point can not be moved). Like in the previous example, points can be added and moved on the source half (red) and corresponding points will be automatically added on the destination half (blue). The difference here is that the destination points will be rotated 180 degrees about the center point. After editing, your partially transformed tile may then look something like this:



By inspection we can tell that the center rotated edge will tile nicely with a copy of itself that is rotated by 180 degrees. But we have created a problem; the top and bottom edges will tesselate only with copies that are translated but NOT rotated. Fortunately in this case we can remedy that problem by translating the newly center rotated edge to the last remaining unedited edge on the opposite side of the square. To do this, select the **translate** tool from the **transformation** pulldown. It will prompt you for a source edge which can be satisfied by clicking any point on the center rotated edge we just made. The polygon will then look like this:



At this point we have transformations on all four edges of our square. We are ready to return to the tile editor to give this a try. To do this, click the **return to tiles** command button. The tile editor will reappear and the newly transformed tile will appear in place of the original tile that we started with. Use the **<u>tile</u> copy –m** tool to make replicas of the tile and snap them into place. Because we have not moved the vertices of the original tile they will snap back together **exactly**. Three copies are sufficient to demonstrate the result. After some **<u>tile</u> recolor** operations the image looks like this:



Maybe these look like carousel ponies to you, or maybe fantastic llamas. Whatever they look like, you can return to the transform editor using the **return to transform** command button to further refine them. If you are satisfied with your result you can easily make additional copies using the **<u>select</u> all** tool followed by **<u>group</u> copy -m**.

When you are completely satisfied with your transformed tile and ready to commit to it, return to the transformation editor and click the **exit** button (or the red X on your window border). Once you exit, the tile is restored permantely to a polygonal state and no further transformation editing can take place on that tile. **EscherSketch** will remind you of this and give you a chance to reconsider. But this is the only way to free up the transformation editor for use on a different tile.

Saving partially transformed tiles

Although tiles cannot reenter the transformation editor once it has been exited, you do have the ability to save, and subsequently restore a partially completed transformation editing session. To do this, simply return to the **tile editor** (by clicking the **return to tiles** command button) without exiting the transformation editor and save your state (using the **<u>files</u>**

save state command). Your active transformation editing session will be saved along with your other tiles (in the .tes file) and will be available to return to the transformation editor when you reload the file. **EscherSketch** will not allow you to load a new .tes file while the transform editor is active.

Description of transformation tools

To select a new transformation, or the same transformation on a different edge, click the **transformations** button and select a tool from the list. (Reselect it even if it already appears to be selected.)

Selecting the **translate** tool (discussed above) prompts you to select a source edge. The appropriate destination edge (that being the one that is both parallel and congruent) is automatically identified by **EscherSketch**, if it exists. If the destination edge already contains points from a previous edit, you will receive a warning message and the edge in question will appear in magenta. If you choose to proceed, the desination edge will be updated with a tanslation of the source edge, recolored in blue. The source edge wil be colored in red, thus enabling edits on that edge. Each edit on the source edge will be automically translated to a corresponding edit on the destination edge. This tool should be used only on polygons that are intended to be reassembled by translation (**copy –m**).

The **center rotation** tool (also discussed previously) is the most versital tool in the sense that it can be applied to any edge of any polygon. After selecting this tool you will be promted to identify an edge to transform. If the edge contains edits that were NOT generated by a previous center rotation operation you will be warned and asked for permission to proceed. The selected edge will be divided into two equal halves, one which is the source half (in red) and the other the destination half (in blue). Edits on the source half will create corresponing changes on the destination half. In general, this transformation should only when the file tile is intended to be reassembled by rotation of 180 degrees. As we have seen in the example, there are exceptions to this rule.

The **vertex rotation** tool prompts you for a source edge. The destination edge must share a common vertex with the source and must be congruent. Like the translate tool, **EscherSketch** will attempt to find an appropriate destination edge. In some cases a seemingly perfect destination edge will be overlooked. This is because the vertex rotation is only capable of transfoming in one direction (either clockwise or counterclockwise depending, oddly, on the direction that the original tile was created). If the destination edge is not recognized, reselect the vertex rotation tool again and this time select the edge on the opposite side of the vertex to be the source, then make your edits on that edge. Edits on the source edge will be automatically transferred to the destination edge, but rotated about the common vertex by the angle between the source and destination edges. This tool should only be used on tiles that are intended to be reassembled by rotation about the same angle as the vertex rotation, and only on tiles that have rotational symmetry about the same angle.

The **glide reflection** tool first prompts you to identify a source edge, and then again to identify a desitnation edge. The destination edge must be congruent to the source edge. Like all of the other transformation tools, a warning will be issued if the destination edge (temporarily colored in magenta) already contains edits. Edits to the source edge will be automatically transferred to the destination edge after appying a glide reflection. This tool should ony be used on tiles that are intended to be reassembled using a combination of reflection about the axis of the glide reflection and translation.

The **reflect** tool is unique in a number of ways. First, it takes its effect immediately upon selection of the tool. Second, it is the only tool that changes the shape and size of the starting tile (it actually moves the location of the tile's vertices).

Finally, it must be the last transformation applied before returning to the tile editor. It works as follows: **EscherSketch** first tries to identify a legal reflection edge. The only legal reflection edge is one that is both vertical and the rightmost edge of the starting tile (the starting tile must contain no points to the right of the reflection edge) . If a legal reflection edge exists but already contains edits, a warning will be issued that presents you an opportunity to either delete those edits or abort the transformation. These rules may seem highly restrictive, but after a few applications you will realize the tool is perfectly general. (however, the starting tile may need to be rotated or reflected prior to entering the transformation editor to ensure that an appropriate reflection edge exists). Once a tile has been reflected, no further edits can be made on any edge. This ensures that the tile is a true geometric reflection . For this reason an **unreflect** tool is provided. Subsequent attempts to make edits to any edge after a reflection has been applied will initiate a warning with an offer to unreflect the polygon. This transformation should be applied only to tiles that are intended to be reassembled using a combination of reflections and translations.

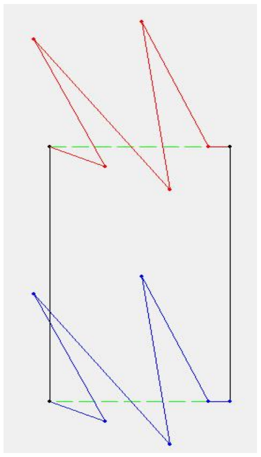
The **views** pulldown contains several checkboxes that allow you to control aspects of the editing pane view.

The **curve mode** tool activates curve editing which will be discussed at length in the next section.

Curve editing

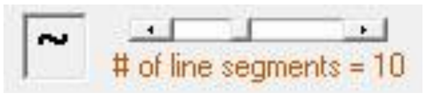
We have seen in our previous example that arbitrarily complex shapes can be applied to edges if we have the patience to add and move a large number of points. This process can be tedious and time consuming, and without extreme care can lead to curves that are jagged or otherwise misshapen. To simplify this process the **EscherSketch transformation editor** provides a rich set of curve editing tools.

If you have never used a curve editor before it will take you a little practice, but you will soon find you are able to create smooth curved edges of any type you want with very few editing steps. Once again I will explain this with an example.

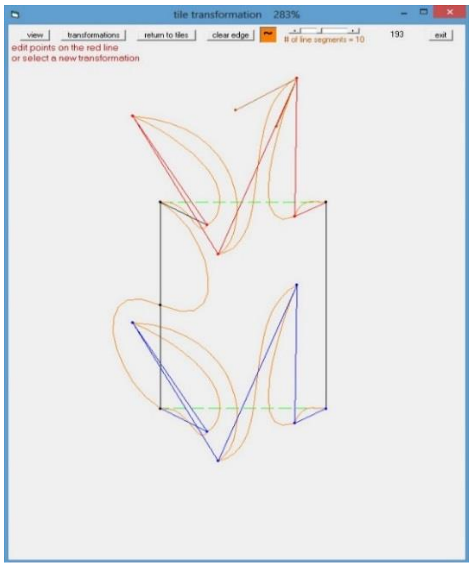


For this example, my target tile will be some sort of creepy tessellating bug. I will start by creating a vertical rectangle in the tile editor, then bring it into the transformation editor by clicking the **transform tile** button and selecting my rectangle. Once again I will use the **translate** transformation and apply it to the top edge of my rectangle. At this point I will rough in some crude shapes representing legs or antennae. My tile then looks like this:

I have added only 5 points to the source edge and that should be sufficient to accomplish what I want to achieve. To get started adding curves to the red edge I will activate the curve editor by clicking on the **curve mode** button in the tool bar (the button marked with a "**~**"). The button will

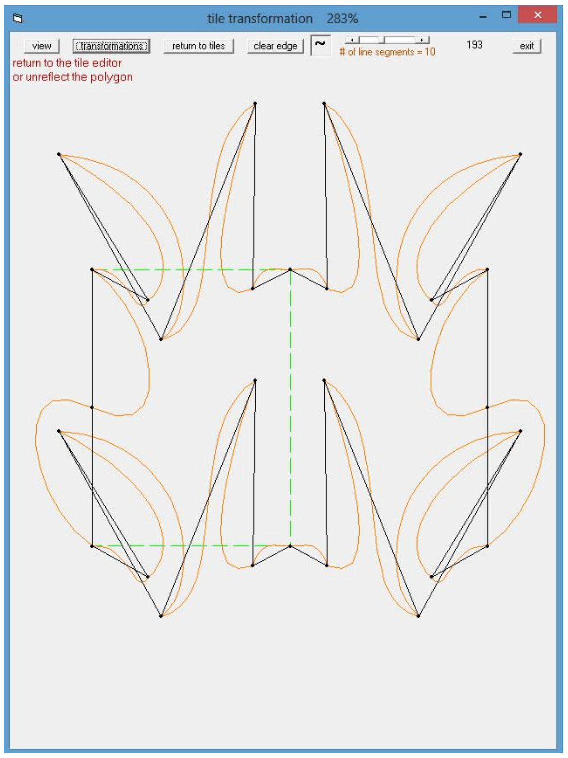


light up in orange to remind you that you are in curve editing mode. Once in the curve editor you can do most of the things you can do otherwise, such as select a new transformation (there must always be an active, or red edge to apply curves), return to the tile editor and back, move (red) vertices, resize your window or zoom in or out. The only things you cannot do is add or delete points. You must exit the curve editor to do that by clicking again on the **curve mode** tool button. You will also see a scroll bar on the tool bar which we might use later to smooth our curve(s). We are now ready to initiate some curves on the source line. To do this simply (while the curve editor is active) click on each of the red points in turn. As you do you will see an orange curve appear for each line segment. This curve becomes the outline of your tile when you return to the tile editor. After clicking all of them your tile will look like this:



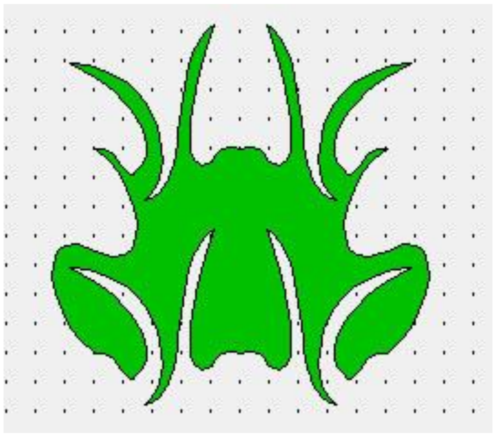
You will also notice a pair of *control points* in brown. These are points that you can move to alter the shape of the curve. I will offer a more detailed description of these control points shortly, but for now it is best just to experiment with them. Click and drag a control point and you will see a corresponding change in the orange line nearest the control point. A few things should become apparent: no matter where you move the control points, the resulting curve (in orange) will always pass through a red vertex, thus these vertices become *anchor points* for our curve. Also, the curve will always pass through the anchor points at a tangent angle determined by the control point. To help visualize this a brown tangent line will always be drawn from the anchor point to the control point. Finally the control points themselves act as "magnets" that tend to draw the orange curve toward that point. Play with this a little and the action of the control points on the curve will become very natural and intuitive, and most of all predictable.

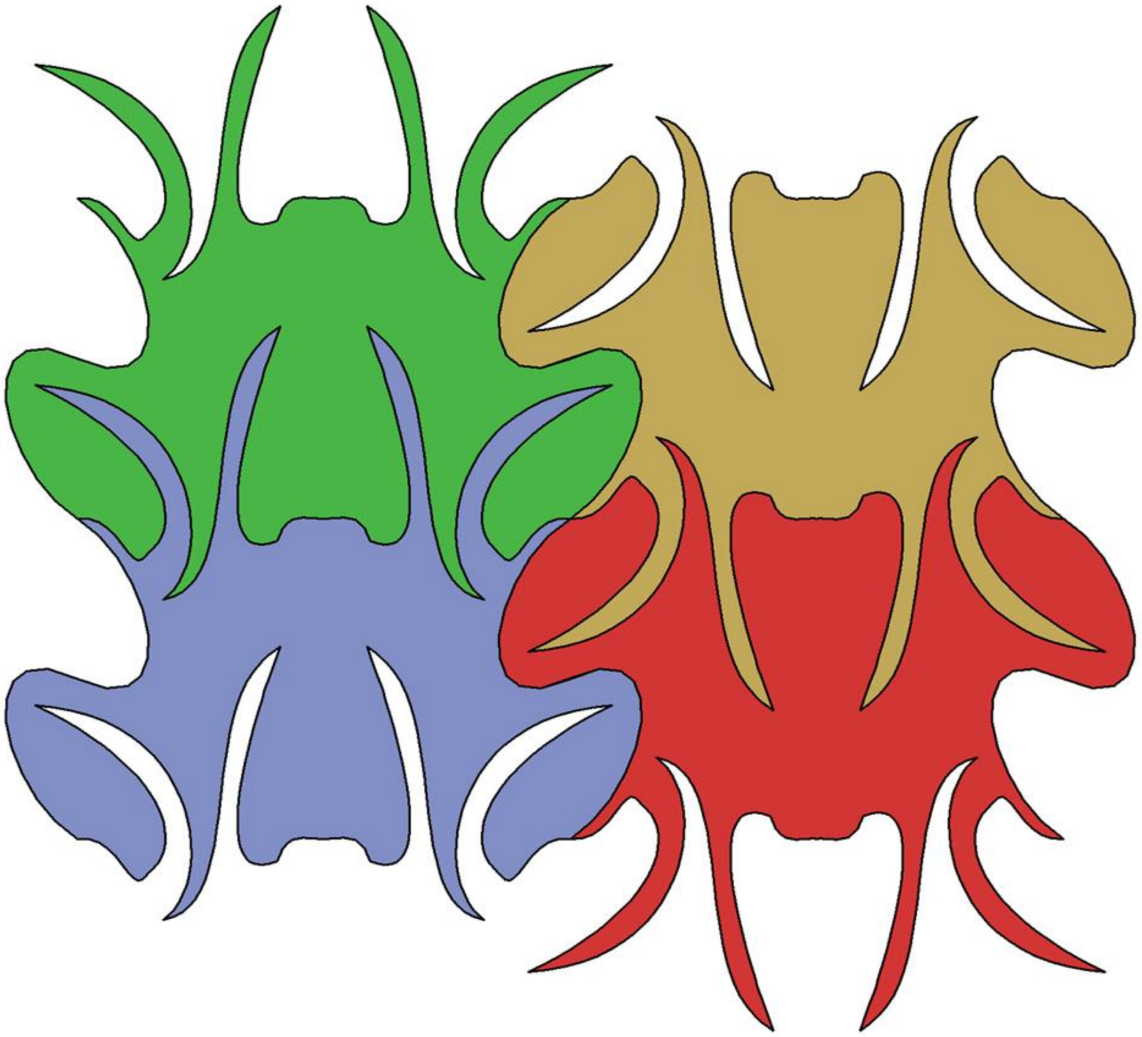
We are now ready to complete our bug using only the five anchor points we already established:



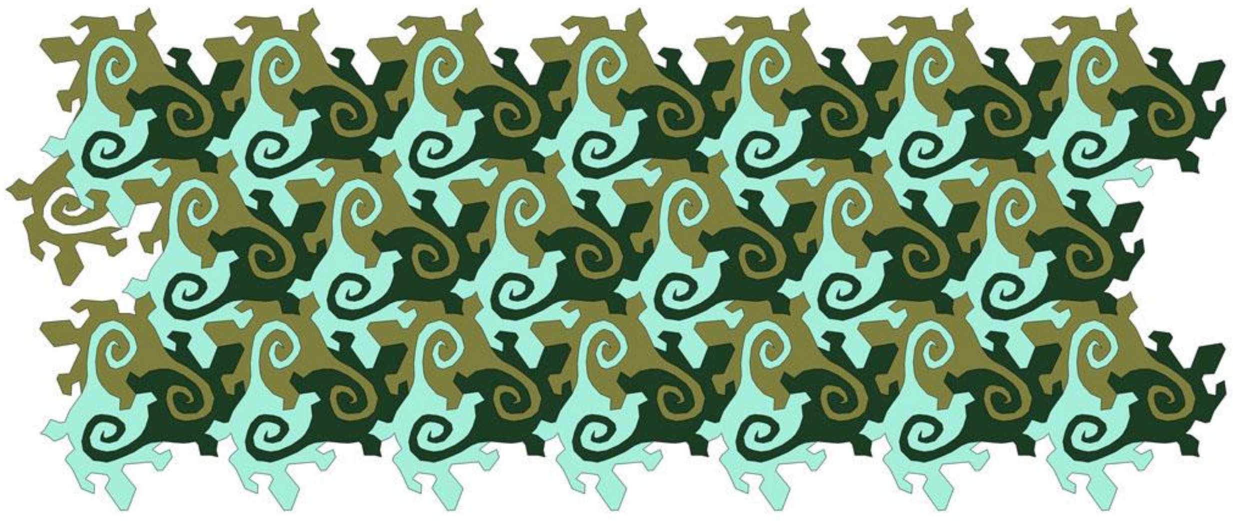
After adjusting all eight of the curve control points and making some minor moves of the anchor points I added a center rotation transformation to the left edge of the rectangle. All of this was done without ever leaving the curve editor. A Few words on technique are in order here: To bring up a different set of control points for editing, simply click on a new anchor (red) point to see the control points associated with that vertex. You can also click on any red line to expose the two control points on either side of that line. You can move any red anchor point simply by clicking and dragging it just as you would outside of the curve editor, and the associated control points will move with the anchor point. You can return control points to their default (initial) positions by right clicking any anchor point or red line segment.

To complete my bug I will finally apply a **reflection** transformation which provides me an identical second half that is reflected about the right edge of the rectangle. We can now return to the tile editor to see our finished tile:





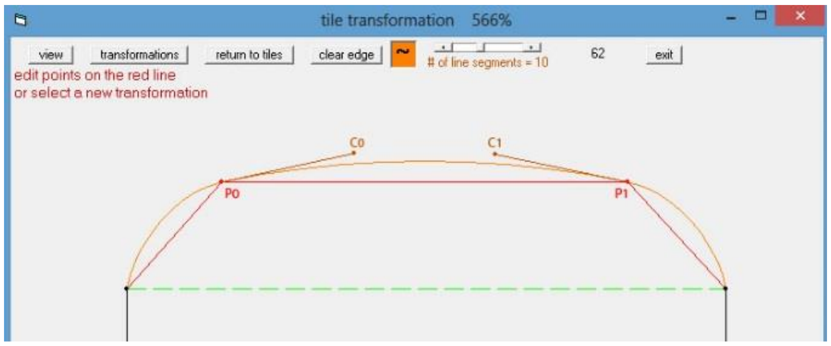
A picture of our tessellated bug with expanded detail shows the smooth curves and sharp points created by the curve editor



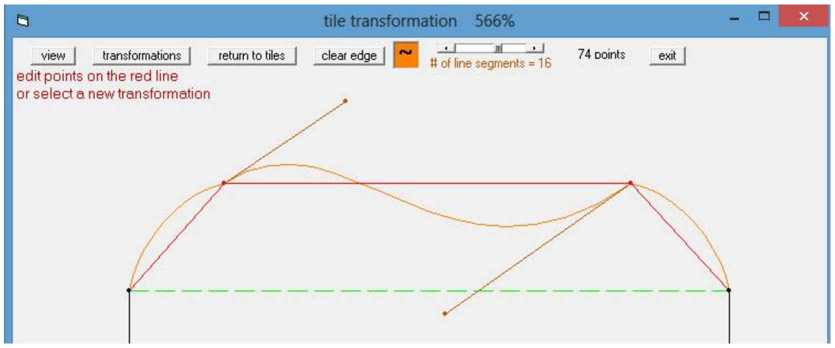
Turtles with spiraling tails created from a hexagon using vertex rotation transformations

Detailed Description of the Curve Editor

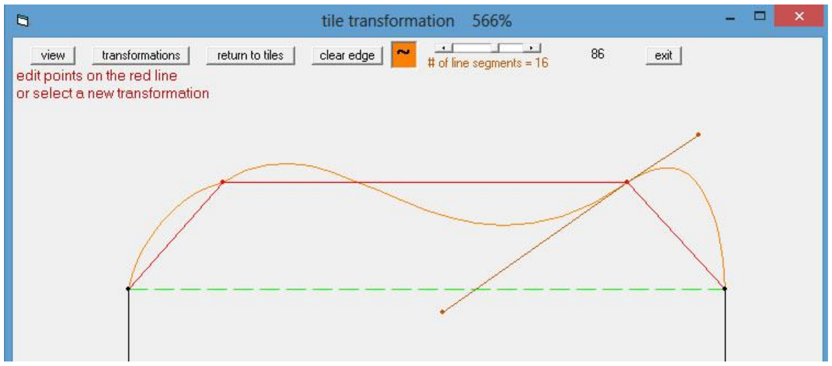
The type of curves created by **EscherSketch** are commonly known as *cubic Bezier curves*, named for Pierre Bezier who developed them for use in CAD for the automobile body design industry. A cubic Bezier curve is defined by four points: two anchor points which are the endpoints of the curve, and two control points that generally do not lie on the curve. To display the control points on a line, start the curve editor. You must have an active transformation with an editable (red) edge. Click the red line. This sets initial locations of the control points (C0 and C1) at locations that ensure a smooth curve through the anchor points (P0 and P1). The curve starts at point P0 and proceeds to the right at a tangent angle defined by control point C0 (that is, parallel to the brown line from P0 to C0). The curve (in orange) is drawn toward the control points as if by magnetism. The curve terminates on point P1 at a tangent angle determined by control point C1.



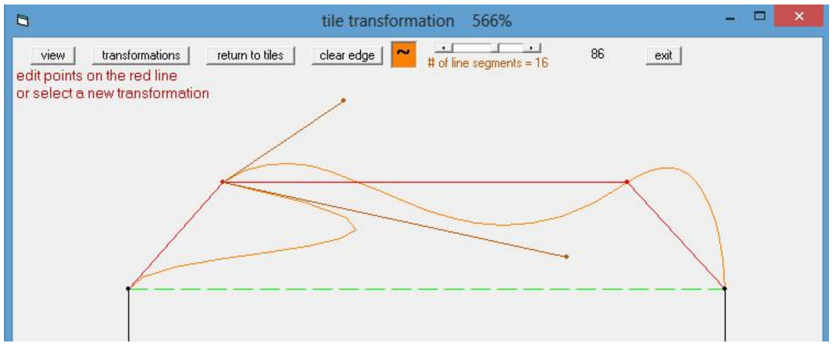
Control points, represented by brown dots can be moved by clicking and dragging them. The curve will follow the control points as depicted by the frame to the right.



Control points can also be displayed by clicking any red anchor point (vertex), which will show the control points on either side of the vertex as depicted on the right. Anchor points can also be moved by dragging them, and the associated control points will move along with them. In this case I have moved a control point to adjust the tangent angle on the rightmost curve segment to match that of its neighbor to its left, thus creating a smooth transition through their common anchor point.



In the last frame I clicked on the other (leftmost) anchor point and adjusted a control point to pull the leftmost curve segment in thus creating a concave shape similar to the bug's antenna of the previous example.



Control points can be reset to their default (Initial) locations by right clicking any red vertex or line segment (which deletes the existing curve) then clicking on the same point (which reestablishes a curve that resembles the initial one we started with).

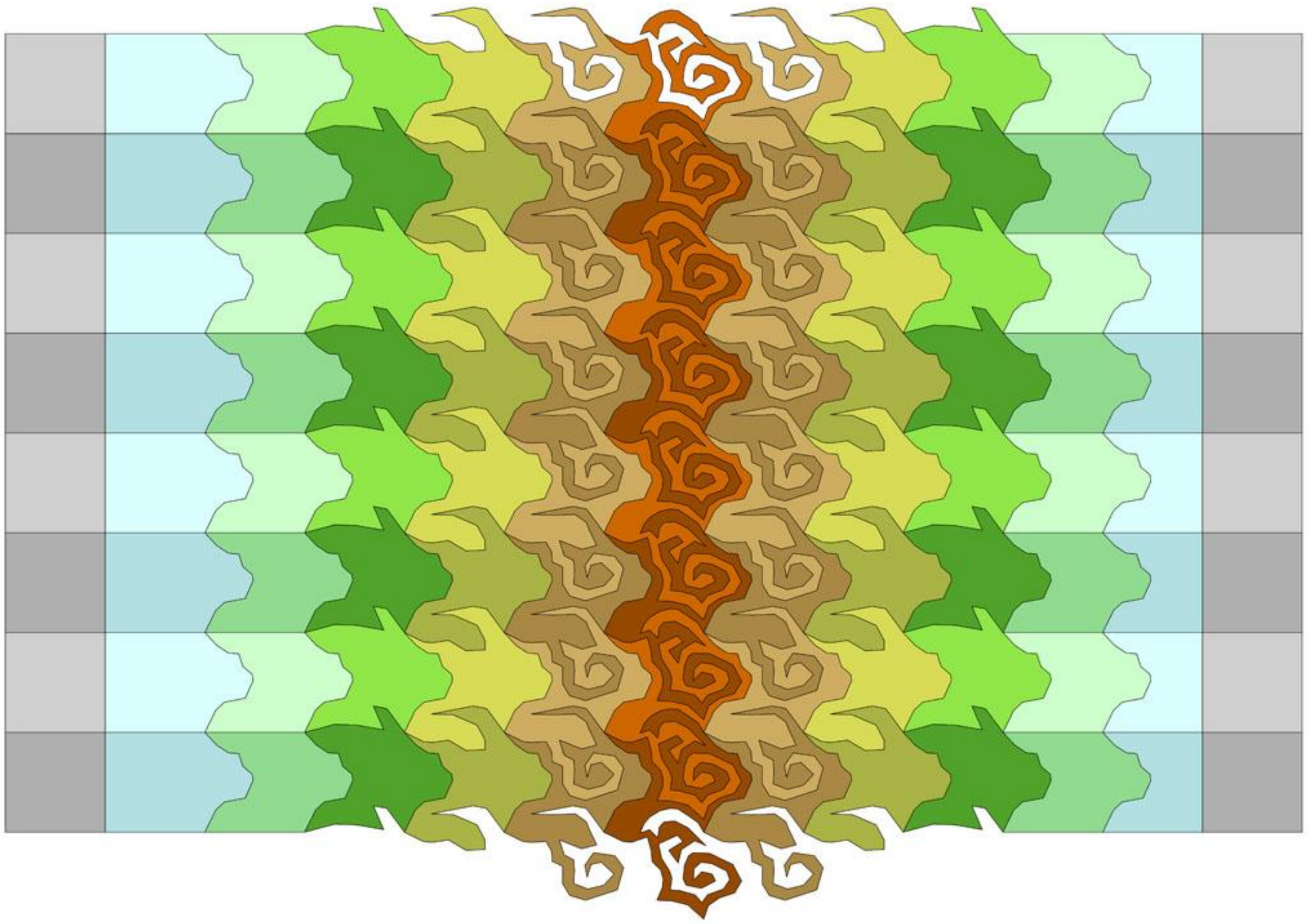
New vertex (anchor) points cannot be added to the drawing during curve editing, nor can existing points be deleted. To do this, temporarily exit the curve editor by clicking the orange curve command button in the toolbar.

As you can probably see, curves are actually constructed from a series of short line segments. By default the number of segments is set to 10. For some more complex curves you may want to increase the number of segments to get a smoother curve. To do this click on the red line segment associated with that curve and adjust the scroll bar in the tool bar (this scroll bar is only visible while curve editing is active).

Note that all along we have been clicking on only red lines and points to select the control point we wish to edit. These red objects clearly associate with curves and control points, but **EscherSketch** will NOT respond to any attempt to select an orange curve. (Do you notice a common theme here between all components of the transformation editor?)

Finally, the right mouse button performs a handy function in the curve editor. While a vertex control pair is visible (exposed by clicking a vertex), the pair can be moved in tandem by dragging one of the points with the right mouse button. The opposite point will be locked to the point you are dragging and will move in the opposite direction at the same angle.

All of this may sound a little complicated at first, but give it try and you will find it quickly becomes easy and intuitive, and it leads to great results in your final tile.

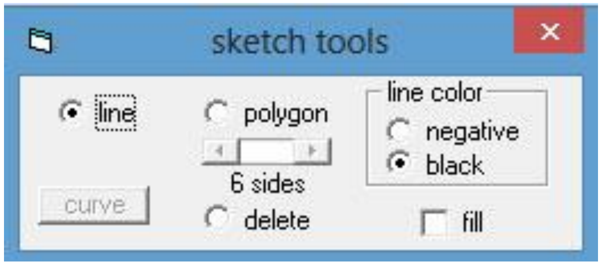


A creative tessellation formed using a combination of tile editor and transformation editor tools

Decorating your tile

There is one last set of tool functions available in the transformation editor, that being the **sketch** editor. This allows you to draw rudimentary lines, curves and polygons on your tile, say to give your tile creature eyes, scales, feathers or whatever is appropriate to make it more artistic and identifiable.

Once you have achieved the basic shape of your complex tile is a good time to start sketching in some details. To do this click on the **sketch** button on your toolbar. A small window will appear on top of the transformation window that looks like this:



This provides you the means to add lines, *polylines* and regular polygons right on top of your tile. The default tool is the **line** tool. This is similar in many respects to creating a new tile in the tile editor, by simply clicking points on your tile. As you click points, new segments will be added to your line drawing. Like the tile editor, you can back out points by returning to the last point and reclicking it. To terminate your line, right click anywhere in the editing pane. Note that the line need not be closed. Once your line is terminated, the curve button will be enabled. This is your one chance to add a curve to each of your line segments (but it is by no means necessary to take this step if you intended to present straight lines). Click this button and a single control point will appear as a brown dot on the center of each segment. Move these points around by clicking and dragging to create curves. Like the curve editor, Bezier curves will be created, but this is a little bit simpler because there is only one control point per segment rather than two. When you have completed adding curves, reclick the **curve** button and the curve will be set.

Alternatively you could have created closed a *polyline* by returning to and clicking your starting point rather than right clicking to terminate the line. This creates a polygonal shape that can be filled with color. Just like the regular line, the **curve** button will be enabled to allow you to curve the edges of your polyline. Modify the curves and reclick the **curve** button as before. Selecting the **polygon** tool lets you create a regular polygon. When selected, the scroll bar beneath the tool option will be enabled and you can select any number of sides from 3 (a triangle) to 24 (nearly a circle). To create your polygon, drop your mouse button on the desired center point and drag outward. When you release the mouse button the polygon will be set. Just like the polyline, the **curve** button will be enabled to allow you to curve the edges of your polygon if desired.

Once a shape has been set there is no way to go back and reedit it. But you do have the means to delete it and redraw it if you want. Select the **delete** option and you will see a red handle point appear on each object that you have drawn. You can delete any object by clicking on its handle point.

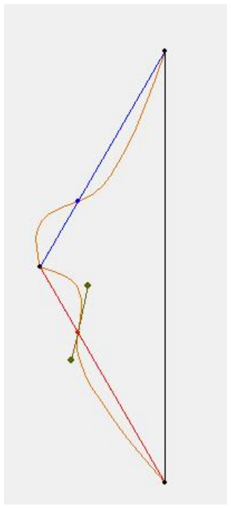
The final set of capabilities in the sketch editor are the **fill** checkbox and the **line color** options. These should be thought of as attributes of the lines or shapes you create with the other tools and must be set up <u>before</u> you enter the shape. Checking **fill** will cause any polygon or polyline to be filled with a color that is the *color negative* of whatever tile it belongs to. If this box is not checked the shape will have a transparent center. The **line color** options refer to any lines (or curves) you draw, and to the outlines of any polygons or polylines. There are only two options, either black or the negative color of whatever tile it is drawn on. Negative colors are represented by grey in the transform editor, but will show as colors once returned to the tile editor.

You can draw over any polygon even if it is filled, but remember that the order they are sketched is the order that they will be drawn, meaning newer sketches hide older ones beneath them.

It is also possible to create the same complex tile with two or more different sketches. Say for example you have a tessellating butterfly and you want two or more different wing markings on them. Do this as follows: Create the butterfly and sketch in the first set of markings. Then return to the tile editor and make a second copy of your tile (using the **<u>tiles</u> copy –m** tool). Then return to the transform editor, select the **sketch** tools and delete the first set of markings, then sketch another set. Return to the tile editor again and make a copy. Repeat as many times as necessary to create the number of butterfly variations you desire.

Examples

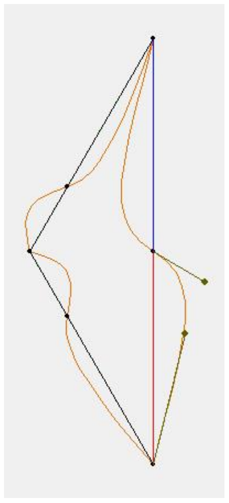
Example 1: Hummingbird

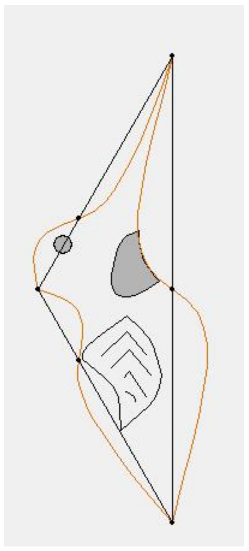


This is a simple example. You can start with a stock tile from the primitives. With the hexagonal grid in view (**<u>grids</u> hexagonal**), load primitives (**<u>files</u> load prims**) and click on **<u>transform tile</u>**. Click on the 120 degree isosceles triangle which is the lime green triangle in the second row.

Once in the transform editor, select the vertex rotation from the **transforms** pulldown. Add a single point as indicated on the image to the left. Now apply a curve to the source edge by clicking on the curve tool ("~") then selecting the red vertex you just added. Adjust the curve control points to create something like you see in the picture.

On the remaining edge you will apply a **center rotation** transformation. This will add an unmovable point on the center line and it will not be necessary to add any more points. (Since you do not need to add points it is not even necessary to exit the curve editor). Apply a curve to the source edge by clicking anywhere on the red half of the line an adjusting the control points until you see something like you see in the next picture.



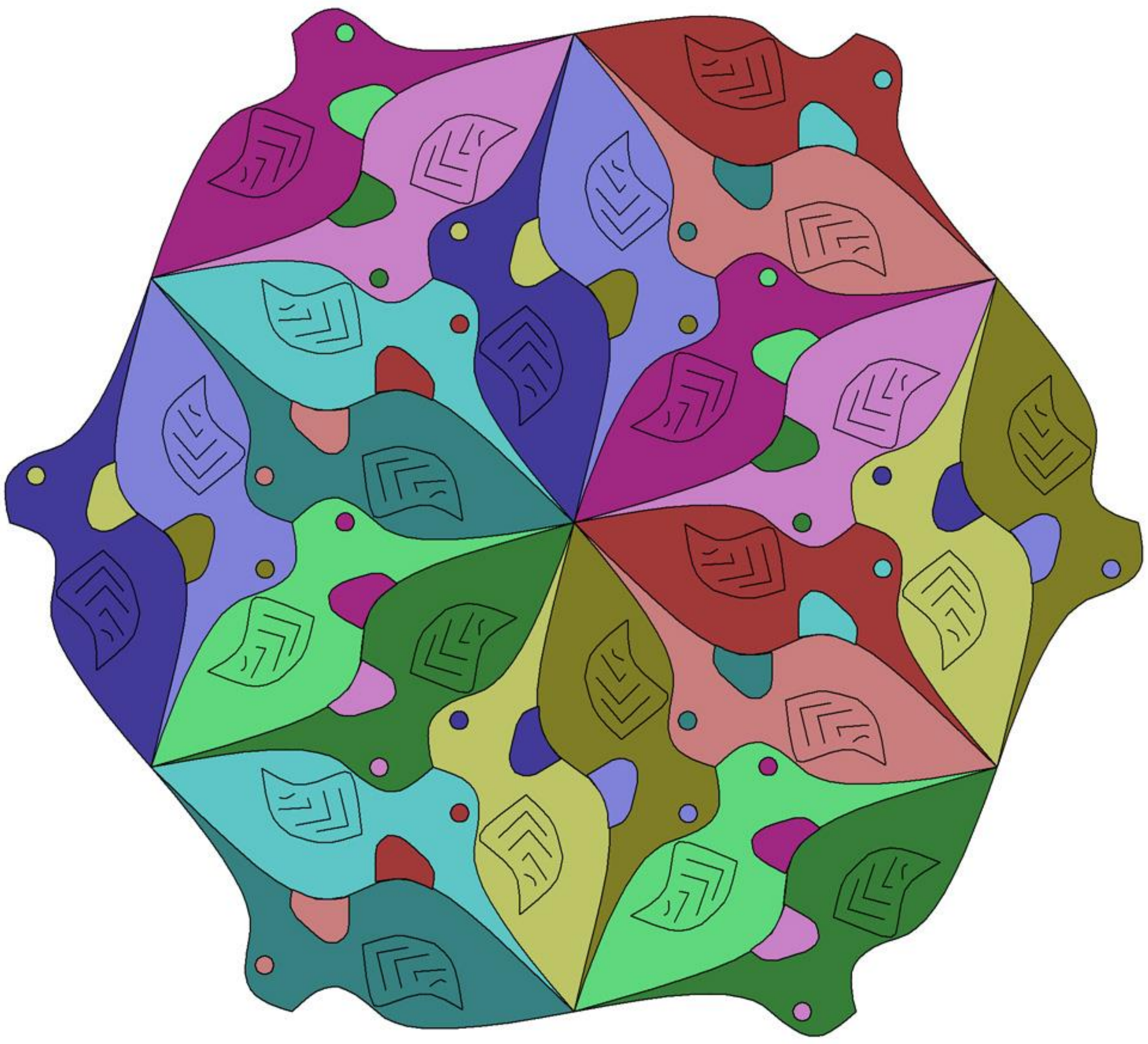


Now it remains to sketch in a few details to give your hummingbird some character. Click on the **sketch** button to start the sketch editor. In my example I used the **line** tool to create the outline of a wing. I completed a polyline by returning to the starting point, then immediately clicked on the **curve** tool (in the sketch control panel) to create the final shape. I sketched in the indication of some feathers on the wing using the line tool and right clicking each time to complete each of the two segment lines. The characteristic throat crest was created with the line tool with the fill option checked. To match the contour of the bird's throat, I included a few points directly on the orange line. Then the sketch curve tool could be used to dial in a throat crest that hugs the bird's contour. I added the eye with the **polygon** tool with a large number of sides, (the number of sides can be adjusted with the scroll bar once the polygon tool has been selected) again with the fill option enabled.

With the hummingbird complete it is now time to return to the tile editor to assemble your tessellation. First you will need to eliminate any tiles on the editing pane that will not belong to your final tessellation. There are many ways to do this, but if you left the transformation editor active you can use a simple trick to make this easy: Use **select all** followed by **group delete**. This works because **EscherSketch** will not delete a tile that is currently active for transformation.

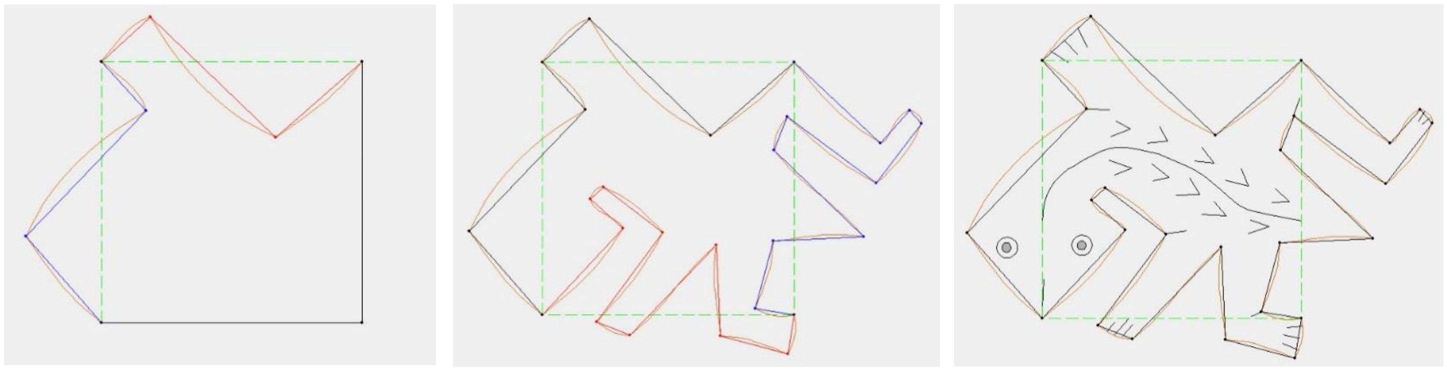
Since we created the edges using all rotation transformations it make sense that we would reassemble it using rotated copies (**copy –m** and **rotate –rm**). Its good practice not to use the original tile in your assembly, especially if you might return to the transformation editor. This is because subsequently returning from the transform editor will overwrite that tile in its original position and rotation, thus messing up your tessellation (of course you could fix this). Instead, build your tessellations using all copies of the original tile in a separate area of the editing pane.

It's fun to figure out for yourself how to assemble your tessellation, it's a little like building a puzzle. You will notice as you recolor your hummingbirds that the filled areas in your sketches change as well, and will always be the negative of the hummingbird's fill color. My tessellated hummingbirds look like this:

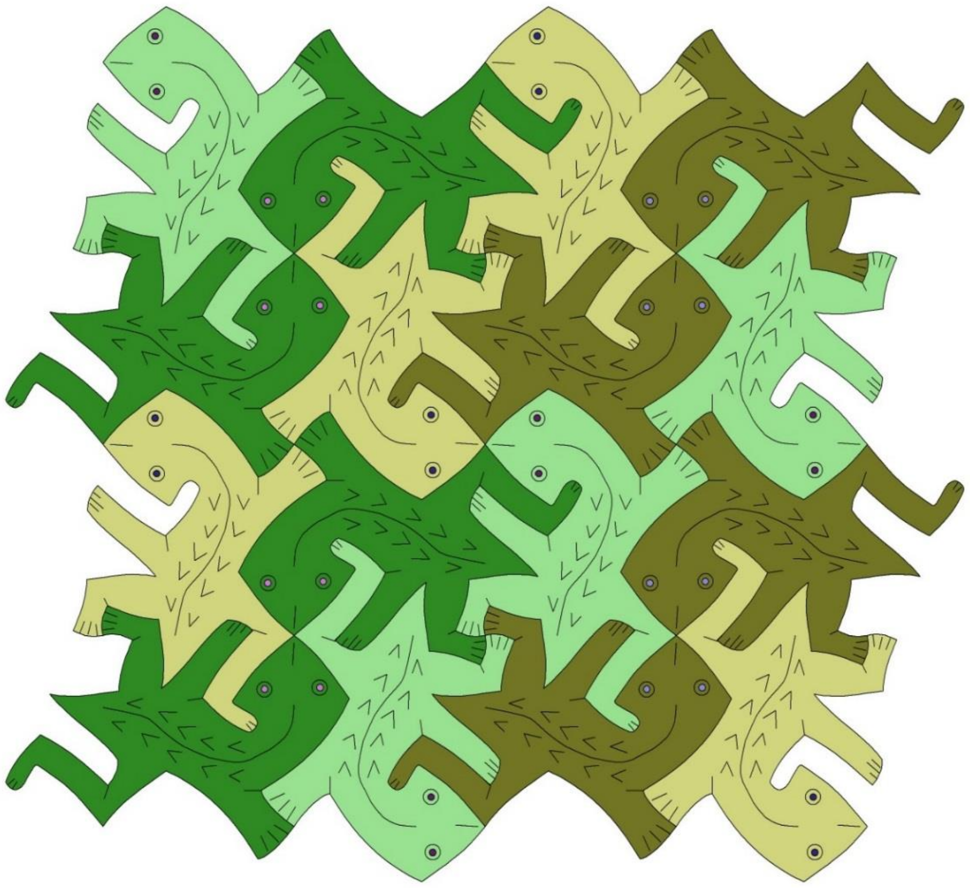


Example 2: Creeping lizards

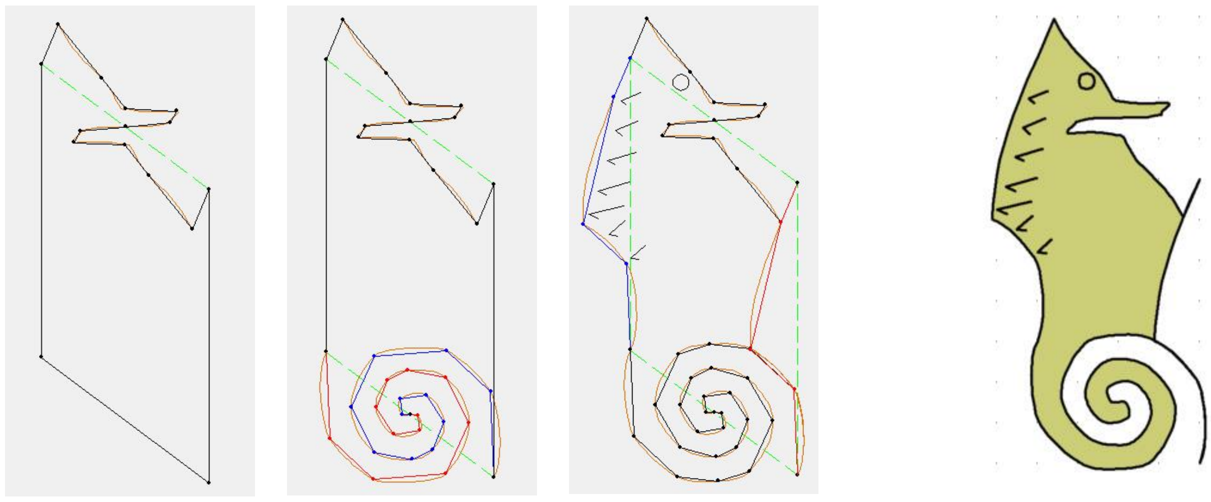
This tessellation starts with a perfect square in the rectangular grid. Squares make good starting points because they can accommodate ALL of the possible edge transformations (as do perfect hexagons and equilateral quadrilaterals). This one uses all vertex rotation transformations.



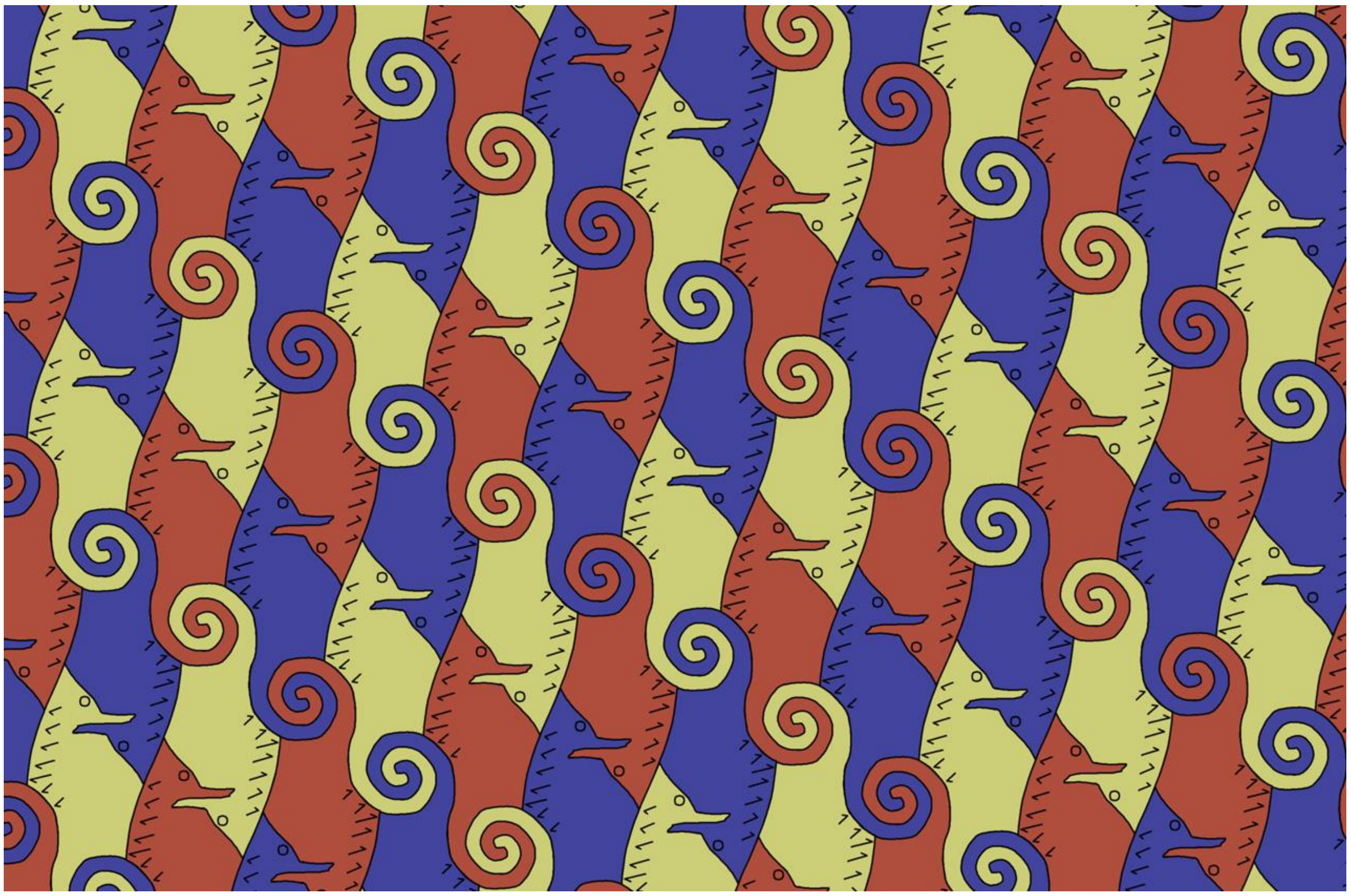
In the first frame, apply a vertex rotation from the top edge to the left edge (or from left to top depending on how the square was created). In the second frame apply another vertex rotation involving the remaining two edges. For me this took a few trips back and forth to the tile editor until I was satisfied with the shape. (Don't forget you can always return to an edge to refine it, but be careful not to overwrite an edge you did not intend to overwrite). Then a few sketches will complete the lizard. Back in the tile editor this tessellates using only copy and rotate commands which is not surprising considering the way the edges were transformed.   My tessellation looks like this:



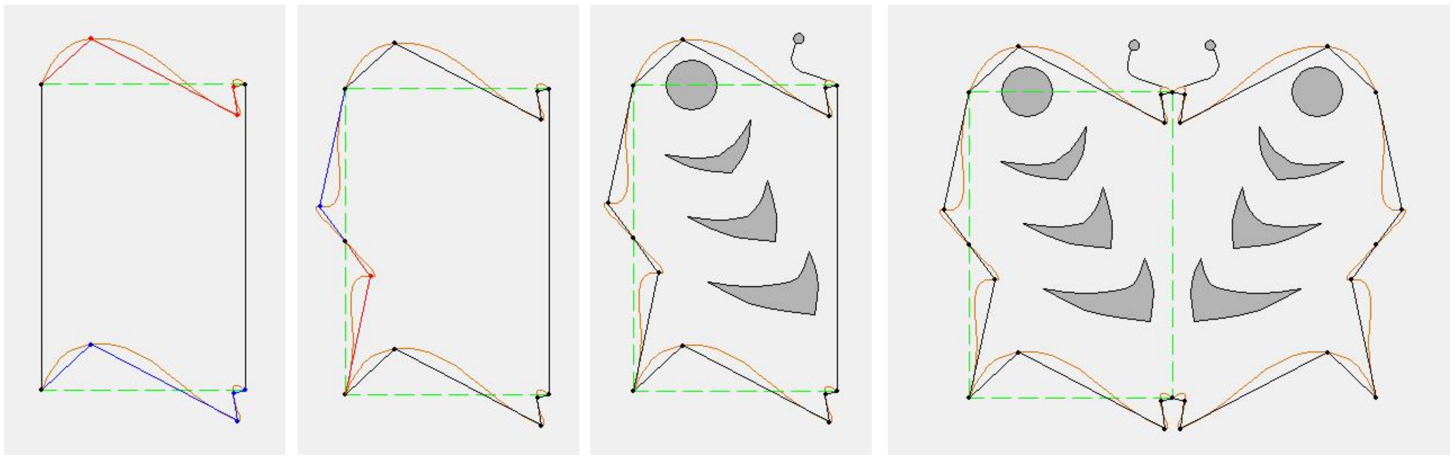
Example 3: Escher's seahorse



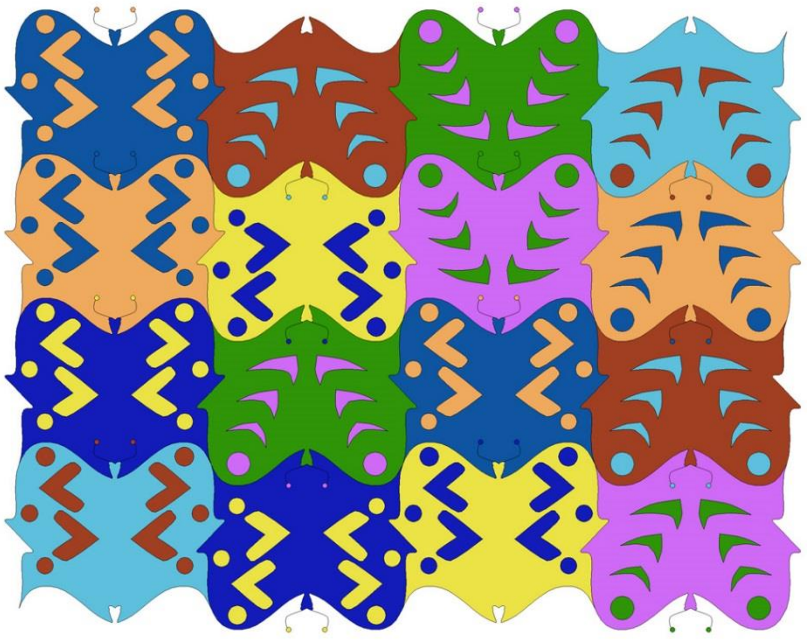
The seahorse is a little harder to figure out because the vertex points of the starting tile are not apparent in the final figure. But we can tell by inspection of the head and tail that these edges were created using center rotation transformations along the top and bottom of a parallelogram. The two vertical edges need to be completed using a translation, but they seem to create unwanted appendages on the top and bottom of the right edge. The solution is to create points on the right vertical edge that fall on top of corresponding points on the top and bottom edges. This effectively hides the anomaly in a line that will be hidden in the tessellation by the coincident right edge of its neighbor. Take a look:



Example 4: butterflies

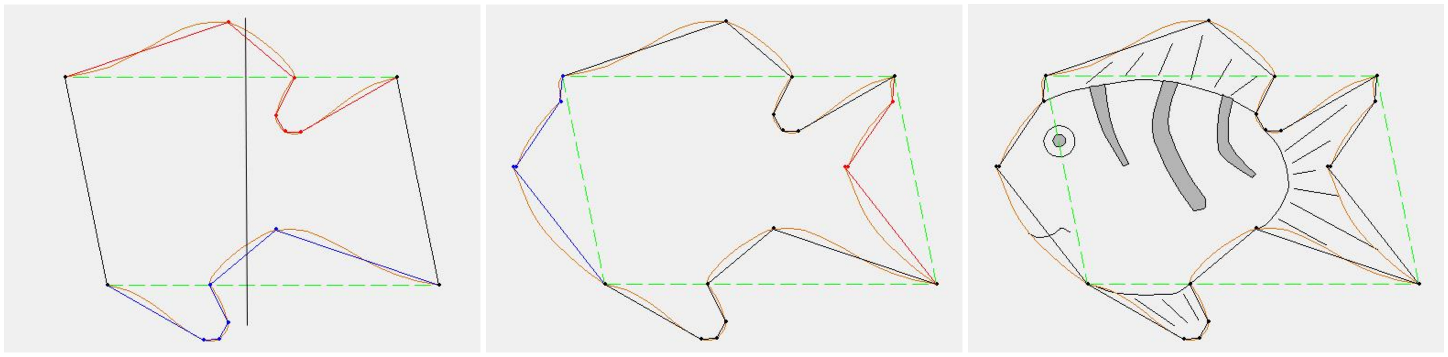


This tessellation is fairly easy to make and uses some of the **EscherSketch** features touched on earlier. The first transformation is a translation from the top edge to the bottom edge. Next we apply a center rotation to the left edge leaving the right edge unedited. After a few sketches using filled polygons and polylines we reflect the image (**<u>transformations</u> reflect** tool) about the vertical right edge; the sketches reflect along with the edges.

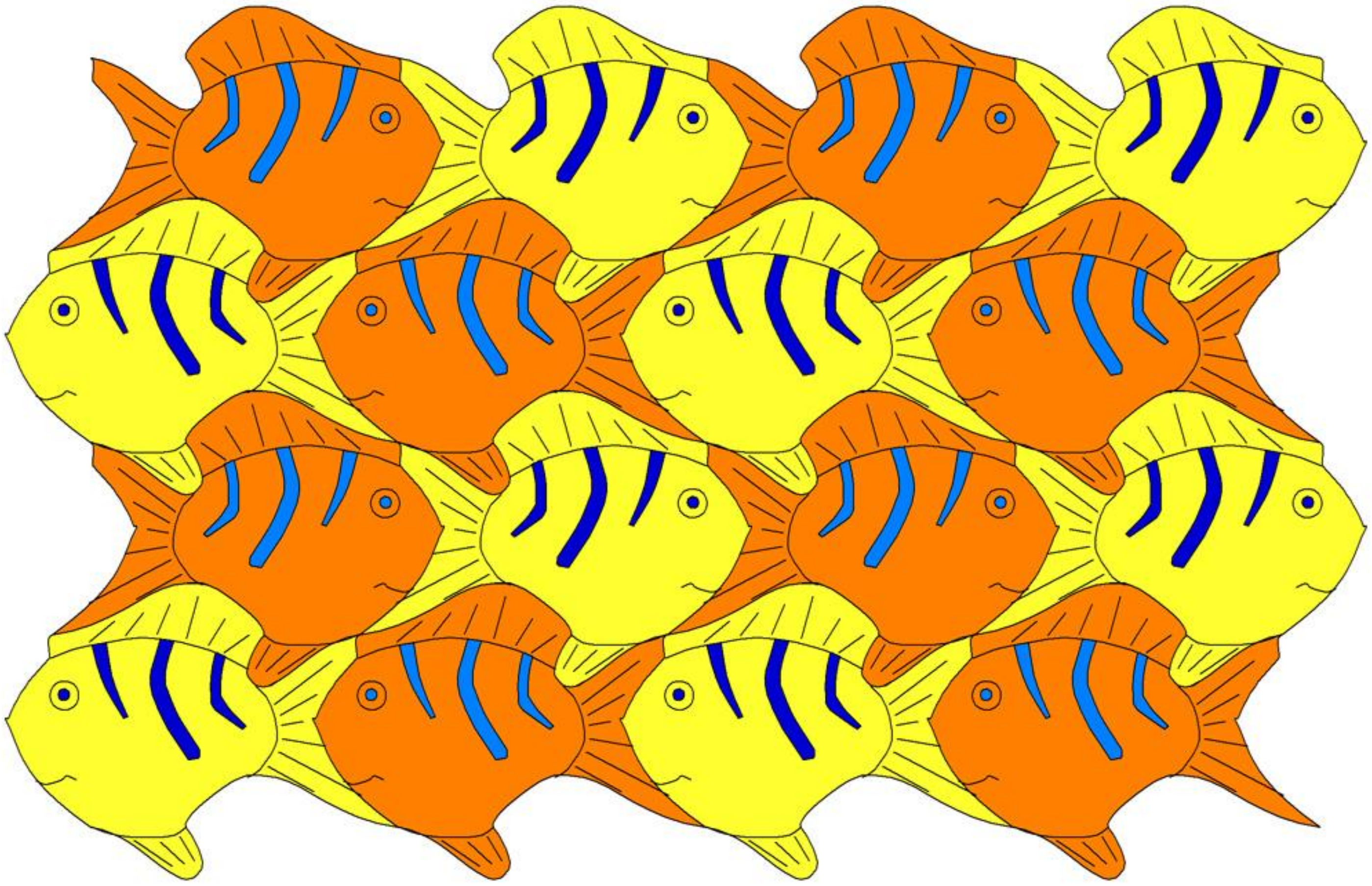


The tessellation goes back together with a combination of 180 degree rotations and translations. There are a few things to take note of: First the antennae were sketched outside of the main body of the butterfly. They show on top of the neighboring butterfly in the tessellation. Sketches will always draw on top of the image. Second, there are two distinct wing patterns in this drawing. That can be done as follows: Complete the butterfly with the first set of sketches then return to the tile editor and make a copy (**<u>tile</u> copy –m**). Then return to the transform editor, delete the first set of sketches and create new ones (if the tile is reflected, as is the case in this example you will need to unreflect it using **<u>transformations</u> unreflect** before these modifications can be made). Once that's complete, re-reflect it and return to the tile editor and make a copy of it. This procedure can be repeated as many times as necessary to create the number of different patterns you desire. The last thing to make note of is the use of inverse color. As we expected, the filled polygons in the sketch are the inverse, or "negative" color of the wings. But some of the butterflies are the exact negative color of another butterfly. This is accomplished using the **color chip**. Click the chip to acquire a color from a filled polygon sketch (by simply clicking a point of the color you desire), then use the **<u>tile</u> recolor** tool to apply that color to another butterfly. So for example we have light blue butterflies with dark red markings and dark red butterflies with light blue markings.

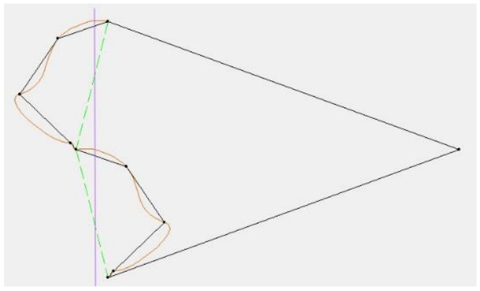
Example 5: clown fish

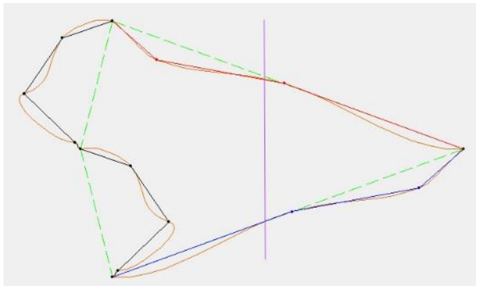


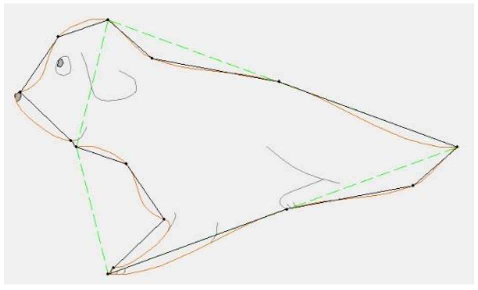
In this example we use the glide reflection transformation. The glide reflect tool differs from most of the other tools in that it requires you to select a source edge, then prompts you to select a destination edge. The only requirement for the destination edge is that it be congruent to (the same length as) the source. They need not be either parallel or adjacent. In this example we started with a parallelogram and applied a glide reflection from the top edge to the bottom. The transformation works as follows: points on the source edge are reflected about a line that pass through the center of the source and is perpendicular to the line from which the source points are derived (the edge of the starting tile). The reflected points are then translated (copied and moved) to the destination edge. Not surprisingly, our tessellation goes back together using a combination of reflected and translated copies of our fish:



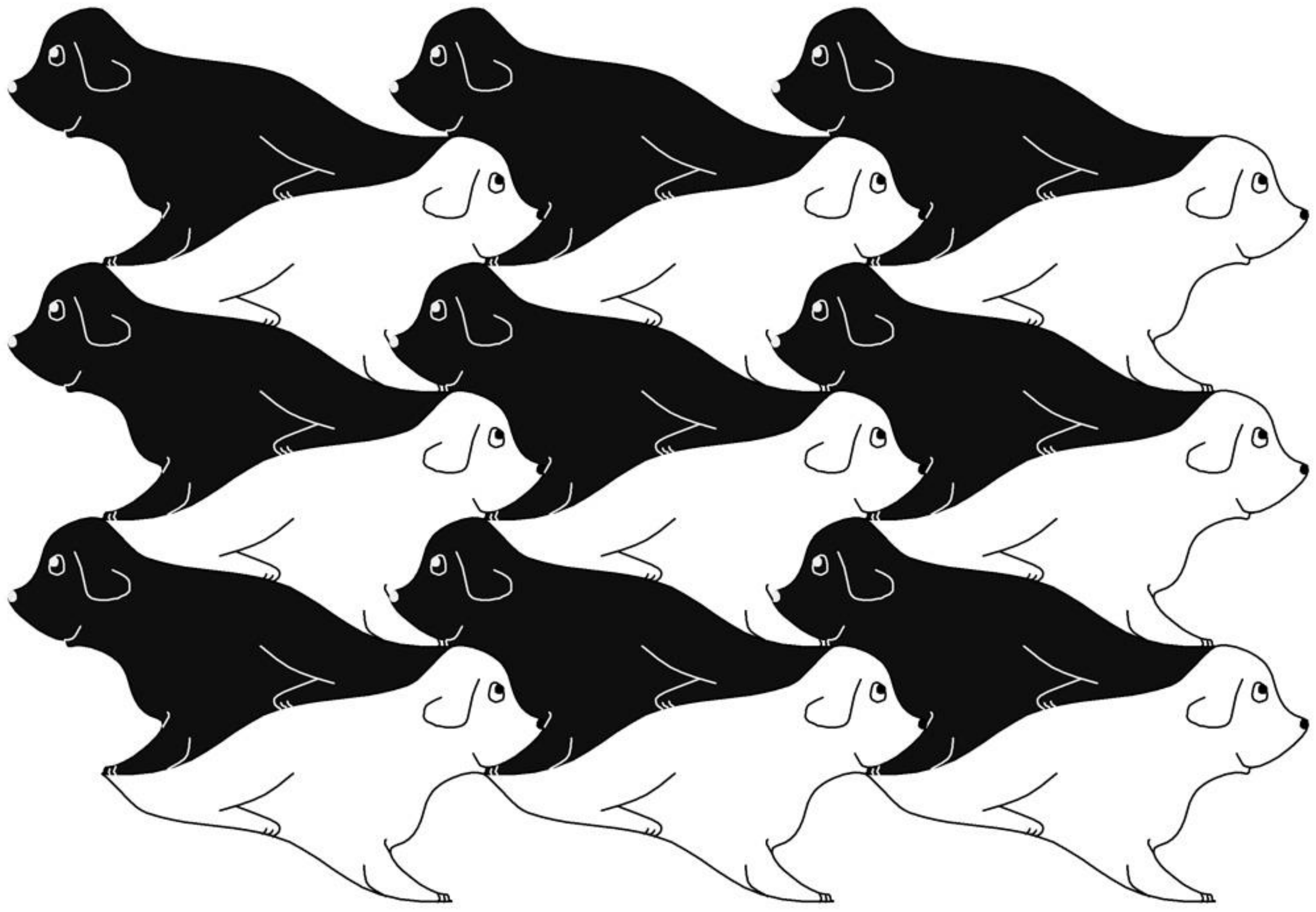
Example 6: puppies



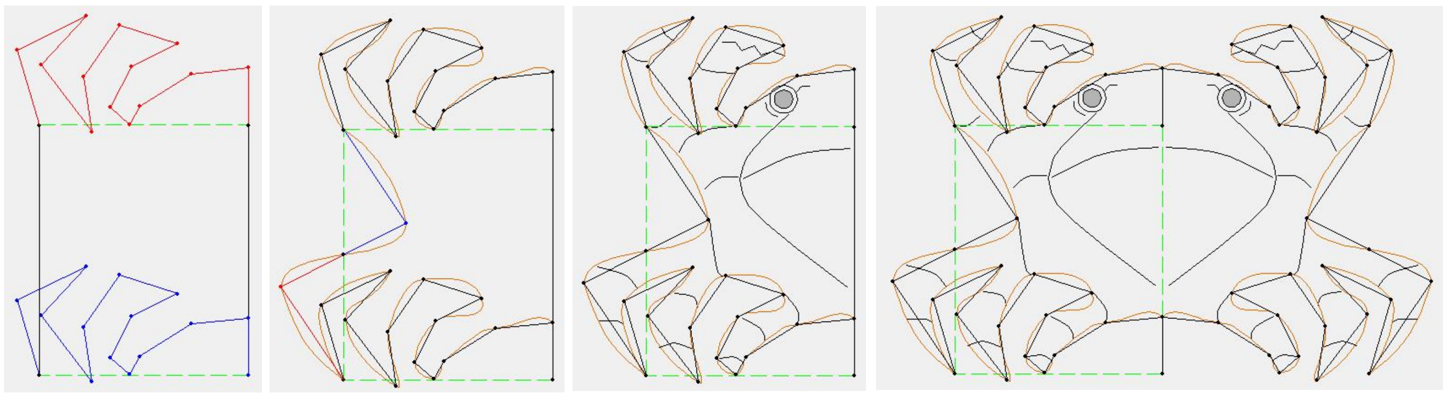


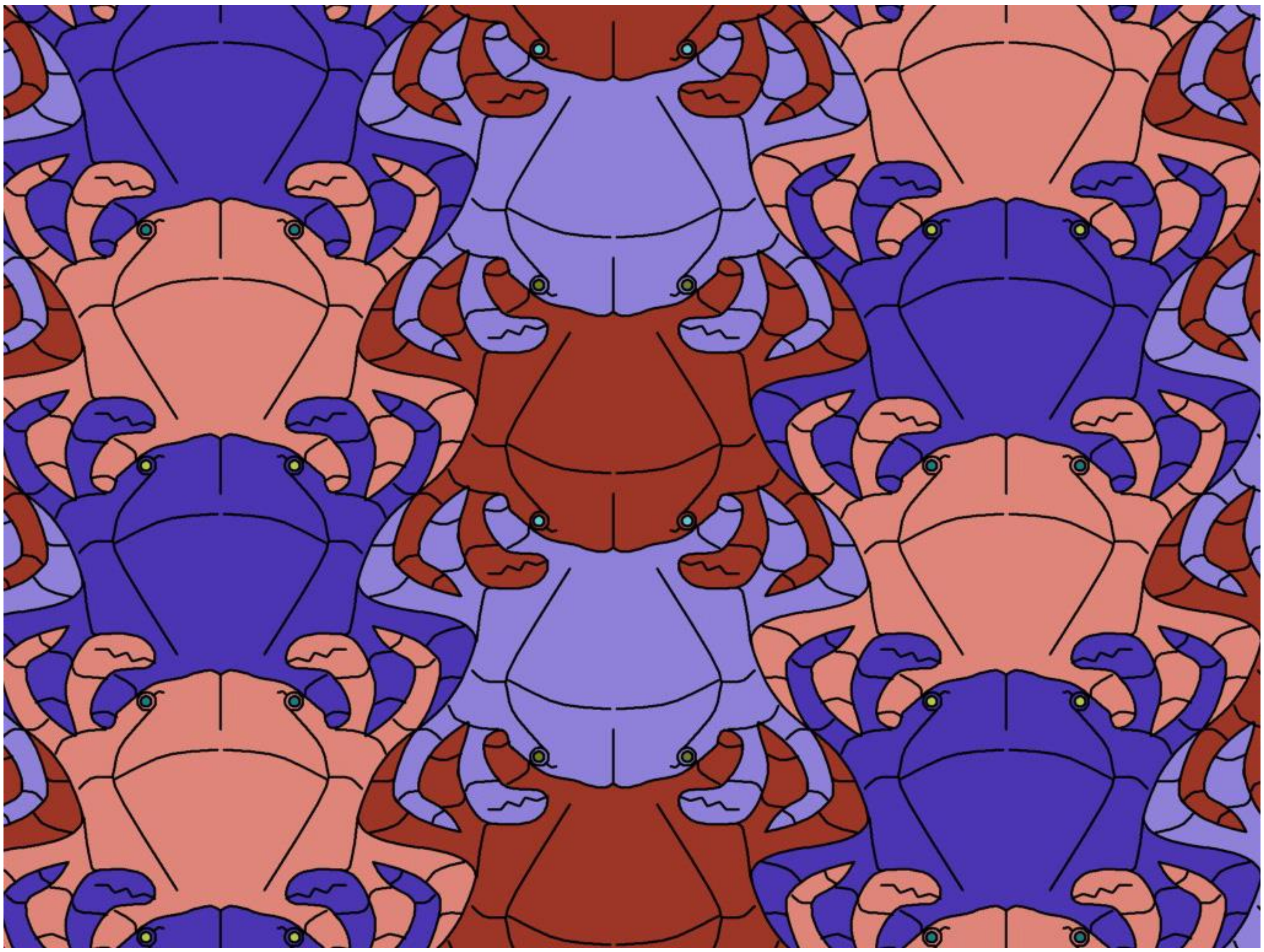


This example starts with a kite polygon and uses glide reflections on two edge pairs. These pairs are not parallel but they are congruent. First the puppies face is sketched in the upper left edge, then glide reflected to the lower left edge to form his chest and part of his front paws. In this case the source points are reflected about a line that connects the center points of the source and destination before being translated to the destination edge. The same type of transformation is applied to the remaining pair of edges. Notice that all the lines and filled objects were sketched using the negative color option. This works particularly well for very dark or light tiles. In this case I have black and white in mind:

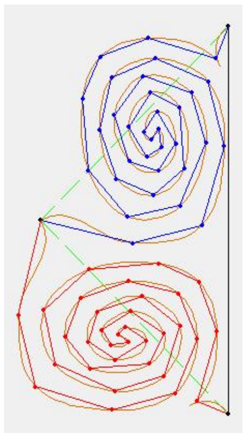


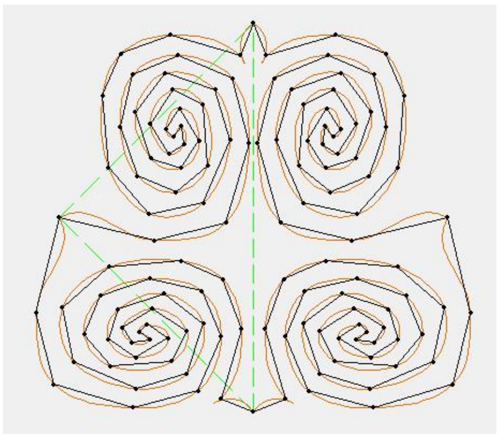
Example 7: crab


1: translate top to bottom; 2: center rotation on left edge, add curves; 3: sketch details; 4: reflect

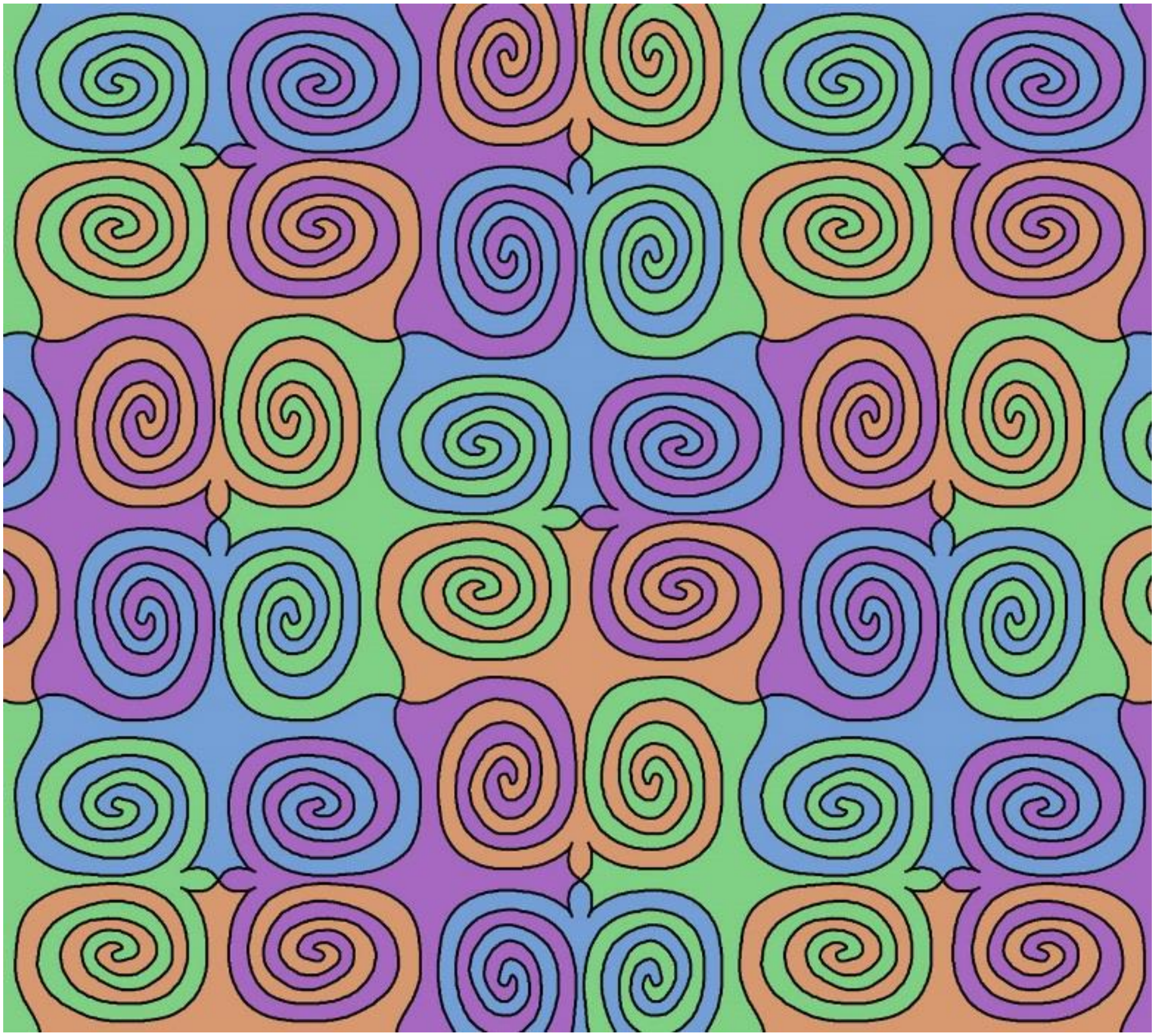


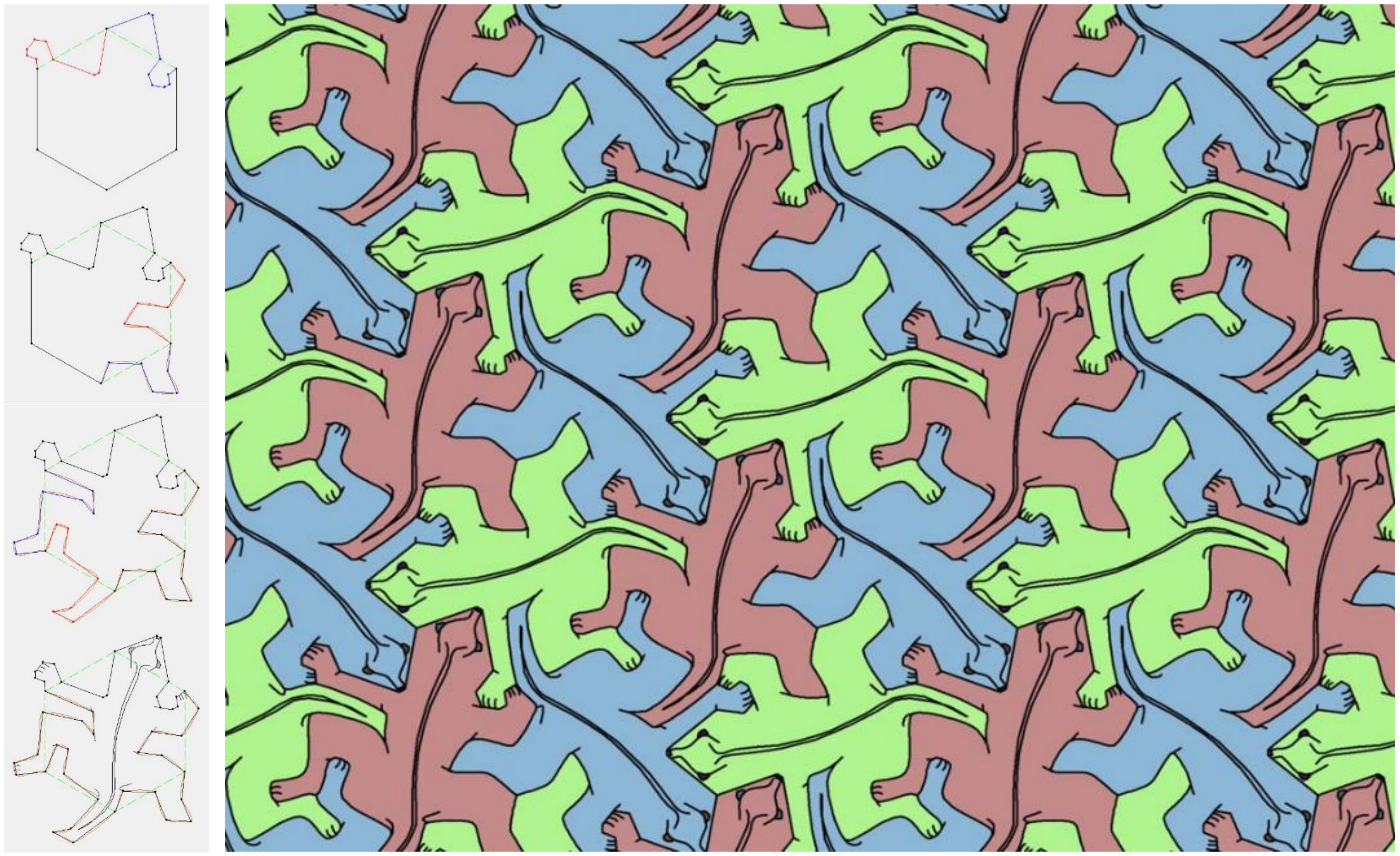
Example 8: just for fun

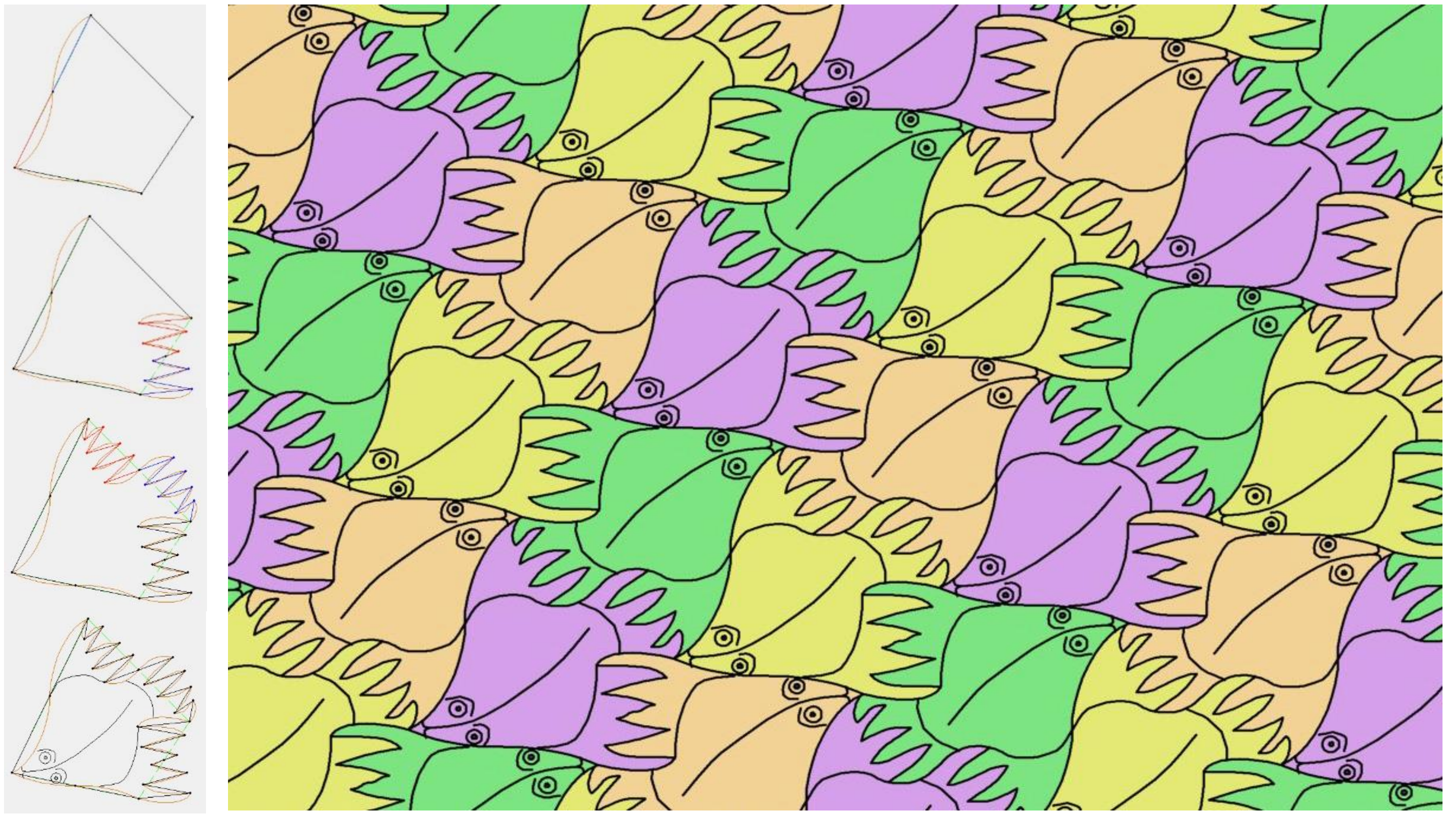




This is just an abstract design made from an isosceles right triangle. It looks like it might have been a lot of work to adjust all the curves, but once the anchor points are set it takes only one click in the curve editor on each point to establish the default curves. That's all. Only one curl was actually drawn; the second one is the result of a vertex rotation and the other two were created with the reflect tool. It shows that you can make your edges as complex as you like and they will snap back together seamlessly as long as you stay within the rules that govern the world of tessellations.

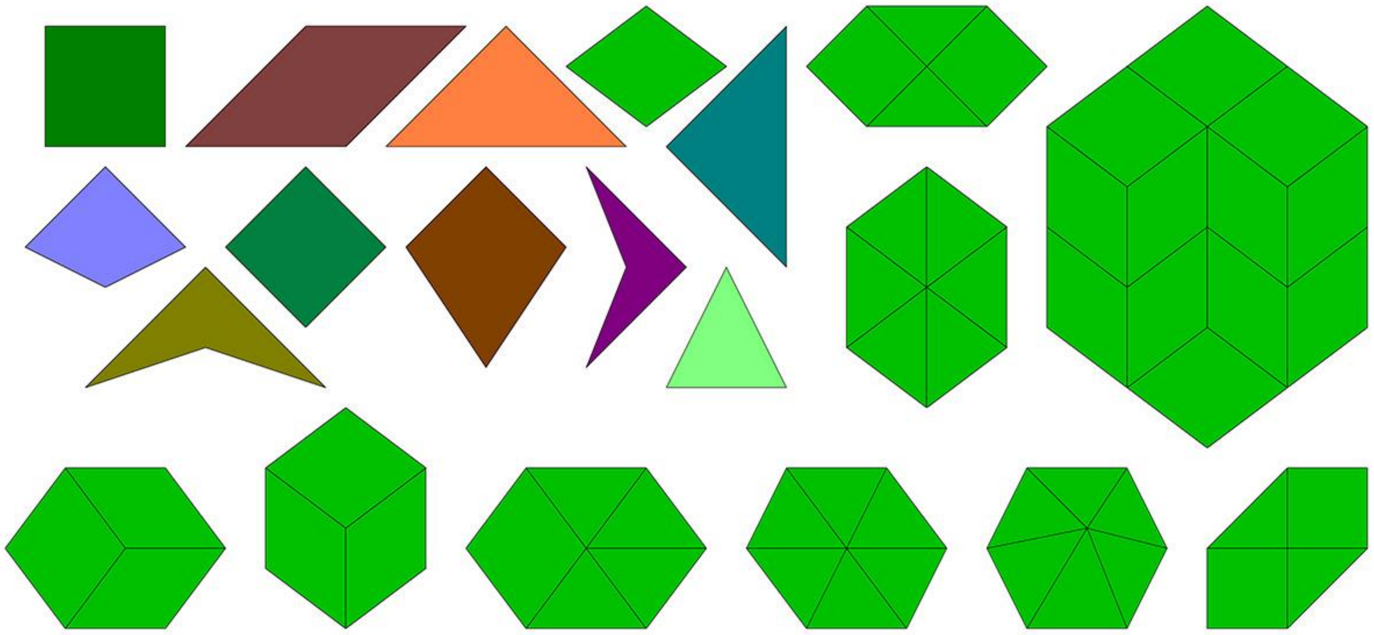



Another Escher lizard, this one created from three vertex rotation transformations on a hexagon.

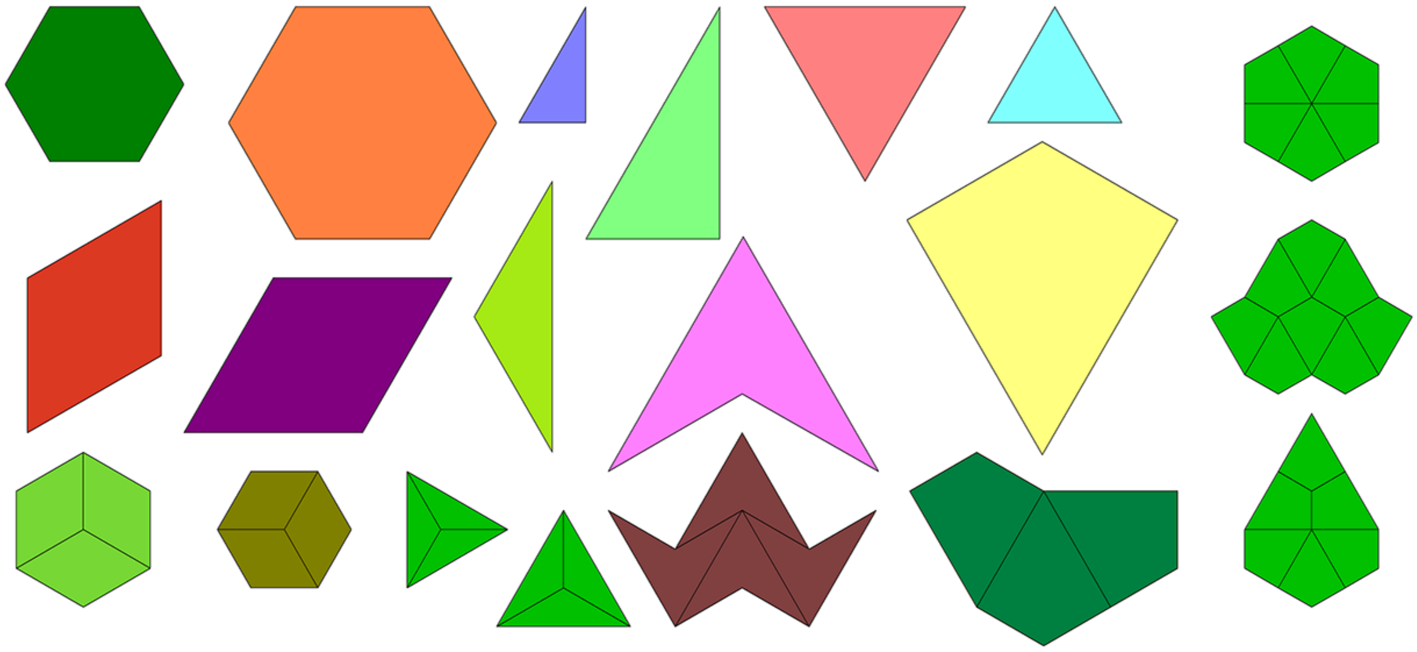

Another fish, this one created from a scalene quadliratel using four center rotation transformations

Appendix The primitives

EscherSketch allows you to load a file *of primitives* that contain tiles that tessellate the plane and each form a useful starting point for generation of a complex tile in the transformation editor. To load this file use **<u>files</u> load prims** in the tile editor.



In the rectangular grid there is a variety of squares, isosceles and right triangles, kites and arrows. The green subdivided hexagons consist of other useful triangles, and they can be traced around to generate hexagons that tessellate.



The primitives on the hexagonal grid feature hexagons, equilateral triangles and parallelograms, and various other triangles, kites and arrows that feature 30, 60 and 120 degree angles. There's also some small tessellations that indicate ways to reassemble them.