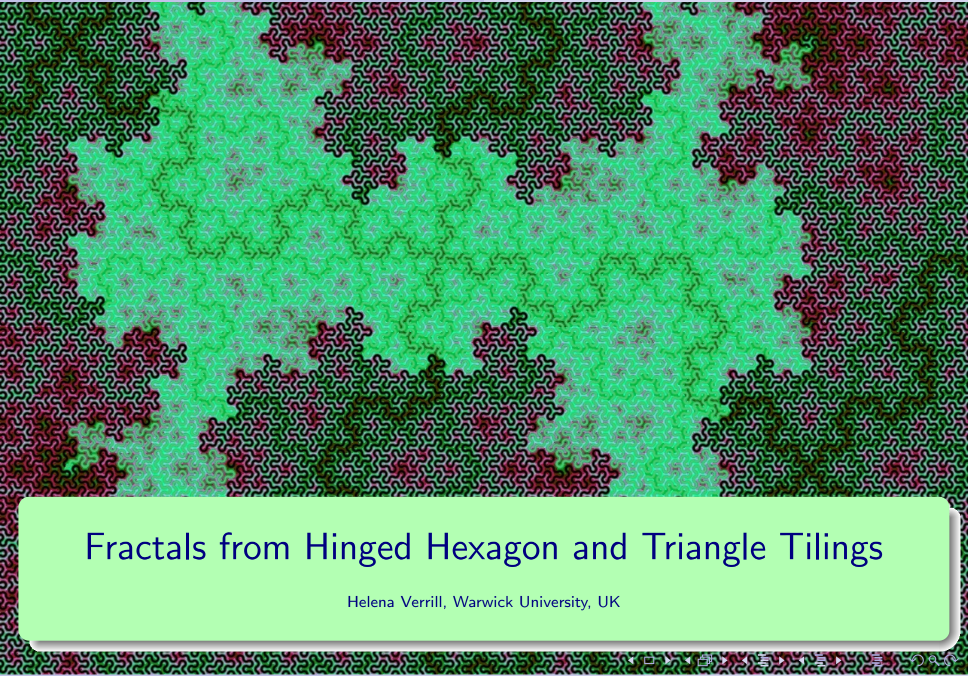
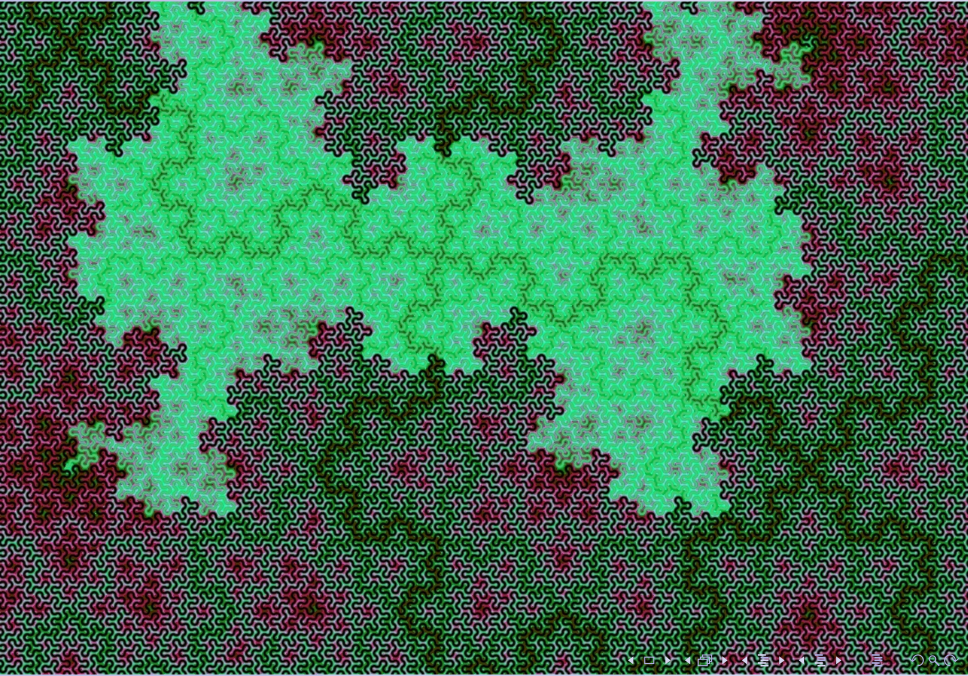


Fractals from Hinged Hexagon and Triangle Tilings

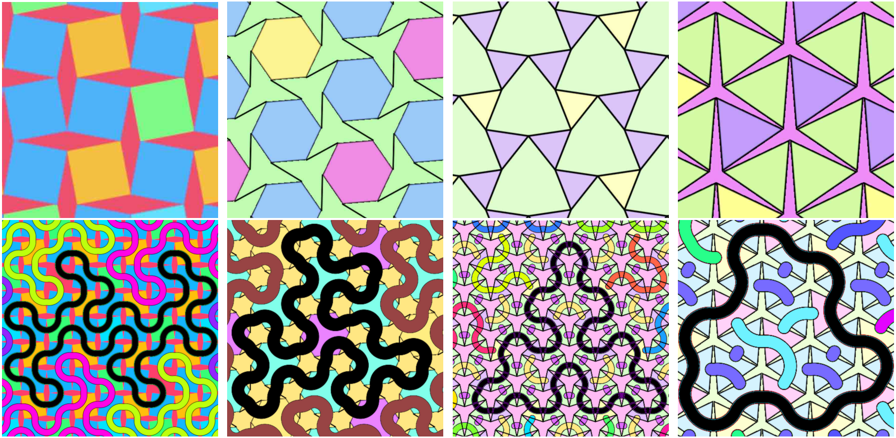
Helena Verrill, Warwick University, UK



Summary of Talk

(Graphics from `https://www.mathamaze.co.uk/Truchet2/`)

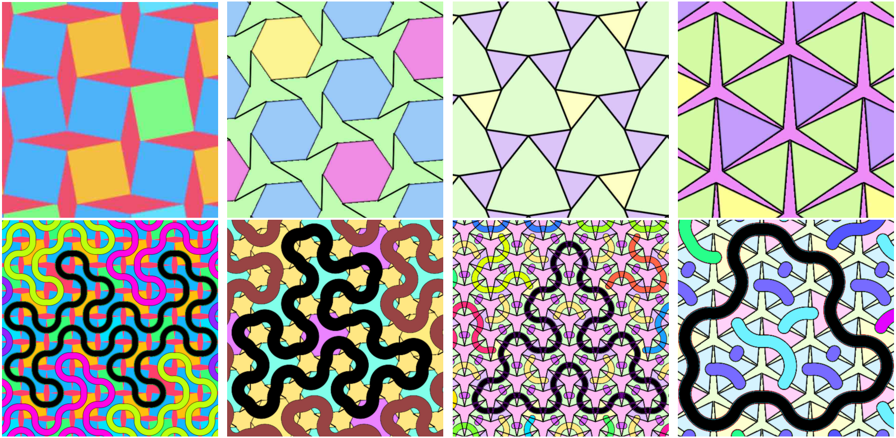
Take an initial tiling, e.g., of squares, hexagons or triangles.



Summary of Talk

(Graphics from `https://www.mathamaze.co.uk/Truchet2/`)

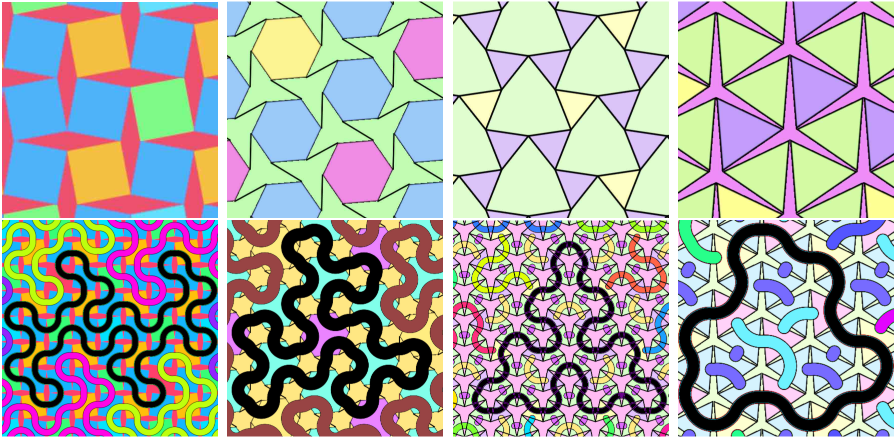
Decorate each tile with arcs;



Summary of Talk

(Graphics from `https://www.mathamaze.co.uk/Truchet2/`)

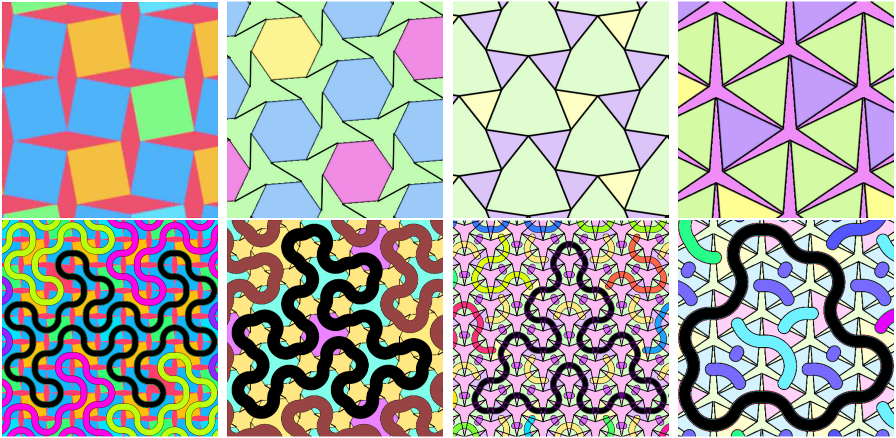
Decorate each tile with arcs; hinge to obtain new tiling;



Summary of Talk

(Graphics from `https://www.mathamaze.co.uk/Truchet2/`)

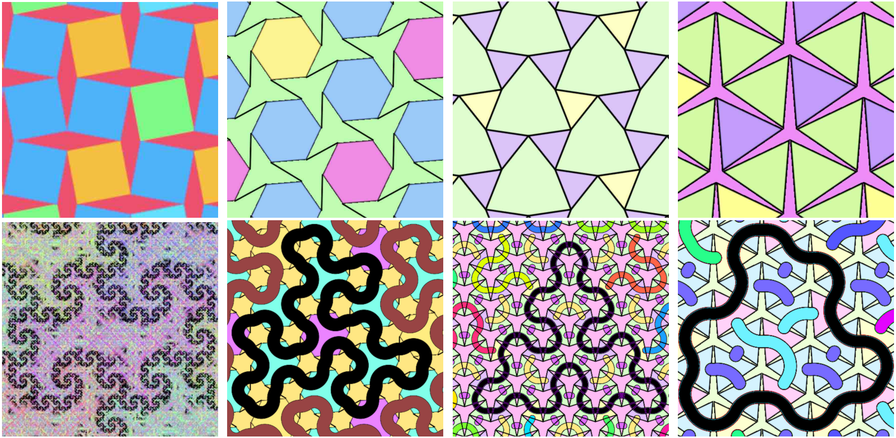
Decorate each tile with arcs; hinge to obtain new tiling; repeat



Summary of Talk

(Graphics from `https://www.mathamaze.co.uk/Truchet2/`)

Decorate each tile with arcs; hinge to obtain new tiling; repeat

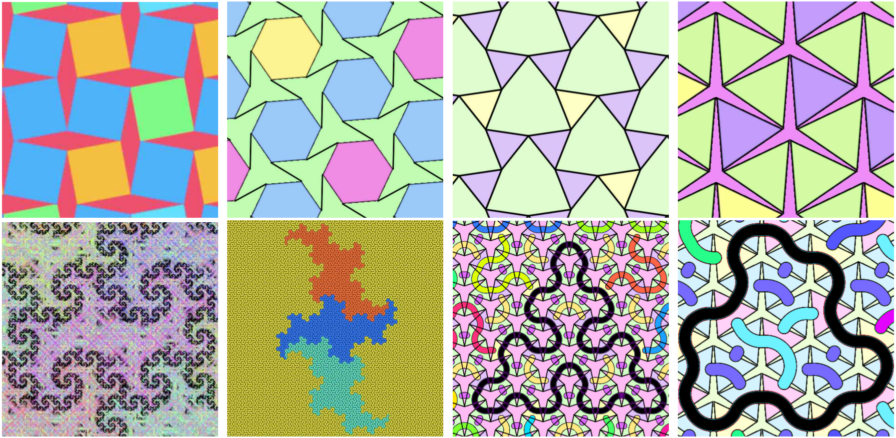


Heighway dragon

Summary of Talk

(Graphics from `https://www.mathamaze.co.uk/Truchet2/`)

Decorate each tile with arcs; hinge to obtain new tiling; repeat

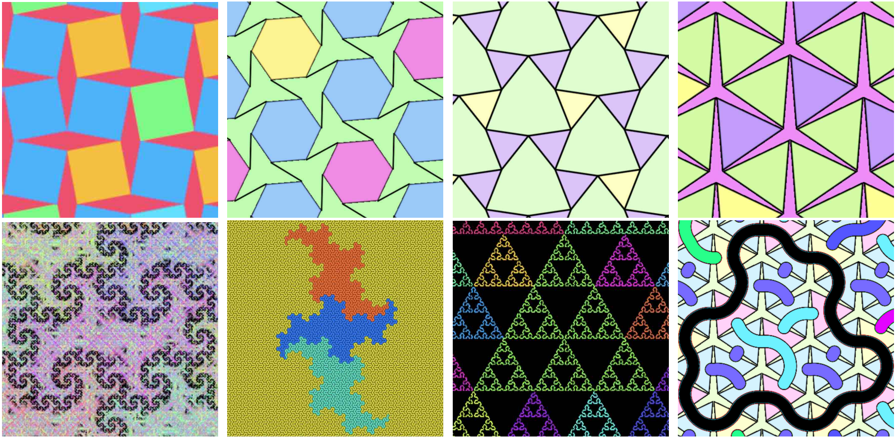


Heighway dragon • Davis/Knuth terdragon

Summary of Talk

(Graphics from `https://www.mathamaze.co.uk/Truchet2/`)

Decorate each tile with arcs; hinge to obtain new tiling; repeat

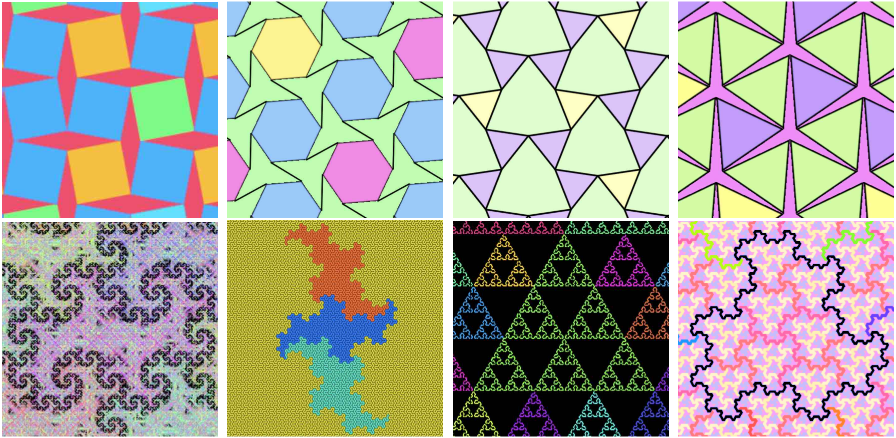


Heighway dragon • Davis/Knuth terdragon • Sierpinski triangle

Summary of Talk

(Graphics from https://www.mathamaze.co.uk/Truchet2/)

Decorate each tile with arcs; hinge to obtain new tiling; repeat

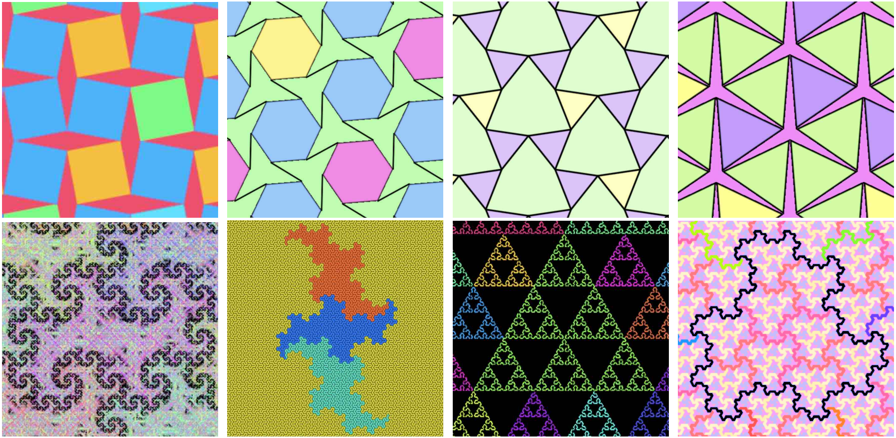


Heighway dragon • Davis/Knuth terdragon • Sierpinski triangle • Terdragon b'dary

Summary of Talk

(Graphics from `https://www.mathamaze.co.uk/Truchet2/`)

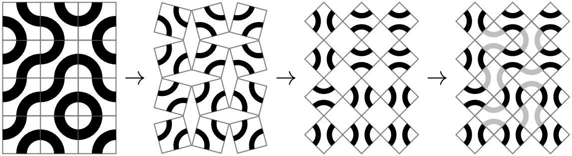
Decorate each tile with arcs; hinge to obtain new tiling; repeat



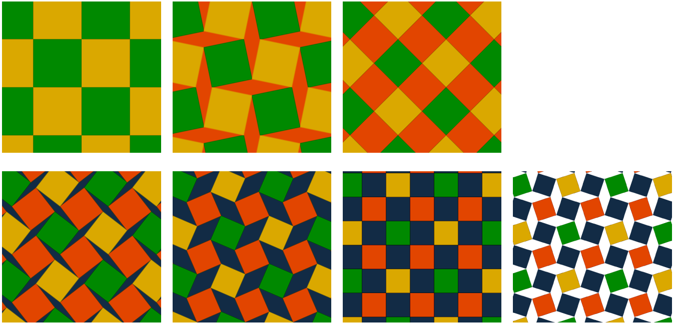
Heighway dragon • Davis/Knuth terdragon • Sierpinski triangle • Fudgeflake

Recall from last year: Hinged squares

Hinged
Truchet tiling

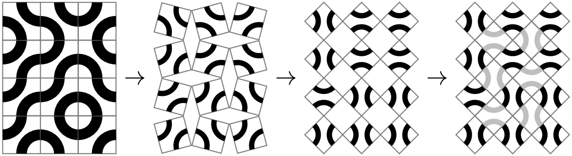


Hinged tiling;
rotate 45°;
scale by $\sqrt{2}$;
background
becomes
foreground

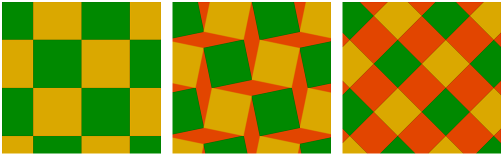


Recall from last year: Hinged squares

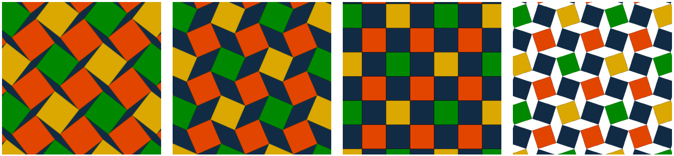
Hinged
Truchet tiling



Hinged tiling;
rotate 45°;
scale by $\sqrt{2}$;
background
becomes
foreground

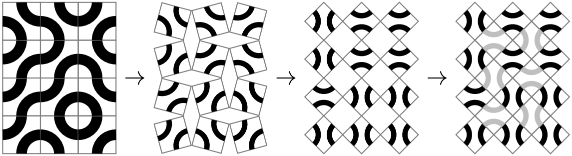


End result:
Heighway
fractal dragon

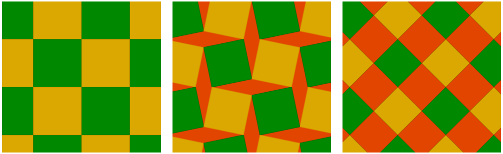


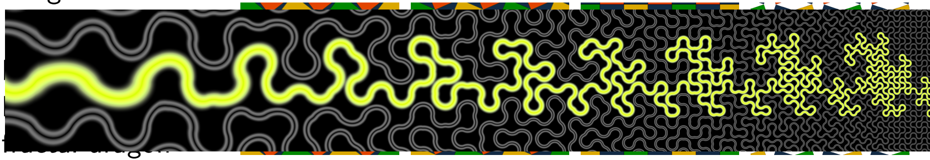
Recall from last year: Hinged squares

Hinged
Truchet tiling

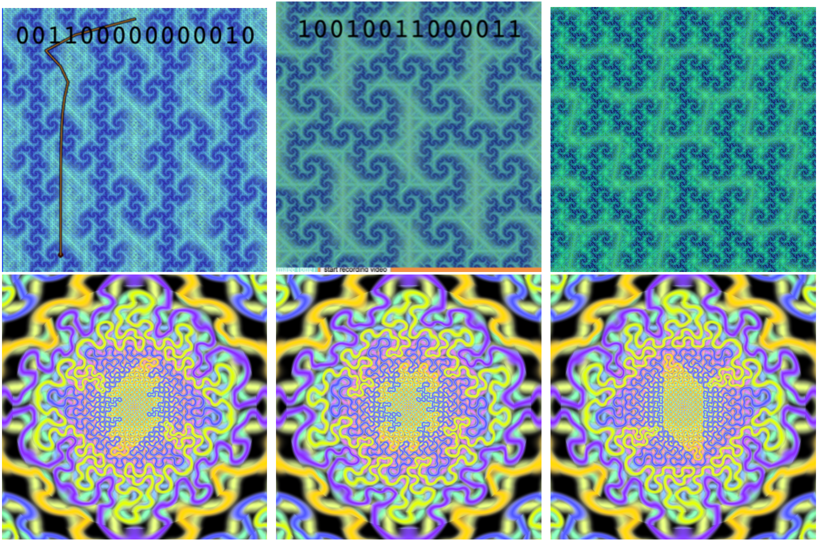


Hinged tiling;
rotate 45°;
scale by $\sqrt{2}$;
background
becomes
foreground





(this is from last year's talk; spot Heighway's dragon)



What about other hinged tilings?

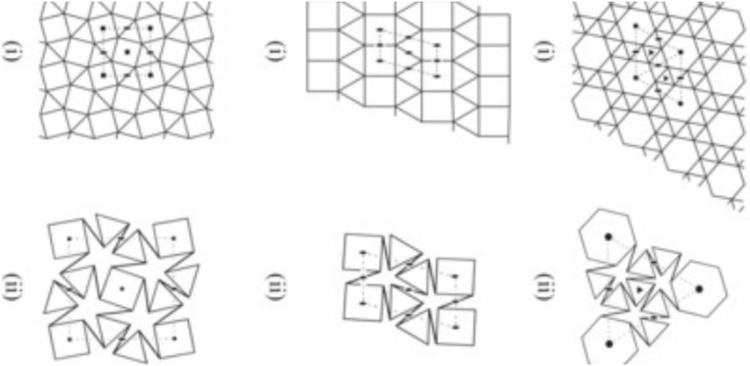
- There are many hinged tilings

What about other hinged tilings?

- There are many hinged tilings
- E.g., if you want to try this at home, see e.g.

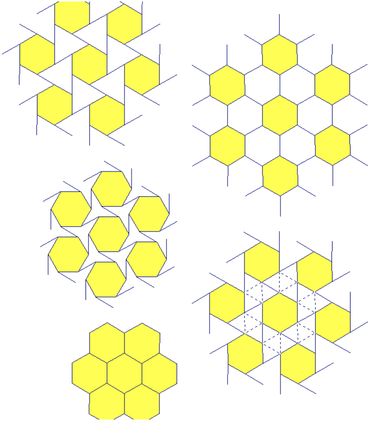
What about other hinged tilings?

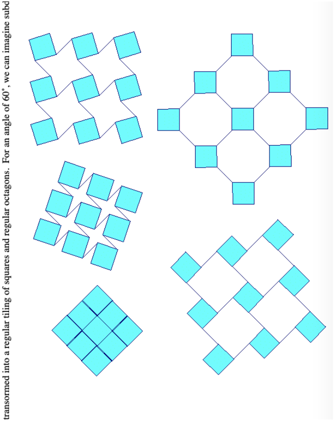
- There are many hinged tilings
- E.g., if you want to try this at home, see e.g. T. Tarnai, P. Fowler, S. Guest and F. Kovacs. “Equiauxetic Hinged Archimedean Tilings.” Symmetry (Basel),vol, 14 (2), 2022. https://www.mdpi.com/2073-8994/14/2/232



What about other hinged tilings?

- There are many hinged tilings
- E.g., if you want to try this at home, see e.g. or Alfinio Flores, “Hinged tilings” page, https://www.public.asu.edu/~aaafp/tiling/hingedtilingtext.html





What about other hinged tilings?

- There are many hinged tilings
- E.g., if you want to try this at home, see e.g. Tarnai et al, Flores,...
- Most don't open out to the same tiling (as the squares do)

What about other hinged tilings?

- There are many hinged tilings
- E.g., if you want to try this at home, see e.g. Tarnai et al, Flores,. . .
- Most don't open out to the same tiling (as the squares do)
- But some can be subdivided after opening, so the iteration process can be applied.

What about other hinged tilings?

- There are many hinged tilings
- E.g., if you want to try this at home, see e.g. Tarnai et al, Flores,...
- Most don't open out to the same tiling (as the squares do)
- But some can be subdivided after opening, so the iteration process can be applied.
- Candidates: Regular hexagon and triangle tilings.

What about other hinged tilings?

- There are many hinged tilings
- E.g., if you want to try this at home, see e.g. Tarnai et al, Flores,. . .
- Most don't open out to the same tiling (as the squares do)
- But some can be subdivided after opening, so the iteration process can be applied.
- Candidates: Regular hexagon and triangle tilings.
- Question: What happens? What fractals will we get?

What about other hinged tilings?

- There are many hinged tilings
- E.g., if you want to try this at home, see e.g. Tarnai et al, Flores,...
- Most don't open out to the same tiling (as the squares do)
- But some can be subdivided after opening, so the iteration process can be applied.
- Candidates: Regular hexagon and triangle tilings.
- Question: What happens? What fractals will we get?
- We will see

What about other hinged tilings?

- There are many hinged tilings
- E.g., if you want to try this at home, see e.g. Tarnai et al, Flores,...
- Most don't open out to the same tiling (as the squares do)
- But some can be subdivided after opening, so the iteration process can be applied.
- Candidates: Regular hexagon and triangle tilings.
- Question: What happens? What fractals will we get?
- We will see (1) fractal space filling curves;

What about other hinged tilings?

- There are many hinged tilings
- E.g., if you want to try this at home, see e.g. Tarnai et al, Flores,. . .
- Most don't open out to the same tiling (as the squares do)
- But some can be subdivided after opening, so the iteration process can be applied.
- Candidates: Regular hexagon and triangle tilings.
- Question: What happens? What fractals will we get?
- We will see (1) fractal space filling curves; (2) relations between fractals;

What about other hinged tilings?

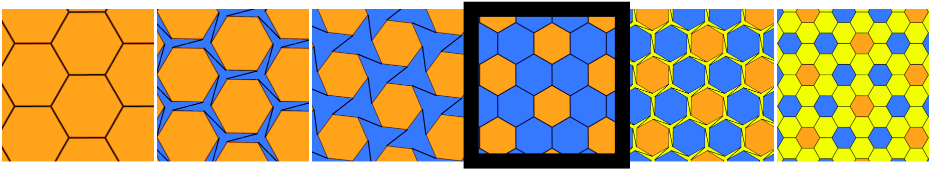
- There are many hinged tilings
- E.g., if you want to try this at home, see e.g. Tarnai et al, Flores,. . .
- Most don't open out to the same tiling (as the squares do)
- But some can be subdivided after opening, so the iteration process can be applied.
- Candidates: Regular hexagon and triangle tilings.
- Question: What happens? What fractals will we get?
- We will see (1) fractal space filling curves; (2) relations between fractals; (3) boundaries of space filling curves.

What about other hinged tilings?

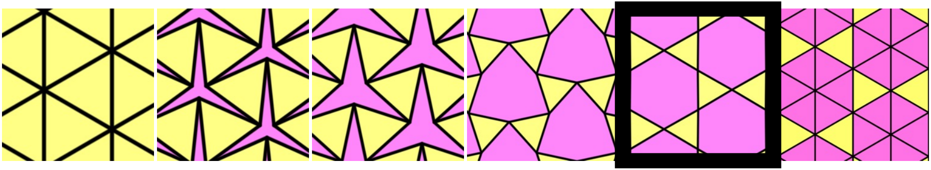
- There are many hinged tilings
- E.g., if you want to try this at home, see e.g. Tarnai et al, Flores,. . .
- Most don't open out to the same tiling (as the squares do)
- But some can be subdivided after opening, so the iteration process can be applied.
- Candidates: Regular hexagon and triangle tilings.
- Question: What happens? What fractals will we get?
- We will see (1) fractal space filling curves; (2) relations between fractals; (3) boundaries of space filling curves.
- The following slides contain snapshots from javscript programs, which are used to clarify ideas. Please follow along at: `www.mathamaze.co.uk/Truchet3/`

The hinged tilings

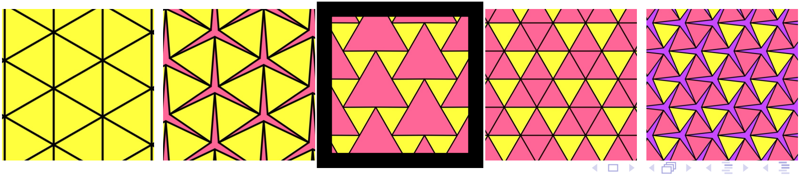
Hexagons with links: (rotate hexagons through 30°)



Triangles (1) (rotate triangles through 60°)

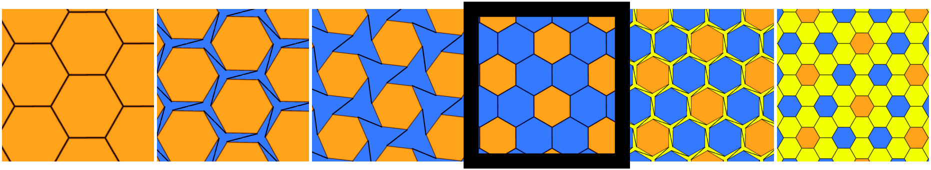


Triangles (2) (rotate triangles through 30°)

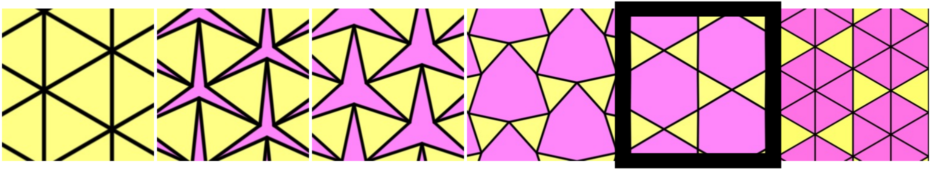


The hinged tilings

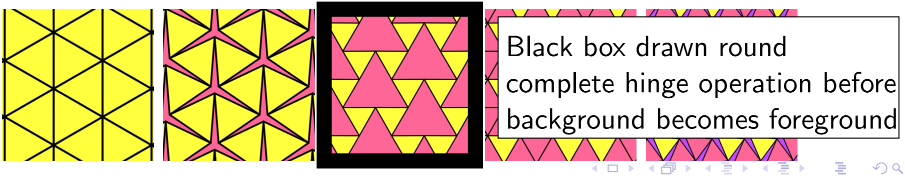
Hexagons with links: (rotate hexagons through 30°)



Triangles (1) (rotate triangles through 60°)



Triangles (2) (rotate triangles through 30°)



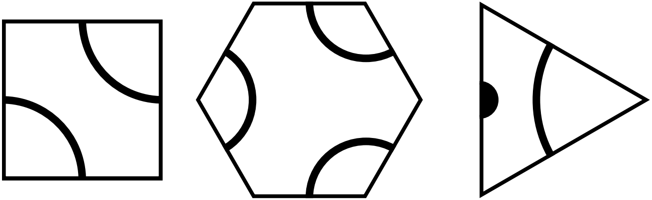
Black box drawn round complete hinge operation before background becomes foreground

Truchet designs and fractals

- We use the following tile designs:

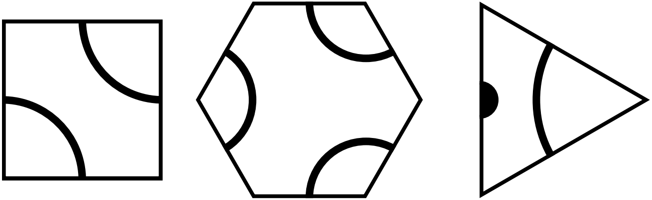
Truchet designs and fractals

- We use the following tile designs:



Truchet designs and fractals

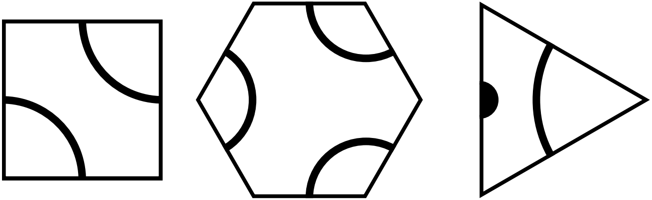
- We use the following tile designs:



- Inspired by the Smith (Truchet) tile design (square case);

Truchet designs and fractals

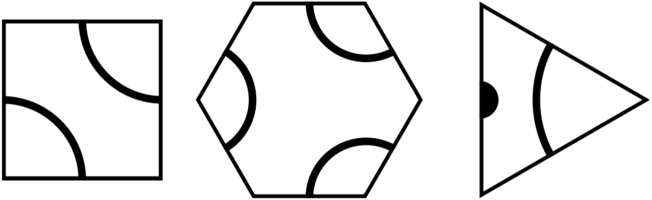
- We use the following tile designs:



- Inspired by the Smith (Truchet) tile design (square case);
- Truchet had the idea of putting together a lot of identical tiles at different orientations; Smith had the idea of using circle arcs
- Just hinge a design with these tiles; add more such tiles when in the "open" position. Repeat.

Truchet designs and fractals

- We use the following tile designs:



- Inspired by the Smith (Truchet) tile design (square case);
- Truchet had the idea of putting together a lot of identical tiles at different orientations; Smith had the idea of using circle arcs
- Just hinge a design with these tiles; add more such tiles when in the "open" position. Repeat.
- Example programs at: `www.mathamaze.co.uk/Truchet3/`

Binary operation sequences; continuous vs discrete

- Overview of next few slides:

Binary operation sequences; continuous vs discrete

- Overview of next few slides:
- We have a **continuous hinging operation**, starting from closed, at $t = 0$, ending at "open", $t = 1$.

Binary operation sequences; continuous vs discrete

- Overview of next few slides:
- We have a **continuous hinging operation**, starting from closed, at $t = 0$, ending at "open", $t = 1$.
- ("open" is when you decided you've finished your hinging and want to replace background with new forground tiles etc; this is your choice; depending on what works)

Binary operation sequences; continuous vs discrete

- Overview of next few slides:
- We have a **continuous hinging operation**, starting from closed, at $t = 0$, ending at "open", $t = 1$.
- ("open" is when you decided you've finished your hinging and want to replace background with new forground tiles etc; this is your choice; depending on what works)
- In each case, we can rotate **clockwise** or **counterclockwise**; denote this by 0 or 1. So, an operation sequence can be described as a **binary string** e.g. 001010111 etc.

Binary operation sequences; continuous vs discrete

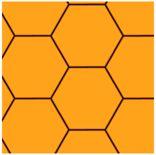
- Overview of next few slides:
- We have a **continuous hinging operation**, starting from closed, at $t = 0$, ending at "open", $t = 1$.
- ("open" is when you decided you've finished your hinging and want to replace background with new forground tiles etc; this is your choice; depending on what works)
- In each case, we can rotate **clockwise** or **counterclockwise**; denote this by 0 or 1. So, an operation sequence can be described as a **binary string** e.g. 001010111 etc.
- The image at stage $t = 0, 1, 2, 3$ would be the appearance after **discrete replacements** corresponding to strings: $\varnothing$, 0, 00, 001, etc

Binary operation sequences; continuous vs discrete

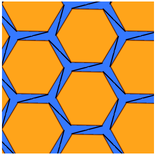
- Overview of next few slides:
- We have a **continuous hinging operation**, starting from closed, at $t = 0$, ending at "open", $t = 1$.
- ("open" is when you decided you've finished your hinging and want to replace background with new forground tiles etc; this is your choice; depending on what works)
- In each case, we can rotate **clockwise** or **counterclockwise**; denote this by 0 or 1. So, an operation sequence can be described as a **binary string** e.g. 001010111 etc.
- The image at stage $t = 0, 1, 2, 3$ would be the appearance after **discrete replacements** corresponding to strings: $\varnothing$, 0, 00, 001, etc
- Image at a non integer value of t corresponds to an **intermediate** position; i.e., **continuous** interpolation of the discrete replacement rule by the hinging process.

Hexagon case

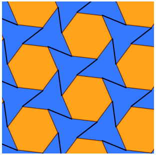
- Continuous process:



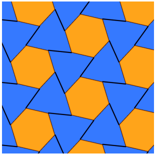
$t = 0$



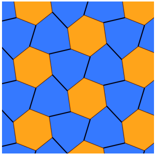
$t = 0.1$



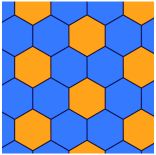
$t = 0.3$



$t = 0.5$



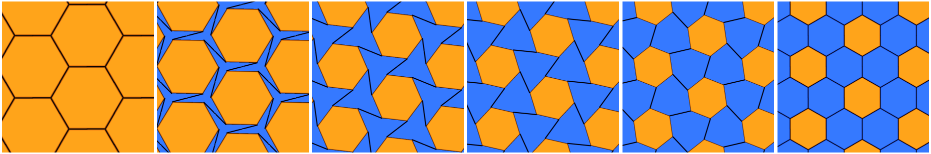
$t = 0.8$



$t = 1$

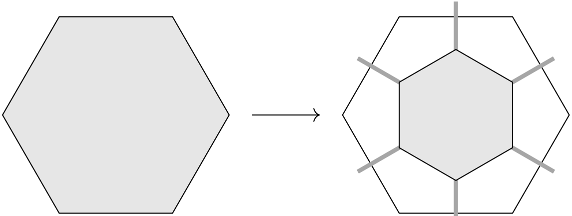
Hexagon case

- Continuous process:



$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

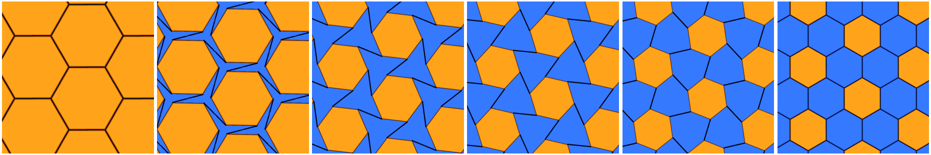
- Discrete process:



$t = 0$ $t = 1$

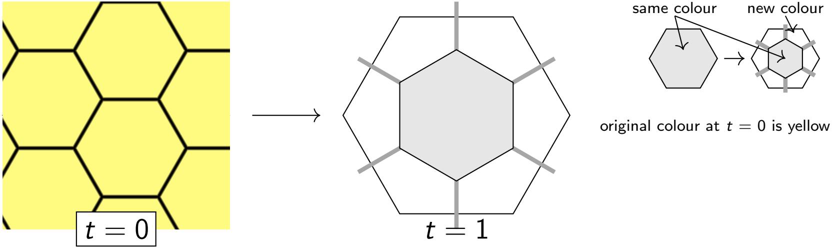
Hexagon case

- Continuous process:



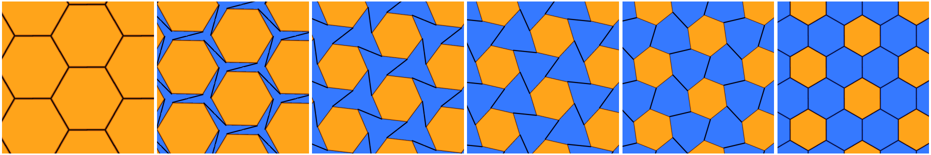
$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

- Discrete process:



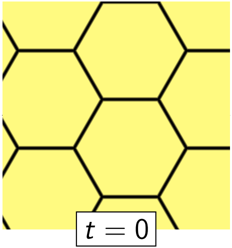
Hexagon case

- Continuous process:

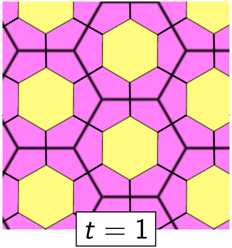


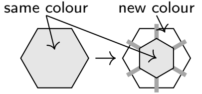
$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

- Discrete process:



$\longrightarrow$

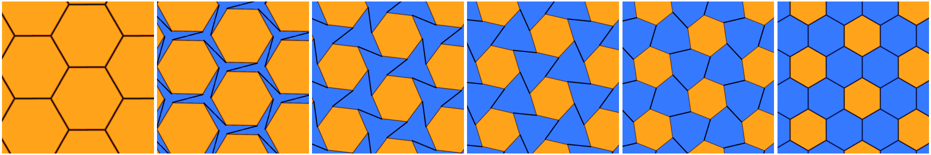




original colour at $t = 0$ is yellow
new colour at $t = 1$ is pink

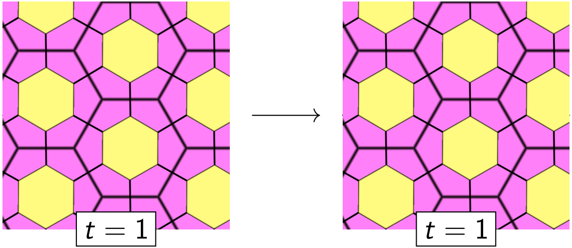
Hexagon case

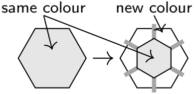
- Continuous process:



$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

- Discrete process:

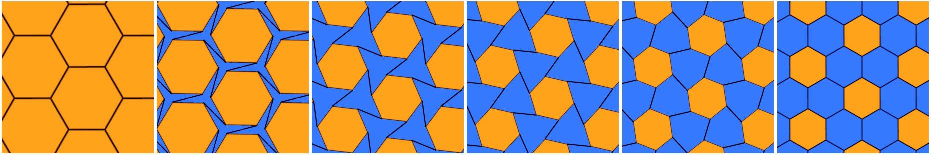




original colour at $t = 0$ is yellow
new colour at $t = 1$ is pink

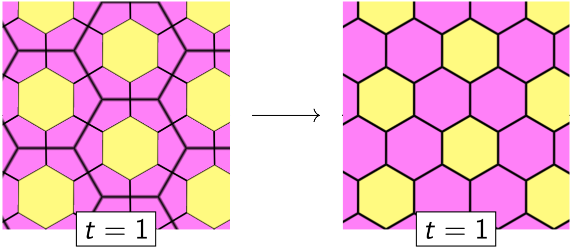
Hexagon case

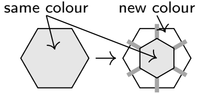
- Continuous process:



$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

- Discrete process:

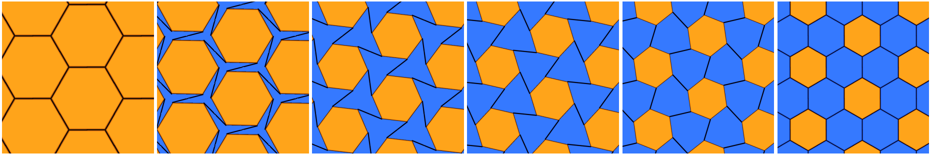




original colour at $t = 0$ is yellow
new colour at $t = 1$ is pink

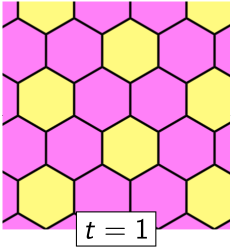
Hexagon case

- Continuous process:

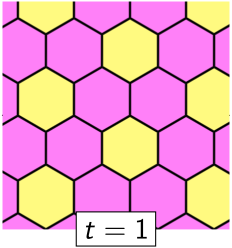


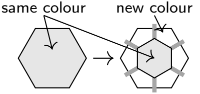
$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

- Discrete process:



$\longrightarrow$

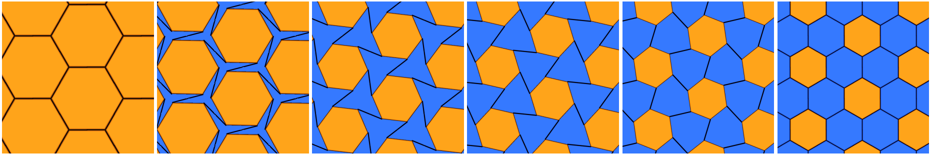




original colour at $t = 0$ is yellow
new colour at $t = 1$ is pink

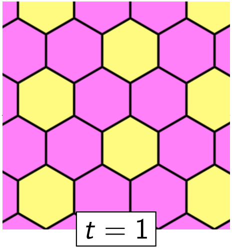
Hexagon case

- Continuous process:

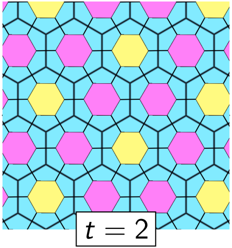


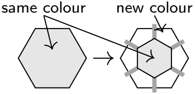
$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

- Discrete process:



→

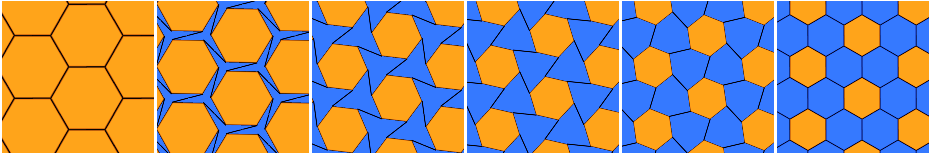




original colour at $t = 0$ is yellow
new colour at $t = 1$ is pink
new colour at $t = 2$ is blue

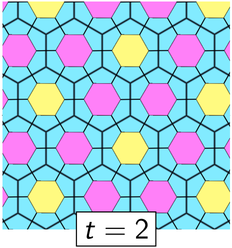
Hexagon case

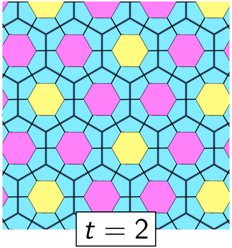
- Continuous process:

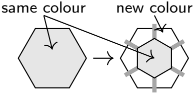


$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

- Discrete process:



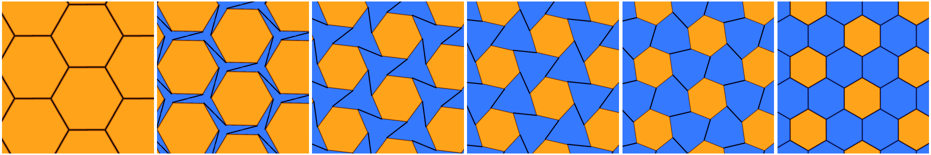




original colour at $t = 0$ is yellow
new colour at $t = 1$ is pink
new colour at $t = 2$ is blue

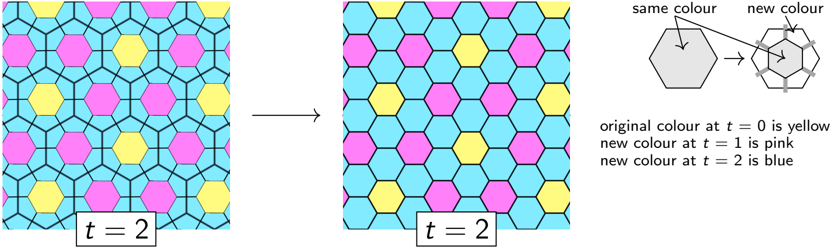
Hexagon case

- Continuous process:



$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

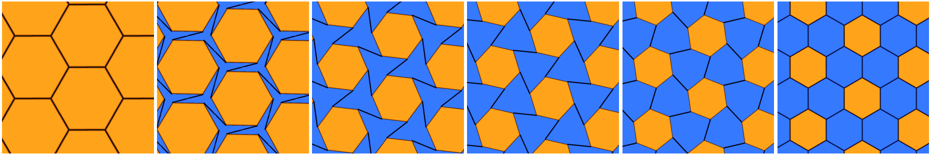
- Discrete process:



original colour at $t = 0$ is yellow
new colour at $t = 1$ is pink
new colour at $t = 2$ is blue

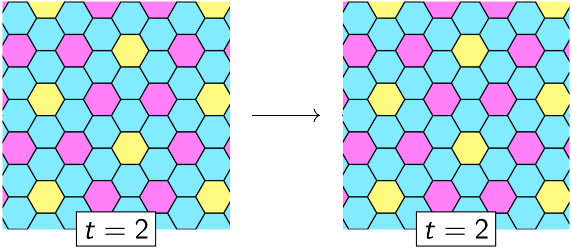
Hexagon case

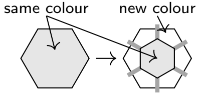
- Continuous process:



$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

- Discrete process:

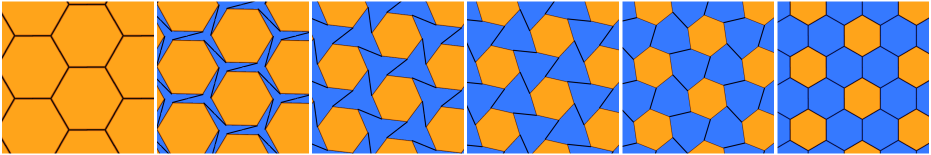




original colour at $t = 0$ is yellow
new colour at $t = 1$ is pink
new colour at $t = 2$ is blue

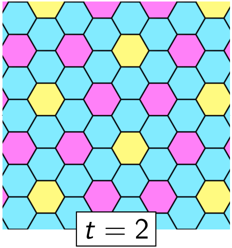
Hexagon case

- Continuous process:

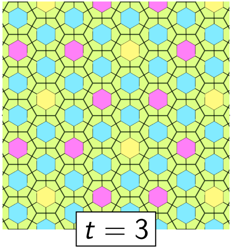


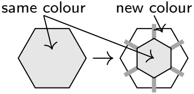
$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

- Discrete process:



$\longrightarrow$

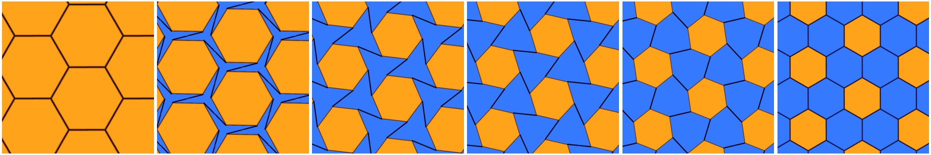




original colour at $t = 0$ is yellow
new colour at $t = 1$ is pink
new colour at $t = 2$ is blue
new colour at $t = 3$ is green

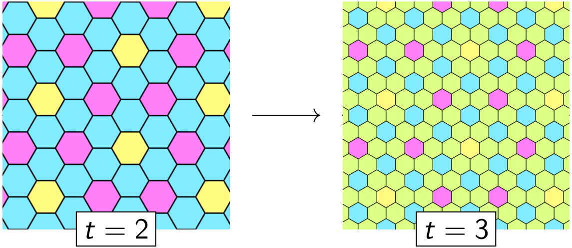
Hexagon case

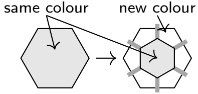
- Continuous process:



$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

- Discrete process:

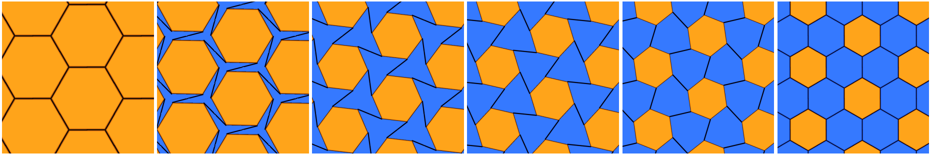




original colour at $t = 0$ is yellow
new colour at $t = 1$ is pink
new colour at $t = 2$ is blue
new colour at $t = 3$ is green

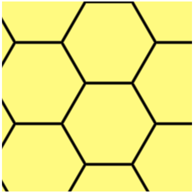
Hexagon case

- Continuous process:



$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

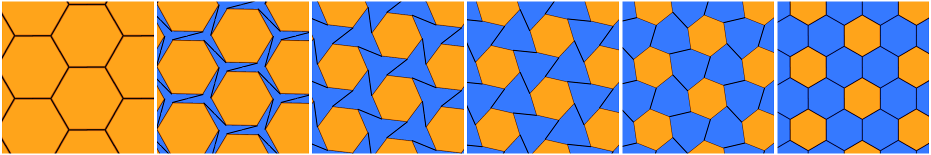
- Discrete process:



$t = 0$

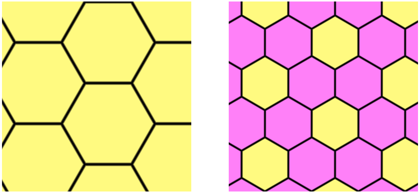
Hexagon case

- Continuous process:



$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

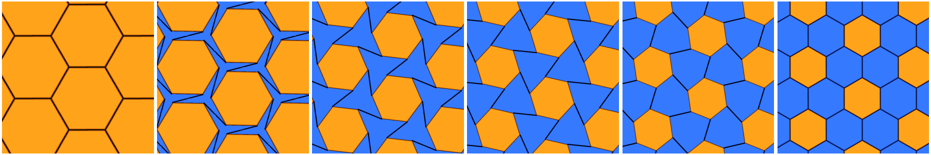
- Discrete process:



$t = 0$ $t = 1$

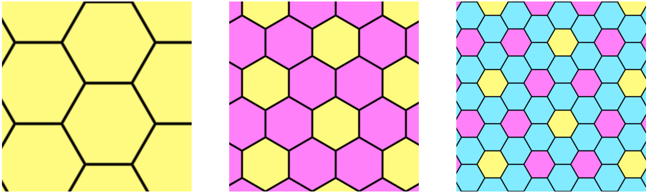
Hexagon case

- Continuous process:



$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

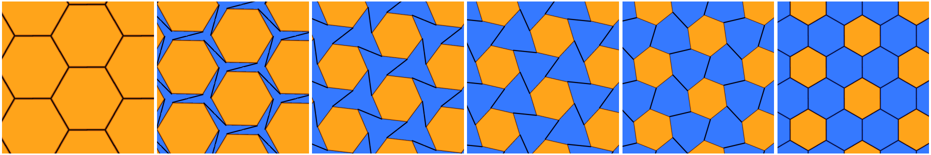
- Discrete process:



$t = 0$ $t = 1$ $t = 2$

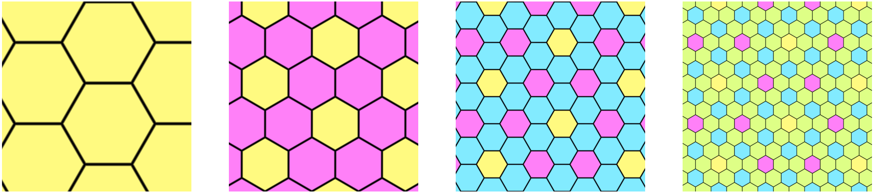
Hexagon case

- Continuous process:



$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

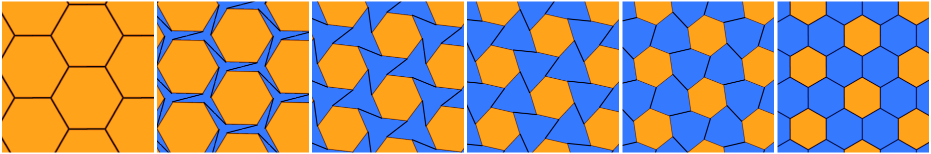
- Discrete process:



$t = 0$ $t = 1$ $t = 2$ $t = 3$

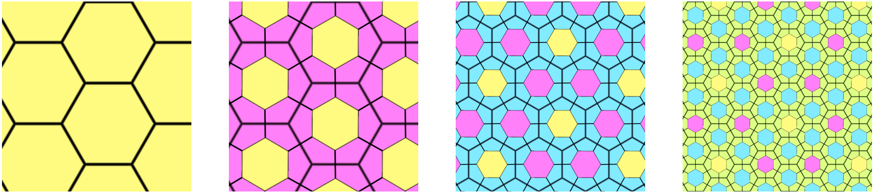
Hexagon case

- Continuous process:



$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

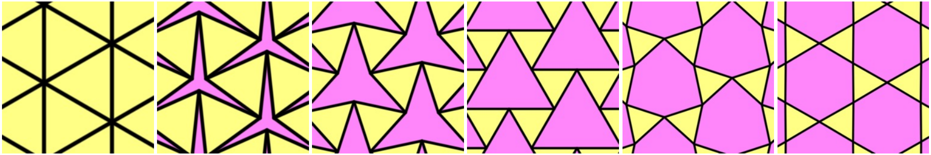
- Discrete process: (with lines to show replaced tiles)



$t = 0$ $t = 1$ $t = 2$ $t = 3$

Triangle 1 case

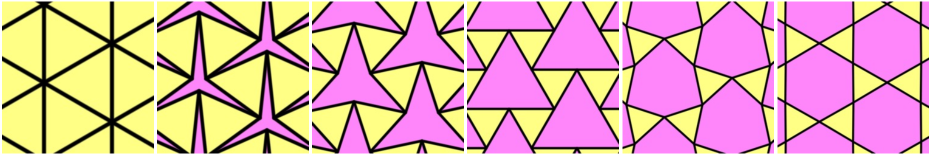
- Continuous process:



$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

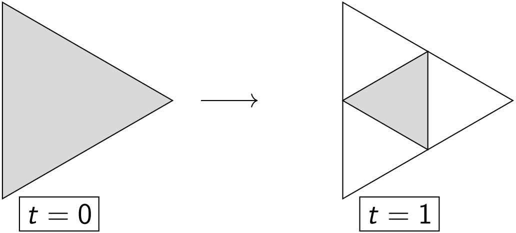
Triangle 1 case

- Continuous process:



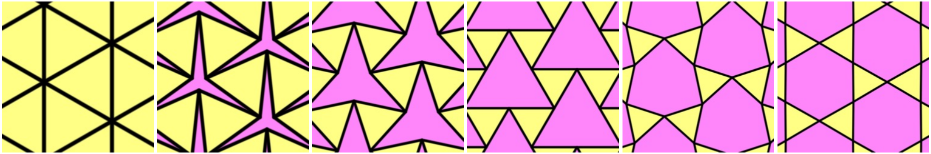
$t = 0$	$t = 0.1$	$t = 0.3$	$t = 0.5$	$t = 0.8$	$t = 1$

- Discrete process:



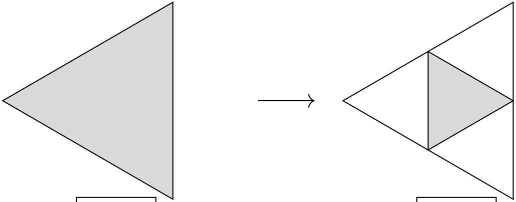
Triangle 1 case

- Continuous process:



$t = 0$ $t = 0.1$ $t = 0.3$ $t = 0.5$ $t = 0.8$ $t = 1$

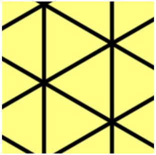
- Discrete process: (there are two orientations of triangles)

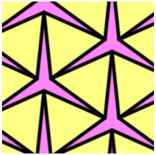


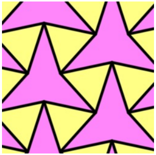
$t = 0$ $\longrightarrow$ $t = 1$

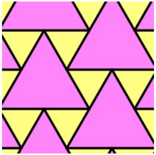
Triangle 1 case

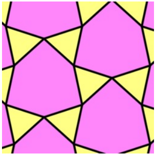
- Continuous process:

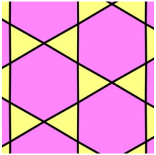

$t = 0$


$t = 0.1$

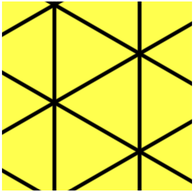

$t = 0.3$

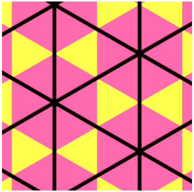

$t = 0.5$

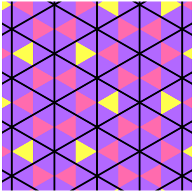

$t = 0.8$

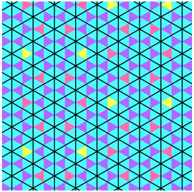

$t = 1$

- Discrete process:


$t = 0$

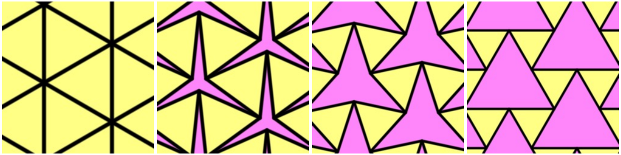

$t = 1$


$t = 2$


$t = 3$

Triangle 2 case

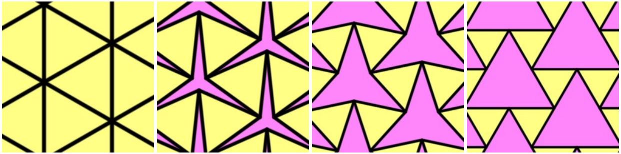
- Continuous process:



$t = 0$ $t = 0.2$ $t = 0.6$ $t = 1$

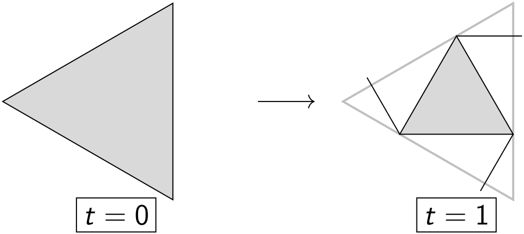
Triangle 2 case

- Continuous process:



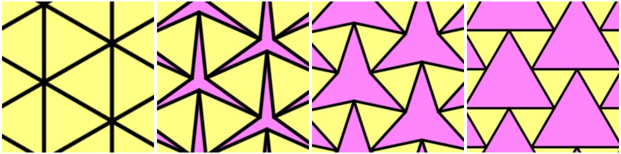
$t = 0$ $t = 0.2$ $t = 0.6$ $t = 1$

- Discrete process: (also mirror image of this triangle)



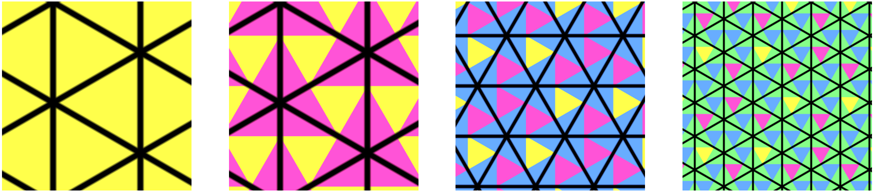
Triangle 2 case

- Continuous process:



$t = 0$ $t = 0.2$ $t = 0.6$ $t = 1$

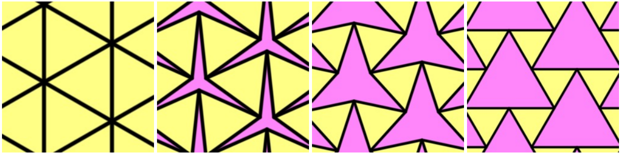
- Discrete process: (note original tile boundaries are not boundaries after operation)



$t = 0$ $t = 1$ $t = 2$ $t = 3$

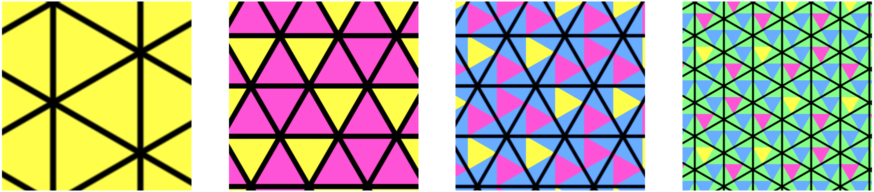
Triangle 2 case

- Continuous process:



$t = 0$ $t = 0.2$ $t = 0.6$ $t = 1$

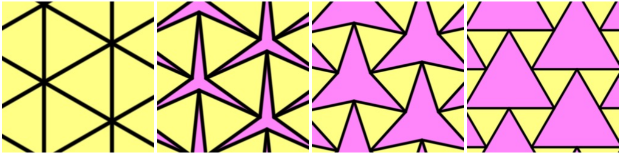
- Discrete process: (note original tile boundaries are not boundaries after operation)



$t = 0$ $t = 1$ $t = 2$ $t = 3$

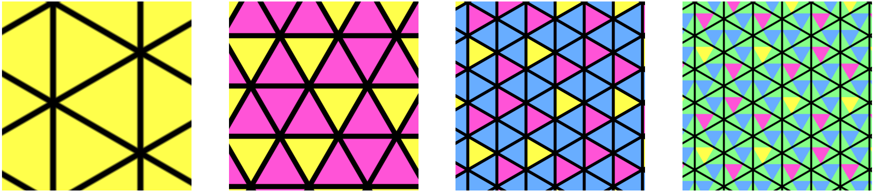
Triangle 2 case

- Continuous process:



$t = 0$ $t = 0.2$ $t = 0.6$ $t = 1$

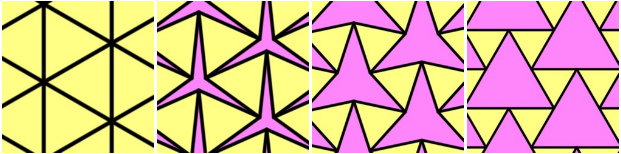
- Discrete process: (note original tile boundaries are not boundaries after operation)



$t = 0$ $t = 1$ $t = 2$ $t = 3$

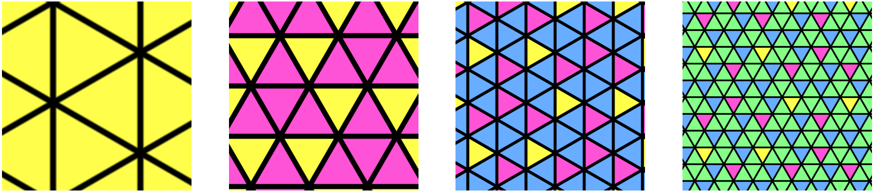
Triangle 2 case

- Continuous process:



$t = 0$ $t = 0.2$ $t = 0.6$ $t = 1$

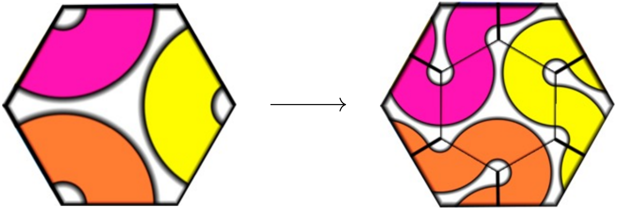
- Discrete process: (note original tile boundaries are not boundaries after operation)



$t = 0$ $t = 1$ $t = 2$ $t = 3$

Now add the Truchet / Smith designs

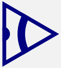
- Hexagons: (individual tile, one operation)



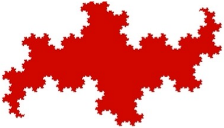
Now add the Truchet / Smith designs

- Hexagons: (several tiles, several operations)

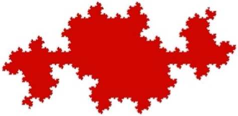
$t = 0$ $t = 0.4$ $t = 1$ $t = 1.6$ $t = 2$

Now add the Truchet / Smith designs  

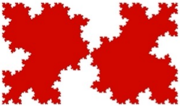
- Hexagons: (many operations ⇒ Fractiles! (terdragon))



0000000 . . .



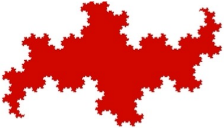
001001001 . . .



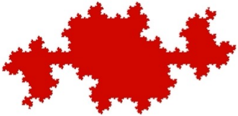
01010101 . . .

Now add the Truchet / Smith designs

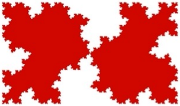
- Hexagons:



0000000 . . .

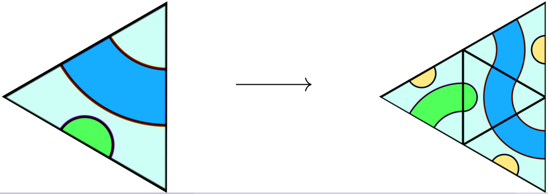


001001001 . . .

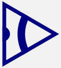


01010101 . . .

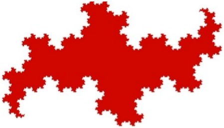
- Triangles, case 1: (individual tile, one operation)



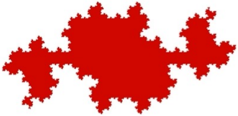
Now add the Truchet / Smith designs

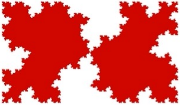
- Hexagons:



0000000 . . .



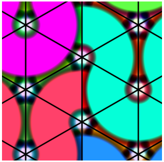
001001001 . . .



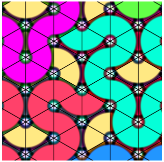
01010101 . . .

- Triangles, case 1: (several tiles, several operations)

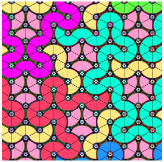
$t = 0$



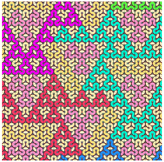
$t = 1$



$t = 2$



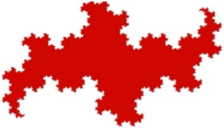
$t = 3$



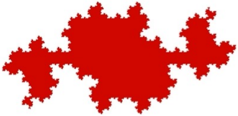
$t = 4$

Now add the Truchet / Smith designs

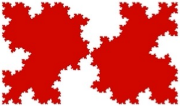
- Hexagons:



0000000 . . .

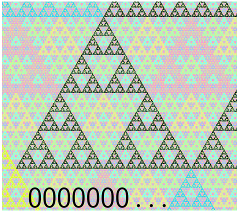


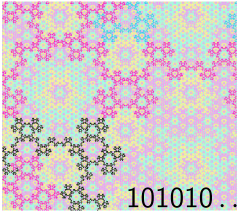
001001001 . . .

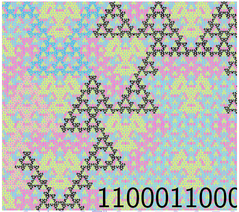


01010101 . . .

- Triangles, case 1: (many operations ⇒ Sierpinski variations)

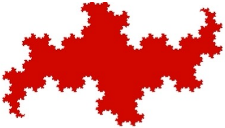




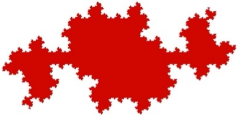


Now add the Truchet / Smith designs

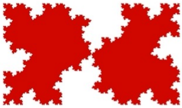
- Hexagons:



0000000 . . .

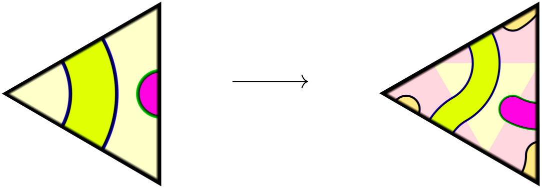


001001001 . . .



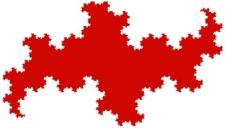
01010101 . . .

- Triangles, case 2: (individual tile, one operation)

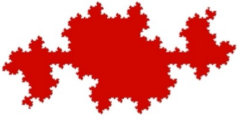


Now add the Truchet / Smith designs

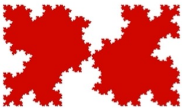
- Hexagons:



0000000 . . .

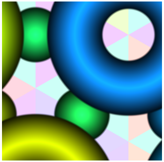


001001001 . . .

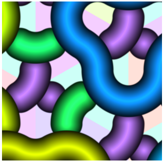


01010101 . . .

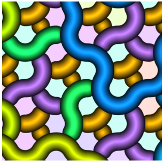
- Triangles, case 2: (several tiles, several operations)



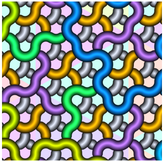
$t = 0$



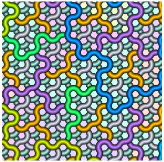
$t = 1$



$t = 2$

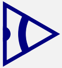


$t = 3$

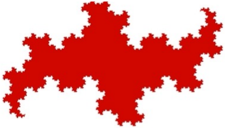


$t = 4$

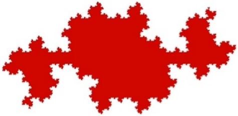
Now add the Truchet / Smith designs



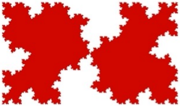
- Hexagons:



0000000 . . .

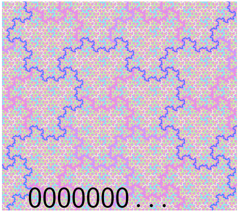


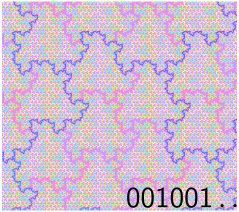
001001001 . . .

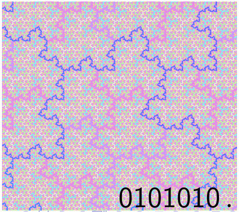


01010101 . . .

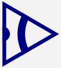
- Triangles, case 2: (many operations ⇒ terdragon boundary)



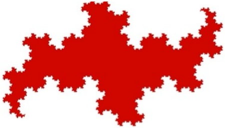




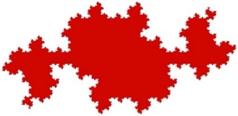
Now add the Truchet / Smith designs

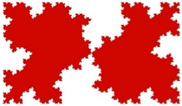
- Hexagons:



0000000 . . .

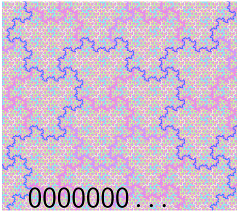


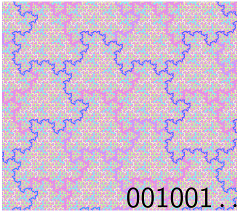
001001001 . . .

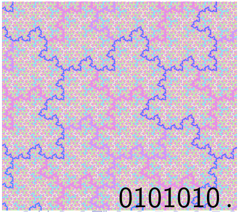


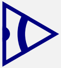
01010101 . . .

- Triangles, case 2: (many operations ⇒ fudge flake)

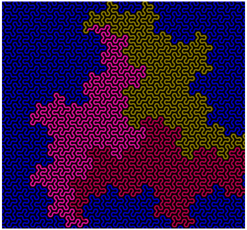
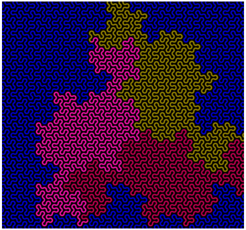
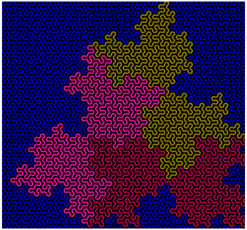




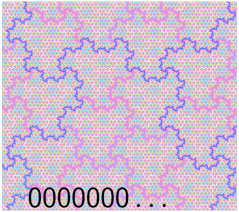


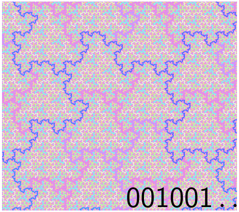
Now add the Truchet / Smith designs  

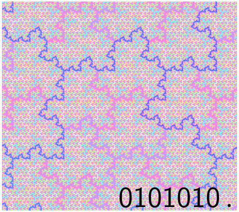
- Hexagons:

- Triangles, case 2: (many operations ⇒ fudge flake)

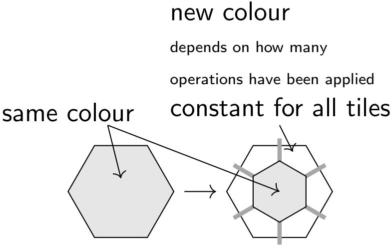




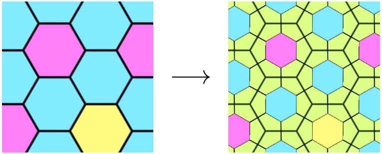


Digression: Fun with undecorated hinged hexagon tiles

- How should we colour the tiles? <u>“correct”</u>:

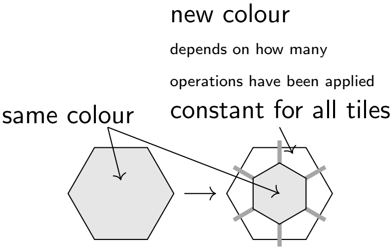


example: new colour is green

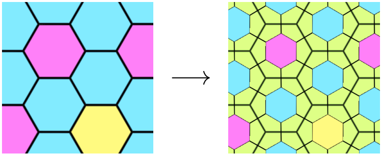


Digression: Fun with undecorated hinged hexagon tiles

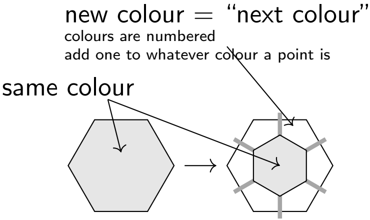
- How should we colour the tiles? "correct":



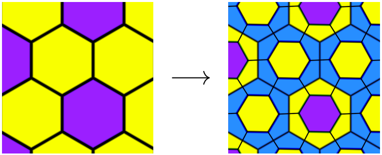
example: new colour is green



- "wrong":



(0) purple; (1) yellow; (2) blue



Digression: Fun with undecorated hinged hexagon tiles

- How should we colour the tiles? "correct":

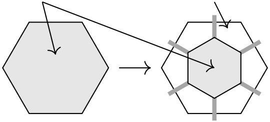
new colour

depends on how many

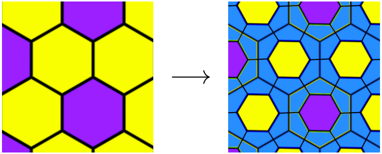
operations have been applied

constant for all tiles

same colour



example: new colour is blue



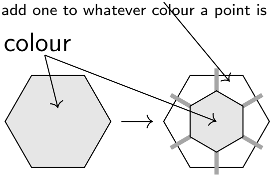
- "wrong":

new colour = "next colour"

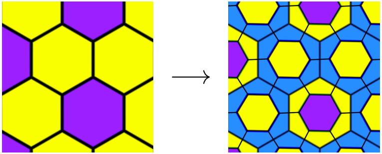
colours are numbered

add one to whatever colour a point is

same colour



(0) purple; (1) yellow; (2) blue

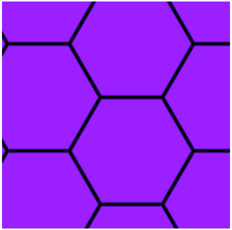


Koch snowflake

- Let's apply the "wrong" "colour + 1" rule:

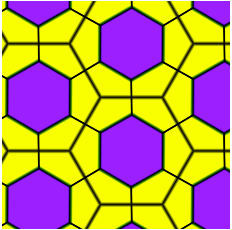
Koch snowflake

- Let's apply the "wrong" "colour + 1" rule:
-



$t = 0$

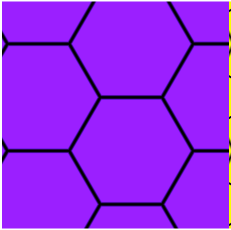
Koch snowflake

- Let's apply the "wrong" "colour + 1" rule:
- 

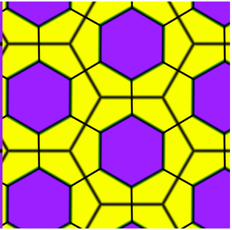
$t = 1$

Koch snowflake

- Let's apply the "wrong" "colour + 1" rule:
-

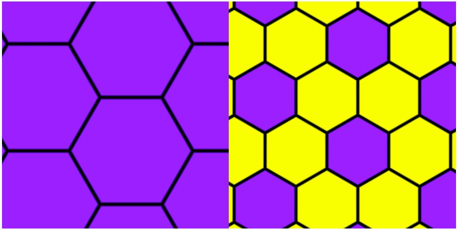


$t = 0$



$t = 1$

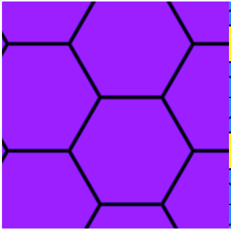
Koch snowflake

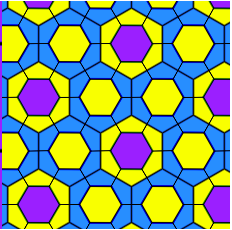
- Let's apply the "wrong" "colour + 1" rule:
- 

$t = 0$ $t = 1$

Koch snowflake

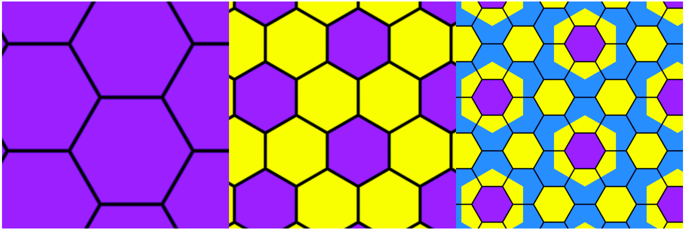
- Let's apply the "wrong" "colour + 1" rule:
-





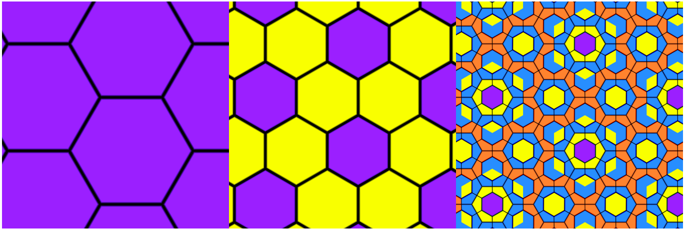
$t = 0$ $t = 2$

Koch snowflake

- Let's apply the "wrong" "colour + 1" rule:
- 

$t = 0$ $t = 1$ $t = 2$

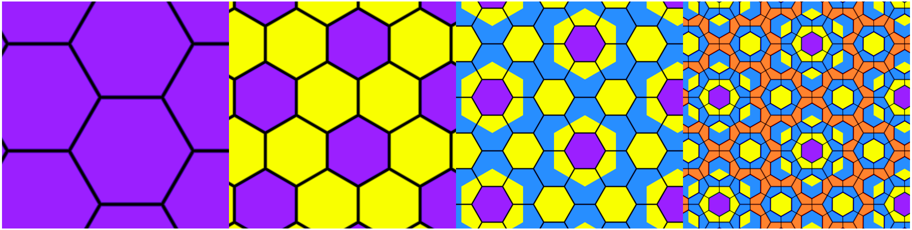
Koch snowflake

- Let's apply the "wrong" "colour + 1" rule:
- 

$t = 0$ $t = 1$ $t = 3$

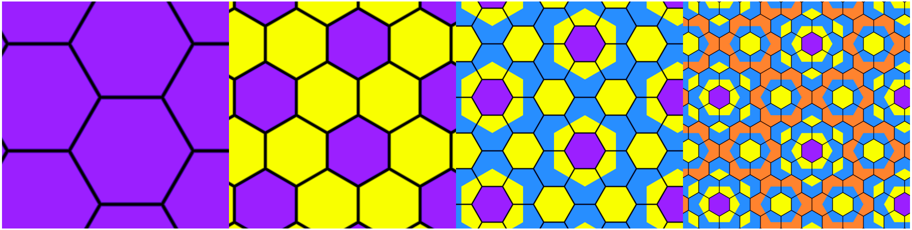
Koch snowflake

- Let's apply the "wrong" "colour + 1" rule:
-



$t = 0$ $t = 1$ $t = 2$ $t = 3$

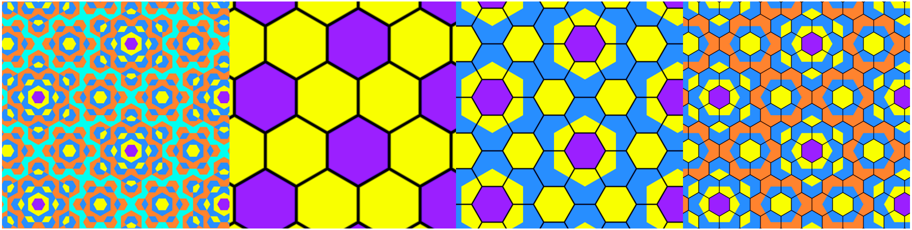
Koch snowflake

- Let's apply the "wrong" "colour + 1" rule:
- 

$t = 0$ $t = 1$ $t = 2$ $t = 3$

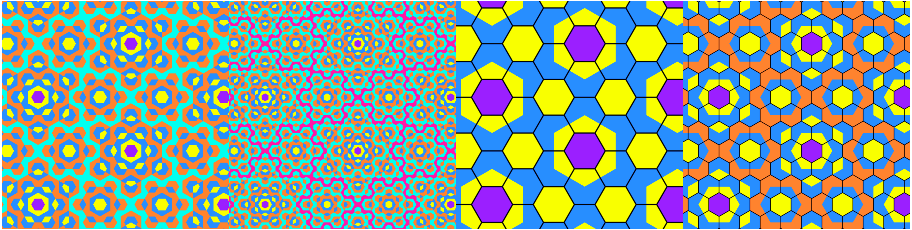
Koch snowflake

- Let's apply the "wrong" "colour + 1" rule:
-



$t = 4$ $t = 1$ $t = 2$ $t = 3$

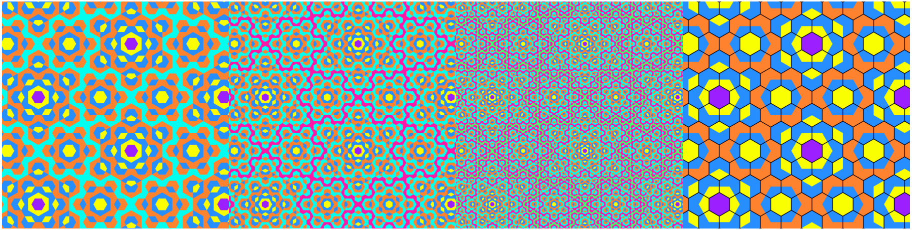
Koch snowflake

- Let's apply the "wrong" "colour + 1" rule:
- 

$t = 4$ $t = 5$ $t = 2$ $t = 3$

Koch snowflake

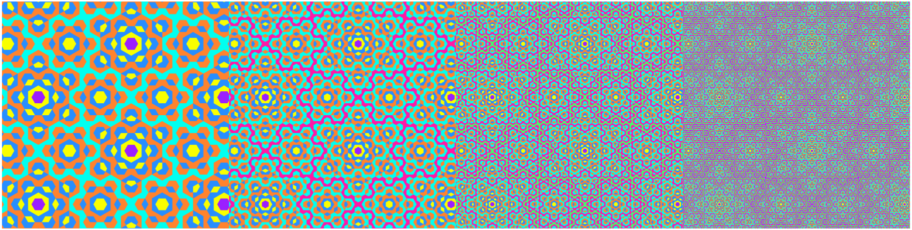
- Let's apply the "wrong" "colour + 1" rule:
-



$t = 4$ $t = 5$ $t = 6$ $t = 3$

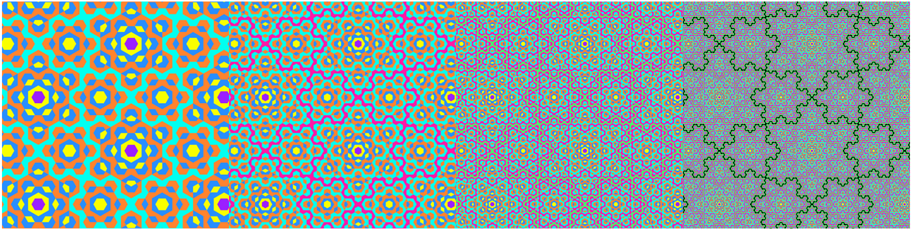
Koch snowflake

- Let's apply the "wrong" "colour + 1" rule:
-



$t = 4$ $t = 5$ $t = 6$ $t = 7$

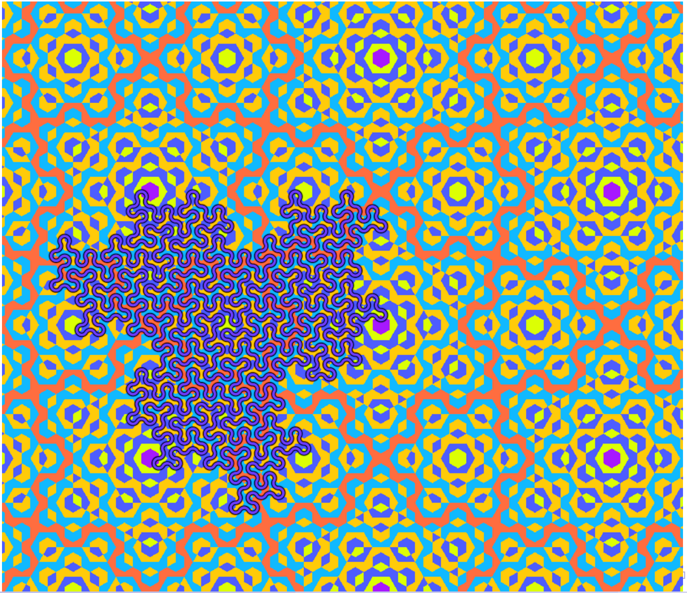
Koch snowflake

- Let's apply the "wrong" "colour + 1" rule:
- 

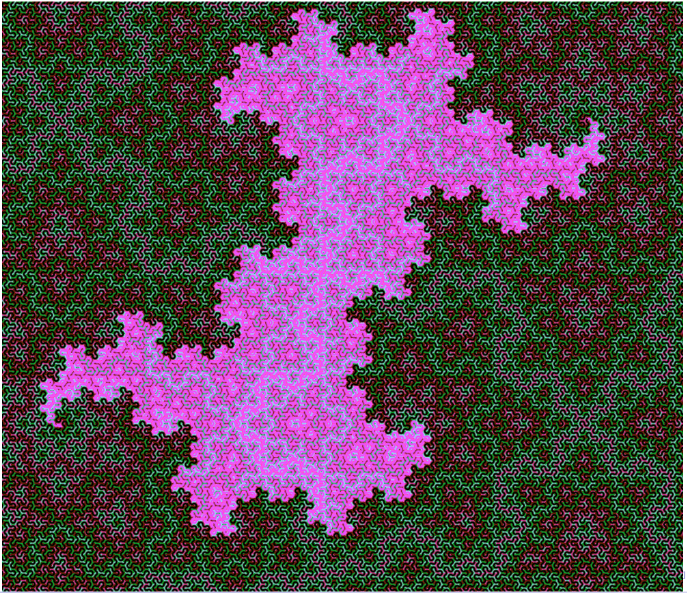
$t = 4$ $t = 5$ $t = 6$ $t = 7$

- A tessellation of Koch snowflakes appears

Terdragon and Koch snowflake examples



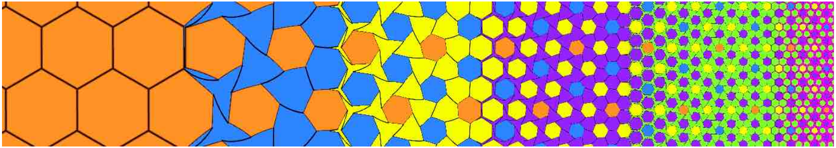
Terdragon and Koch snowflake examples



Left out; hints of ideas

No time to cover:

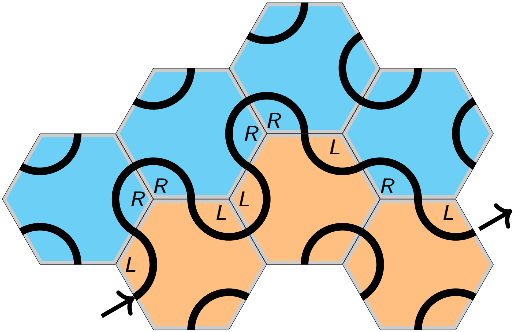
- Varying operation iteration level accross image



Left out; hints of ideas

No time to cover:

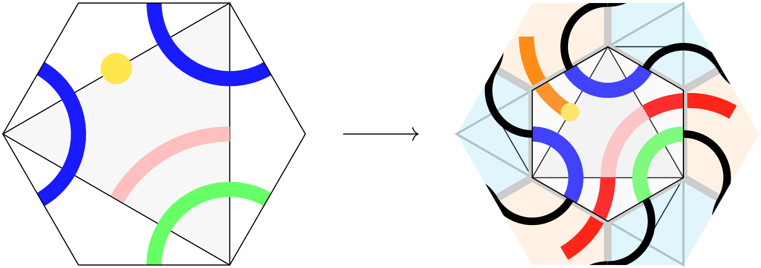
- Varying operation iteration level accross image
- L-systems, used to describe these fractals (replacement rule; symbols describe paths)



Left out; hints of ideas

No time to cover:

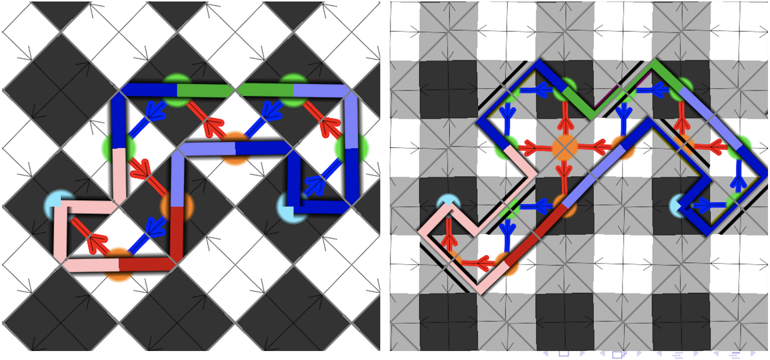
- Varying operation iteration level accross image
- L-systems, used to describe these fractals (replacement rule; symbols describe paths)
- How the terdragon and its boundary are related via the tilings



Left out; hints of ideas

No time to cover:

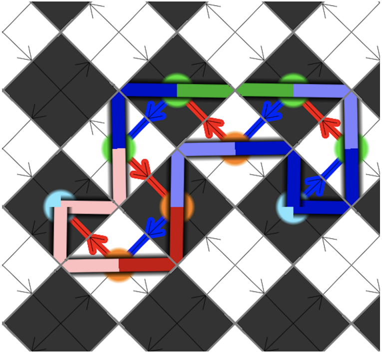
- Varying operation iteration level accross image
- L-systems, used to describe these fractals (replacement rule; symbols describe paths)
- How the terdragon and its boundary are related via the tilings
- Extention to get boundary L-systems for Heighway dragon and certain square grid based space filling curves

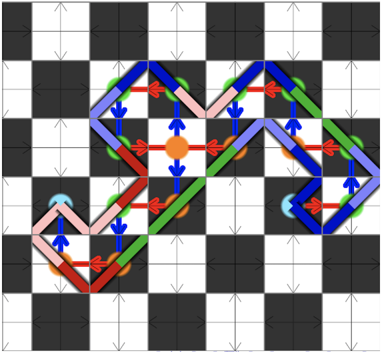


Left out; hints of ideas

No time to cover:

- Varying operation iteration level accross image
- L-systems, used to describe these fractals (replacement rule; symbols describe paths)
- How the terdragon and its boundary are related via the tilings
- Extention to get boundary L-systems for Heighway dragon and certain square grid based space filling curves

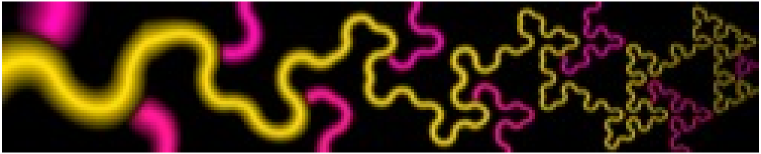




Left out; hints of ideas

No time to cover:

- Varying operation iteration level accross image
- L-systems, used to describe these fractals (replacement rule; symbols describe paths)
- How the terdragon and its boundary are related via the tilings
- Extention to get boundary L-systems for Heighway dragon and certain square grid based space filling curves
- You can read more at `https://www.mathamaze.co.uk/Truchet3/`



End of talk

Thank you for listening!