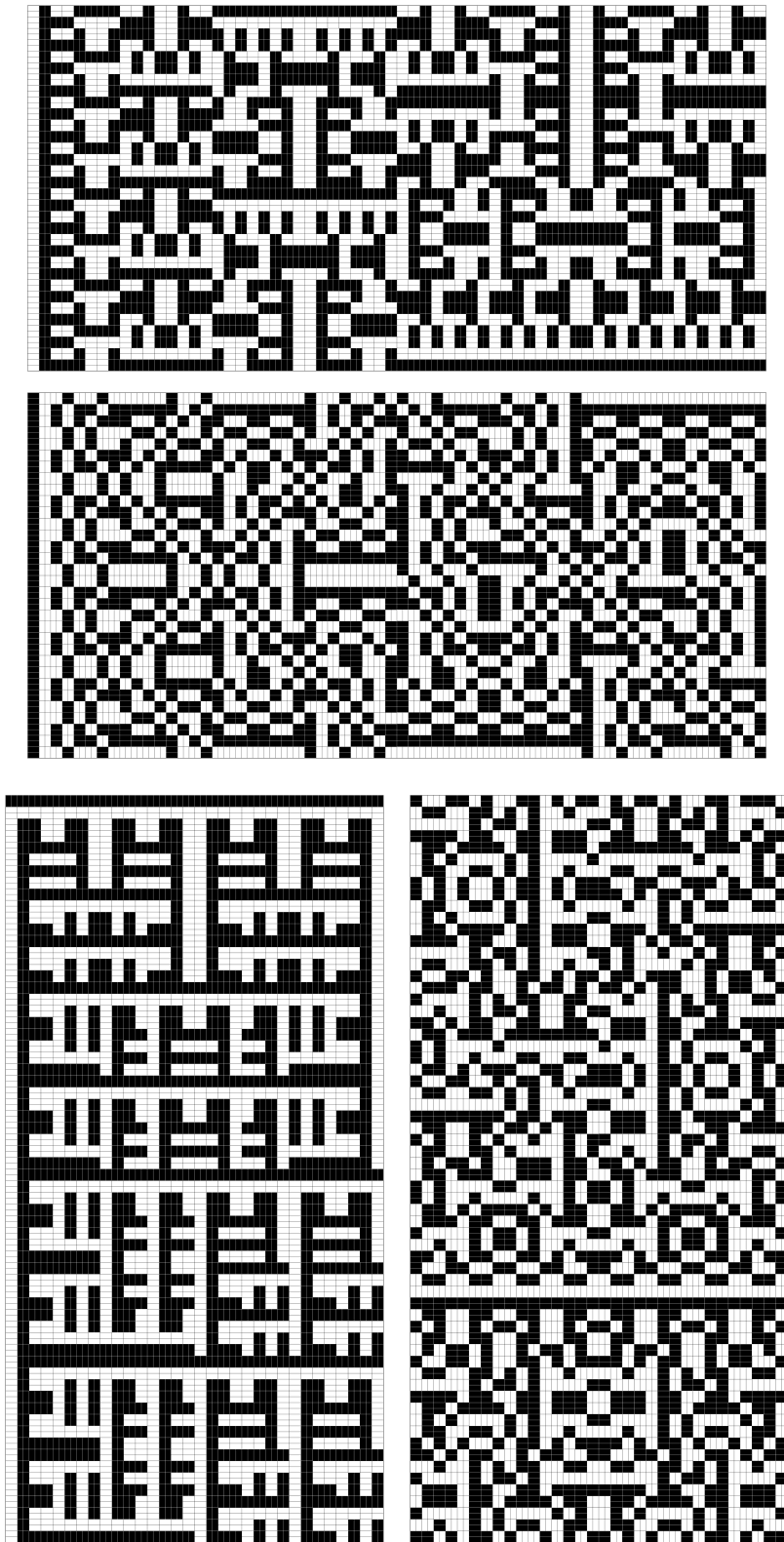


a lot of tiles (trivial scan)

4096 variations on 256 tiles dedicated to Chaim Goodman-Strauss and Omar Lopez
a performance-installation



a lot of tiles (trivial scan)

4096 variations on 256 tiles dedicated to Chaim Goodman-Strauss and Omar Lopez
a performance-installation

michael winter (cdmx, mx and nashville, usa; 2018)

Program notes

This piece is based on a set of rectangle substitution tilings (commonly referred to as tessellations) explored by Chaim Goodman-Strauss in his seminal paper *Lots of aperiodic sets of tiles*. A rectangle substitution tiling is generated by dissecting a rectangle into 4 smaller rectangles, which are then dissected into 8 even smaller rectangles, and so on. Rectangles can produce both periodic and non-periodic tilings and thus are not strictly aperiodic. A strictly aperiodic set of tiles consists of a group of geometric shapes that submit only non-periodic tilings. However, Goodman-Strauss shows in his paper that the group of rectangle substitution tilings used for this piece can be represented by 25380 aperiodic tilings using subsets of 211 non-rectangular tiles. Prior to this discovery, only a handful of strictly aperiodic tilings were known. That is, in one fell swoop, Goodman-Strauss extended the number of known aperiodic tilings by some 500+ fold.

In order to proceed with his proof of the newly discovered 25380 aperiodic tilings, Goodman-Strauss outlines how to generate the 256 rectangle substitution tilings which they represent. This piece extends his method of generating these 256 rectangle tilings to create 4096 sonic and visual variations. The parenthetical in the title ‘trivial scan’ refers to the method of sonification. A given tiling is further dissected into squares (groups of two rectangles side-by-side) which are then scanned / read such that the orientation of the pairs of subtiles which form the squares (whether they are grouped vertically or horizontally) determines sonic parameters of the piece.

Setting

The piece consists of both a visualization and sonification of the tilings generated and projected by a program written in SuperCollider. The program has a ‘continuous play’ mode for an installation setting that generates random tilings from the 4096 variations in succession. Live performers can also accompany the electronic realization of a tiling as detailed below. This allows the installation to be occasionally augmented by live performances and enables any subset of the variations (preferably at least 3) to be played in a concert setting as well. For the latter, projection of the visualization is optional.

Generally, the setting should be dark and the sonification should be clear and present.

Tile variations (transform code)

Each of the 4046 variations can be represented by a transform code of 7 digits. The first 4 digits generate the 256 archetypal tilings (visualizations of these are appended to this document) by determining the orientation of each of the 4 subtiles relative to its parent tile. This is the method outlined by Goodman Straus in his paper. The 5th and 6th digits determine the orientation and mirroring of the entire tiling. As mentioned in the program note, each tiling is further dissected into squares and scanned. The read-head always scans from left to right and top to bottom. Thus, the 5th and 6th digits of the transform code ultimately result in different read sequences. The 7th digit determines the color inversion of the subtiles: the color (black / white) based on the orientation (vertical / horizontal). These are turned into numeric values for sonic parameters.

Tile sonification (the scan)

The tiling is generated up to a depth of 6 substitutions / hierarchical levels (hls). While the tile visualization is hopefully self-explanatory, the sonification warrants detailed instructions.

hl 6 (fundamental): The maximum depth of 6 substitutions of the tiling is sonified by a highly controlled tremolo on a pitched tone sounding a low g (2 octaves and a perfect fourth below middle c or a frequency of approximately 49 hertz). The scanned white and black values map to amplitudes blurred by an exponential lag (see the SuperCollider UGen ‘Lag’). This means that amplitude curves are not linear but logarithmic.

The computer generates this tone with a sine wave oscillator. The sine tone may be doubled or replaced by a low reed woodwind instrument (such as a bass clarinet, baritone saxophone, bassoon, or contrabassoon) sounding the lowest g comfortably playable in the instrument’s range. In the score, which is engraved using the Lilypond typesetting language, a gray-scale curved gradient shows the amplitude of the tremolo such that completely black denotes the maximum volume and completely white denotes silence. Below the gradient is a rhythmic notation where the duration of each note corresponds to the amount of time before the curve changes direction (from louder to softer or softer to louder). The tempo for the quarter note strictly equals 120 beats per minute making each variation last just over 4 minutes and 15 seconds. Below each note is a numeric value indicating the target amplitude from the preceding note between: between 0 (silent) and 9 (maximum volume). For longer note durations, the curve will potentially reach the loudest or softest level before the duration indicated by the note (that is, before the volume changes direction again). The very first note of each tiling may indicate a silence if the upper left corner is a white tile which results in a starting amplitude of 0. If the upper left corner of the tile is black, this part starts at full scale / volume immediately with no fade in. However, each variation ends with a fade out some time before the end of the final measure. Performers should use the electronic tone from the application as a reference for the dynamic profile of the tremolo.

In order to breath, the performer may fade to silence on any note over which a decrease in amplitude occurs and then reenter fading in on any note in which an increase in amplitude occurs.

This should be the most present sonic element of the piece; loud and clear.

hl 6 (harmonics): The fundamental is colored by a set of 3 harmonics (the 5th, 9th, and 13th) that sonify adjacent rows of the tiling. In this sense, the piece contains a sort of a canon. The fundamental sonifies the tiling starting on the 1st row, the 5th harmonic sonifies the tiling starting on the 2nd row, and so on. While the blurring method is the same, the maximum amplitudes of each harmonic decrease directly in correspondence with the harmonic number (1 divided by the harmonic number squared). By default these parts are not included in the score as they are not played by live performers. However, there are lines in the Lilypond file that can be uncommented to view the parts. (Note that the amplitude gradient for these parts is normalized and does not represent the decreasing amplitudes of the harmonics.)

The dynamic of this part should be well below the fundamental (see more in the explanation of the SuperCollider interface). That is, the correspondence between the harmonic number and the decrease in amplitude is maintained between the harmonics, but not in relations to the fundamental, which should sound much louder. Though it is highly preferred to play the piece with the synthesized sounds, a strictly acoustic realization could be played with the harmonics omitted.

hl 5 and 4 (high noise and low noise, respectively): High and low noises sonify the 5th and 4th substitutions / hls, respectively. The SuperCollider application synthesizes these with filtered white noise: using a high pass filter with a cutoff of 5000 hertz for the high noise and a low pass filter with a cutoff of 300 hertz for the low noise. Unlike hl 6, the black / white values are not blurred. A white value results in a soft noise and a black value results in a slightly louder noise. That is, there is no smooth transition, the volume change is abrupt and binary.

Like the fundamental of hl 6, these parts can be doubled / replaced by acoustic instruments such as percussion. Pitched instruments can also be used as long as the resulting noise does not have a clearly defined pitch. In the score, the notes of these parts are accompanied by a numeric value, 0 or 1, denoting the two states (soft or slightly louder, respectively) for the duration of that note. Like the electronics, the change should be abrupt; highlighting the binary nature of the underlying structure. Performers are free to explore different methods of sonifying the two states by different types of sounds (not just considering the dynamic parameter).

These parts should be considered as secondary to the fundamental of hl 6 and any decision moving away from the synthesized version of high and low noises should maintain the subtlety of these parts in relation to the fundamental of hl 6.

SuperCollider program

The application contains a graphical user interface (gui) and a window for the visualization of the tiling. The gui has three tabbed panes shown on the following page: a ‘model’ pane, a ‘transport’ pane, and a ‘mixer’ pane. These are explained in more detail below. To launch the application, execute `supercollider/a_lot_of_tiles_trivial_scan_main.scd` in SuperCollider after booting the server (on Linux, press `ctrl+enter` with the cursor anywhere within the code block to execute the code).

The source code for the application is appended at the end of this score and can also be downloaded from a git repository at https://www.github.com/mwinter80/a_lot_of_tiles_trivial_scan. The generation of this document (using LaTeX) contains a version date in order to help track changes and the git repository will also detail commit changes. The piece was written using SuperCollider version 3.9.0 and Lilypond version 2.19.81.

Model pane: This pane allows the user to manipulate the transform code (which automatically updates the tile visualization), control the layout of the tile visualization, and provides display options for the tile visualization window. There are three additional buttons on the bottom right to start a ‘continuous play’ mode for an installation settings (note that this will disable the transform code buttons), to generate the Lilypond files for the current tiling / transform, and to export an image of the current tiling / transform.

There are four layout options: one for each of the hls and another view that embeds the hls together. The display options allow the user to enter into fullscreen mode (while the escape key will exit fullscreen mode). Deselecting the ‘window decorator’ button will remove the window border from the tile visualization window. This can be used to launch several instances of the application and project more than one tile visualization using the same projector.

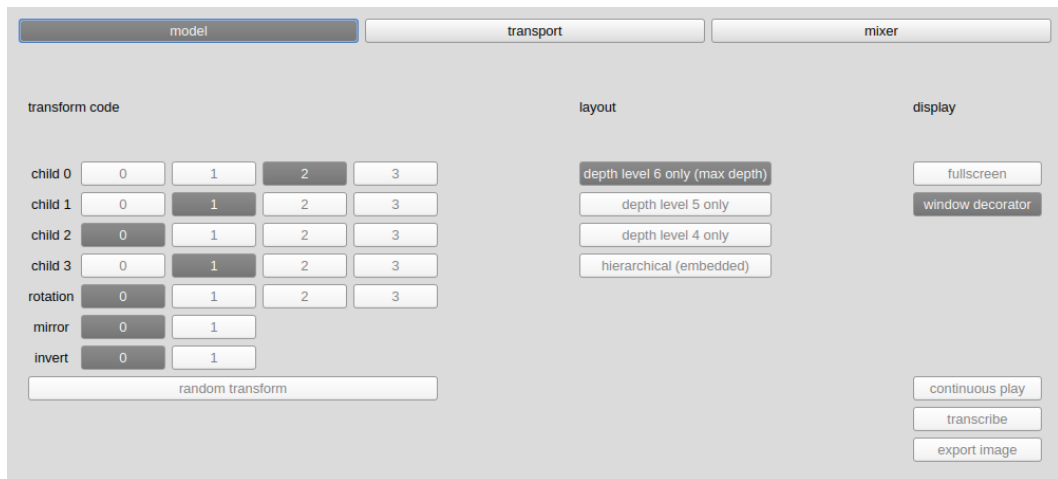
The ‘transcribe’ button will generate all the Lilypond files for the master score and the parts in the `score/transform_<code>_score/lilypond/` folder (where `<code>` is the digit sequence of the transform code; for example, `score/transform_2101000_score/lilypond/`). It will also attempt to run a Lilypond executable if it is installed in order to automatically create pdfs of the score in the folder `score/transform_<code>_score/pdf/`. Scores of 4 of the variations are included in this document.

Transport pane: This pane has playback controls for the current tiling / transform allowing the user to start from different locations. When the ‘play’ button is pressed, the application will check if the necessary audio files exist. If not, it will generate them in the `audio/transform_<code>_audio/` folder (this will delay the start a small amount of time). The generated audio files can be used for playback in an alternative environment such as a digital audio workstation. Note that if the application is in ‘continuous play’ mode, the audio files will be deleted as to not consume too much storage space.

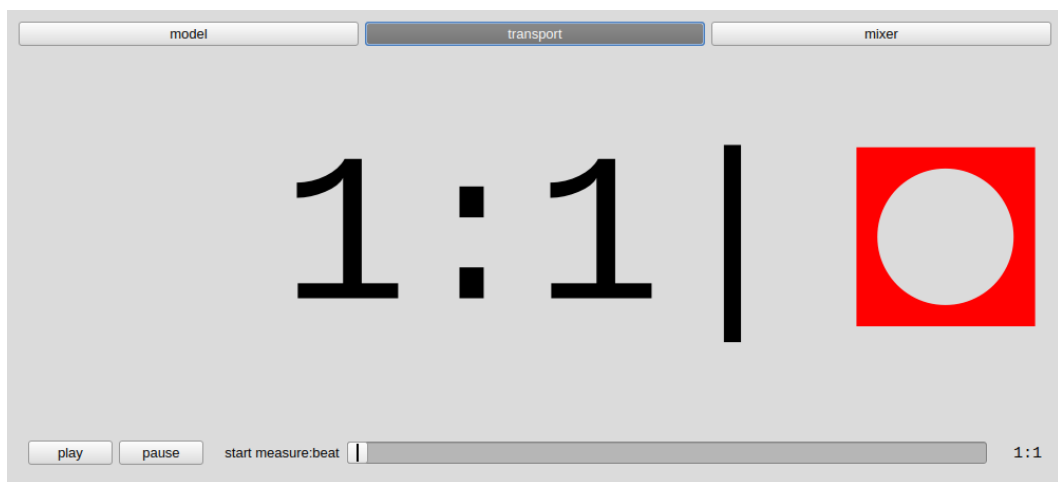
The pane also contains a visual metronome which displays the current measure and beat with respect to the score. When the ‘play’ button is pressed, the metronome will give two measures of silence before starting.

Mixer pane: This pane provides controls for playback levels of the different sonic elements allowing the user to ensure that the fundamental of hl 6 is the most present sonic element.

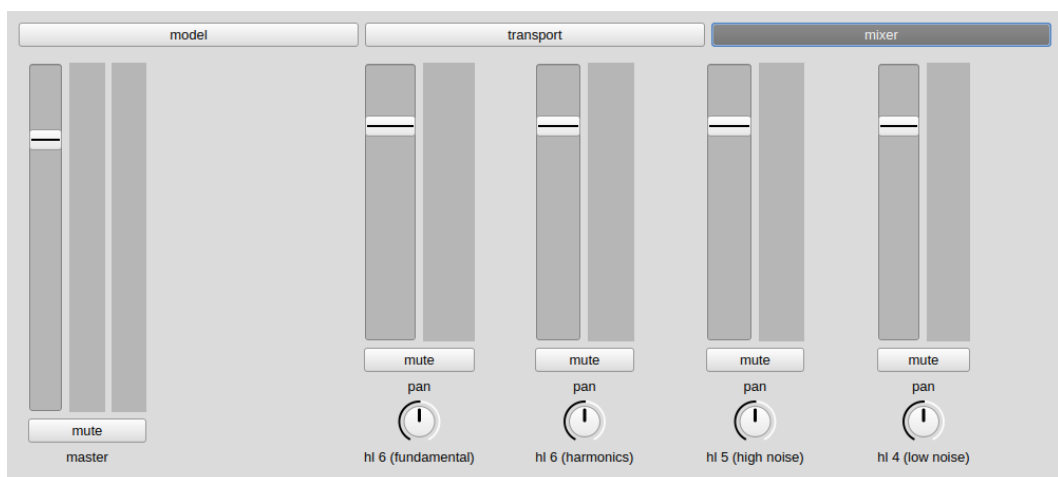
I would like to extend a special thanks to James Harkins and Thomas Morley for their generous help with SuperCollider and Lilypond, respectively.



model pane



transport pane



mixer pane

a lot of tiles (trivial scan)

2101000

michael winter

(cdmx, mx and nashville, usa; 2018)

master score - 2101000

hl 6 (harm 1)

hl 5 (high noise)

hl 4 (low noise)

0 0 3 1 8 3 5 2 5 2 9 4 6 2 5 2 8 3 5 2 5 2 8 3 5 2 5

⑤

hl6 (h1)

hl5 (hn)

hl4 (ln)

1 7 1 7 3 8 0 7 3 8 1 7 3 8 1 7 3

⑨

hl6 (h1)

hl5 (hn)

hl4 (ln)

8 5 6 2 5 2 8 3 8 5 7 4 6 4 6 2 5 3 5 3 5 3 8 3 8 3 5 2 5 2 4 2 4 2 8 3

⑬

hl6 (h1)

hl5 (hn)

hl4 (ln)

8 5 9 3 6 2 5 2 4 2 4 3 5 3 5 3 5 2 5 3 5 3 5 3 5 5 2 5 2 4 2 4 2 8 3 8 3 5 2 5 2 4

⑪

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑫

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑬

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑭

hl6 (h1)

hl5 (hn)

hl4 (ln)

③③

hl6 (h1)

hl5 (hn)

hl4 (ln)

③⑦

hl6 (h1)

hl5 (hn)

hl4 (ln)

④①

hl6 (h1)

hl5 (hn)

hl4 (ln)

④⑤

hl6 (h1)

hl5 (hn)

hl4 (ln)

④9

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑤3

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑤7

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑥1

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑥5

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑥9

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑦3

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑦7

hl6 (h1)

hl5 (hn)

hl4 (ln)

81

hl6 (h1)

hl5 (hn)

hl4 (ln)

85

hl6 (h1)

hl5 (hn)

hl4 (ln)

89

hl6 (h1)

hl5 (hn)

hl4 (ln)

93

hl6 (h1)

hl5 (hn)

hl4 (ln)

97

hl6 (h1)

hl5 (hn)

hl4 (ln)

101

hl6 (h1)

hl5 (hn)

hl4 (ln)

105

hl6 (h1)

hl5 (hn)

hl4 (ln)

109

hl6 (h1)

hl5 (hn)

hl4 (ln)

113

hl6 (h1)

hl5 (hn)

hl4 (ln)

117

hl6 (h1)

hl5 (hn)

hl4 (ln)

121

hl6 (h1)

hl5 (hn)

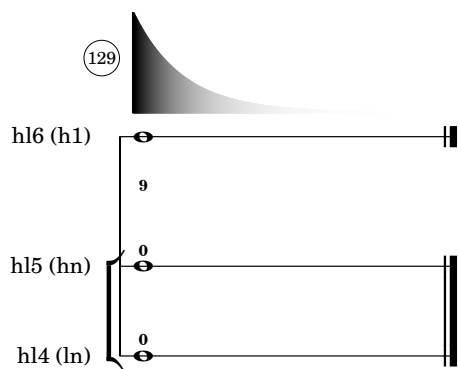
hl4 (ln)

125

hl6 (h1)

hl5 (hn)

hl4 (ln)



a lot of tiles (trivial scan)

2101000

michael winter

(cdmx, mx and nashville, usa; 2018)

part - hl 6 - 2101000

hl 6
(harm 1)

0 0 3 1 8 3 5 2 5 2 9 4 6 2 5 2 8 3 5 2 5 2 8 3 5 2 5

⑤

hl6 (h1)

1 7 1 7 3 8 0 7 3 8 1 7 3 8 1 7 3

⑨

hl6 (h1)

8 5 6 2 5 2 8 3 8 5 7 4 6 4 6 2 5 3 5 3 5 3 8 3 8 3 5 2 5 2 4 2 4 2 8 3

⑬

hl6 (h1)

8 5 9 3 6 2 5 2 4 2 4 3 5 3 5 3 5 2 5 3 5 3 5 3 5 2 5 2 4 2 4 2 8 3 8 3 5 2 5 2 4

⑰

hl6 (h1)

1 4 1 8 5 7 4 7 5 6 2 5 1 4 0 4 1 4 1 4 3 7 4 6 4 8 3 5 2 5 2 8 5 7 4 7 5 6

⑳

hl6 (h1)

2 8 0 4 2 6 4 6 2 8 5 9 6 8 3 5 3 7 4 6 0 7 3 8 0 4 2 6 4 6

㉔

hl6 (h1)

2 5 2 4 2 4 0 7 4 9 6 8 0 3 1 4 1 4 2 4 2 4 2 4

㉘

hl6 (h1)

0 8 3 9 6 7 2 4 0 4 1 4 2 9 4 8 3 8 3

33

hl6 (h1)

9 6 7 3 8 3 5 2 5 2 4 2 8 3 8 3 9 4 8 3 8 3

37

hl6 (h1)

9 6 9 1 7 3 8 3 5 2 5 2 4 2 4 2 4 0 3 1 4 1 4 1 4 2 4 2 4

41

hl6 (h1)

0 3 1 4 1 8 3 8 1 7 3 8 0 4 2 6 4 6 0 7 3 8 0 4 2 6 4 6

45

hl6 (h1)

2 8 3 5 2 5 2 4 2 8 3 5 2 5 2 8 5 7 4 7 5 6 4 9 3 5 2 5 2 8 5 7 4 7 5 6

49

hl6 (h1)

2 5 2 8 5 7 4 7 5 6 4 8 3 5 2 5 2 8 3 5 2 5 2 4 2 8 3 8 3 5 2 5 2 4

53

hl6 (h1)

1 7 0 4 2 6 4 6 0 7 3 8 1 7 3 8 3 5 2 5 2 4 2 4 2 8 3

57

hl6 (h1)

8 5 6 2 5 2 5 0 3 1 4 1 4 2 4 2 4 2 8 3 8 1 7 3 8 1 7 3

61

hl6 (h1)

8 5 9 3 9 4 8 3 8 3 5 2 5 2 4 2 8 3 5 2 5 2 8 3 5 2 5

⑥5

hl6 (h1)

1 4 1 8 3 5 2 5 2 9 6 9 3 6 3 8 3 7 3 8 5 7 3 8

⑥9

hl6 (h1)

3 8 1 7 3 8 0 3 1 4 1 4 3 5 2 5 2 6 2 5 2 4 3 5 2 5 2 4

⑦3

hl6 (h1)

2 4 2 4 2 8 3 8 5 7 4 6 4 6 2 5 3 5 3 5 3 5 3 8 0 7 0 7 0 7

⑦7

hl6 (h1)

3 8 3 5 2 5 2 4 2 4 3 5 3 5 3 5 2 5 3 5 3 5 3 5 3 6 2 8 5 7 3 9 4 6 4 8 3 5

⑧1

hl6 (h1)

2 5 2 8 5 7 4 7 5 6 2 5 1 4 0 4 1 4 1 4 1 8 5 7 3 9 4 6 4 8 3 5

⑧5

hl6 (h1)

2 7 0 4 2 6 4 6 2 8 5 9 6 8 3 8 0 7 0 7 0 7

⑧9

hl6 (h1)

3 5 2 5 2 4 0 7 4 9 6 8 3 5 2 5 2 4 3 5 2 5 2 6 2 5 2 4 3 5 2 5 2 4

⑨3

hl6 (h1)

2 8 3 9 6 7 2 4 0 4 1 4 1 8 3 5 3 8 3 7 3 8 5 7 3 8

97

hl6 (h1)

3 5 2 8 3 5 2 5 2 4 2 8 3 8 3 5 2 5 1 4 1 4 1 4 1 4 1 4 1 4

101

hl6 (h1)

1 7 1 7 3 8 3 5 2 5 2 4 2 4 2 8 5 8 5 8 5 8 3 8 5 8 5 8 5

105

hl6 (h1)

8 5 7 3 5 2 8 3 8 1 7 3 8 1 7 4 8 5 8 5 8 3 8 5 8 5 8 5

109

hl6 (h1)

8 5 9 3 6 2 5 2 4 2 8 3 5 2 5 2 8 3 5 1 4 1 4 1 4 1 4 1 4 1 4

113

hl6 (h1)

1 4 1 8 5 7 4 7 5 6 4 8 3 5 2 5 2 8 5 7 4 6 4 6 4 6 4 6 3 6 3 6 3 7 4 6 4 6 4 6 4 6 3 6 3 6 3 6

117

hl6 (h1)

2 7 0 4 2 6 4 6 0 7 3 8 0 4 2 5 3 5 3 5 3 6 3 6 3 6 3 7 4 6 4 6 4 6 4 6 3 6 3 6 3 6

121

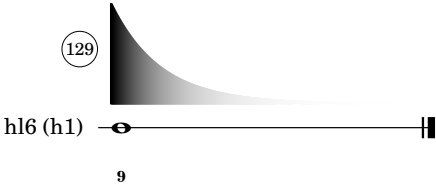
hl6 (h1)

2 5 2 4 2 4 0 3 1 4 1 4 2 4 2 4 2 4

125

hl6 (h1)

0 8 3 9 4 8 3 8 3



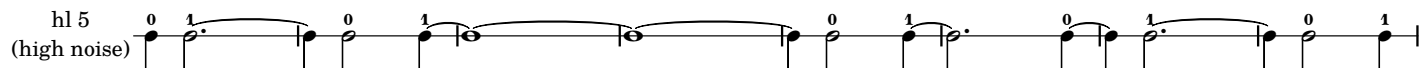
a lot of tiles (trivial scan)

2101000

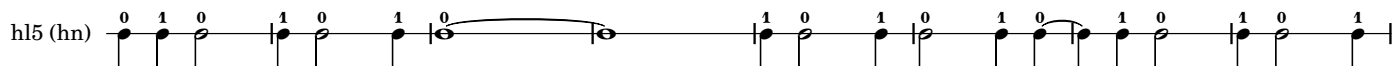
michael winter

(cdmx, mx and nashville, usa; 2018)

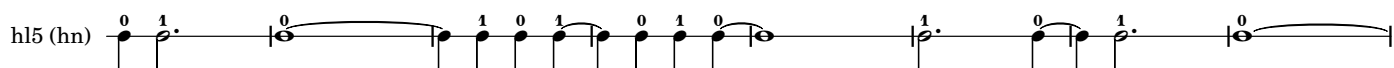
part - hl 5 - 2101000



⑨



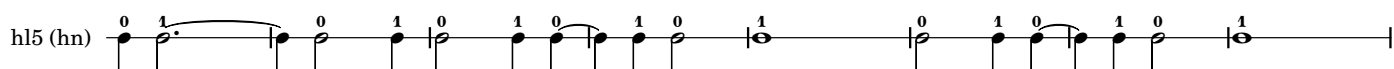
⑰



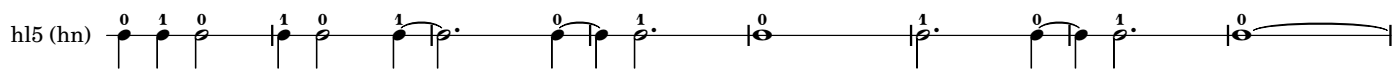
⑳



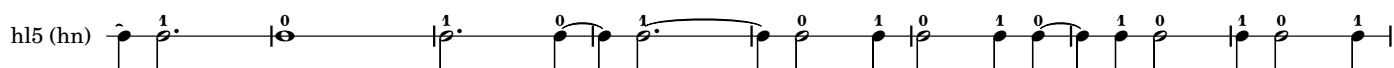
㉓



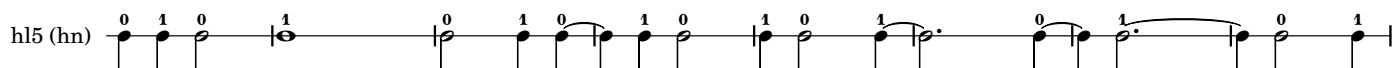
㉕



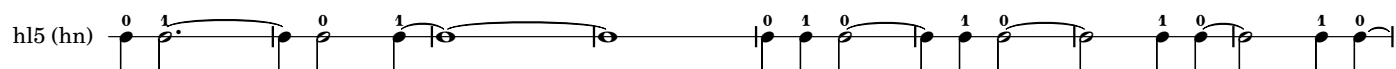
㉙



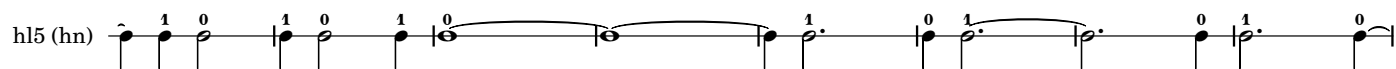
㉟



65



73



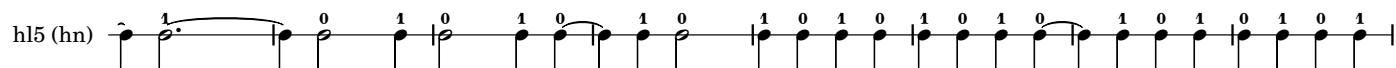
81



89



97



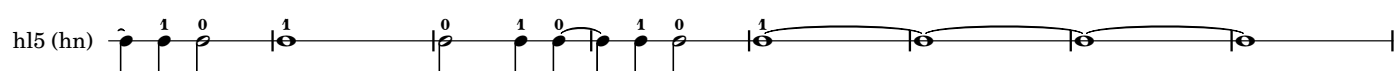
105



113



121



(part - hl 5 - 2101000)

129

hl5 (hn) $\overset{0}{\bullet} \text{---} \text{H}$

a lot of tiles (trivial scan)

2101000

michael winter

(cdmx, mx and nashville, usa; 2018)

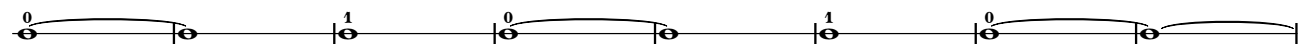
part - hl 4 - 2101000

hl 4
(low noise)



⑨

hl4 (ln)




⑰

hl4 (ln)



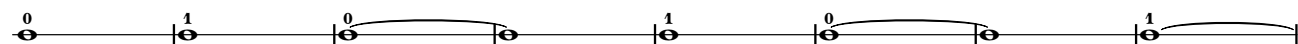
②⑤

hl4 (ln)



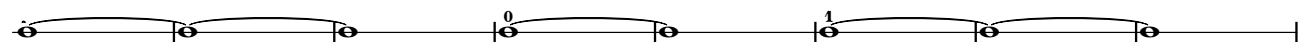
③③

hl4 (ln)



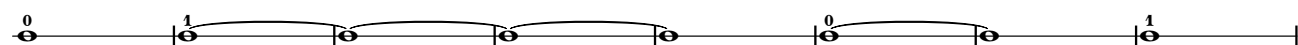
④①

hl4 (ln)




④⑨

hl4 (ln)



⑤⑦

hl4 (ln)



65

73

81

89

Diagram illustrating a 1D lattice structure with 7 sites. The sites are labeled with binary values: 1, 0, 1, 0, 1, 0. The first site is labeled $hl4 (ln)$ and has a dot above it. Curved lines connect the 3rd and 4th sites, and the 6th and 7th sites.

97

105

113

121

129

hl4 (ln) $\frac{0}{\mathbf{e}} \mathbf{H}$

a lot of tiles (trivial scan)

0111211

michael winter

(cdmx, mx and nashville, usa; 2018)

master score - 0111211

h1 6 (harm 1)

h1 5 (high noise)

h1 4 (low noise)

Measure 1: 9 9 4 6 2 5

Measure 2: 0 3 1 4

Measure 3: 0 3 1 4 1 4 3 5 2 5

Measure 4: 0 3 1 4 1 4

⑤

h1 6 (h1)

h1 5 (hn)

h1 4 (ln)

Measure 5: 0 3 2 4 3 7 4

Measure 6: 9 6 7 4

Measure 7: 9 6 7 5 7 5 6 4 6 4

Measure 8: 9 6 8 5 7 4

⑨

h1 6 (h1)

h1 5 (hn)

h1 4 (ln)

Measure 9: 9 6 7 5 6 4 8 3 5 3 5 3 7 3 7 2 7 4 6 4 6 4 8 5 7 4 6 2 7 5 6 4 6 4 7 3 7 3 7 2 7 2

Measure 10: 1 0 1 0 1 0 1 0 1 0 1 0 0 1 0 1 0

Measure 11: 1 0 1 0 1 0 1 0 1 0 1 0 0 1 0 1 0

Measure 12: 1 0 1 0 1 0 1 0 1 0 1 0 0 1 0 1 0

⑬

h1 6 (h1)

h1 5 (hn)

h1 4 (ln)

Measure 13: 7 2 5 3 5 1 6 4 6 2 5 3 7 3 7 4 6 2 5 3 5 2 5 2 4 3 7 1 4 3 5 2 5 3 7 3 7 2 7 2 7

Measure 14: 1 0 1 0 1 0 1 0 1 0 1 0 0 1 0 1 0

Measure 15: 1 0 1 0 1 0 1 0 1 0 1 0 0 1 0 1 0

Measure 16: 1 0 1 0 1 0 1 0 1 0 1 0 0 1 0 1 0

⑪

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑫

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑮

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑲

hl6 (h1)

hl5 (hn)

hl4 (ln)

③③

hl6 (h1)

hl5 (hn)

hl4 (ln)

③⑦

hl6 (h1)

hl5 (hn)

hl4 (ln)

④①

hl6 (h1)

hl5 (hn)

hl4 (ln)

④⑤

hl6 (h1)

hl5 (hn)

hl4 (ln)

④9

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑤3

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑤7

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑥1

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑥5

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑥9

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑦3

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑦7

hl6 (h1)

hl5 (hn)

hl4 (ln)

81

hl6 (h1)

hl5 (hn)

hl4 (ln)

85

hl6 (h1)

hl5 (hn)

hl4 (ln)

89

hl6 (h1)

hl5 (hn)

hl4 (ln)

93

hl6 (h1)

hl5 (hn)

hl4 (ln)

97

hl6 (h1)

hl5 (hn)

hl4 (ln)

101

hl6 (h1)

hl5 (hn)

hl4 (ln)

105

hl6 (h1)

9 6 7 4 6 4 8 5 7 4 6 2 6 4 6 2 5 3 8 5 6 4 6 4 7 4 6 2 5 3 7 4 6 2 5 3 7 4 6 4 6 4 8 5 7 4 6 2

hl5 (hn)

1 0 1 0 1 0 1 0 1 0 1 0 1 0 1

hl4 (ln)

109

hl6 (h1)

6 2 5 3 5 2 5 2 4 3 7 4 6 2 6 2 5 2 4 3 5 2 6 2 6 2 7 2 7 2 7 2 5 3 5 2 5 2 4 3 7

hl5 (hn)

0 1 0 1 0 1 0 1 0 1

hl4 (ln)

0 1 0

113

hl6 (h1)

hl5 (hn)

hl4 (ln)

117

hl6 (h1)

hl5 (hn)

hl4 (ln)

121

hl6 (h1)

hl5 (hn)

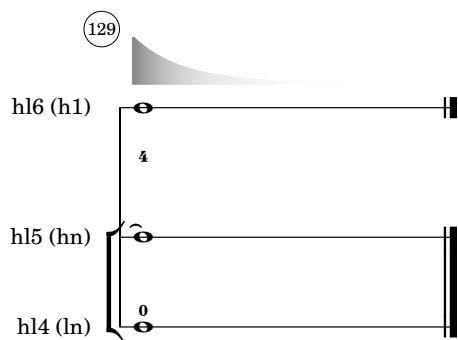
hl4 (ln)

125

hl6 (h1)

hl5 (hn)

hl4 (ln)



a lot of tiles (trivial scan)

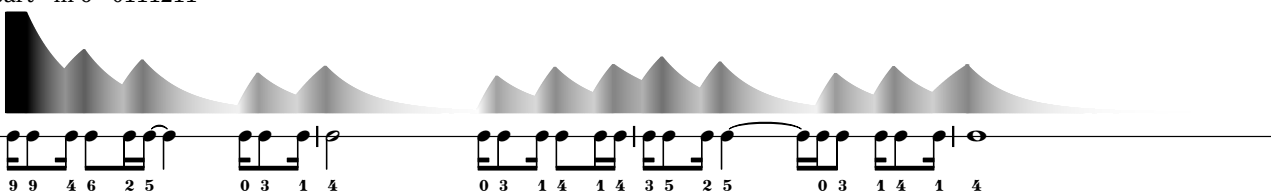
0111211

michael winter

(cdmx, mx and nashville, usa; 2018)

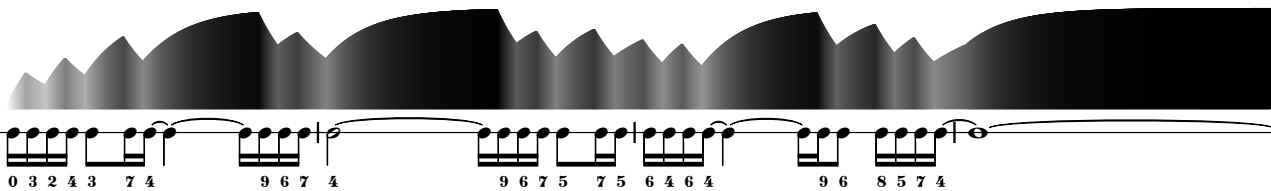
part - hl 6 - 0111211

hl 6
(harm 1)



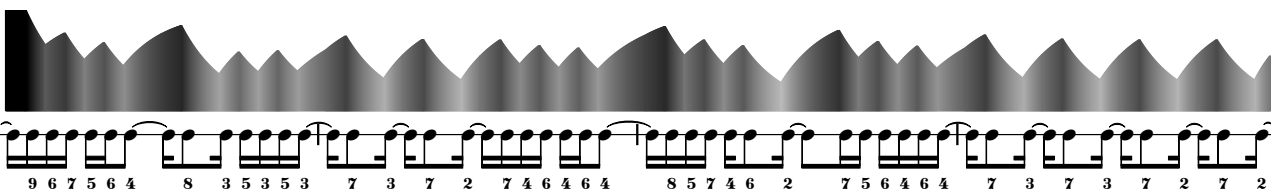
⑤

hl6 (h1)



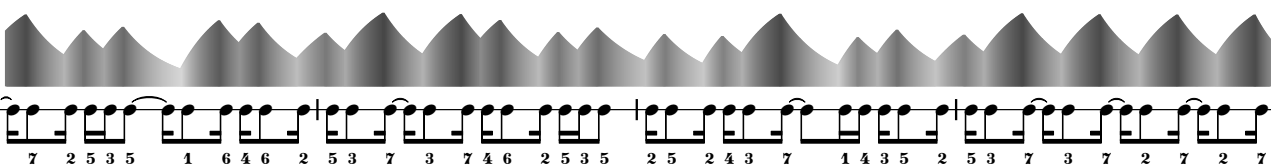
⑨

hl6 (h1)



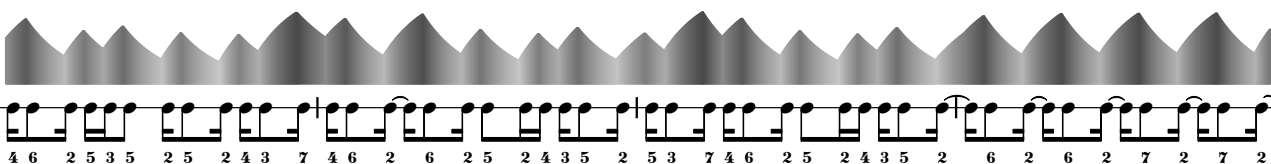
⑬

hl6 (h1)



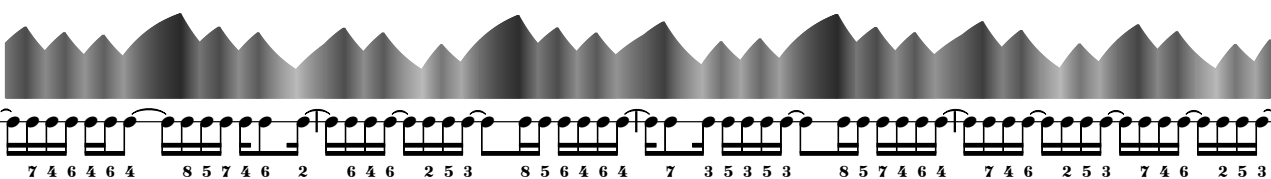
⑰

hl6 (h1)



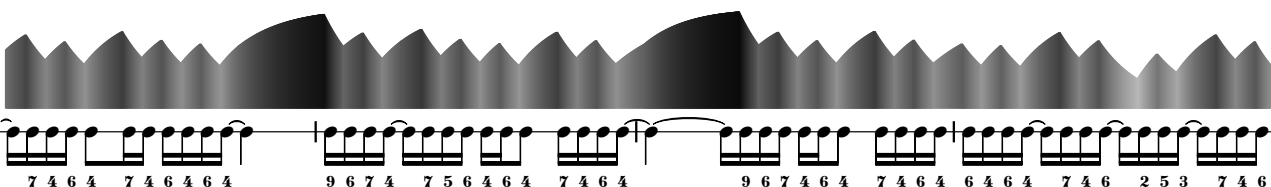
⑲

hl6 (h1)



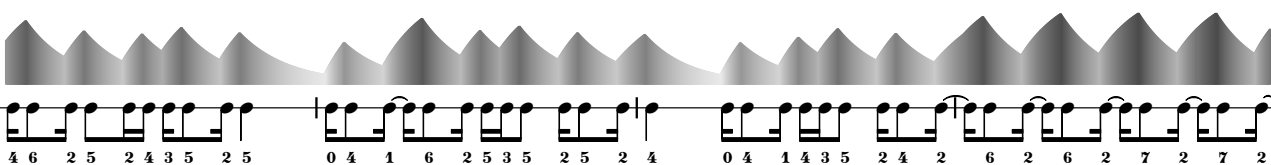
⑳

hl6 (h1)



㉑

hl6 (h1)



③③

hl6 (h1)

7 2 5 2 4 3 5 2 5 0 4 1 4 1 4 3 5 2 6 2 6 2 5 2 4 3 5 2 5 0 4 1 4 1 4 2 4 2

③⑦

hl6 (h1)

6 4 6 4 7 4 6 4 6 4 9 6 7 4 7 5 6 4 6 4 7 4 6 4 6 4 6 4 7 4 6 4 6 4 9 6 7 4 7 5 7 5 7 5 6 4

④①

hl6 (h1)

7 4 6 4 6 4 8 5 7 4 6 2 6 4 6 4 6 4 8 5 7 2 5 3 7 4 6 4 6 4 8 5 7 4 6 2 6 4 6 4 6 4 8 5 7 4 6 4

④⑤

hl6 (h1)

7 3 5 3 5 2 5 2 4 3 7 4 6 2 5 3 5 2 5 2 6 2 6 2 5 3 5 2 5 2 4 3 7 4 6 2 5 3 5 1 4 2 5 2

④⑨

hl6 (h1)

6 2 5 3 5 1 6 4 6 2 5 2 4 3 5 2 5 3 7 3 7 4 6 2 5 3 5 1 6 4 6 2 5 2 4 3 5 2 6 2 5 3 5

⑤③

hl6 (h1)

2 5 3 5 3 5 3 8 3 5 3 5 3 8 5 6 4 6 4 7 3 7 3 7 4 6 4 6 4 8 3 5 3 5 3 8 5 7 4 6 4 7 4 6 4 6 4

⑤⑦

hl6 (h1)

8 5 7 4 7 5 9 6 7 4 6 4 7 4 6 4 9 6 7 5 7 5 9 6 7 4 6 4 7 4 6 4 7 4 6 4 7 4

⑥①

hl6 (h1)

7 3 5 2 5 0 3 1 4 2 5 2 4 2 4 0 3 1 4 1 4 0 3 1 4 2 5 2 4 2 6 2 5 2 4

65

hl6 (h1)

3 5 2 5 2 4 0 3 1 4 2 5 2 4 2 4 0 3 1 4 1 6 2 5 2 4 3 5 2 5 0 3 1 4 1

69

hl6 (h1)

6 4 6 4 7 4 9 6 7 4 6 4 7 4 6 4 9 6 8 5 7 4 7 5 6 4 7 4 6 4 6 4 9 6 8 5 7 4

73

hl6 (h1)

7 5 6 4 6 4 8 3 5 3 5 3 8 5 7 4 6 4 7 3 7 2 8 5 6 4 6 4 7 4 6 4 6 4 8 5 7 4 6 2 7 5 6 4 6 4

77

hl6 (h1)

7 3 5 3 5 1 6 4 6 2 5 2 4 3 5 2 5 3 7 3 7 1 4 3 5 2 6 2 5 3 5 2 5 2 4 3 7 1 4 3 5 2

81

hl6 (h1)

6 2 5 3 5 2 5 2 4 3 7 4 6 2 5 3 5 2 5 2 6 2 6 2 5 3 5 1 4 2 5 2 4 3 7 4 6 2 5 2 4 3 5 2

85

hl6 (h1)

6 4 6 4 6 4 8 5 7 4 6 2 6 4 6 4 6 4 8 5 7 2 5 3 7 4 6 4 6 4 8 5 7 4 6 4 7 3 5 3 5 3 8 5 7 4 6 4

89

hl6 (h1)

7 4 6 4 7 4 6 4 6 4 9 6 7 4 7 5 6 4 6 4 7 4 6 4 6 4 6 4 7 4 7 5 7 5 6 4 9 6 7 4 6 4 7 4 6 4

93

hl6 (h1)

7 3 5 2 5 3 5 2 5 0 4 1 4 1 4 3 5 2 6 2 6 2 5 2 4 2 4 2 4 2 4 0 4 1 4 2 5 2 4 2

97

hl6 (h1)

6 2 5 2 4 3 5 2 5 0 4 1 6 2 5 3 5 2 5 2 6 2 6 2 7 2 7 2 7 2 5 2 4 3 5 2 5

101

hl6 (h1)

0 4 2 5 3 7 4 6 4 6 4 9 6 7 4 7 5 6 4 6 4 7 4 6 4 6 4 6 4 7 4 6 2 5 3 7 4 6 4 6 4 6 4 7 4 6 4 6 4

105

hl6 (h1)

9 6 7 4 6 4 8 5 7 4 6 2 6 4 6 2 5 3 8 5 6 4 6 4 7 4 6 2 5 3 7 4 6 2 5 3 7 4 6 2 5 3 7 4 6 4 6 4 8 5 7 4 6 2

109

hl6 (h1)

6 2 5 3 5 2 5 2 4 3 7 4 6 2 6 2 5 2 4 3 5 2 6 2 6 2 7 2 7 2 7 2 5 3 5 2 5 2 4 3 7

113

hl6 (h1)

4 6 2 5 3 5 1 6 4 6 2 5 3 7 3 7 4 6 2 5 3 5 1 6 2 6 2 7 2 7 4 6 2 5 3 5 1 6 4 6 2

117

hl6 (h1)

6 4 6 4 6 4 8 3 5 3 5 3 7 3 7 3 7 4 6 4 6 4 8 3 7 3 7 2 7 2 7 4 6 4 6 4 8 3 5 3 5 3

121

hl6 (h1)

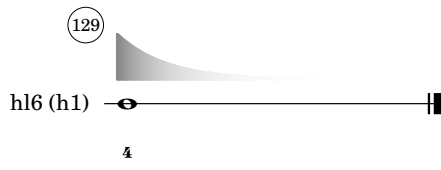
7 4 6 4 7 4 9 6 7 4 9 6 7 4 7 5 9 6 7 5 7 5 9 6 7

125

hl6 (h1)

4 6 2 5 2 4 0 3 1 4 0 3 1 4 1 4 0 3 1 4 1 4 0 3 1

(part - hl 6 - 0111211)



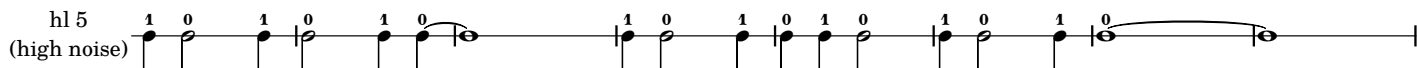
a lot of tiles (trivial scan)

0111211

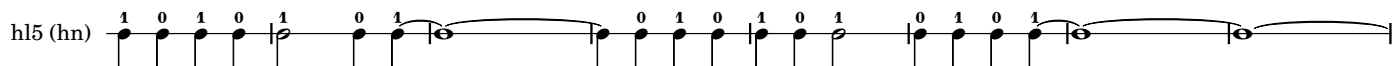
michael winter

(cdmx, mx and nashville, usa; 2018)

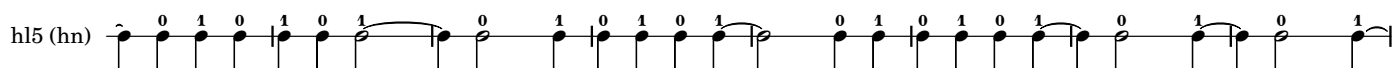
part - hl 5 - 0111211



⑨



⑰



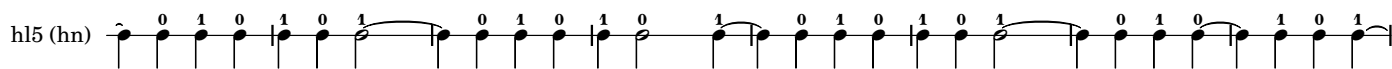
②⑤



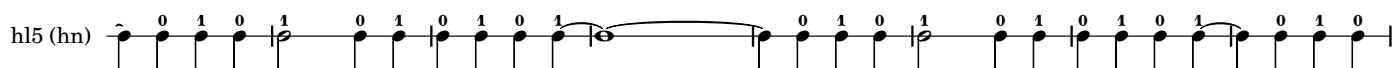
③③



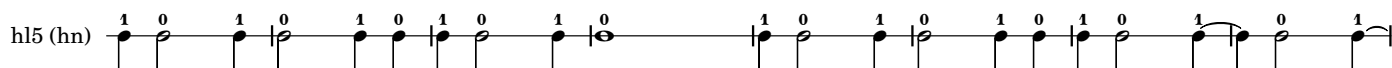
④①



④⑨



⑤⑦



(part - hl 5 - 0111211)

65

73

[illegible]

(81)

(89)

97

105


(113)

(121)

hl5 (hn)

(part - hl 5 - 0111211)

129

hl5 (hn) 

a lot of tiles (trivial scan)

0111211

michael winter

(cdmx, mx and nashville, usa; 2018)

part - hl 4 - 0111211

hl 4
(low noise)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a circular head. The notes are labeled with '1' or '0' above them. The sequence of notes is 1, 0, 1, 0, 1, 0, 1, 0. There are curved lines (arcs) connecting the second and third notes, and the fourth and fifth notes. The staff ends with a vertical bar line.

⑨

hl4 (ln)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a circular head. The notes are labeled with '1' or '0' above them. The sequence of notes is 1, 0, 1, 0, 1, 0, 1, 0. There are curved lines (arcs) connecting the second and third notes, the fourth and fifth notes, and the sixth and seventh notes. The staff ends with a vertical bar line.

⑪

hl4 (ln)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a circular head. The notes are labeled with '1' or '0' above them. The sequence of notes is 1, 0, 1, 0, 1, 0, 1, 0. There are curved lines (arcs) connecting the fifth and sixth notes, and the seventh and eighth notes. The staff ends with a vertical bar line.

⑮

hl4 (ln)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a circular head. The notes are labeled with '1' or '0' above them. The sequence of notes is 0, 1, 0, 1, 0, 1, 0, 1. There are curved lines (arcs) connecting the fourth and fifth notes, the fifth and sixth notes, the sixth and seventh notes, and the seventh and eighth notes. The staff ends with a vertical bar line.

⑰

hl4 (ln)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a circular head. The notes are labeled with '1' or '0' above them. The sequence of notes is 0, 1, 1, 0, 1, 0, 1, 0. There are curved lines (arcs) connecting the seventh and eighth notes, and the eighth note is connected to the first note of the next measure (indicated by a line extending beyond the bar line). The staff ends with a vertical bar line.

⑲

hl4 (ln)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a circular head. The notes are labeled with '1' or '0' above them. The sequence of notes is 0, 1, 1, 0, 1, 0, 1, 0. There are curved lines (arcs) connecting the sixth and seventh notes, and the seventh and eighth notes. The staff ends with a vertical bar line.

⑳

hl4 (ln)

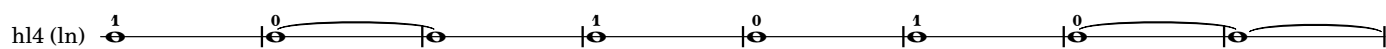
A musical staff with a single line. It contains eight notes, each with a vertical stem and a circular head. The notes are labeled with '1' or '0' above them. The sequence of notes is 0, 1, 1, 0, 1, 0, 1, 0. There are curved lines (arcs) connecting the second and third notes, and the sixth and seventh notes. The staff ends with a vertical bar line.

㉑

hl4 (ln)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a circular head. The notes are labeled with '1' or '0' above them. The sequence of notes is 1, 0, 1, 0, 1, 0, 1, 0. There are curved lines (arcs) connecting the second and third notes, the fifth and sixth notes, and the seventh and eighth notes. The staff ends with a vertical bar line.

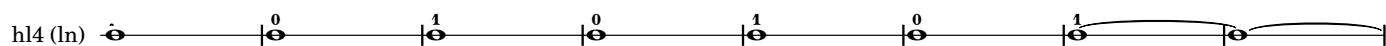
(65)



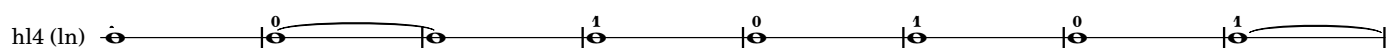
(73)



(81)



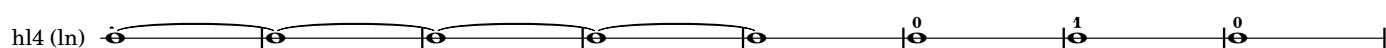
(89)



(97)



(105)



(113)



(121)



(part - hl 4 - 0111211)

129

hl4 (ln) $\frac{0}{\mathbf{e}} \mathbf{H}$

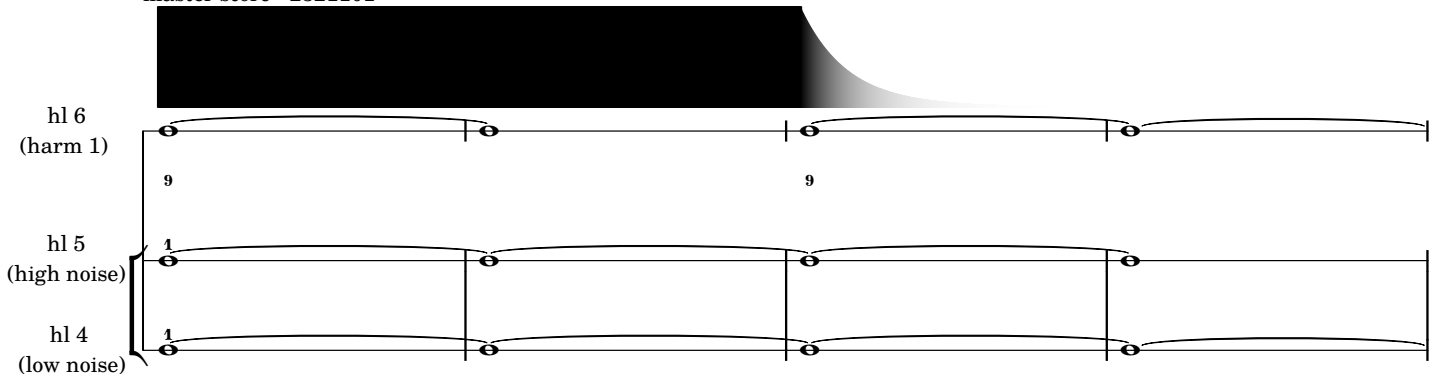
a lot of tiles (trivial scan)

2321101

michael winter

(cdmx, mx and nashville, usa; 2018)

master score - 2321101

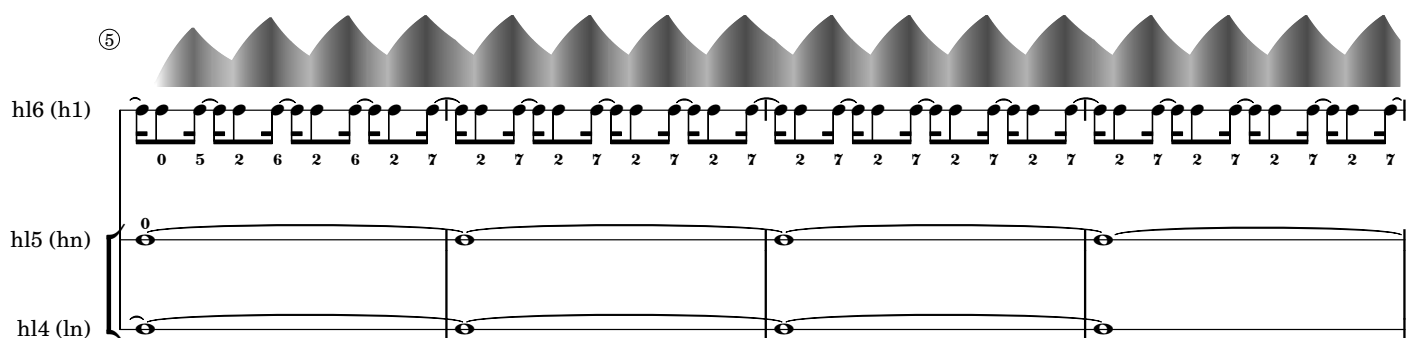


hl 6 (harm 1)

hl 5 (high noise)

hl 4 (low noise)

⑤

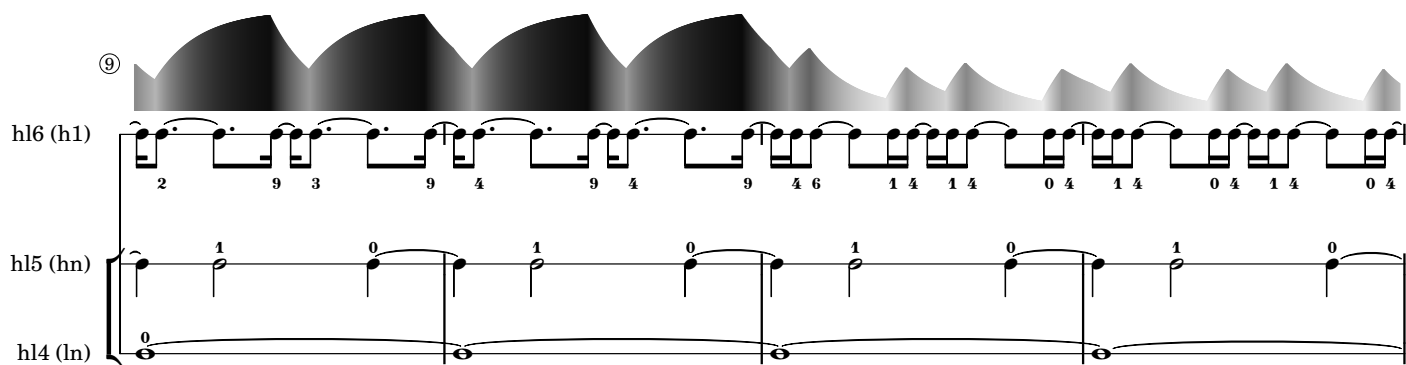


hl6 (h1)

hl5 (hn)

hl4 (ln)

⑨

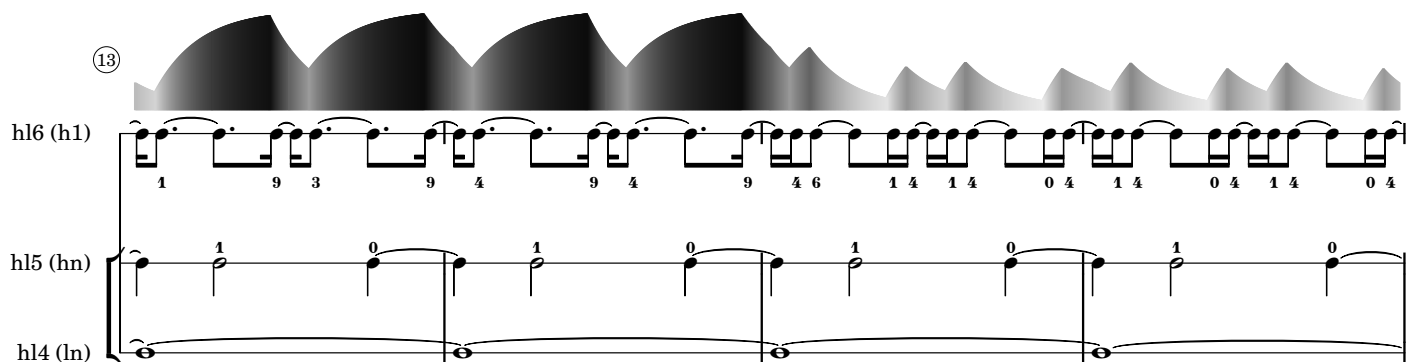


hl6 (h1)

hl5 (hn)

hl4 (ln)

⑬



hl6 (h1)

hl5 (hn)

hl4 (ln)

⑪

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑫

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑮

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑲

hl6 (h1)

hl5 (hn)

hl4 (ln)

③③

hl6 (h1)

hl5 (hn)

hl4 (ln)

③⑦

hl6 (h1)

hl5 (hn)

hl4 (ln)

④①

hl6 (h1)

hl5 (hn)

hl4 (ln)

④⑤

hl6 (h1)

hl5 (hn)

hl4 (ln)

④⑨

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑤③

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑤⑦

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑥①

hl6 (h1)

hl5 (hn)

hl4 (ln)

65

hl6 (h1)

hl5 (hn)

hl4 (ln)

69

hl6 (h1)

hl5 (hn)

hl4 (ln)

73

hl6 (h1)

hl5 (hn)

hl4 (ln)

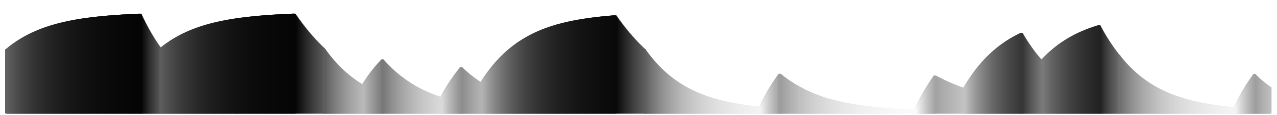
77

hl6 (h1)

hl5 (hn)

hl4 (ln)

81



hl6 (h1)

hl5 (hn)

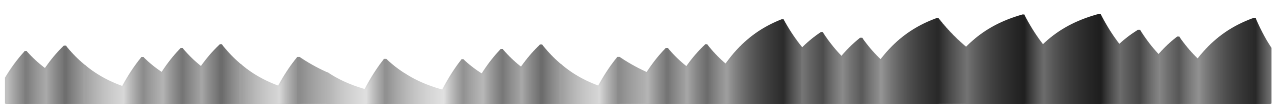
hl4 (ln)

6 9 6 9 2 5 1 4 2 9 0 3 0 3 2 7 5 8 0 3

1 0 1 0 1 0 1 0 1 0

1 0 1 0

85



hl6 (h1)

hl5 (hn)

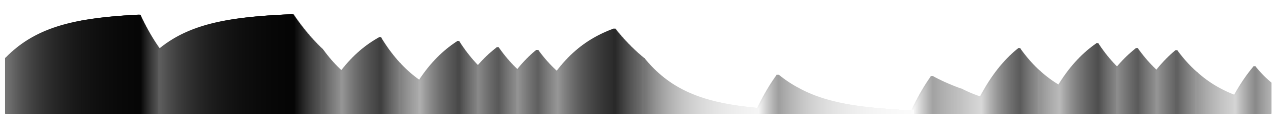
hl4 (ln)

2 5 3 5 1 4 2 5 3 5 1 4 1 4 1 4 2 5 3 5 1 4 2 5 3 5 8 5 6 4 6 4 8 5 8 5 8 5 7 4 6 4 8

1 0 1 0 1 0 1 0 1 0

1 0

89



hl6 (h1)

hl5 (hn)

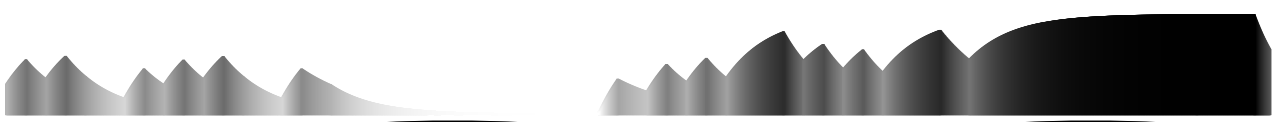
hl4 (ln)

5 9 6 9 4 7 3 7 4 6 4 6 4 8 0 3 0 3 1 6 2 6 4 6 4 6 1 4

1 0 1 0 1 0 1 0

1 0 1 0

93



hl6 (h1)

hl5 (hn)

hl4 (ln)

2 5 3 5 1 4 2 5 3 5 1 4 0 3 2 4 3 5 3 8 5 6 4 6 4 8 5 9

1 0 1 0 1 0 1 0

1 0

97

hl6 (h1)

hl5 (hn)

hl4 (ln)

101

hl6 (h1)

hl5 (hn)

hl4 (ln)

105

hl6 (h1)

hl5 (hn)

hl4 (ln)

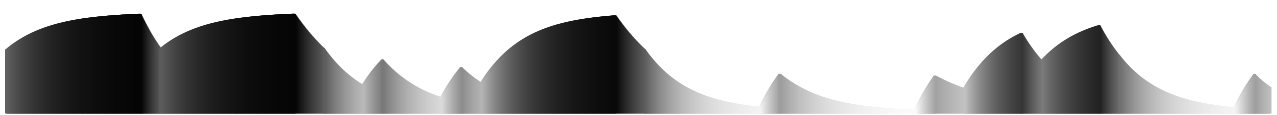
109

hl6 (h1)

hl5 (hn)

hl4 (ln)

113



hl6 (h1)

6 9 6 9 2 5 1 4 2 9 0 3 0 3 2 7 5 8 0 3

hl5 (hn)

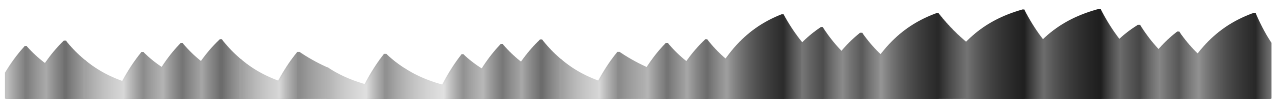
1 0 1 0 1 0 1 0 1 0

hl4 (ln)

1 0 1 0

Detailed description: This system covers measures 113 to 116. The spectrogram at the top shows a series of peaks. The hl6 (h1) staff features a complex melodic line with many slurs and ties, and a sequence of numbers below it. The hl5 (hn) staff has a simpler line with notes and fingerings. The hl4 (ln) staff has a single note with a long sustain.

117



hl6 (h1)

2 5 3 5 1 4 2 5 3 5 1 4 1 4 1 4 2 5 3 5 1 4 2 5 3 5 8 5 6 4 6 4 8 5 8 5 8 5 7 4 6 4 8

hl5 (hn)

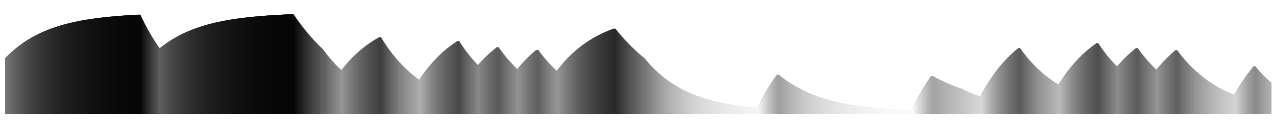
1 0 1 0 1 0 1 0 1 0

hl4 (ln)

1 0

Detailed description: This system covers measures 117 to 120. The spectrogram shows a more active texture. The hl6 (h1) staff has a very busy melodic line with many slurs and ties, and a long sequence of numbers. The hl5 (hn) staff has a line with notes and fingerings. The hl4 (ln) staff has a single note with a long sustain.

121



hl6 (h1)

5 9 6 9 4 7 3 7 4 6 4 6 4 8 0 3 0 3 1 6 2 6 4 6 4 6 1 4

hl5 (hn)

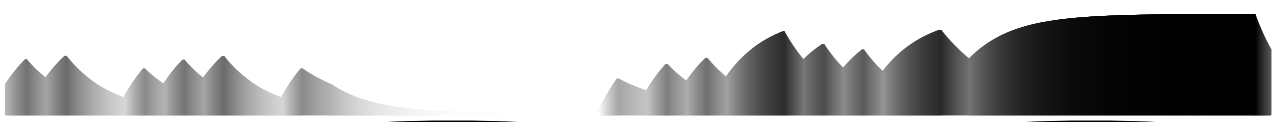
1 0 1 0 1 0 1 0

hl4 (ln)

1 0 1 0

Detailed description: This system covers measures 121 to 124. The spectrogram shows a series of peaks. The hl6 (h1) staff features a complex melodic line with many slurs and ties, and a sequence of numbers. The hl5 (hn) staff has a simpler line with notes and fingerings. The hl4 (ln) staff has a single note with a long sustain.

125



hl6 (h1)

2 5 3 5 1 4 2 5 3 5 1 4 0 3 2 4 3 5 3 8 5 6 4 6 4 8 5 9

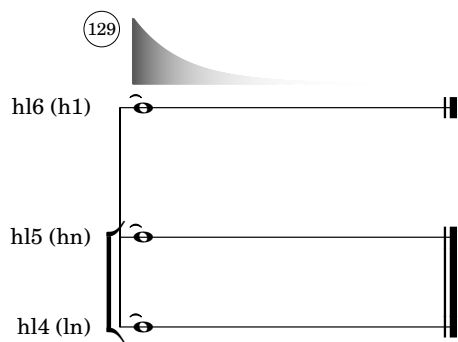
hl5 (hn)

1 0 1 0 1 0 1 0

hl4 (ln)

1 0 1 0

Detailed description: This system covers measures 125 to 128. The spectrogram shows a series of peaks. The hl6 (h1) staff features a complex melodic line with many slurs and ties, and a sequence of numbers. The hl5 (hn) staff has a simpler line with notes and fingerings. The hl4 (ln) staff has a single note with a long sustain.



a lot of tiles (trivial scan)

2321101

michael winter

(cdmx, mx and nashville, usa; 2018)

part - hl 6 - 2321101

hl 6
(harm 1)

9

9

⑤

hl6 (h1)

0 5 2 6 2 6 2 7 2 7 2 7 2 7 2 7 2 7 2 7 2 7 2 7 2 7 2 7

⑨

hl6 (h1)

2 9 3 9 4 9 4 9 4 6 1 4 1 4 0 4 1 4 0 4 1 4 0 4

⑬

hl6 (h1)

1 9 3 9 4 9 4 9 4 6 1 4 1 4 0 4 1 4 0 4 1 4 0 4

⑰

hl6 (h1)

1 9 4 9 4 6 0 3 1 4 0 3

⑳

hl6 (h1)

1 4 1 4 2 6 4 6 1 4 1 4 1 4 2 6 4 6 1 4 1 7 5 6 4 7 4 6 4 8 3 8 5 6 4 7 5 6 4 8

㉔

hl6 (h1)

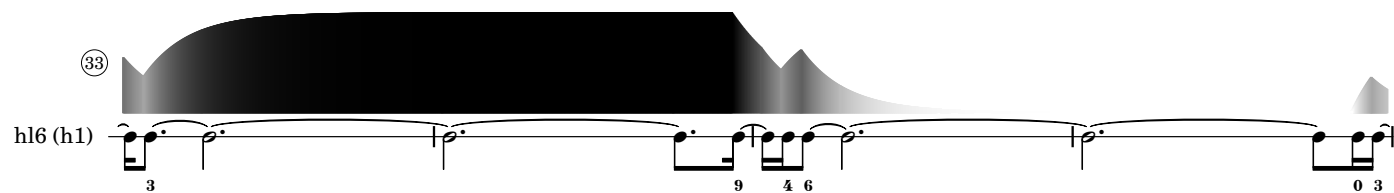
3 9 4 9 4 6 0 3 1 4 0 3

㉘

hl6 (h1)

1 4 1 4 2 6 4 6 1 4 1 4 1 4 2 6 4 6 1 4 1 7 5 6 4 7 4 6 4 8 3 8 5 6 4 7 5 6 4 8

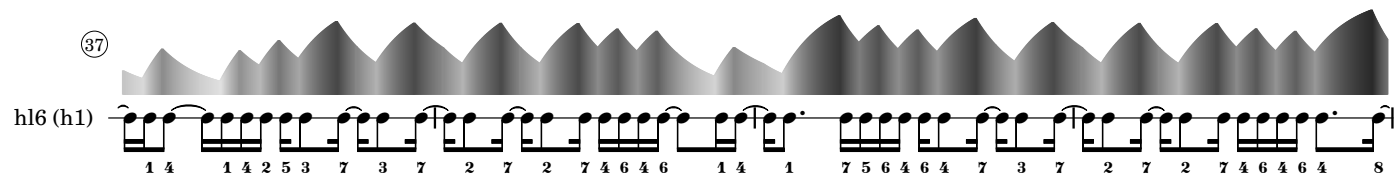
33



hl6 (h1)

3

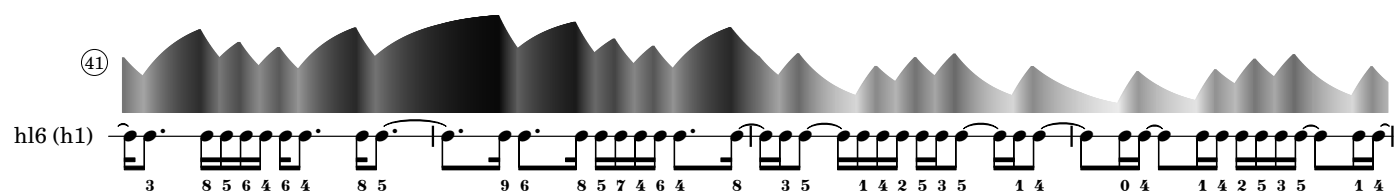
37



hl6 (h1)

1 4 1 4 2 5 3 7 3 7 2 7 2 7 4 6 4 6 1 4 1 7 5 6 4 6 4 7 3 7 2 7 2 7 4 6 4 6 4 8

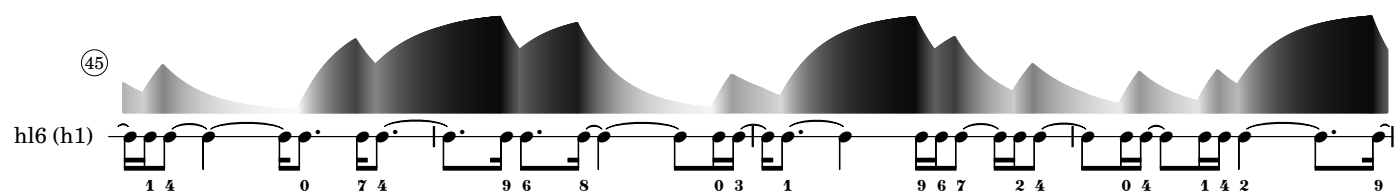
41



hl6 (h1)

3 8 5 6 4 6 4 8 5 9 6 8 5 7 4 6 4 8 3 5 1 4 2 5 3 5 1 4 0 4 1 4 2 5 3 5 1 4

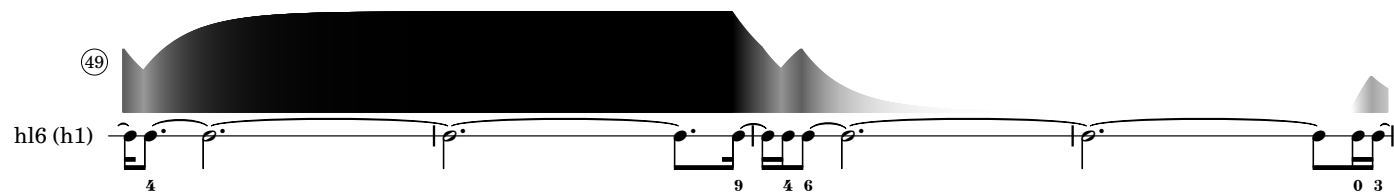
45



hl6 (h1)

1 4 0 7 4 9 6 8 0 3 1 9 6 7 2 4 0 4 1 4 2 9

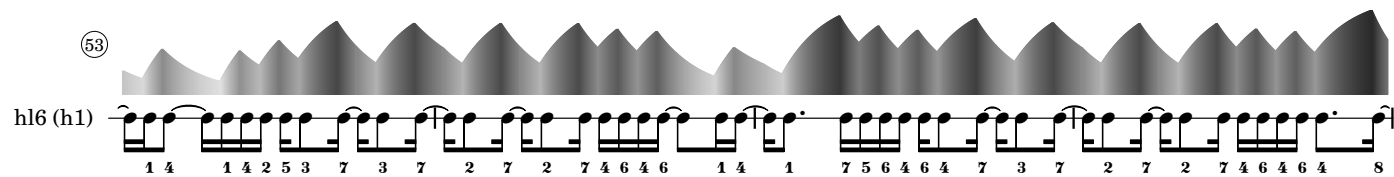
49



hl6 (h1)

4

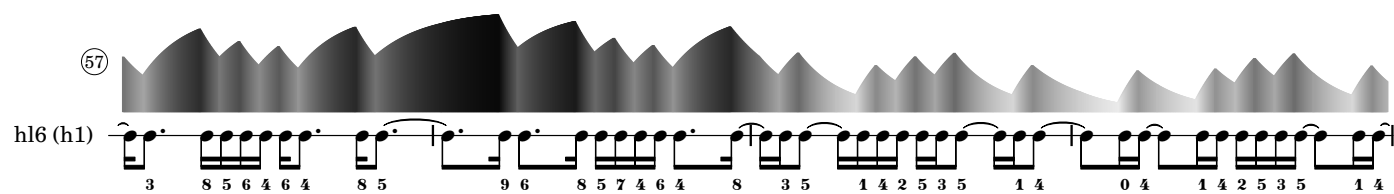
53



hl6 (h1)

1 4 1 4 2 5 3 7 3 7 2 7 2 7 4 6 4 6 1 4 1 7 5 6 4 6 4 7 3 7 2 7 2 7 4 6 4 6 4 8

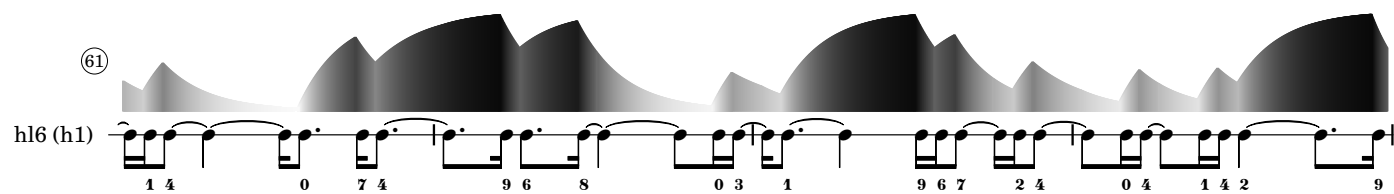
57



hl6 (h1)

3 8 5 6 4 6 4 8 5 9 6 8 5 7 4 6 4 8 3 5 1 4 2 5 3 5 1 4 0 4 1 4 2 5 3 5 1 4


61



hl6 (h1)

1 4 0 7 4 9 6 8 0 3 1 9 6 7 2 4 0 4 1 4 2 9

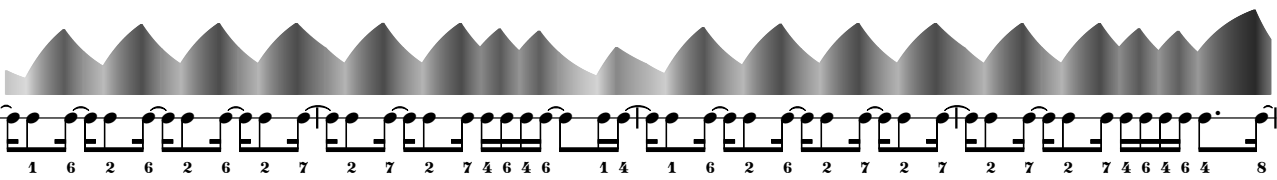
65



hl6 (h1)

6 9 0 3

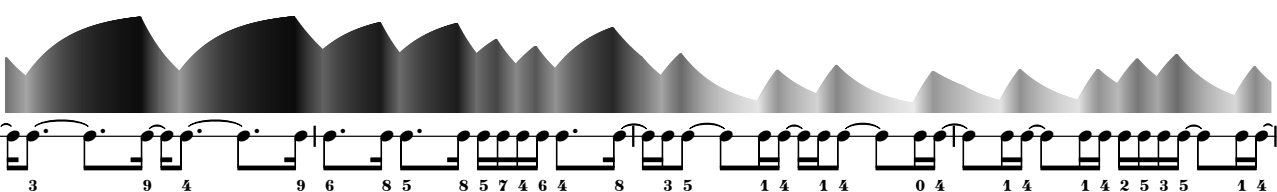
69



hl6 (h1)

1 6 2 6 2 6 2 7 2 7 2 7 4 6 4 6 1 4 1 6 2 6 2 7 2 7 2 7 2 7 4 6 4 6 4 8

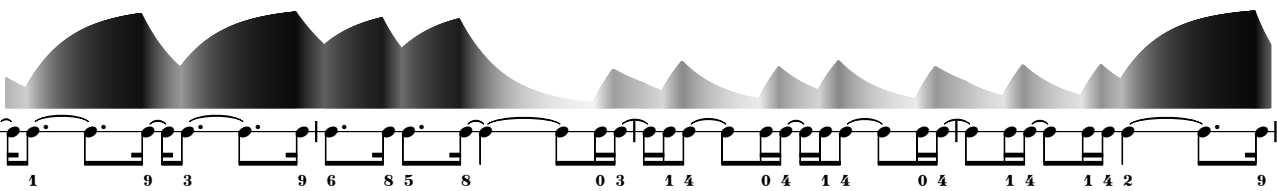
73



hl6 (h1)

3 9 4 9 6 8 5 8 5 7 4 6 4 8 3 5 1 4 1 4 0 4 1 4 1 4 2 5 3 5 1 4

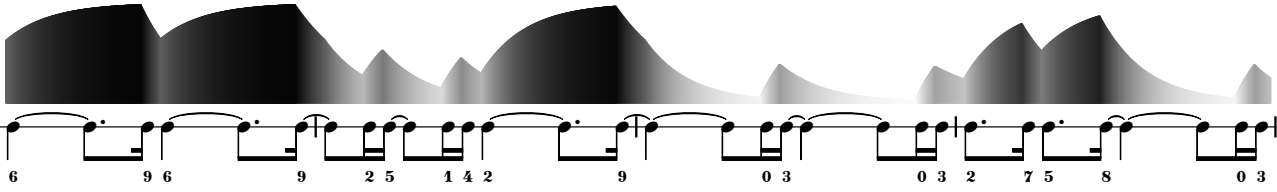
77



hl6 (h1)

1 9 3 9 6 8 5 8 0 3 1 4 0 4 1 4 0 4 1 4 1 4 2 9

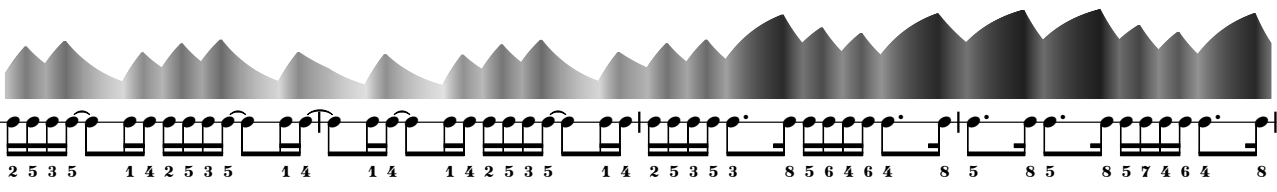
81



hl6 (h1)

6 9 6 9 2 5 1 4 2 9 0 3 0 3 2 7 5 8 0 3

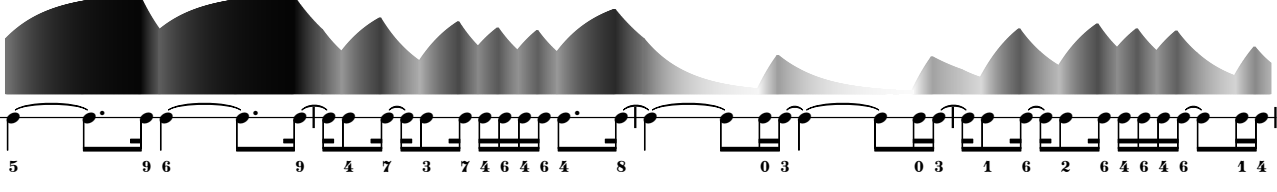
85



hl6 (h1)

2 5 3 5 1 4 2 5 3 5 1 4 1 4 1 4 2 5 3 5 1 4 2 5 3 5 3 8 5 6 4 6 4 8 5 8 5 8 5 7 4 6 4 8

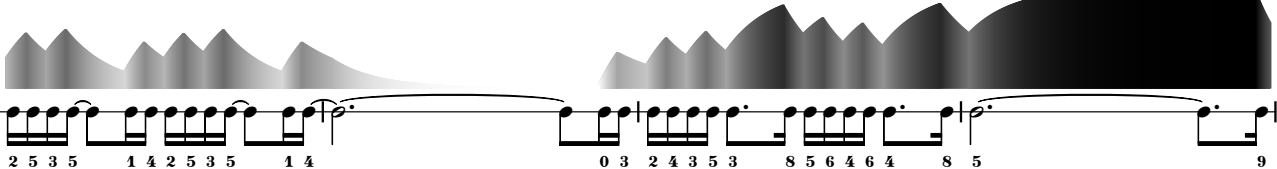
89



hl6 (h1)

5 9 6 9 4 7 3 7 4 6 4 6 4 8 0 3 0 3 1 6 2 6 4 6 4 6 1 4

93



hl6 (h1)

2 5 3 5 1 4 2 5 3 5 1 4 0 3 2 4 3 5 3 8 5 6 4 6 4 8 5 9

97

hl6 (h1)

6 9 0 3

101

hl6 (h1)

1 6 2 6 2 6 2 7 2 7 2 7 4 6 4 6 1 4 1 6 2 6 2 7 2 7 2 7 2 7 4 6 4 6 4 8

105

hl6 (h1)

3 9 4 9 6 8 5 8 5 7 4 6 4 8 3 5 1 4 1 4 0 4 1 4 1 4 2 5 3 5 1 4

109

hl6 (h1)

1 9 3 9 6 8 5 8 0 3 1 4 0 4 1 4 0 4 1 4 1 4 2 9

113

hl6 (h1)

6 9 6 9 2 5 1 4 2 9 0 3 0 3 2 7 5 8 0 3

117

hl6 (h1)

2 5 3 5 1 4 2 5 3 5 1 4 1 4 1 4 2 5 3 5 1 4 2 5 3 5 3 8 5 6 4 6 4 8 5 8 5 8 5 7 4 6 4 8

121

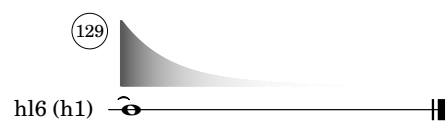
hl6 (h1)

5 9 6 9 4 7 3 7 4 6 4 6 4 8 0 3 0 3 1 6 2 6 4 6 4 6 1 4

125

hl6 (h1)

2 5 3 5 1 4 2 5 3 5 1 4 0 3 2 4 3 5 3 8 5 6 4 6 4 8 5 9



a lot of tiles (trivial scan)

2321101

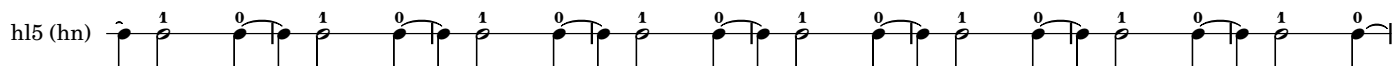
michael winter

(cdmx, mx and nashville, usa; 2018)

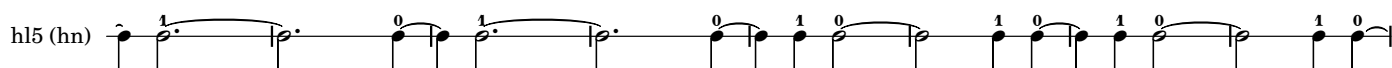
part - hl 5 - 2321101



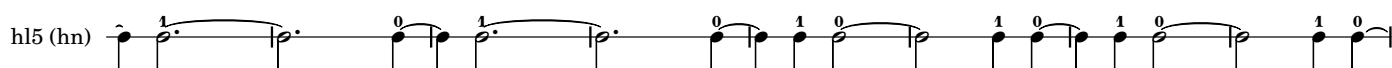
⑨



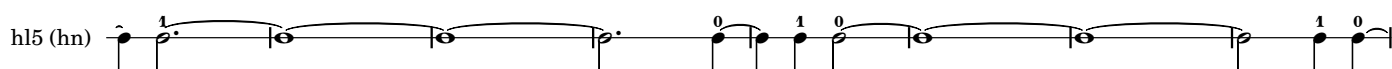
⑰



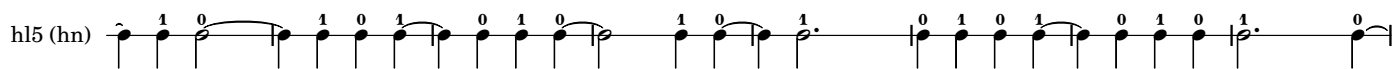
⑳



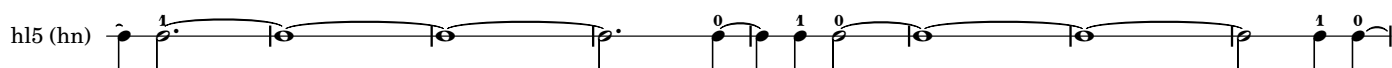
㉓



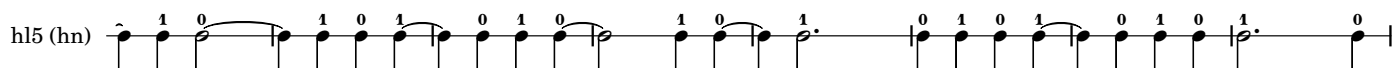
㉑



㉗



㉙



65

73

(81)

(89)

hl5 (hm)

97


105

(113)

(121)

hl5 (hn)

129

hl5 (hn) 

a lot of tiles (trivial scan)


2321101

michael winter

(cdmx, mx and nashville, usa; 2018)

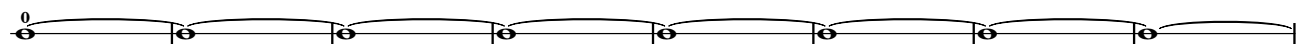
part - hl 4 - 2321101

hl 4
(low noise)



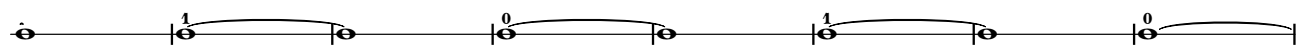
⑨

hl4 (ln)




⑰

hl4 (ln)



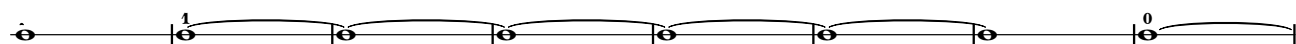
②⑤

hl4 (ln)




③③

hl4 (ln)




④①

hl4 (ln)




④⑨

hl4 (ln)

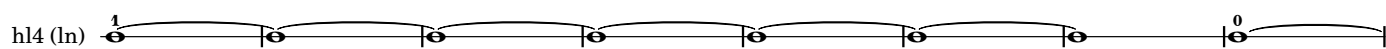


⑤⑦

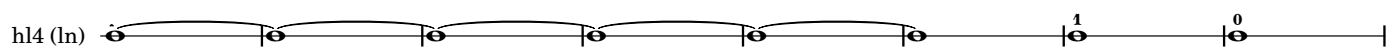
hl4 (ln)



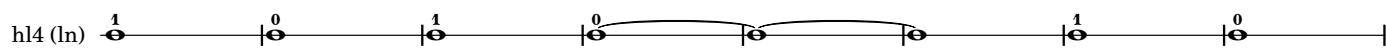
65



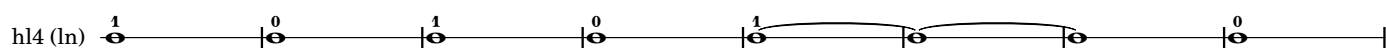
73



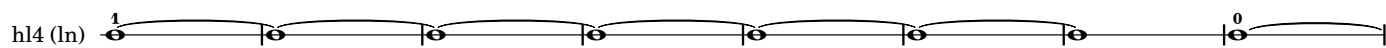
81



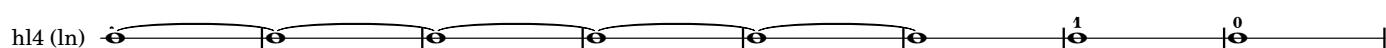
89



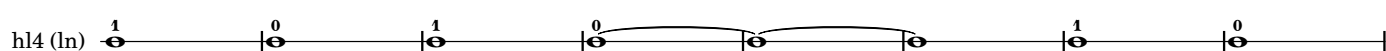
97



105




113



121



129

hl4 (ln) 

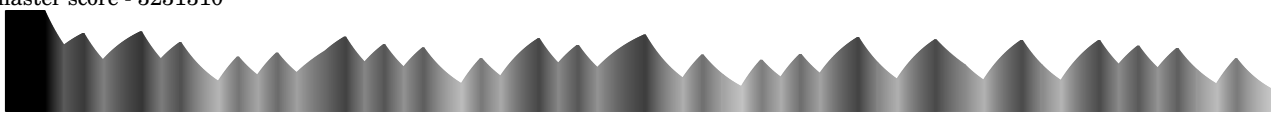
a lot of tiles (trivial scan)

3231310

michael winter

(cdmx, mx and nashville, usa; 2018)

master score - 3231310




hl 6
(harm 1)

hl 5
(high noise)

hl 4
(low noise)

9 9 6 7 5 7 5 6 2 5 3 5 3 7 4 6 4 6 2 5 3 7 4 6 4 7 3 5 2 5 3 5 3 7 3 7 3 7 2 7 4 6 4 6 2 5

⑤



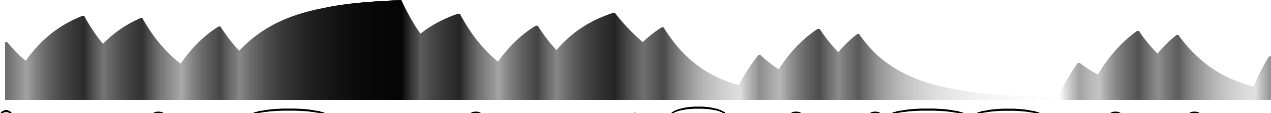
hl6 (h1)

hl5 (hn)

hl4 (ln)

2 4 0 4 2 5 0 4 2 5 2 4 3 5 1 4 2 5 0 4 2 7 5 7 5 9 5 8 3 7 4 9 5 8 5 8

⑨



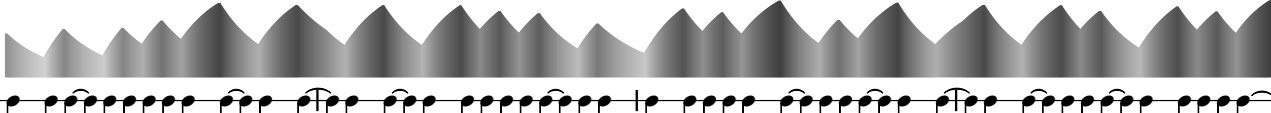
hl6 (h1)

hl5 (hn)

hl4 (ln)

3 8 5 8 3 7 4 9 6 8 3 7 4 8 5 7 1 4 2 6 4 6 0 3 2 6 4 6 1

⑬



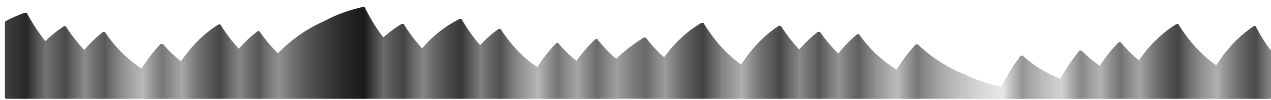
hl6 (h1)

hl5 (hn)

hl4 (ln)

4 1 4 1 4 3 5 3 7 3 7 2 7 2 7 4 6 4 6 2 5 2 6 4 6 4 7 3 5 3 7 3 7 3 7 4 6 2 6 4 6 4

⑪




hl6 (h1)

hl5 (hn)

hl4 (ln)

8 5 7 4 6 2 5 3 7 4 6 4 8 5 7 4 7 5 6 2 5 3 5 3 5 3 7 3 7 4 6 4 6 2 5 0 4 1 4 3 5 3 7 3 7

⑮



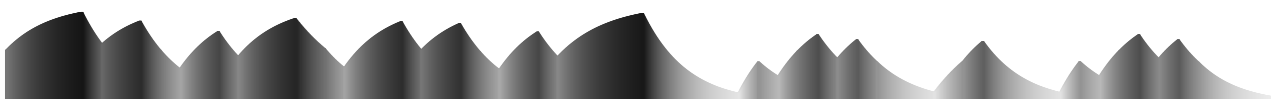
hl6 (h1)

hl5 (hn)

hl4 (ln)

2 5 3 5 1 4 2 5 0 6 1 4 2 5 0 4 2 5 2 6 4 8 5 8 5 8 3 8 5 8 5 9 5 8

⑲



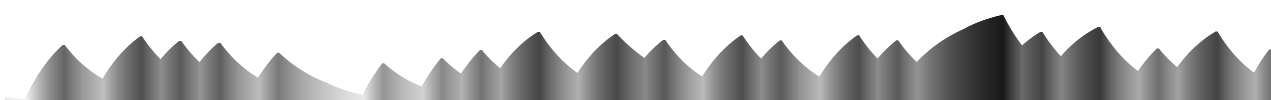
hl6 (h1)

hl5 (hn)

hl4 (ln)

5 9 5 8 3 7 4 8 3 8 5 8 3 7 4 9 1 4 2 6 4 6 1 6 1 4 2 6 4 6

⑳



hl6 (h1)

hl5 (hn)

hl4 (ln)

0 5 2 6 4 6 4 6 2 5 0 4 1 4 3 5 3 7 3 7 4 6 2 6 4 6 2 6 4 6 4 8 5 7 4 7 3 5 3 7 3

③③

hl6 (h1)

hl5 (hn)

hl4 (ln)

③⑦

hl6 (h1)

hl5 (hn)

hl4 (ln)

④①

hl6 (h1)

hl5 (hn)

hl4 (ln)

④⑤

hl6 (h1)

hl5 (hn)

hl4 (ln)

④9

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑤3

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑤7

hl6 (h1)

hl5 (hn)

hl4 (ln)

⑥1

hl6 (h1)

hl5 (hn)

hl4 (ln)

65

hl6 (h1)

hl5 (hn)

hl4 (ln)

69

hl6 (h1)

hl5 (hn)

hl4 (ln)

73

hl6 (h1)

hl5 (hn)

hl4 (ln)

77


hl6 (h1)

hl5 (hn)

hl4 (ln)

-6-

97



hl6 (h1)

hl5 (hn)


hl4 (ln)

3 5 3 7 4 6 2 8 5 7 5 7 5 6 4 7 3 5 3 7 3 5 3 7 3 7 4 6 1 4 2 5 2 4 2 4 3 5 3 7 3 7

0 1 0 1 0 1 0 1 0 1

1 0 1

101



hl6 (h1)

hl5 (hn)

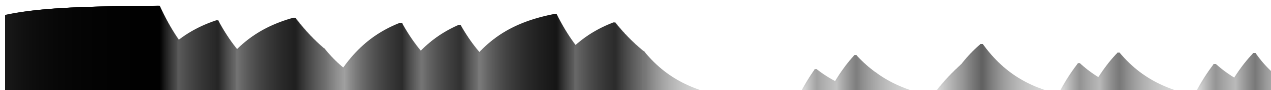
hl4 (ln)

4 6 4 6 2 5 3 5 2 5 2 4 3 7 1 4 2 6 4 6 0 5 2 6 4 7 4 6 4 7 3 8 5 7 3 7 4

0 1 0 1 0 1 0 1 0 1 0 1

0 1 0

105



hl6 (h1)

hl5 (hn)


hl4 (ln)

9 6 8 5 8 3 8 5 8 5 9 5 8 0 3 2 5 0 6 1 4 2 5 0 4 2 5

1 0 1 0 1 0 1 0 1

1 0 0

109



hl6 (h1)

hl5 (hn)

hl4 (ln)

2 6 2 6 4 6 4 6 2 5 0 4 1 4 3 5 3 7 3 7 4 6 2 6 2 5 3 7 4 6 4 8 5 7 4 7 5 6 2 5 3 5 3

0 1 0 1 0 1 0 1

1 0 1

113

hl6 (h1)

hl5 (hn)

hl4 (ln)

117

hl6 (h1)

hl5 (hn)

hl4 (ln)

121

hl6 (h1)

hl5 (hn)

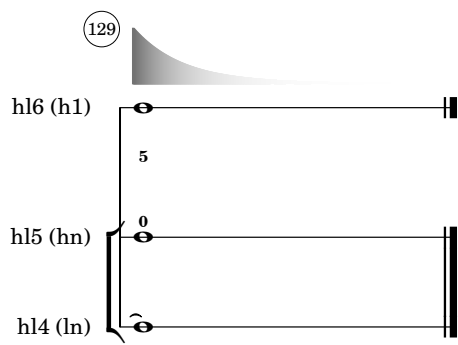
hl4 (ln)

125

hl6 (h1)

hl5 (hn)

hl4 (ln)



a lot of tiles (trivial scan)

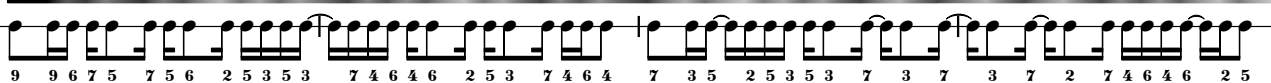
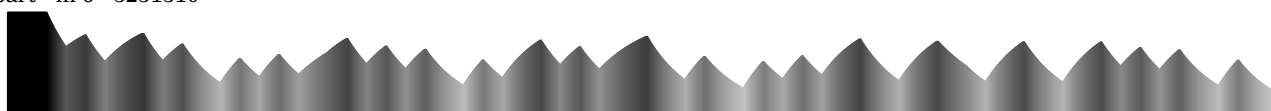
3231310

michael winter

(cdmx, mx and nashville, usa; 2018)

part - hl 6 - 3231310

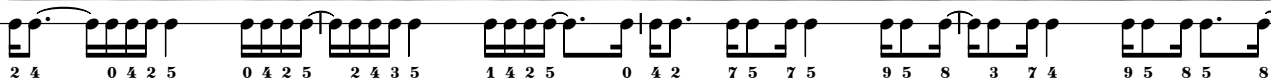
hl 6
(harm 1)



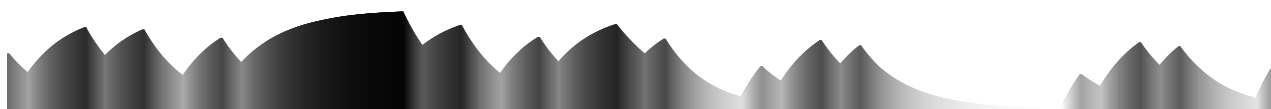
⑤



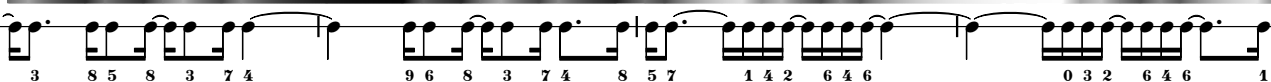
hl6 (h1)



⑨



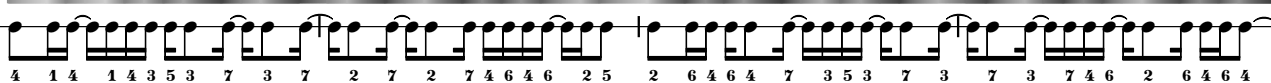
hl6 (h1)



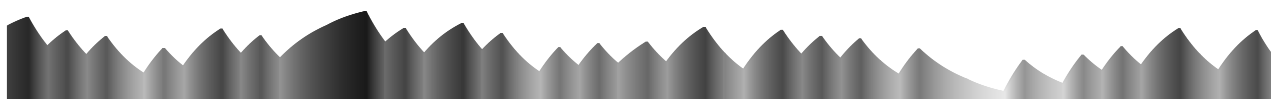
⑬



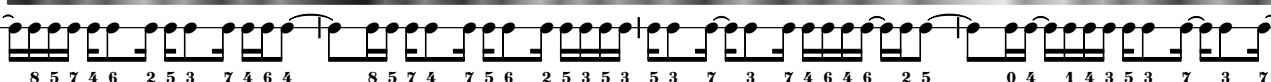
hl6 (h1)



⑰



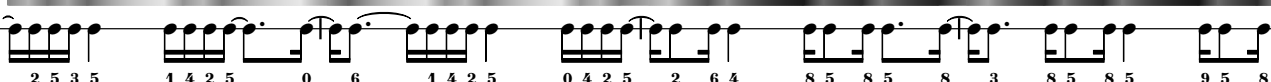
hl6 (h1)



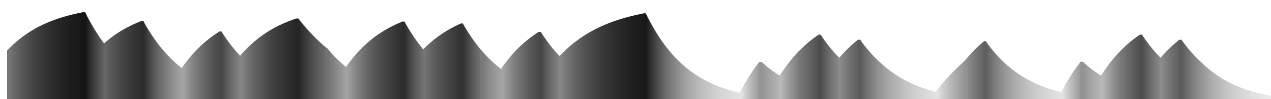
⑳



hl6 (h1)



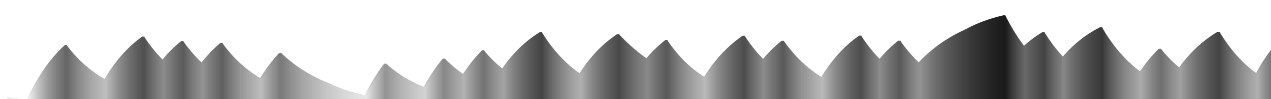
㉔



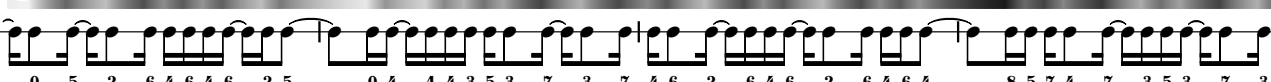
hl6 (h1)



㉘



hl6 (h1)



33

hl6 (h1)

5 3 5 3 7 4 6 2 5 3 7 4 6 2 5 3 7 4 6 2 5 3 7 4 6 2 6 2 7 2 7 2 7 2 7 2 7 2 7

37

hl6 (h1)

4 6 4 6 2 5 3 7 4 6 2 5 3 7 4 6 2 5 3 5 3 7 3 7 3 7 2 7 2 7 2 7 2 7 2 7

41

hl6 (h1)

4 9

45

hl6 (h1)

0 5 2 6 2 6 2 7 2 7 2 7 2 7 2 7 4 6 2 6 2 6 2 7 2 7 2 7 2 7 2 7 2

49

hl6 (h1)

8 5 6 4 7 3 7 4 6 4 8 5 7 4 7 3 7 4 6 4 7 3 5 2 5 3 7 4 6 2 5 0 4 1 4 3 7 4 6 2 5

53

hl6 (h1)

2 4 0 4 2 6 4 6 1 6 1 4 2 6 4 6 1 4 2 8 5 7 3 7 4 8 3 8 5 8 3 7 4 8

57

hl6 (h1)

3 7 4 6 4 8 5 7 4 7 3 7 4 6 4 8 5 7 4 7 5 6 4 6 2 5 0 4 1 4 3 7 4 6 2 5 0 4 1 4 3

61

hl6 (h1)

7 4 6 1 6 1 4 2 6 4 6 1 6 1 4 2 5 3 7 4 8 3 8 5 8 3 7 4 8 3 8 5 8

65

hl6 (h1)

5 6 2 6 4 6 2 6 4 6 4 7 4 7 5 8 3 5 3 7 4 6 2 6 2 7 4 6 4 6 2 5 2 4 3 5 0 4 2 6 2 6

69

hl6 (h1)

0 3 2 6 4 6 1 6 4 6 2 5 2 4 3 5 2 5 3 9 6 8 3 7 4 8 3 7 4 6 4 7 4 7 3 7

73

hl6 (h1)

3 7 4 9 5 8 5 8 3 8 5 8 5 9 6 7 4 6 1 4 2 5 0 6 1 4 2 5

77

hl6 (h1)

0 5 2 6 4 6 4 6 2 5 0 4 1 4 3 5 3 7 3 7 4 6 4 6 4 6 2 5 3 7 4 6 4 8 5 7 4 7 5 6 2 6 2

81

hl6 (h1)

7 4 6 4 6 2 5 3 7 4 6 4 8 3 7 3 8 5 6 4 7 3 7 3 7 4 6 4 6 2 5 1 6 2 6 1 4 1 4 3

85

hl6 (h1)

5 3 5 3 5 1 4 2 5 0 4 0 3 0 3 2 5 3 7 4 9 5 8 5 8 5 9 6 8 5 8

89

hl6 (h1)


5 9 5 8 3 7 4 8 3 7 3 7 2 7 4 6 4 7 0 4 2 6 4 6 1 6 4 6 2 5 3 7 4 6 2 5

93

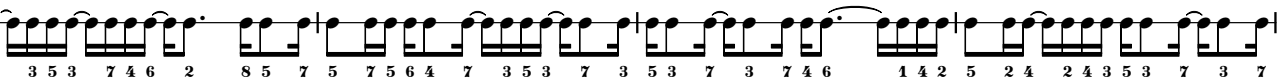
hl6 (h1)

1 6 2 6 4 6 4 6 2 5 1 6 2 6 4 6 4 6 1 6 2 6 4 6 2 6 4 6 4 7 4 6 4 7 4 6 2 6 4 8

97




hl6 (h1)

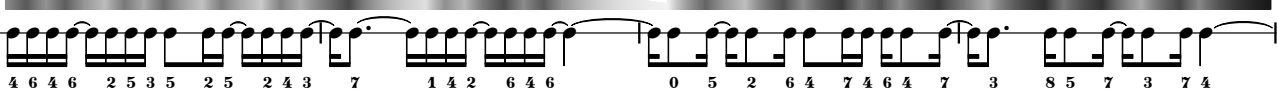


3 5 3 7 4 6 2 8 5 7 5 7 5 6 4 7 3 5 3 7 3 5 3 7 3 7 4 6 1 4 2 5 2 4 2 4 3 5 3 7 3 7

101

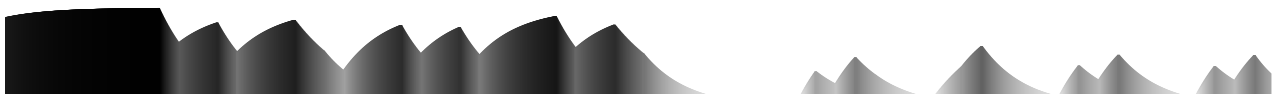


hl6 (h1)

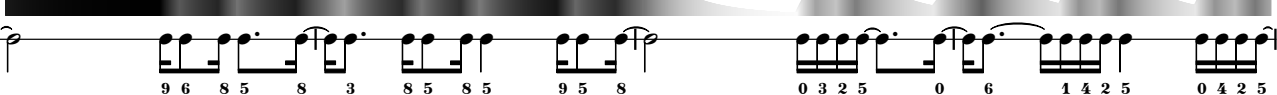


4 6 4 6 2 5 3 5 2 5 2 4 3 7 1 4 2 6 4 6 0 5 2 6 4 7 4 6 4 7 3 8 5 7 3 7 4

105

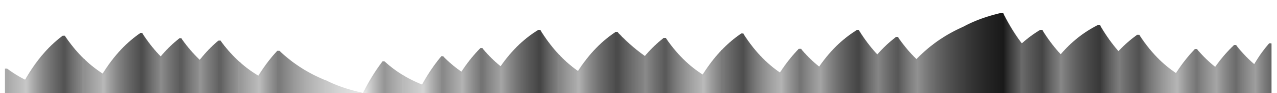


hl6 (h1)

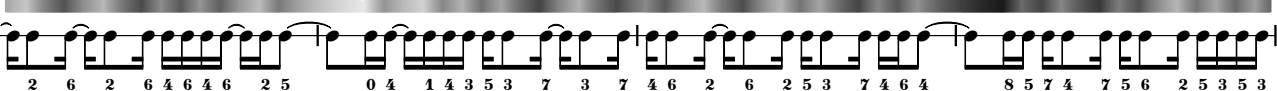


9 6 8 5 8 3 8 5 8 5 9 5 8 0 3 2 5 0 6 1 4 2 5 0 4 2 5

109

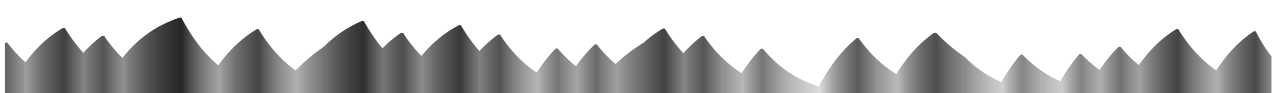


hl6 (h1)




2 6 2 6 4 6 4 6 2 5 0 4 1 4 3 5 3 7 3 7 4 6 2 6 2 5 3 7 4 6 4 8 5 7 4 7 5 6 2 5 3 5 3

113




hl6 (h1)

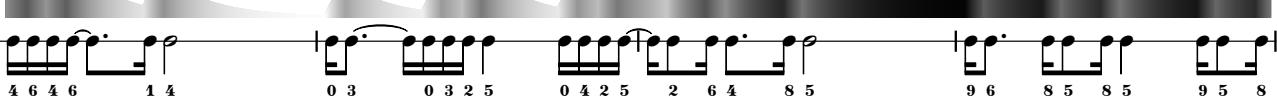


5 3 7 4 6 4 8 3 7 3 8 5 6 4 7 5 6 2 5 3 5 3 7 4 6 2 5 1 6 2 6 1 4 1 4 3 5 3 7 3 7

117

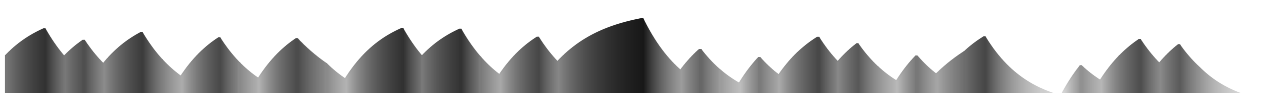


hl6 (h1)

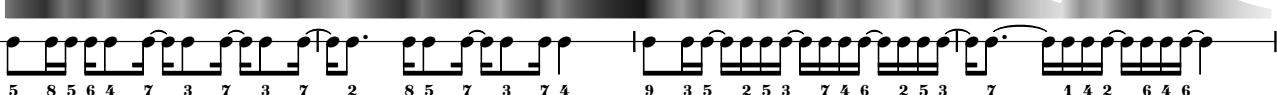


4 6 4 6 1 4 0 3 0 3 2 5 0 4 2 5 2 6 4 8 5 9 6 8 5 8 5 9 5 8

121

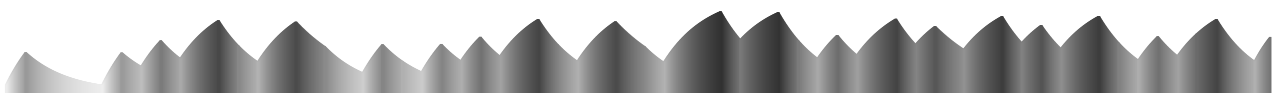


hl6 (h1)

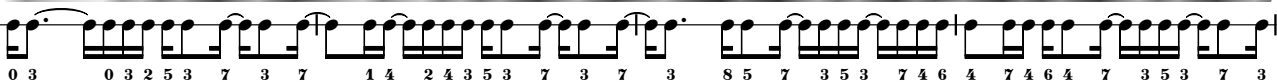


5 8 5 6 4 7 3 7 3 7 2 8 5 7 3 7 4 9 3 5 2 5 3 7 4 6 2 5 3 7 1 4 2 6 4 6

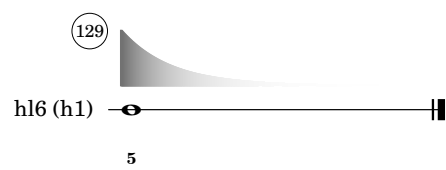
125



hl6 (h1)



0 3 0 3 2 5 3 7 3 7 1 4 2 4 3 5 3 7 3 7 3 8 5 7 3 5 3 7 4 6 4 7 4 6 4 7 3 5 3 7 3



a lot of tiles (trivial scan)

3231310

michael winter

(cdmx, mx and nashville, usa; 2018)

part - hl 5 - 3231310

hl 5
(high noise)

⑨

17

[illegible]

25

33

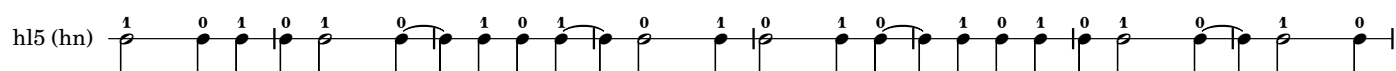
[illegible]

41

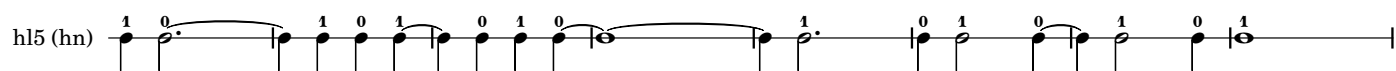
49

57

65



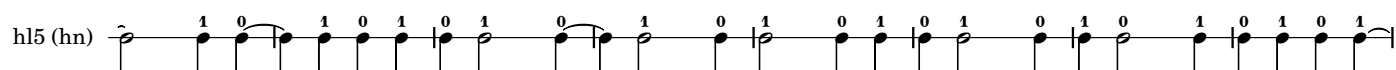
73



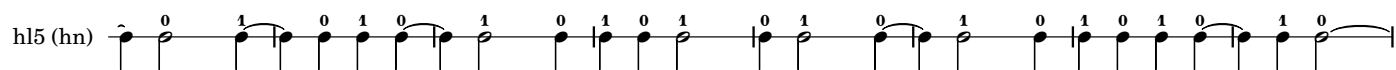
81



89



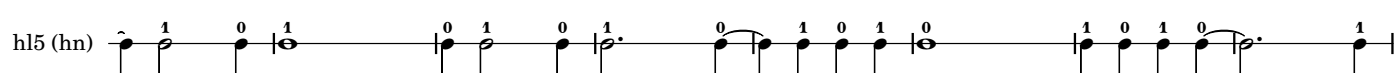
97



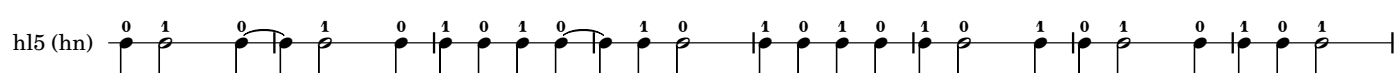
105



113



121



(part - hl 5 - 3231310)

129

hl5 (hn) $\overset{0}{\mathfrak{e}}\text{H}$

a lot of tiles (trivial scan)

3231310

michael winter

(cdmx, mx and nashville, usa; 2018)

part - hl 4 - 3231310

hl 4
(low noise)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a horizontal line. The notes are labeled with the sequence 0, 1, 0, 1, 0, 1, 0, 1. The first and third notes are marked with a '0' above them, and the second, fourth, sixth, and eighth notes are marked with a '1' above them. The first and third notes are connected by a slur, and the fourth and sixth notes are connected by a slur. The staff ends with a double bar line.

⑨

hl4 (ln)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a horizontal line. The notes are labeled with the sequence 1, 0, 1, 0, 1, 0, 1, 0. The first and third notes are marked with a '1' above them, and the second, fourth, sixth, and eighth notes are marked with a '0' above them. The first and third notes are connected by a slur, and the fourth and sixth notes are connected by a slur. The staff ends with a double bar line.

⑰

hl4 (ln)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a horizontal line. The notes are labeled with the sequence 1, 0, 1, 0, 1, 0, 1, 0. The first and third notes are marked with a '1' above them, and the second, fourth, sixth, and eighth notes are marked with a '0' above them. The first and third notes are connected by a slur, and the fourth and sixth notes are connected by a slur. The staff ends with a double bar line.

②⑤

hl4 (ln)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a horizontal line. The notes are labeled with the sequence 0, 1, 0, 1, 0, 1, 0, 1. The first and third notes are marked with a '0' above them, and the second, fourth, sixth, and eighth notes are marked with a '1' above them. The first and third notes are connected by a slur, and the fourth and sixth notes are connected by a slur. The staff ends with a double bar line.

③③

hl4 (ln)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a horizontal line. The notes are labeled with the sequence 1, 0, 1, 0, 1, 0, 1, 0. The first and third notes are marked with a '1' above them, and the second, fourth, sixth, and eighth notes are marked with a '0' above them. The first and third notes are connected by a slur, and the fourth and sixth notes are connected by a slur. The staff ends with a double bar line.

④①

hl4 (ln)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a horizontal line. The notes are labeled with the sequence 0, 1, 0, 1, 0, 1, 0, 1. The first and third notes are marked with a '0' above them, and the second, fourth, sixth, and eighth notes are marked with a '1' above them. The first and third notes are connected by a slur, and the fourth and sixth notes are connected by a slur. The staff ends with a double bar line.

④⑨

hl4 (ln)

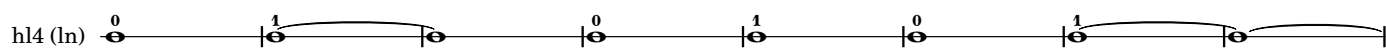
A musical staff with a single line. It contains eight notes, each with a vertical stem and a horizontal line. The notes are labeled with the sequence 1, 0, 1, 0, 1, 0, 1, 0. The first and third notes are marked with a '1' above them, and the second, fourth, sixth, and eighth notes are marked with a '0' above them. The first and third notes are connected by a slur, and the fourth and sixth notes are connected by a slur. The staff ends with a double bar line.

⑤⑦

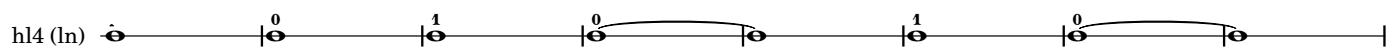
hl4 (ln)

A musical staff with a single line. It contains eight notes, each with a vertical stem and a horizontal line. The notes are labeled with the sequence 1, 0, 1, 0, 1, 0, 1, 0. The first and third notes are marked with a '1' above them, and the second, fourth, sixth, and eighth notes are marked with a '0' above them. The first and third notes are connected by a slur, and the fourth and sixth notes are connected by a slur. The staff ends with a double bar line.

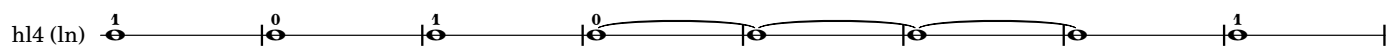
65



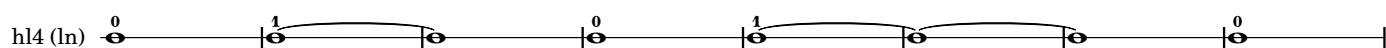
73



81



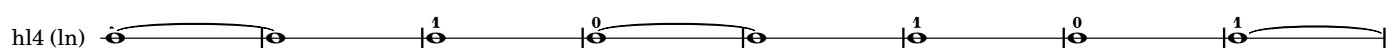
89



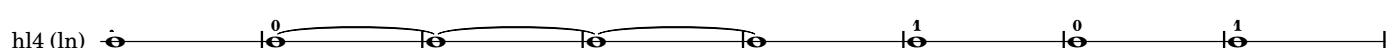
97



105




113



121



129

hl4 (ln) 

a_lot_of_tiles_trivial_scan_main.scd

```

1 (
2 // MAIN LAUNCH (loads necessary files and definitions)
3
4 var appEnvironment, cond;
5
6 s.boot;
7
8 appEnvironment = Environment.make;
9 appEnvironment.push;
10
11 // load
12 "a_lot_of_tiles_trivial_scan_tiler.scd".loadRelative;
13 "a_lot_of_tiles_trivial_scan_visualizer.scd".loadRelative;
14 "a_lot_of_tiles_trivial_scan_sonifier.scd".loadRelative;
15 "a_lot_of_tiles_trivial_scan_gui.scd".loadRelative;
16 "a_lot_of_tiles_trivial_scan_transcriber.scd".loadRelative;
17
18 // init
19 "dir = thisProcess.nowExecutingPath.dirname;
20 "loadedTransform = nil;
21 "transform = [2, 1, 0, 1, 0, 0, 0];
22 "tileMap = `mapAll.value(6 /*max depth*/, `transform);
23 "layoutState = 0;
24 "dur = 0.125;
25 "continuousPlay = false;
26
27 // launch
28 "launchTileVisualizer.value;
29 "launchGui.value;
30
31 appEnvironment.pop;
32 )

```

a_lot_of_tiles_trivial_scan_tiler.scd

```

1 (
2 var substitute, resamp, read, readPos = [0, 0];
3
4 // this function is the tiler
5 // it is essentially the generator of the underlying structure of the piece and all the variations
6 substitute = {arg parent, transform; var s, t, u, v, sub;
7   s = 4.collect({arg i; parent[0] + ((parent[i] - parent[0]) / 2)});
8   t = [parent[2] + ((parent[3] - parent[2]) / 2), parent[2], s[3], s[2]];
9   u = [t[0], s[1], t[0] + ((parent[1] - s[1]) / 2), s[1] + ((parent[1] - s[1]) / 2)];
10  v = [parent[1], parent[3], u[3], u[2]];
11  sub = [s, t, u, v].collect({arg child, i; if(transform[i] == nil, {nil},
12    {4.collect({arg j; child[(j.asDigits(2, 2) + transform[i].asDigits(2, 2)) % 2].convertDigits(2)})}));
13    sub.removeEvery([nil])});
14
15 // these functions either tile a specific depth (mapDepth) or up to a specific depth (mapAll)
16 "mapDepth = {arg depth, transform; var tiles = [[0, 0], [2, 0], [0, 1], [2, 1]];
17   depth.do({arg i; tiles = tiles.collect({arg tile; substitute.value(tile, transform)}).flatten});
18   [tiles]);
19
20 "mapAll = {arg depth, transform; var tiles = Array.fill(depth + 1, {1});
21   tiles[0].add([0, 0], [2, 0], [0, 1], [2, 1]);
22   depth.do({arg i; tiles[i].do({arg tile; tiles[i + 1] = tiles[i + 1].addAll(substitute.value(tile, transform))});
23     tiles});
24
25 // utility to resample the matrix output by `matricize
26 resamp = {arg matrixOrig, resampLevel; var mult, matrixNew = [];
27   mult = pow(2, resampLevel);
28   matrixOrig.do({arg rowOrig; var rowNew = [];
29     rowOrig.do({arg elem; mult.do({rowNew = rowNew.add(elem)}));
30     mult.do({matrixNew = matrixNew.add(rowNew)}));
31   matrixNew;
32
33 // function to turn the tile map into a matrix of 1s and 0s by reading blocks of 2 vertical or horizontal tiles
34 "matricize = {
35   arg tileMap, depth = 6, resampLevel = 0, transform;
36   var rotationState = transform[4], mirrorState = transform[5], invertState = transform[6], matrix;
37   matrix = Array.fill2D(pow(2, depth - 1), pow(2, depth), {0});
38   tileMap[tileMap.size - 1].do({arg tile; var transpose, xMin, xMax, yMin, yMax;
39     transpose = (tile * pow(2, depth)).flop;
40     xMin = minItem(transpose[0]); xMax = maxItem(transpose[0]);
41     yMin = minItem(transpose[1]); yMax = maxItem(transpose[1]);
42     if(xMin + yMin % 2 == 1, {if(xMax - xMin == 1,
43       {matrix[yMin / 2][xMin / 2] = (invertState + 1) % 2}, {matrix[yMin / 2][xMin / 2] = invertState}}));
44     if(resampLevel > 0, {matrix = resamp.value(matrix, resampLevel)});
45     matrix = switch(rotationState, 0, {matrix}, 1, {matrix.flop},
46       2, {matrix.reverse.flop.reverse.flop}, 3, {matrix.reverse.flop});
47     if(mirrorState == 1, {matrix = matrix.flop.reverse.flop});
48     matrix;
49
50 // tape reading model that reads matrix from `genData function
51 // wordShift and letterShift are only used if readPos is not reset elsewhere.
52 read = {
53   arg matrix, wordLen = 1, wordShift = [0, 1], letterShift = [0, 1], wrap = true;
54   var width = matrix.shape[1], height = matrix.shape[0], startReadPos = readPos, word = [];
55   wordLen.do({arg i;
56     word = word.add(matrix[readPos[0] % height][readPos[1] % width]);
57     readPos = readPos + letterShift;
58     readPos = startReadPos + wordShift; word.convertDigits(2)});
59
60 // read the matrix - this data is ultimately used to generate the piece
61 "genData = {arg matrix, transform, partSize, shift = 0, pad = 16, minScale = 0, maxScale = 1;
62   var width = matrix.shape[1], height = matrix.shape[0], res;
63   readPos = [0, 0];
64   res = partSize.collect({arg p; height.collect({arg i;
65     readPos[0] = 1 + p + shift;
66     width.collect({arg j; var val = read.value(matrix;
67       minScale + ((maxScale - minScale) * val)});
68     }).flatten.addAll(Array.fill(pad, {0}))});
69 }

```

a.lot_of_tiles.trivial_scan_sonifier.scd

```

1  (
2  // synth for hl 6
3  SynthDef(\hl6.sine, {arg freq, amp = 1, pos = 0;
4    Out.ar(0, Pan2.ar(SinOsc.ar(freq, 0, Lag.kr(amp, 2)), pos))
5  }).store;
6
7  // synth for hl 5
8  SynthDef(\hl5.high.noise, {arg amp = 1, pos = 0;
9    Out.ar(0, Pan2.ar(HPF.ar(WhiteNoise.ar(Lag.kr(amp * (1 / 128), 0.05)), 5000), pos))
10  }).store;
11
12  // synth for hl 4
13  SynthDef(\hl4.low.noise, {arg amp = 1, pos = 0;
14    Out.ar(0, Pan2.ar(LPF.ar(WhiteNoise.ar(Lag.kr(amp * (1 / 4), 0.05)), 300), pos))
15  }).store;
16
17  // synth for amp curves for score
18  SynthDef(\lamp, {arg freq, amp = 1;
19    Out.ar(0, Lag.kr(K2A.ar(amp), 2))
20  }).store;
21
22  // synth for playback
23  SynthDef(\play, {arg sinePlayer1, sinePlayer2, highNoisePlayer, lowNoisePlayer,
24    sineBuf1 = 0, sineBuf2 = 1, highNoiseBuf = 2, lowNoiseBuf = 3,
25    eTracks, eTracksPanned, eMaster,
26    eVol = #[0.8, 0.8, 0.8, 0.8], eMute = #[1, 1, 1, 1], ePan = #[0, 0, 0, 0], masterVol = 1, masterMute = 1,
27    playRate = 0, startPos = 0, startTrig = 0, hash;
28    var dStartTrig, phasor, countOff, imp, curBeat;
29
30    countOff = PulseCount.kr(impulse.kr(4), startTrig) * startTrig;
31    dStartTrig = countOff > 17;
32    phasor = Phasor.ar(dStartTrig, Select.kr(playRate * dStartTrig, [0, BufRateScale.kr(sineBuf1)]),
33      0, BufFrames.kr(sineBuf1), startPos * BufFrames.kr(sineBuf1));
34    curBeat = ((A2K.kr(phasor) / BufFrames.kr(sineBuf1)) * BufDur.kr(sineBuf1) * 4).trunc;
35    curBeat = Select.kr(dStartTrig, [countOff - 18, curBeat]);
36
37    sinePlayer1 = PlayBuf.ar(2, sineBuf1, playRate * dStartTrig, dStartTrig, startPos * BufFrames.kr(sineBuf1));
38    sinePlayer2 = PlayBuf.ar(2, sineBuf2, playRate * dStartTrig, dStartTrig, startPos * BufFrames.kr(sineBuf2));
39    highNoisePlayer = PlayBuf.ar(2, highNoiseBuf, playRate * dStartTrig, dStartTrig, startPos * BufFrames.kr(highNoiseBuf));
40    lowNoisePlayer = PlayBuf.ar(2, lowNoiseBuf, playRate * dStartTrig, dStartTrig, startPos * BufFrames.kr(lowNoiseBuf));
41
42    eTracks = {arg i;
43      [Mix.new(sinePlayer1), Mix.new(sinePlayer2), Mix.new(highNoisePlayer), Mix.new(lowNoisePlayer)][i] * eVol[i] * eMute[i] ! 4;
44    eTracksPanned = {arg i; Pan2.ar(eTracks[i], ePan[i]) ! 4;
45    eMaster = Mix.new(eTracksPanned) * masterVol * masterMute * dStartTrig.lag;
46    Out.ar(0, eMaster);
47
48    // optional click — uncomment and send to an output not used
49    //Out.ar(2, 10 * BPF.ar(WhiteNoise.ar * EnvGen.kr(Env.perc(0.01, 0.1), curBeat % 2 <= 0), 440 * ((curBeat % 8 <= 0) + 1), 0.02));
50    SendTrig.kr(Changed.kr(curBeat), hash, curBeat);
51    imp = Impulse.kr(10);
52    SendReply.kr(imp, '/masterLevel', [Amplitude.kr(eMaster)], hash);
53    SendReply.kr(imp, '/trackLevels', [Amplitude.kr(eTracks)], hash)}.add;
54
55  // generate audio for playback
56  ^genPattern = {arg data, ins, durMult, genAudio = false, cond = nil;
57    var pattern;
58    pattern = Ppar(data.collect({
59      arg seq, p;
60      switch(ins,
61        "hl6.fundamental", {Pmono(\hl6.sine, \freq, 31.midicps,
62          \dur, \dur, \amp, Pseq(seq), \pos, 0)},
63        "hl6.harmonics", {Pmono(\hl6.sine, \freq, 31.midicps + ((31.midicps * 4) * (p + 1)),
64          \dur, \dur, \amp, Pseq(seq * (1 / pow(1 + (p + 1) * 4), 1)), \pos, 0)},
65        "hl5.high.noise", {Pmono(\hl5.high.noise, \dur, \dur * durMult, \amp, Pseq(seq), \pos, 0)},
66        "hl4.low.noise", {Pmono(\hl4.low.noise, \dur, \dur * durMult, \amp, Pseq(seq), \pos, 0)}});
67    if(genAudio, {File.mkdir(~dir + "/" + "." + "audio" + "/" + "transform" + "transform.join + "audio" + "/" + ins + ".wav", \dur * durMult * data[0].size,
68      pattern.render(~dir + "/" + "." + "audio" + "/" + "transform" + "transform.join + "audio" + "/" + ins + ".wav", \dur * durMult * data[0].size,
69      headerFormat: "WAV", sampleRate: s.sampleRate, action: {if(cond != nil, {cond.unhang}})}));
70    pattern;
71  })

```

a_lot_of_tiles_trivial_scan_visualizer.scd

```

1  (
2  var normalizeTile, getMasterRect, tileDrawFunc;
3
4  // provide scaled coordinate for the tiles
5  normalizeTile = {arg child, parent;
6    var parentXMin, parentYMin, parentWidth, parentHeight, p1, p2, p3, p4;
7    parentXMin = minItem(parent.slice(nil, 0));
8    parentYMin = minItem(parent.slice(nil, 1));
9    parentWidth = (maxItem(parent.slice(nil, 0)) - parentXMin);
10   parentHeight = (maxItem(parent.slice(nil, 1)) - parentYMin);
11   p1 = [(minItem(child.slice(nil, 0)) - parentXMin) / parentWidth, (minItem(child.slice(nil, 1)) - parentYMin) / parentHeight];
12   p2 = [(maxItem(child.slice(nil, 0)) - parentXMin) / parentWidth, p1[1]];
13   p3 = [p1[0], (maxItem(child.slice(nil, 1)) - parentYMin) / parentHeight];
14   p4 = [p2[0], p3[1]];
15   [p1, p2, p3, p4]];
16
17 // get the master rectangle for the tiling based on orientation
18 getMasterRect = {arg tileWin, rotationState;
19   var rectWidth, rectHeight;
20   case({(rotationState % 2 == 0) && (tileWin.bounds.width <= (tileWin.bounds.height * 2))}, {
21     rectWidth = tileWin.bounds.width - 20;
22     rectHeight = rectWidth / 2,
23     {(rotationState % 2 == 0) && (tileWin.bounds.width > (tileWin.bounds.height * 2))}, {
24       rectHeight = tileWin.bounds.height - 20;
25       rectWidth = rectHeight * 2,
26       {(rotationState % 2 == 1) && (tileWin.bounds.height <= (tileWin.bounds.width * 2))}, {
27         rectWidth = tileWin.bounds.height - 20;
28         rectHeight = rectWidth / 2,
29         {(rotationState % 2 == 1) && (tileWin.bounds.height > (tileWin.bounds.width * 2))}, {
30           rectWidth = tileWin.bounds.width - 20;
31           rectHeight = rectWidth / 2;
32         Rect.new((tileWin.bounds.width - rectWidth) / 2, (tileWin.bounds.height - rectHeight) / 2, rectWidth, rectHeight));
33
34 // draw the tiling
35 tileDrawFunc = {arg tileWin, tileMap, transform, layoutState;
36   var rotationState = transform[4], mirrorState = transform[5], invertState = transform[6],
37   masterRect, parentRects, depthLevels;
38
39   masterRect = getMasterRect.value(tileWin, rotationState);
40   parentRects = [[masterRect], [], []];
41
42   Pen.width = (masterRect.width + masterRect.height) / 8000;
43   Pen.scale(abs(mirrorState - 1) * 2 - 1, 1);
44   Pen.translate(mirrorState * -1 * tileWin.bounds.width, 0);
45   Pen.rotate(0.5 * rotationState * pi, tileWin.bounds.width / 2, tileWin.bounds.height / 2);
46
47   depthLevels = switch(layoutState, 0, {[6]}, 1, {[5]}, 2, {[4]}, 3, {[4, 5, 6]});
48   depthLevels.do({arg depth, dIndex;
49     tileMap[depth].do({arg childTile, cIndex;
50       var childTileNorm, parentRect, p1, p2, childRect;
51
52       childTileNorm = if(dIndex > 0, {normalizeTile.value(childTile, tileMap[depth - 1][(cIndex / 4).trunc]}), {childTile});
53       parentRect = if(dIndex > 0, {parentRects[dIndex][(cIndex / 4).trunc]}, {parentRects[0][0]});
54       if((layoutState == 3) && (dIndex > 0), {parentRect = parentRect.insetBy(parentRect.width * 0.04, parentRect.height * 0.04)});
55
56       p1 = Point.new(childTileNorm[0][0] * parentRect.width / if(dIndex > 0, {1}, {2}), childTileNorm[0][1] * parentRect.height);
57       p2 = Point.new(childTileNorm[3][0] * parentRect.width / if(dIndex > 0, {1}, {2}), childTileNorm[3][1] * parentRect.height);
58
59       childRect = Rect.fromPoints(p1, p2).moveBy(parentRect.left, parentRect.top);
60       if((layoutState == 3) && (depth < 6), {parentRects[dIndex + 1] = parentRects[dIndex + 1].add(childRect)});
61
62       if(if(invertState == 0, {abs(p1.x - p2.x) > abs(p1.y - p2.y)}, {abs(p1.x - p2.x) < abs(p1.y - p2.y)}),
63         {Pen.color = Color.white; Pen.fillRect(childRect); Pen.color = Color.black; Pen.strokeRect(childRect)},
64         {Pen.color = Color.black; Pen.fillRect(childRect); Pen.color = Color.white; Pen.strokeRect(childRect)});
65     }));
66
67 // launch the visualization
68 launchTileVisualizer = {
69   arg border = true, isLaunch = true;
70   var tileWin;
71   tileWin = Window("tile", if(isLaunch, {Rect(200, 500, 1000, 500)}, {"tileWin.bounds}), true, border);
72   tileWin.view.background(Color.gray(0.2));
73   tileWin.drawFunc = {arg win; tileDrawFunc.value(win, tileMap, transform, layoutState).inEnvir;
74   tileWin.front;
75   tileWin.refresh}
76 )

```

a_lot_of_tiles_trivial_scan_transcriber.scd

```

1 (
2   -transcribe = {arg data, durMult, subName, drawAmps = false;
3
4   data.do({arg seq, p;
5     fork {var basePath, cond, pattern, ampFilePath, ampSampleRate, ampBuf, amps,
6       lilyFile, lastVal, curTime = 0, lilyString = "";
7
8       basePath = "dir" + "/" + "." + "/" + "score" + "/" + "transform" + "/" + "transform.join" + "/" + "score" + "/" + "lilypond";
9
10      // writes the template files for the master score and parts
11      if(File.exists(basePath).not, {arg template, writeTemplate;
12        writeTemplate = {arg template, part; var fileAppend, scoreType, systemsPerPage, lileFile, formattedTemplate;
13          fileAppend = switch(part, nil, {"master.score"}, 6, {"part.h16"}, 5, {"part.h15"}, 4, {"part.h14"});
14          scoreType = switch(part, nil, {"master.score"}, 6, {"part - h1 6"}, 5, {"part - h1 5"}, 4, {"part - h1 4"});
15          lilyFile = File(basePath + "/" + "a_lot_of_tiles_trivial_scan_transform" + "/" + "transform.join
16            ++ " " + fileAppend + ".ly".standardizePath, "w");
17          formattedTemplate = template.replace("`score.type`", scoreType);
18          if(part != nil, {formattedTemplate = formattedTemplate
19            .replace("systems-per-page = 4", "systems-per-page = 8");
20            [4, 5, 6].difference([part]).do({arg p; formattedTemplate = formattedTemplate
21              .replace("%`h1`" ++ p ++ "start`", "%`h1`" ++ p.asString ++ "`");
22              .replace("%`h1`" ++ p.asString ++ "end`", "%`");});
23          if(part != 6, {formattedTemplate = formattedTemplate
24            .replace("unfold 33", "unfold 17").replace("unfold 3", "unfold 7")});});
25          lilyFile.write(formattedTemplate);
26          lilyFile.close();
27
28          File.mkdir(basePath);
29          File.mkdir(basePath + "/" + "includes");
30          File.mkdir(basePath + "/" + "." + "/" + "pdf");
31          template = File.readAllString("dir" + "/" + "." + "/" + "score" + "/" + "score.template"
32            ++ "/" + "a_lot_of_tiles_trivial_scan.score.template.ly".standardizePath);
33          template = template.replace("`transform`", "transform.join");
34
35          writeTemplate.value(template, nil);
36          writeTemplate.value(template, 6);
37          writeTemplate.value(template, 5);
38          writeTemplate.value(template, 4);
39        });
40
41      cond = Condition.new;
42
43      // generate pattern and get amplitude values for h16
44      pattern = Pmono(\lamp, \dur, \dur * durMult, \amp, Pseq(seq));
45      if(drawAmps, {
46        ampFilePath = basePath + "/" + "amp_curve_tmp.h16.harm." + (1 + (p * 4)).asString() + ".wav";
47        pattern.render(ampFilePath, \dur * durMult * seq.size, action: {cond.unhang}); cond.hang;
48        ampSampleRate = SoundFile(ampFilePath).sampleRate;
49        Buffer.readChannel(s, ampFilePath, channels: [0], action: {arg ampBuf;
50          ampBuf.loadToFloatArray(action: {arg array; amps = array; cond.unhang});}); cond.hang;});
51
52      lilyFile = if(data.size > 1, {lilyFile = subName + ".harm." + (1 + (p * 4)).asString(), {subName}};
53      lilyFile = File(basePath + "/" + "includes" + "/" + "a_lot_of_tiles_trivial_scan" + "/" + lilyFile + ".ly".standardizePath, "w");
54
55      // main transcribe code which groups data into beats and creates lilypond code accordingly
56      lastVal = nil;
57      seq.collect({arg v; durMult.collect({v})}).flat.clump(4).do({arg beat, i;
58        beat.separate({arg a, b; a != b}).do({arg group; var noteLength, target = 0, colors = "", lilyNote;
59          noteLength = group.size;
60          lilyNote = switch(noteLength, 1, {"b'16"}, 2, {"b'8"}, 3, {"b'8."}, 4, {"b'4"});
61
62          // lilypond directives to draw amplitude gradients for h16
63          if(drawAmps, {
64            colors = amps[(ampSampleRate * curTime).asInteger)..(ampSampleRate * (curTime + (noteLength / 8))).asInteger];
65            target = colors[0];
66            colors = 1 - (colors.resamp0(noteLength * 50));
67            colors = colors.collect({arg color; color.round(0.0001).asString});
68            colors = "\override Glissando.details.colors = #' " ++ " " ++ colors.join(" ") ++ " "});
69
70            // 0 or 1 above every note based on data
71            if(drawAmps.not, {lilyNote = lilyNote + " " ++ group[0].asString ++ " "});
72
73            // ties notes
74            lilyString = lilyString ++ if(group[0] == lastVal, {" "}, {" "});
75            lilyString = lilyString ++ if((curTime % 2 == 0), {"|"}, {" "});
76            lilyString = lilyString ++ colors ++ lilyNote;
77            lilyString = lilyString ++ if(drawAmps && (group[0] != lastVal),
78              {"-\\tweak Y-offset #-5.5 ." ++ (target.trunc(0.1) * 10).asString}, {" "});
79            lilyString = lilyString ++ if(drawAmps, {"\\n"}, {" "});
80
81            lastVal = group[0];
82            curTime = curTime + (noteLength / 8);
83          });
84        });
85
86      // consolidates tied quarters for h14 and h15
87      if(drawAmps.not, {
88        lilyString.findRegexp("b'4 \\` (0|1) ` b'4 \\` (0|1) ` b'4 \\` (0|1) ` b'4 \\` (0|1)".clump(5).do(
89          {arg match; lilyString = lilyString.replace(match[0][1], "b'1 ~" ++ match[1][1]);});
90        lilyString.findRegexp("b'4 \\` (0|1) ` b'4 \\` (0|1) ` b'4 \\` (0|1)".clump(4).do(
91          {arg match; lilyString = lilyString.replace(match[0][1], "b'2. ~" ++ match[1][1]);});
92        lilyString.findRegexp("b'4 \\` (0|1) ` b'4 \\` (0|1)".clump(3).do(
93          {arg match; lilyString = lilyString.replace(match[0][1], "b'2 ~" ++ match[1][1]);});
94        lilyString.findRegexp(" \\[b'(1|2.|2|4) \\` (0|1)".clump(3).do(
95          {arg match; lilyString = lilyString.replace(match[0][1], " |b'" ++ match[1][1]);});});
96
97      // prettify
98      lilyString = lilyString.replace(" ", "\\n\\n");
99      lilyString = lilyString.replace("\\n", " \\n");
100
101      // consolidates tied quarters for h16 (very tricky)
102      if(drawAmps, {var replaceFirst, tmp;
103        replaceFirst = { arg string, find, replace = "", offset = 0;
104          var index, out = "", array = string, findSize = max(find.size, 1);
105          index = array.find(find, offset: offset);
106          out = out ++ array.keep(index) ++ replace;
107          array = array.drop(index + findSize);
108          out ++ array;});
109
110      tmp = lilyString;
111      lilyString.split($).do({arg bar; var beats, groups, offset = 0;
112        beats = bar.split($);
113        groups = beats.separate({arg a, b; (a.contains("b'4") && a.contains("~") && b.contains("b'4")).not });});
114        groups.do({arg group;
115          if(group.size > 1, {
116            var ampList, noteLength, tweak, tie, replace, find;
117            ampList = group.collect({arg note; note[ (note.find("(") + 1)..(note.find(")") - 1)]}.join(" ");
118            noteLength = switch(group.size, 2, {"2"}, 3, {"2."}, 4, {" "});
119            tweak = if(group[0].contains("tweak"), {group[0][group[0].find("(")\\tweak)..(group[0].find("#-5.5") + 8)], {" "});
120            tie = if(group.last.contains("("), {" "}, {" "});
121            replace = "\\override Glissando.details.colors = #'(" ++ ampList ++ ") b'" ++ noteLength ++ tweak ++ tie;

```

```

122         find = group.join("\n");
123         tmp = replaceFirst.value(tmp, find, replace, offset));
124         offset = lilyString.find("|", offset: offset)}});
125         lilyString = tmp));
126
127
128         // final lilypond directives and write files
129         if(drawAmps, {
130             lilyString = "\override Glissando.stencil = #my-gliss \\addGliss { \\override Glissando.details.thickness = 1.15 " ++ lilyString ++ " }";
131             lilyString = "{ " ++ lilyString ++ " \\bar \"|.\" \\stopStaff \\hide b'l}";
132             lilyFile.write(lilyString);
133             lilyFile.close;
134
135             // this deletes the tmp files and creates the pdfs on the last call.
136             if(drawAmps, {
137                 File.delete(ampFilePath);
138                 if(p == 3, {
139                     "which lilypond".unixCmd({ arg res1, pid1;
140                         if(res1 == 0, {
141                             PathName(basePath).files.do({arg path;
142                                 ("lilypond -o " ++ basePath + "/" + "pdf " ++ path.fullPath).unixCmd({ arg res2, pid2;
143                                     if(res2 != 0, {{var win, text;
144                                         win = Window.new.front;
145                                         text = StaticText(win, Rect(10, 10, 300, 200));
146                                         text.string = "Something went wrong creating the pdfs, \n" ++
147                                             "but the lilypond files likely generated just fine".defer});
148                                         [[done, res2, pid2].postln]]), {{var win, text;
149                                             win = Window.new.front;
150                                             text = StaticText(win, Rect(10, 10, 300, 200));
151                                             text.string = "Something went wrong creating the pdfs. \nLikely Lilypond is not installed, \n" ++
152                                                 "but the Lilypond files probably generated just fine."}.defer});
153                                         [[done, res1, pid1].postln]]))}}));
154             });
155         )

```

a_lot_of_tiles_trivial_scan_gui.scd

```

1  (
2  var modelView, transportView, mixerView, guiLayout, playButton, randButton;
3
4  // model / transform
5  modelView = {var transformGrid, transformButtons, layouts, layoutGrid, layoutButtons,
6  fullScreenButton, decoratorButton, transcribeButton, contPlayButton, exportButton, displayGrid, model;
7
8  // sets transform
9  transformGrid = GridLayout();
10 transformButtons = Array.fill2D(7, 4, {arg r, c; Button().states({
11   [[c.asString, Color.grey, Color.white], [c.asString, Color.white, Color.grey]]});
12 transformButtons.do({arg row, rIndex;
13   row.do({arg button, cIndex;
14     if((rIndex < 5) || (cIndex < 2), {
15       transformGrid.add(button, rIndex, cIndex + 1);
16       button.action = {transformButtons[rIndex].do(
17         {arg b, i; if(button == b, {b.value = 1}, {b.value = 0})});
18       ^transform[rIndex] = cIndex; if(rIndex < 4, {^tileMap = ^mapAll.value(6, ^transform)}; ^tileWin.refresh}.inEnvir)}});
19   transformGrid.add(if(rIndex < 4,
20     {StaticText().string("child " ++ rIndex.asString).align(\center)},
21     {StaticText().string(("rotation", "mirror", "invert")[rIndex - 4])
22       .align(\center)}), rIndex, 0));
23   ^transform.do({arg v, rIndex; transformButtons[rIndex][v].value = 1});
24
25 // sets layout
26 layouts = ["depth level 6 only (max depth)", "depth level 5 only", "depth level 4 only", "hierarchical (embedded)",
27   "", "", ""];
28 layoutGrid = GridLayout();
29 layoutButtons = layouts.collect({arg l, i; Button().states({
30   [[layouts[i], Color.grey, Color.white], [l, Color.white, Color.grey]]});
31 layoutButtons.do({arg button, bIndex; if(bIndex < 4,
32   {layoutGrid.add(button, bIndex, 0);
33   button.action = {layoutButtons.do({arg b; if(button == b, {b.value = 1}, {b.value = 0})};
34     ^layoutState = bIndex; ^tileWin.refresh}}.inEnvir},
35   {layoutGrid.add(StaticText().string(""), bIndex, 0)}});
36 layoutButtons[0].value = 1;
37
38 randButton = Button().states({"random transform", Color.grey, Color.white}).action({
39   ^transform = Array.fill(7, {arg i; if(i < 5, {4.rand}, {2.rand})});
40   ^tileMap = ^mapAll.value(6, ^transform);
41   ^tileWin.refresh;
42   transformButtons.do({arg row; row.do({arg button; button.value = 0})});
43   ^transform.do({arg v, rIndex; transformButtons[rIndex][v].value = 1}).inEnvir;
44
45 // sets display
46 fullScreenButton = Button().states({"fullscreen", Color.grey, Color.white},
47   ["fullscreen", Color.white, Color.grey]).action({arg button;
48   if(button.value == 0, {^tileWin.endFullScreen}, {^tileWin.fullScreen}).inEnvir;
49
50 ^tileWin.view.keyDownAction = {arg doc, char, mod, unicode, keycode, key;
51   case
52   // <ctrl + f> = enter full screen
53   {mod == 262144 && key == 70} {^tileWin.fullScreen; fullScreenButton.value = 1}
54   // <esc> = exit full screen
55   {mod == 0 && key == 16777216} {^tileWin.endFullScreen; fullScreenButton.value = 0}.inEnvir;
56
57 decoratorButton = Button().states(["window decorator", Color.grey, Color.white],
58   ["window decorator", Color.white, Color.grey]).action({arg button;
59   ^tileWin.close;
60   if(button.value == 0, {^launchTileVisualizer.value(false, false)},
61     {^launchTileVisualizer.value(true, false)}.inEnvir;
62 decoratorButton.value = 1;
63
64 // set continuous play option (disables transform buttons when set)
65 contPlayButton = Button().states({"continuous play", Color.grey, Color.white},
66   ["continuous play", Color.white, Color.grey]).action({arg elem;
67   if(elem.value == 0, {^continuousPlay = false;
68     transformButtons.do({arg row; row.do({arg button; button.enabled = true})});
69     randButton.enabled = true, {
70     transformButtons.do({arg row; row.do({arg button; button.enabled = false})});
71     randButton.enabled = false;
72     if(playButton.value == 0, {
73       fork({
74         ^continuousPlay = true;
75         randButton.valueAction = 0;
76         l.wait;
77         playButton.valueAction = 1, AppClock)}}).inEnvir;
78
79 // transcribe
80 transcribeButton = Button().states({"transcribe", Color.grey, Color.white}).action({
81   var matrixH4, matrixH5, matrixH6, dataH4, dataH5, dataH6;
82   matrixH4 = ^matricize.value([ ^tileMap[4]], 4, 0, ^transform);
83   matrixH5 = ^matricize.value([ ^tileMap[5]], 5, 0, ^transform);
84   matrixH6 = ^matricize.value([ ^tileMap[6]], 6, 0, ^transform);
85   dataH4 = ^genData.value(matrixH4, ^transform, 1, 0, 1);
86   dataH5 = ^genData.value(matrixH5, ^transform, 1, 0, 4);
87   dataH6 = ^genData.value(matrixH6, ^transform, 4, 0, 16);
88   ^transcribe.value(dataH4, 16, "hl.4");
89   ^transcribe.value(dataH5, 4, "hl.5");
90   ^transcribe.value(dataH6, 1, "hl.6", true).inEnvir;
91
92 // export tile image
93 exportButton = Button().states({"export image", Color.grey, Color.white}).action({var img;
94   img = if(^transform[4] % 2 == 0, {Image.new(6000, 3000)}, {Image.new(3000, 6000)});
95   img.draw({ ^tileDrawFunc.value(img, ^tileMap, ^transform, ^layoutState)});
96   FileDialog({ arg path; img.write(path[0]), {}, 0, 1}).inEnvir;
97
98 displayGrid = GridLayout();
99 displayGrid.add(fullScreenButton, 0, 0);
100 displayGrid.add(decoratorButton, 1, 0);
101 5.do({arg i; displayGrid.add(StaticText().string(""), i + 2, 0));
102
103 model = GridLayout();
104 model.add(StaticText().string("transform code", 0, 0);
105 model.add(transformGrid, 1, 0); model.add(randButton, 2, 0);
106 model.add(StaticText().string("", 3, 0); model.add(StaticText().string(""), 4, 0);
107 model.add(StaticText().string("", 0, 1);
108 model.add(StaticText().string("layout", 0, 2);
109 model.add(layoutGrid, 1, 2);
110 model.add(StaticText().string("", 0, 3);
111 model.add(StaticText().string("display", 0, 4);
112 model.add(displayGrid, 1, 4); model.add(contPlayButton, 2, 4); model.add(transcribeButton, 3, 4); model.add(exportButton, 4, 4);
113 model;
114
115 // transport
116 transportView = {
117   arg hash; var clockStringFunc, metronomeStringFunc, metronomeColorFunc,
118   transport, posSlider, startPos = 0, startPosText, pauseButton, clock, metronome;
119
120 // update clock and metronome
121 clockStringFunc = {

```

```

122     arg curBeat; var measure, beat;
123     beat = ((curBeat / 2) % 4 + 1).trunc.asString;
124     measure = ((curBeat / 8) + 1).trunc.asString;
125     if (measure.size == 1, {measure = " " ++ measure});
126     if (measure.size == 2, {measure = " " ++ measure});
127     measure ++ ":" ++ beat;
128     // [-30, -105, -104].asAscii and [-30, -105, -113].asAscii are unicode inverse bullet and normal bullet, respectively
129     metronomeStringFunc = {arg curBeat; case {curBeat % 8 < 1} {[-30, -105, -104].asAscii}
130     {curBeat % 2 < 1} {[-30, -105, -113].asAscii} {true} {" "}};
131     metronomeColorFunc = {arg curBeat; case {curBeat % 32 < 1} {Color.red} {curBeat % 8 < 1} {Color.blue} {true} {Color.black}};
132
133     // get current time and call update clock and metronome
134     OSCFunc({arg msg, time; {
135         if (msg[2] == hash, {
136             if (msg[3] > 0, {clock.string = clockStringFunc.value(msg[3])});
137             metronome.stringColor = metronomeColorFunc.value(msg[3]);
138             metronome.string = metronomeStringFunc.value(msg[3]);
139             // if continuous play, when finished, delete all autogenerated files, reselect a transform, and then start
140             if (msg[3] == 1031, {
141                 playButton.valueAction = 0;
142                 fork({
143                     if (continuousPlay, {var baseDir;
144                         baseDir = "dir +/" ++ "." ++ "audio" ++ "transform." ++ "transform.join ++ "audio";
145                         File.delete(baseDir) ++ "hl.6.fundamental.wav");
146                         File.delete(baseDir) ++ "hl.6.harmonics.wav");
147                         File.delete(baseDir) ++ "hl.5.high.noise.wav");
148                         File.delete(baseDir) ++ "hl.4.low.noise.wav");
149                         File.delete(baseDir);
150                         randButton.valueAction = 0;
151                         posSlider.valueAction = 0;
152                         2.wait;
153                         playButton.valueAction = 1
154                     }}, AppClock)}}).inEnvir.defer}.inEnvir, 'tr', s.addr);
155
156     // transport gui items
157     transport = HLayout(
158         // play / stop button
159         playButton = Button().states(["play", Color.black], ["stop", Color.black, Color.grey]).action({arg elem;
160             fork {var cond, baseDir;
161                 cond = Condition.new;
162                 baseDir = "dir +/" ++ "." ++ "audio" ++ "transform." ++ "transform.join ++ "audio";
163                 // generate audio files if they do not exist
164                 if (File.exists(baseDir) ++ "hl.6.fundamental.wav").not ||
165                     File.exists(baseDir) ++ "hl.6.harmonics.wav").not ||
166                     File.exists(baseDir) ++ "hl.5.high.noise.wav").not ||
167                     File.exists(baseDir) ++ "hl.4.low.noise.wav").not, {
168                     var matrixH4, matrixH5, matrixH6, dataH4, dataH5, dataH6,
169                     patternH4, patternH5, patternH6Fund, patternH6Harms;
170                     matrixH4 = "matricize.value(["tileMap[4]], 4, 0, "transform);
171                     matrixH5 = "matricize.value(["tileMap[5]], 5, 0, "transform);
172                     matrixH6 = "matricize.value(["tileMap[6]], 6, 0, "transform);
173                     dataH4 = "genData.value(matrixH4, "transform, 1, 0, 1, 0.5, 1);
174                     dataH5 = "genData.value(matrixH5, "transform, 1, 0, 4, 0.5, 1);
175                     dataH6 = "genData.value(matrixH6, "transform, 4, 0, 16);
176                     patternH4 = "genPattern.value(dataH4, "hl.4.low.noise", 16, true, cond); cond.hang;
177                     patternH5 = "genPattern.value(dataH5, "hl.5.high.noise", 4, true, cond); cond.hang;
178                     patternH6Fund = "genPattern.value([dataH6[0]], "hl.6.fundamental", 1, true, cond); cond.hang;
179                     patternH6Harms = "genPattern.value(dataH6[1..3], "hl.6.harmonics", 1, true, cond); cond.hang});
180
181                     // load buffers if the transform has changed
182                     if ("loadedTransform" != "transform, {var baseDir;
183                         baseDir = "dir +/" ++ "." ++ "audio" ++ "transform." ++ "transform.join ++ "audio";
184                         "buf1 = Buffer.read(s, baseDir) ++ "hl.6.fundamental.wav", action: {cond.unhang}); cond.hang;
185                         "buf2 = Buffer.read(s, baseDir) ++ "hl.6.harmonics.wav", action: {cond.unhang}); cond.hang;
186                         "buf3 = Buffer.read(s, baseDir) ++ "hl.5.high.noise.wav", action: {cond.unhang}); cond.hang;
187                         "buf4 = Buffer.read(s, baseDir) ++ "hl.4.low.noise.wav", action: {cond.unhang}); cond.hang);
188                         "loadedTransform = "transform.deepCopy;
189
190
191                     // play / stop functionality (create synth if it does not exist
192                     if ("play == nil, {play = Synth.new(\play, [\hash, hash, \playRate, 0, \startTrig, 0,
193                     \sineBuf1, "buf1, \sineBuf2, "buf2, \highNoiseBuf, "buf3, \lowNoiseBuf, "buf4]);
194                     pauseButton.value = 0;
195                     if (elem.value == 0, {
196                         clock.string = clockStringFunc.value((startPos * 129).trunc * 8);
197                         "play.set(\playRate, 0, \startTrig, 0,
198                         \sineBuf1, "buf1, \sineBuf2, "buf2, \highNoiseBuf, "buf3, \lowNoiseBuf, "buf4);
199                         clock.string = clockStringFunc.value((startPos * 129).trunc * 8);,{
200                         "play.set(\startPos, startPos, \playRate, 1, \startTrig, 1,
201                         \sineBuf1, "buf1, \sineBuf2, "buf2, \highNoiseBuf, "buf3, \lowNoiseBuf, "buf4)}}.inEnvir.defer}.inEnvir),
202
203
204                     // pause button
205                     pauseButton = Button().states(["pause", Color.black], ["pause", Color.black, Color.grey]).action({arg elem;
206                         if (elem.value == 1, {play.set(\playRate, 0);{play.set(\playRate, 1)}}.inEnvir),
207
208                     // start position slider
209                     StaticText().string(" start measure:beat"),[
210                         posSlider = Slider(bounds: Rect(0, 0, 30, 5)).action({arg pos; var min, sec;
211                         startPosText.string = clockStringFunc.value((pos.value * 129).trunc * 8);
212                         startPos = ((pos.value * 129).trunc * 8) / (129 * 8);
213                         clock.string = clockStringFunc.value((startPos * 129).trunc * 8).inEnvir, stretch: 1],
214                         startPosText = StaticText().string(" 1:1").font(Font("Monaco", 15)), nil);
215
216     HLayout([VLayout(HLayout(
217         clock = StaticText().string(" 1:1").font(Font("Monaco", 200)),
218         StaticText().string("(").font(Font("Monaco", 200)),
219         metronome = StaticText().string([-30, -105, -104].asAscii).font(Font("Monaco", 300)).stringColor(Color.red), nil, transport),
220         alignment: \top));
221
222     // mixer
223     mixerView = {arg hash; var masterIndicators, trackIndicators, master, tracks,
224     eVol = [0.8, 0.8, 0.8, 0.8], eMute = [1, 1, 1, 1], ePan = [0, 0, 0, 0],
225     eNames = ["hl 6 (fundamental)", "hl 6 (harmonics)", "hl 5 (high noise)", "hl 4 (low noise)"];
226
227     // init indicators
228     masterIndicators = {LevelIndicator() ! 2;
229     trackIndicators = {LevelIndicator() ! 4;
230
231     // get amp values for indicators
232     OSCFunc.new({arg msg; {
233         if (msg[2] == hash, {{arg i;
234             masterIndicators[i].value = msg[3 + i].ampdb.linlin(-60, 0, 0, 1) ! 2}}.defer},
235         'masterLevel', s.addr);
236     OSCFunc.new({arg msg; {
237         if (msg[2] == hash, {{arg i;
238             trackIndicators[i].value = msg[3 + i].ampdb.linlin(-60, 0, 0, 1) ! 4}}.defer},
239         'trackLevels', s.addr);
240
241     // master faders
242     master = HLayout(
243         VLayout([HLayout(
244             Slider().value(0.8).action({arg elem;
245                 "play.set(\masterVol, elem.value * 1.25).inEnvir,

```



```

245     masterIndicators[0], masterIndicators[1]), stretch: 2],
246     Button().states([["mute", Color.black], ["mute", Color.black, Color.grey]]).action({arg elem;
247         ^play.set(^masterMute, (1 - elem.value).abs)}.inEnvir),
248     StaticText().string("        master        ").align(\center)), nil);
249 // track faders
250 tracks = {arg part;
251     HLayout(VLayout(HLayout(
252         Slider().value(0.8).action({arg elem;
253             eVol[part] = elem.value * 1.25; ^play.set("eVol", eVol)}.inEnvir),
254         trackIndicators[part]),
255         Button().states([["mute", Color.black], ["mute", Color.black, Color.grey]]).action({arg elem;
256             eMute[part] = (1 - elem.value).abs; ^play.set("eMute", eMute)}.inEnvir),
257         StaticText().string("pan").align(\center),
258         Knob().value(0.5).action({arg elem;
259             ePan[part] = elem.value * 2 - 1; ^play.set("ePan", ePan)}.inEnvir),
260         StaticText().string(eNames[part]).align(\center)), nil)} ! 4;
261     HLayout(master, nil, *tracks));
262
263 // all gui
264 guiLayout = {var tabButtonReset, modelViewButton, transportViewButton, mixerViewButton, hash, tabs;
265     tabButtonReset = {modelViewButton.value = 1; transportViewButton.value = 1; mixerViewButton.value = 1};
266     VLayout(HLayout(
267         modelViewButton = Button().states([["model", Color.white, Color.grey], ["model", Color.black]]).action({
268             {tabButtonReset.value; modelViewButton.value = 0; tabs.index = 0 }.value(0),
269             transportViewButton = Button().states([["transport", Color.white, Color.grey], ["transport", Color.black]]).action({
270                 {tabButtonReset.value; transportViewButton.value = 0; tabs.index = 1 }.value(1),
271                 mixerViewButton = Button().states([["mixer", Color.white, Color.grey], ["mixer", Color.black]]).action({
272                     {tabButtonReset.value; mixerViewButton.value = 0; tabs.index = 2 }.value(1)),
273                 hash = 10000000.rand;
274                 tabs = StackLayout();
275                 tabs.add(Window.new.layout.(mixerView.value(hash)));
276                 tabs.insert(Window.new.layout.(transportView.value(hash)));
277                 tabs.insert(Window.new.layout.(modelView.value(hash)));
278                 tabs.index = 0});
279
280 // launch the gui
281 ^launchGui = {var guiWin;
282     guiWin = Window.new("a lot of tiles (trivial scan) - gui", Rect(200, 25, 1000, 450)).front;
283     guiWin.layout = guiLayout.value;
284     guiWin.front}
285 )

```

```

1 \version "2.19.81"
2
3 % systems-per-page, unfold vars, and strings delimited by tildes
4 % are changed by the Supercollider transcriber
5
6 \paper {
7   # (set-paper-size "a4" 'portrait)
8   top-margin = 1 \cm
9   bottom-margin = 1 \cm
10  left-margin = 2.5 \cm
11
12  top-system-spacing =
13  #' ((basic-distance . 25 )
14    (minimum-distance . 25 )
15    (padding . 0 )
16    (stretchability . 0))
17
18  last-bottom-spacing =
19  #' ((basic-distance . 15 )
20    (minimum-distance . 15 )
21    (padding . 0 )
22    (stretchability . 0))
23
24  % manually change systems-per-page to 2 if the harmonics of h1 6 are uncommented below
25  systems-per-page = 4
26
27  print-page-number = ##t
28  oddHeaderMarkup = \markup { \on-the-fly #not-first-page "("score.type" - "transform") }
29  evenHeaderMarkup = \markup { \on-the-fly #not-first-page "("score.type" - "transform") }
30  oddFooterMarkup = \markup { \fill-line {
31    \on-the-fly #not-first-page
32    \concat {
33      "-"
34      \fontsize #1.5
35      \on-the-fly #print-page-number-check-first
36      \fromproperty #'page:page-number-string
37      "-"}}}
38  evenFooterMarkup = \markup { \fill-line {
39    \on-the-fly #not-first-page
40    \concat {
41      "-"
42      \fontsize #1.5
43      \on-the-fly #print-page-number-check-first
44      \fromproperty #'page:page-number-string
45      "-"}}}
46 }
47
48 \header {
49   title = \markup { \italic {a lot of tiles (trivial scan)}}
50   subtitle = \markup { \normal-text {"transform"}}
51   composer = \markup { \left-column {"michael winter" "(cdmx, mx and nashville, usa; 2018)"} }
52   piece = \markup { \normal-text {"score.type" - "transform"}}
53   tagline = ""
54 }
55
56 # (set-global-staff-size 15)
57
58 \layout {
59   indent = 0.0 \cm
60   line-width = 17 \cm
61   ragged-last = ##t
62
63   \context {
64     \Score
65     \override BarNumber.extra-offset = #' (0 . 5)
66     \override BarNumber.stencil =
67     # (make-stencil-circler 0.1 0.25 ly:text-interface::print)
68   }
69   \context {
70     \Staff
71     \override StaffSymbol.line-count = #1
72     \override Clef.stencil = #point-stencil
73     \override Clef.space-alist.first-note = #' (extra-space . 1)
74     \remove "Time-signature-engraver"
75   }
76   \context {
77     \Voice
78     \override Glissando.minimum-length = #0
79     \override Glissando.layer = 500
80     \override Glissando.breakable = ##t
81     \override Glissando.bound-details =
82     #' ((right
83       (attach-dir . -1)
84       (end-on-accidental . #f)
85       (padding . 0))
86       (right-broken
87       (padding . 0.5))
88       (left-broken
89       (padding . 0.5))
90       (attach-dir . 1))
91     (left
92       (attach-dir . -1)
93       (padding . -0.25)
94       (start-at-dot . #f)))
95   }
96 }
97
98 % this draws a curve and a gradient
99 # (define (make-grey-filled-curve-stencil-list x-coords colors half-thick rl)
100 (if (null? (cdr x-coords))
101   rl
102   (make-grey-filled-curve-stencil-list
103     (cdr x-coords)
104     (cdr colors)
105     half-thick
106     (cons
107       (stencil-with-color
108         (make-filled-box-stencil
109           (interval-widen (cons (car x-coords) (cdr x-coords)) 0.1)
110           (cons (- half-thick) (- 9 (* 10 (car colors))))))
111       (list (car colors) (car colors) (car colors)))
112     rl)))
113
114 % this draws a just a line with a gradient
115 # (define (make-grey-filled-box-stencil-list x-coords colors half-thick rl)
116 (if (null? (cdr x-coords))
117   rl
118   (make-grey-filled-box-stencil-list
119     (cdr x-coords)
120     (cdr colors)
121     half-thick

```

```

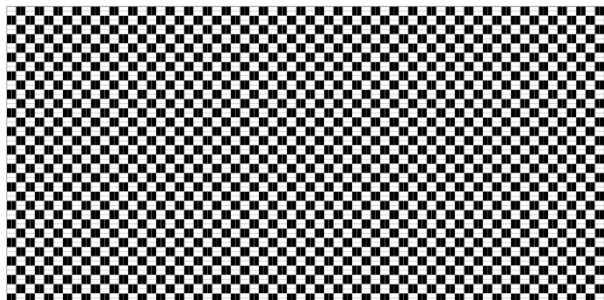
122 (cons
123   (stencil-with-color
124     (make-filled-box-stencil
125       (interval-widen (cons (car x-coords) (cadr x-coords)) 0.1)
126       (cons (- half-thick) half-thick))
127     (list (car colors) (car colors) (car colors)))
128   rl)))
129
130 #define my-gliss
131 (lambda (grob)
132   (if (ly:stencil? (ly:line-spanner::print grob))
133     (let* ((stencil (ly:line-spanner::print grob))
134            (X-ext (ly:stencil-extent stencil X))
135            (Y-ext (ly:stencil-extent stencil Y))
136            (Y-length (- (cdr Y-ext) (car Y-ext)))
137            (left-bound-info (ly:grob-property grob 'left-bound-info))
138            (left-Y (assoc-get 'Y left-bound-info))
139            (thick
140              (assoc-get 'thickness (ly:grob-property grob 'details) 0.5))
141            (layout (ly:grob-layout grob))
142            (blot (ly:output-def-lookup layout 'blot-diameter))
143            (right-bound (ly:spanner-bound grob RIGHT))
144            (right-par (ly:grob-parent right-bound X))
145            (stem
146              (if (grob::has-interface right-par 'note-column-interface)
147                  (ly:grob-object right-par 'stem)
148                  '()))
149            (stem-stencil
150              (if (ly:grob? stem)
151                  (ly:grob-property stem 'stencil)
152                  #f))
153            (stem-thick
154              (if (ly:stencil? stem-stencil)
155                  (interval-length (ly:stencil-extent stem-stencil X))
156                  0))
157            (corr-delta-X (- (interval-length X-ext)
158                             Y-length
159                             blot
160                             stem-thick
161                             ;; mmh, why this value??
162                             -0.05))
163            (colors
164              (assoc-get 'colors (ly:grob-property grob 'details)
165                          (list 0 .5 .8)))
166            (steps
167              (length colors))
168            (raw-x-coords
169              (iota (1+ (abs steps)) 0 (/ corr-delta-X (abs steps))))
170            (x-coords
171              (map
172                (lambda (e)
173                  (+ (car X-ext) Y-length blot e))
174                raw-x-coords)))
175
176    ;; create a flat glissando
177    (ly:grob-set-nested-property! grob '(right-bound-info Y) left-Y)
178
179    ;; return the stencil of added boxes
180    (ly:stencil-translate-axis
181      (apply
182        ly:stencil-add
183        ;; change this to make-grey-filled-box-stencil-list to eliminate curve and just use gradient
184        (make-grey-filled-curve-stencil-list
185          x-coords
186          colors
187          thick
188          '()))
189      ;; the actual offset is TODO, hardcoded here
190      3.5
191      Y))
192    #f)))
193
194 #define (add-gliss m)
195 (case (ly:music-property m 'name)
196   ((NoteEvent)
197    (set! (ly:music-property m 'articulations)
198          (append
199            (ly:music-property m 'articulations)
200            (list (make-music 'GlissandoEvent))))
201    m)
202   (else #f)))
203
204 addGliss =
205 #define-music-function (music)
206 (ly:music?)
207 (map-some-music add-gliss music))
208
209
210 \new Score
211 \with {proportionalNotationDuration = #(ly:make-moment 1 16)}
212 <<
213 \new StaffGroup \with{
214   \override StaffGrouper.staff-staff-spacing =
215   #'((basic-distance . 13)
216      (minimum-distance . 13)
217      (padding . 0)
218      (stretchability . 0))
219   \override StaffGrouper.staffgroup-staff-spacing =
220   #'((basic-distance . 13)
221      (minimum-distance . 13)
222      (padding . 0)
223      (stretchability . 0))
224 }
225 <<
226
227 % uncomment these lines to view harmonics of h1 6
228 % also change systems-per-page to 2 manually above
229 %{
230 \new Staff \with {
231   instrumentName = \markup \center-column {"h1 6 " " (harm 13) " }
232   shortInstrumentName = #"h16 (h13) "
233 }
234 <<
235 \repeat unfold 33 { \repeat unfold 3 { s1 \noBreak } s1 \break }
236 \include "includes/a.lot.of.tiles.trivial.scan.h1.6.harm.13.ly"
237 >>
238 \new Staff \with {
239   instrumentName = \markup \center-column {"h1 6 " " (harm 9) " }
240   shortInstrumentName = #"h16 (h9) "
241 }
242 <<
243 \repeat unfold 33 { \repeat unfold 3 { s1 \noBreak } s1 \break }
244 \include "includes/a.lot.of.tiles.trivial.scan.h1.6.harm.9.ly"

```

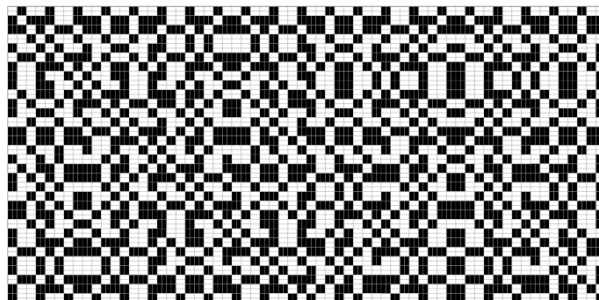
```

245 >>
246 \new Staff \with {
247   instrumentName = \markup \center-column {"h1 6 " "(harm 5)  "}
248   shortInstrumentName = #"h16 (h5)  "
249 }
250 <<
251 \repeat unfold 33 { \repeat unfold 3 { s1 \noBreak } s1 \break }
252 \include "includes/a.lot.of.tiles.trivial.scan.h1.6.harm.5.ly"
253 >>
254 %}
255
256 %--h16-start--
257 \new Staff \with {
258   instrumentName = \markup \center-column {"h1 6 " "(harm 1)  "}
259   shortInstrumentName = #"h16 (h1)  "
260 }
261 <<
262 \repeat unfold 33 { \repeat unfold 3 { s1 \noBreak } s1 \break }
263 \include "includes/a.lot.of.tiles.trivial.scan.h1.6.harm.1.ly"
264 >>
265 %--h16-end--
266 >>
267
268 \new StaffGroup \with{
269   \override StaffGrouper.staff-staff-spacing =
270   #'((basic-distance . 9)
271      (minimum-distance . 9)
272      (padding . 0)
273      (stretchability . 0))
274 }
275 <<
276 %--h15-start--
277 \new Staff \with {
278   instrumentName = \markup \center-column {"h1 5" "(high noise)" }
279   shortInstrumentName = #"h15 (hn)  "
280 }
281 <<
282 \repeat unfold 33 { \repeat unfold 3 { s1 \noBreak } s1 \break }
283 \include "includes/a.lot.of.tiles.trivial.scan.h1.5.ly"
284 >>
285 %--h15-end--
286
287 %--h14-start--
288 \new Staff \with {
289   instrumentName = \markup \center-column {"h1 4 " "(low noise)" }
290   shortInstrumentName = #"h14 (ln)  "
291 }
292 <<
293 \repeat unfold 33 { \repeat unfold 3 { s1 \noBreak } s1 \break }
294 \include "includes/a.lot.of.tiles.trivial.scan.h1.4.ly"
295 >>
296 %--h14-end--
297 >>
298 >>

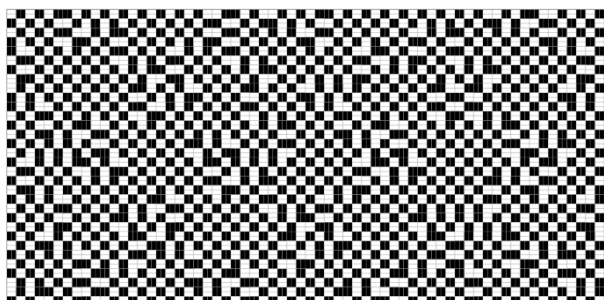
```



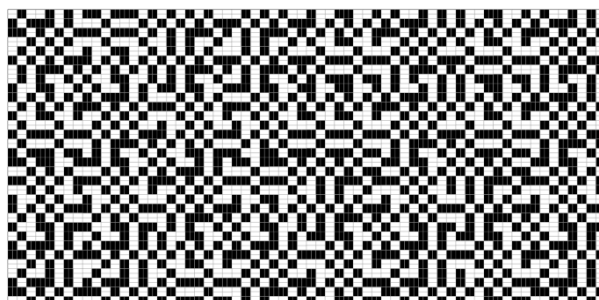
0000



0001



0002



0003



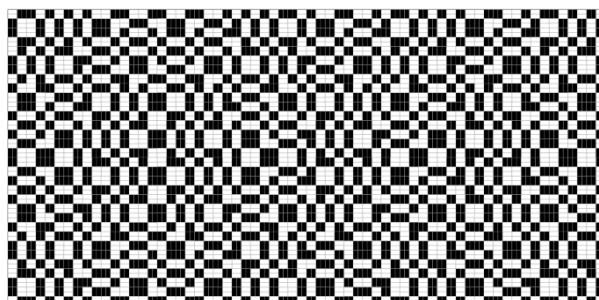
0010



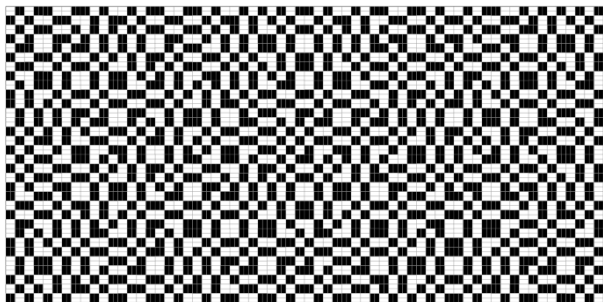
0011



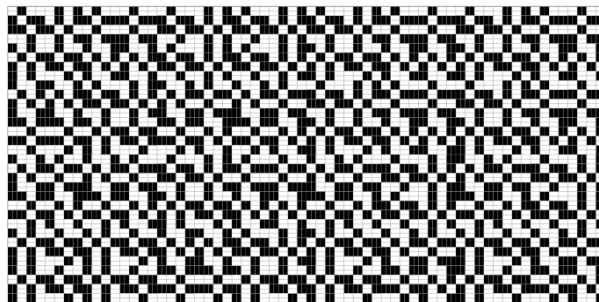
0012



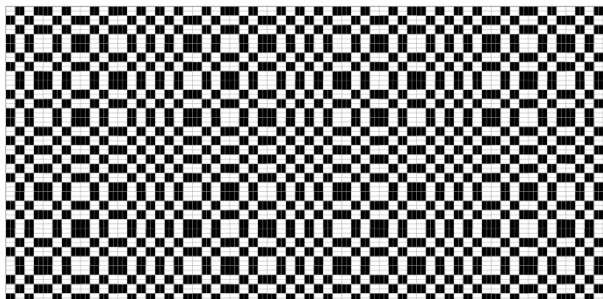
0013



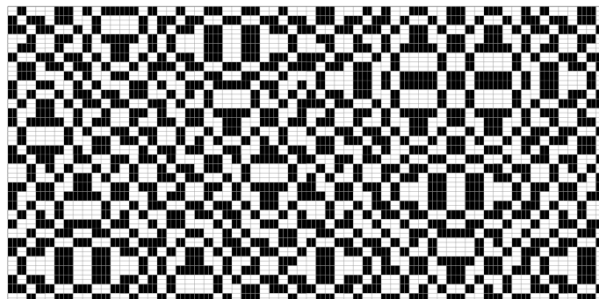
0020



0021



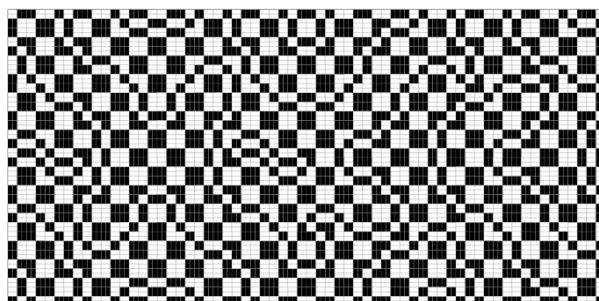
0022



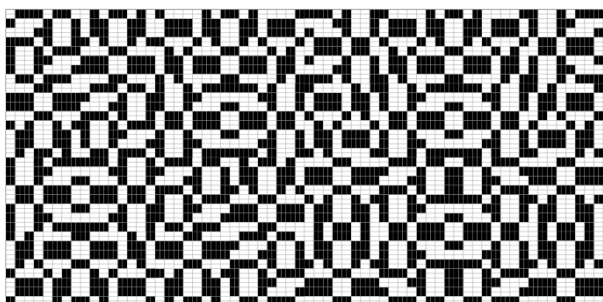
0023



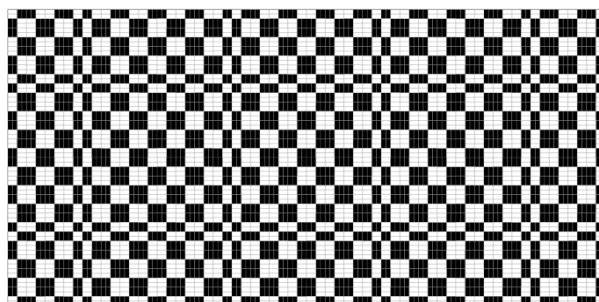
0030



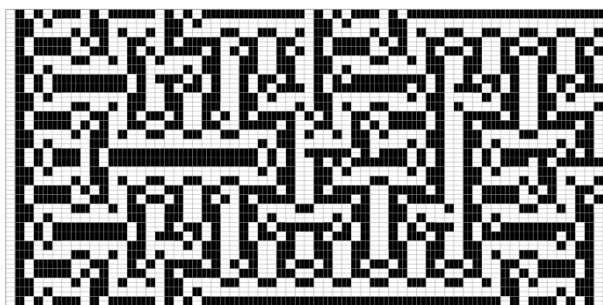
0031



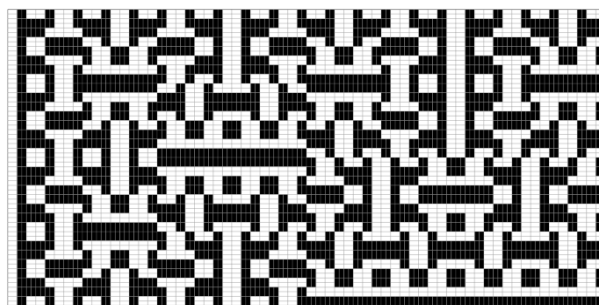
0032



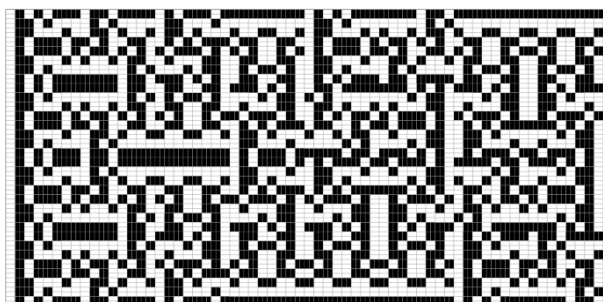
0033



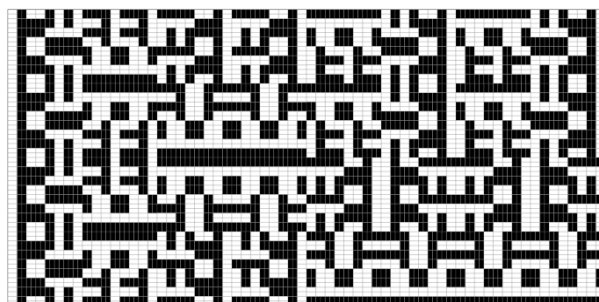
0100



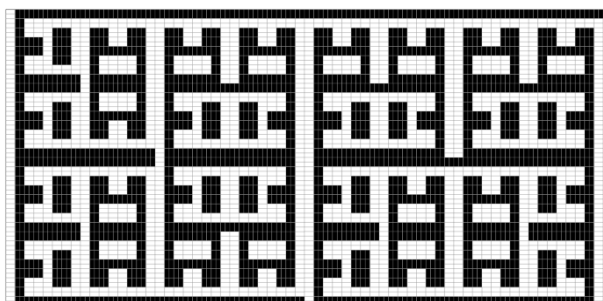
0101



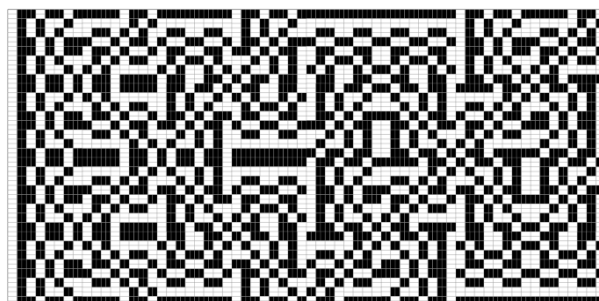
0102



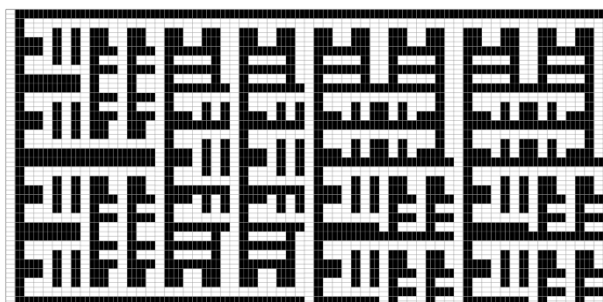
0103



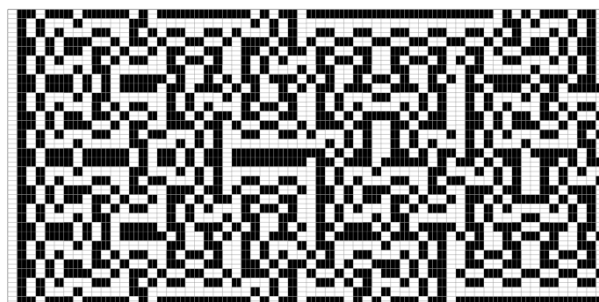
0110



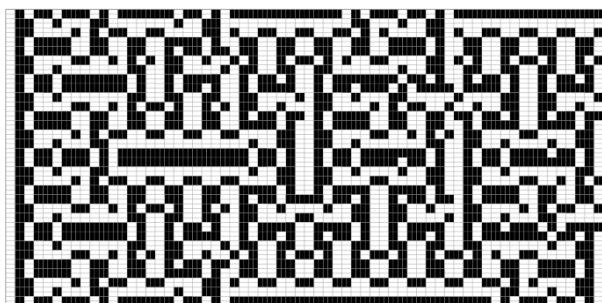
0111



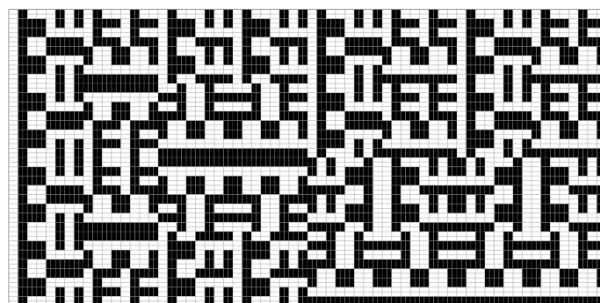
0112



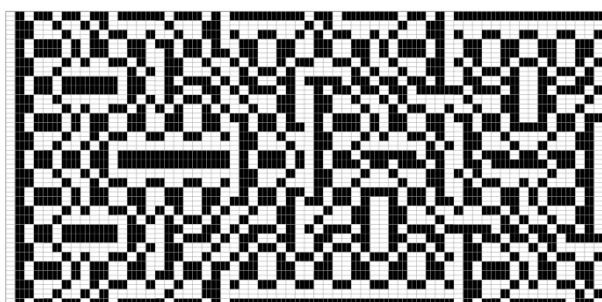
0113



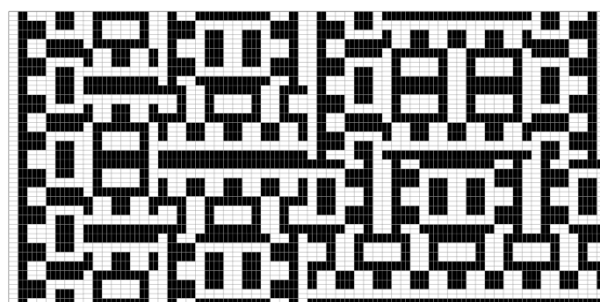
0120



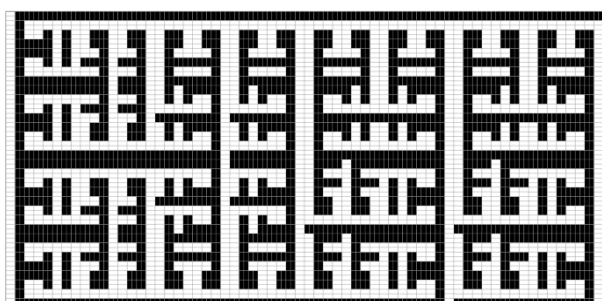
0121



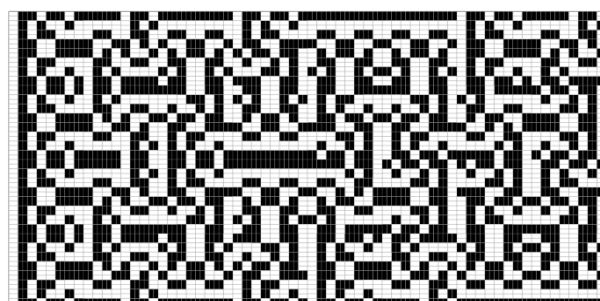
0122



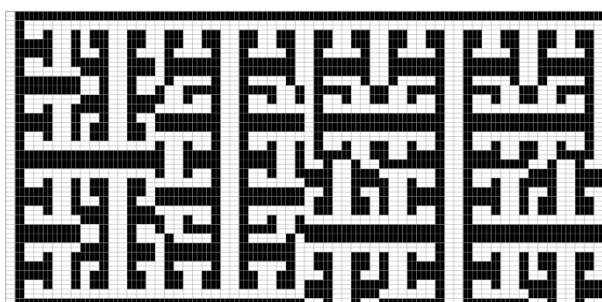
0123



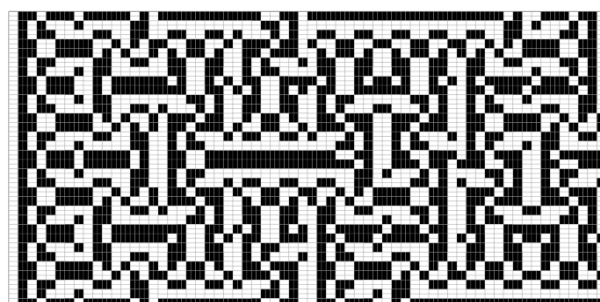
0130



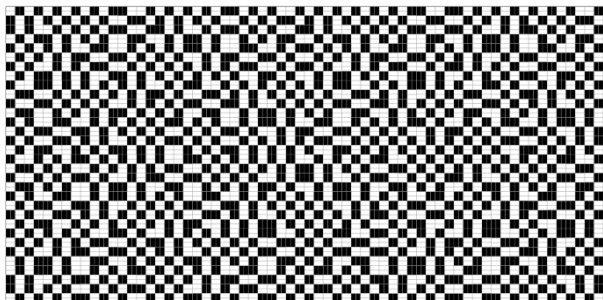
0131



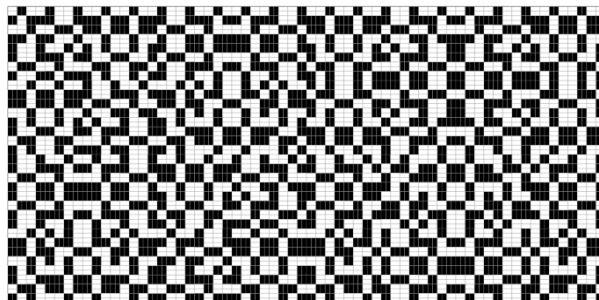
0132



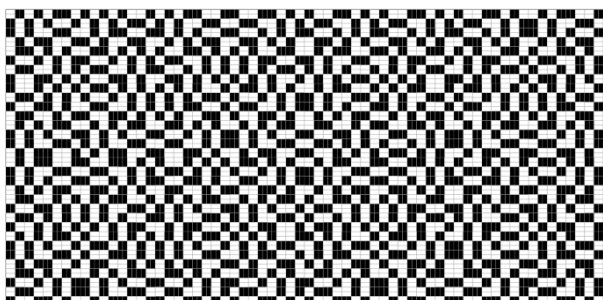
0133



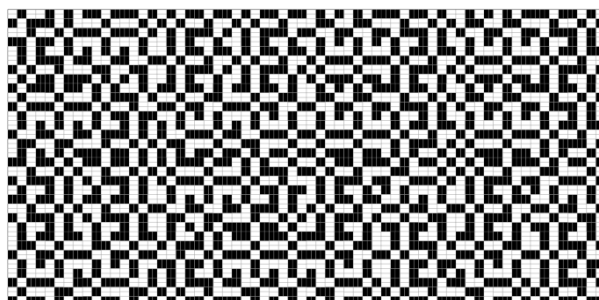
0200



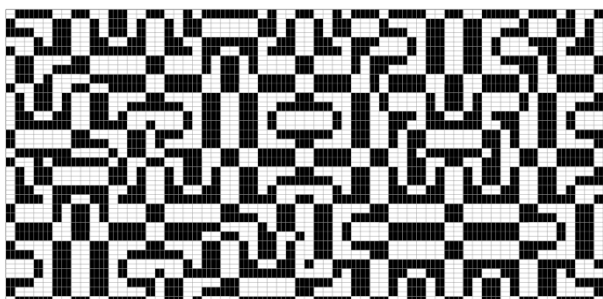
0201



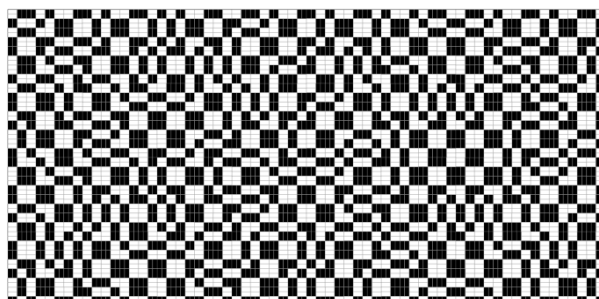
0202



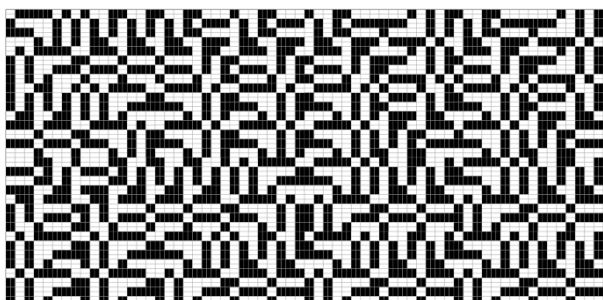
0203



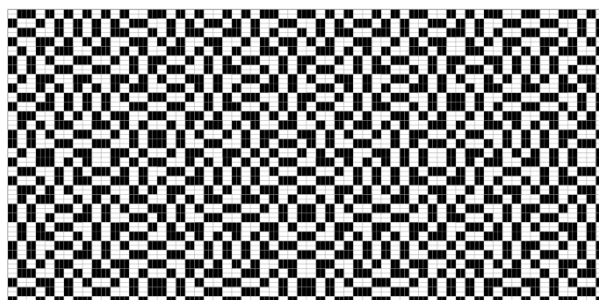
0210



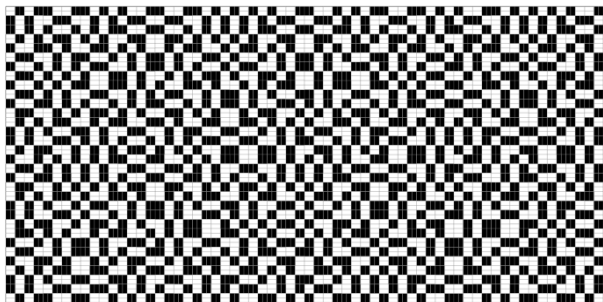
0211



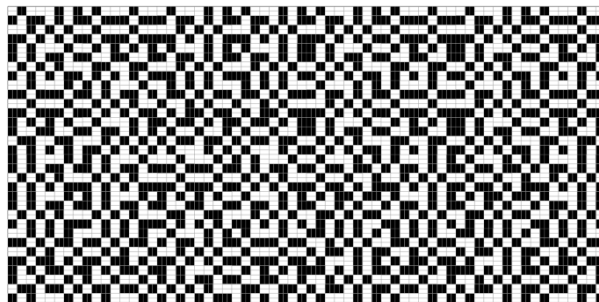
0212



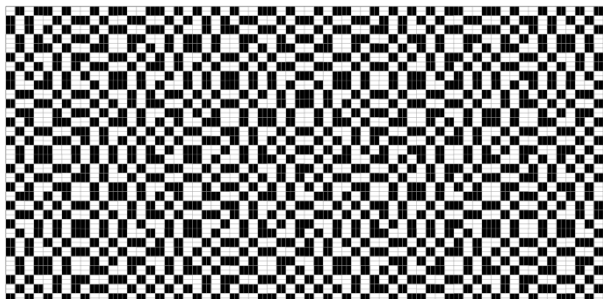
0213



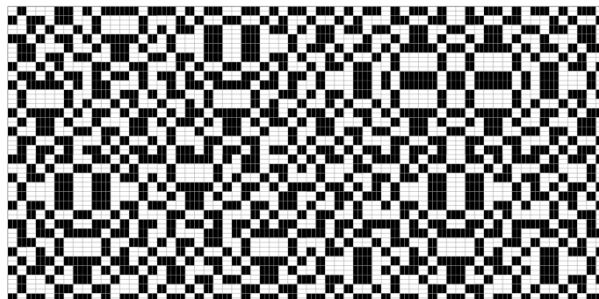
0220



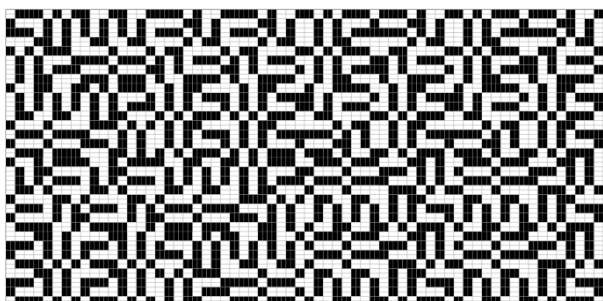
0221



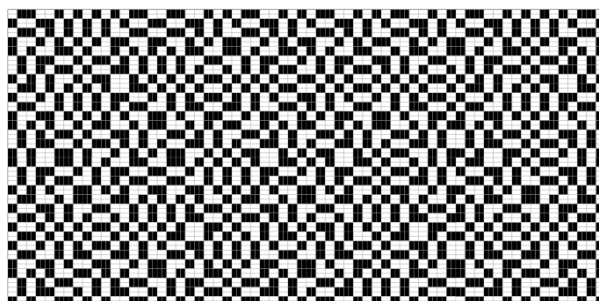
0222



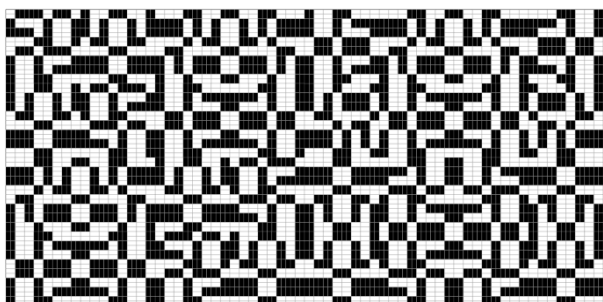
0223



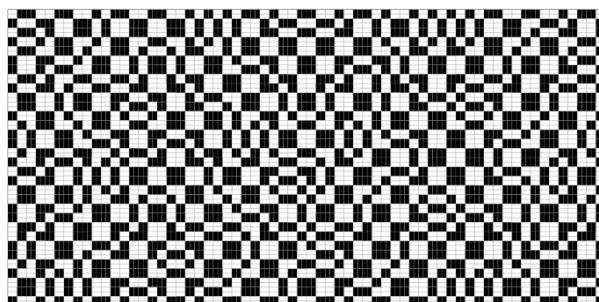
0230



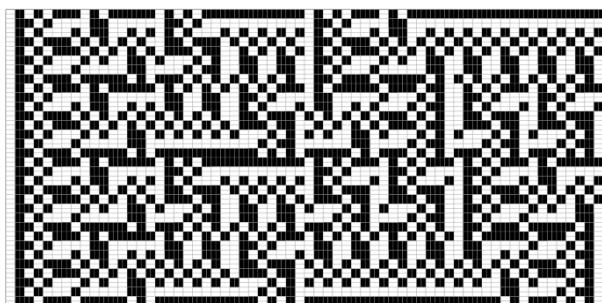
0231



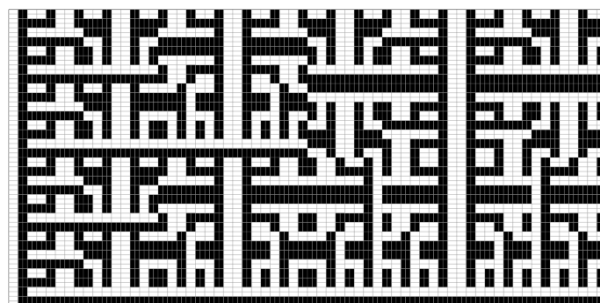
0232



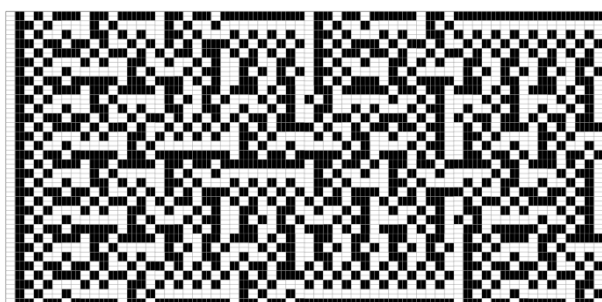
0233



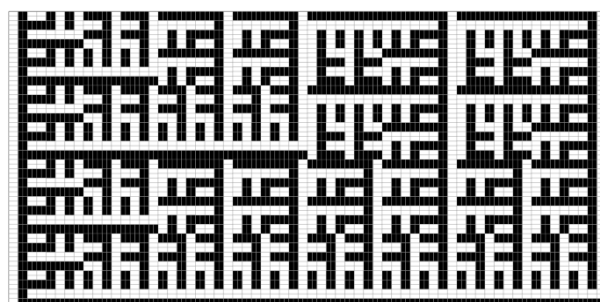
0300



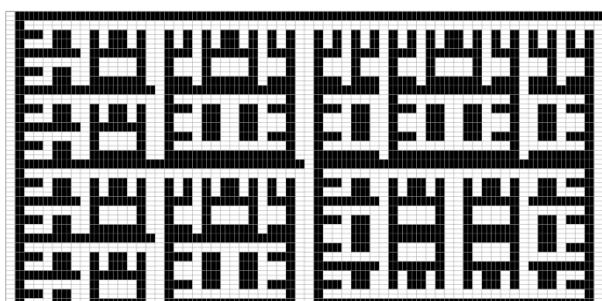
0301



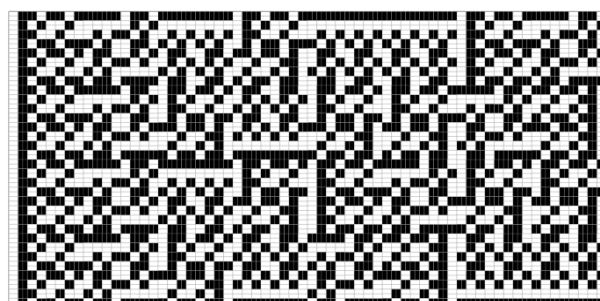
0302



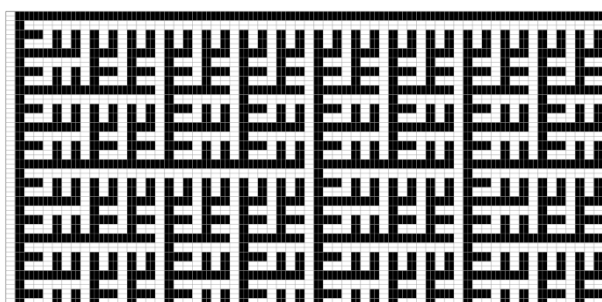
0303



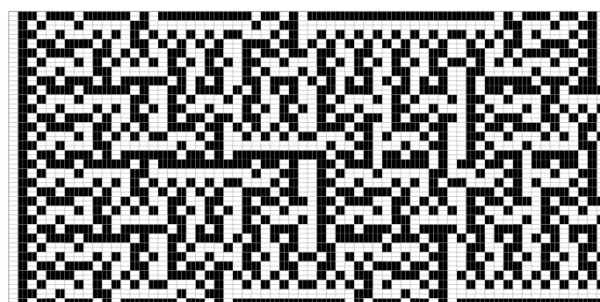
0310



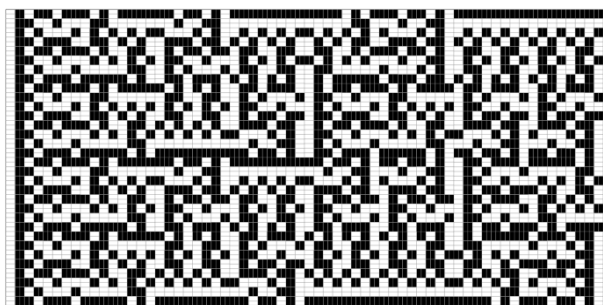
0311



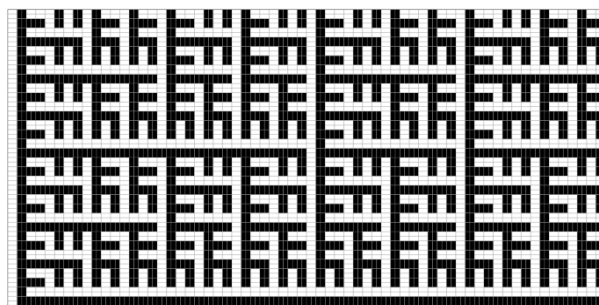
0312



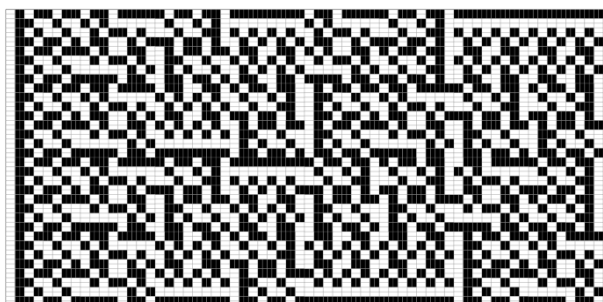
0313



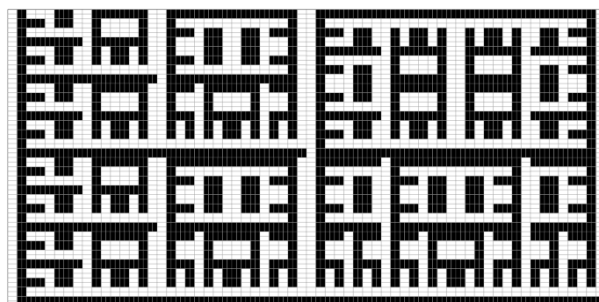
0320



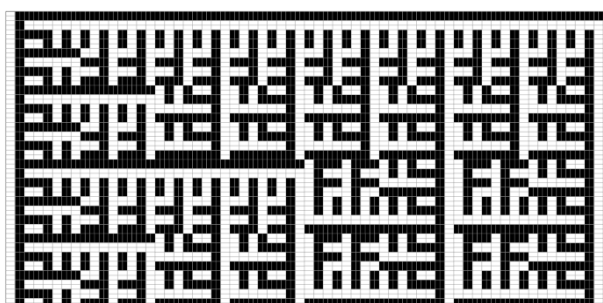
0321



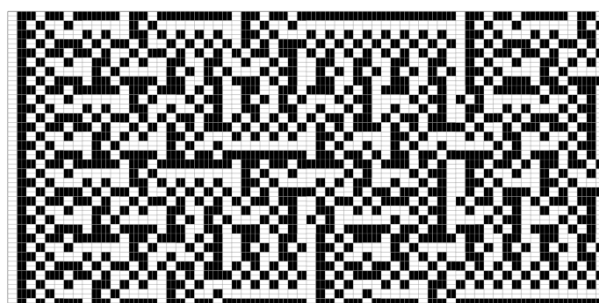
0322



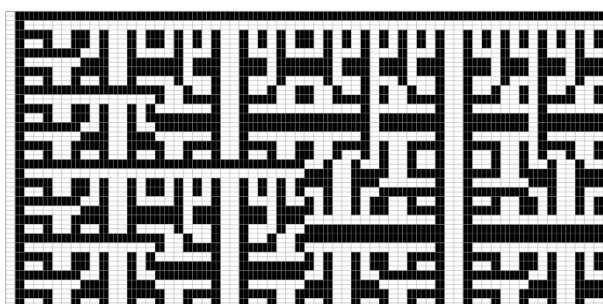
0323



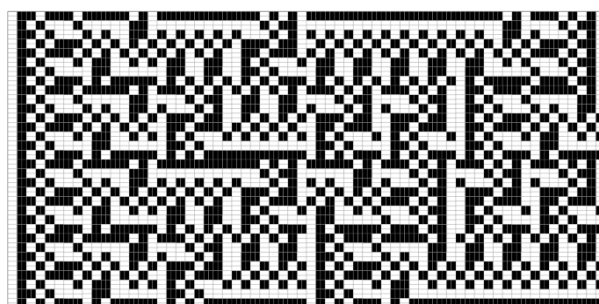
0330



0331



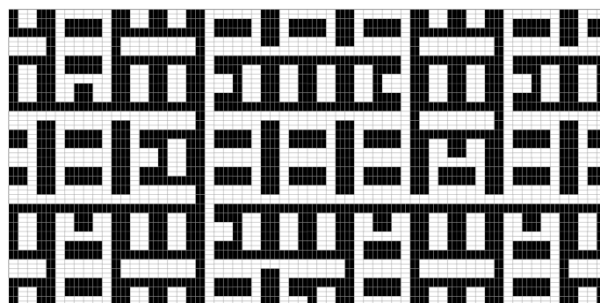
0332



0333



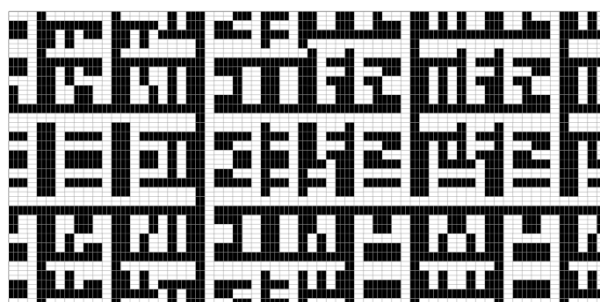
1000



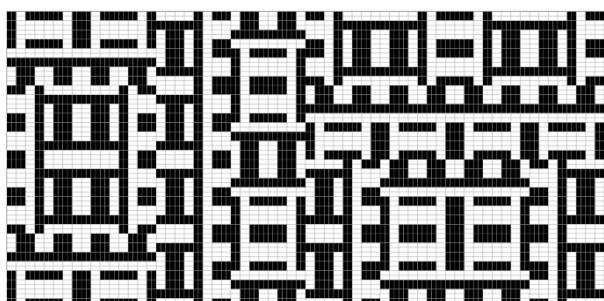
1001



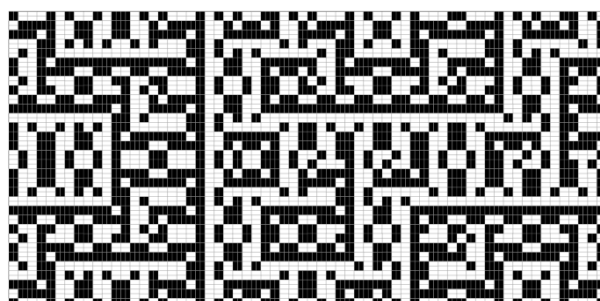
1002



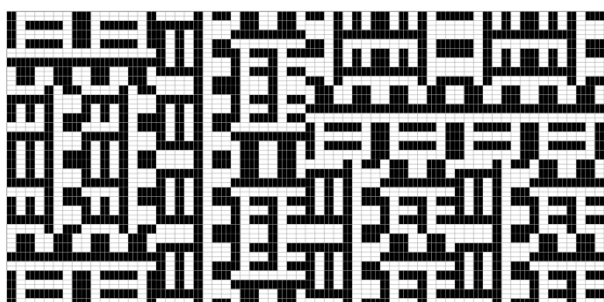
1003



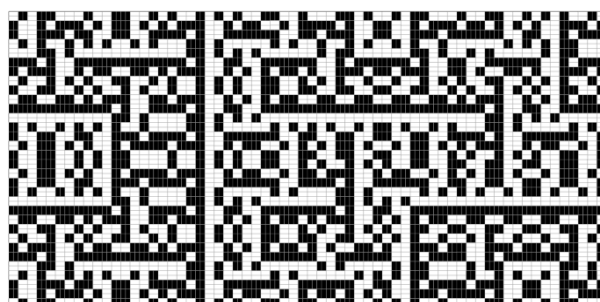
1010



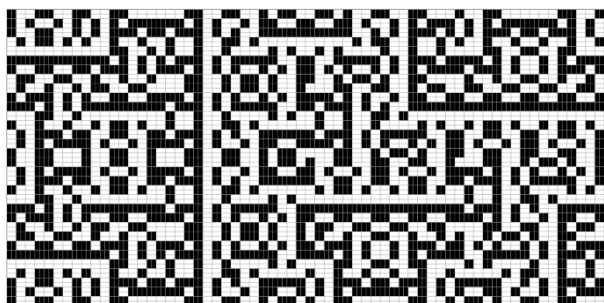
1011



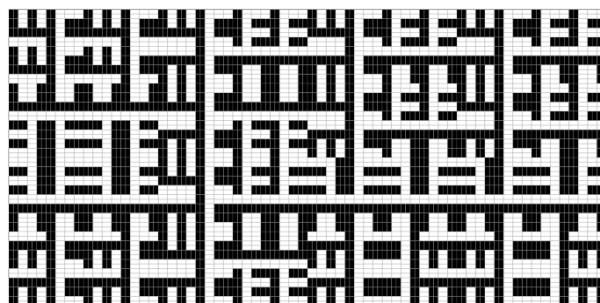
1012



1013



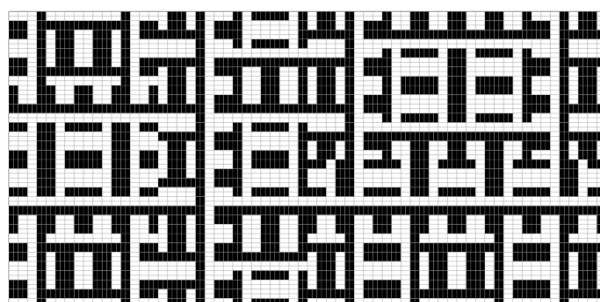
1020



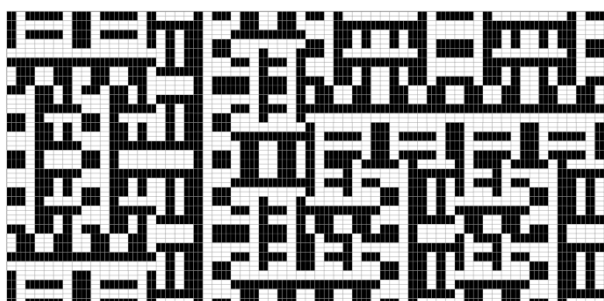
1021



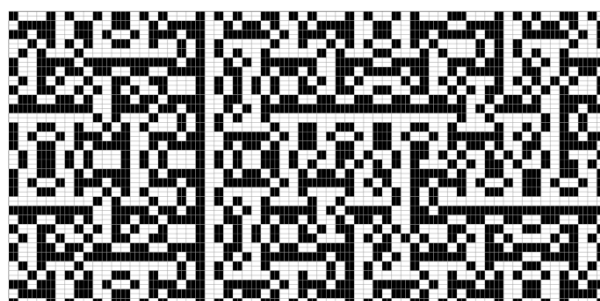
1022



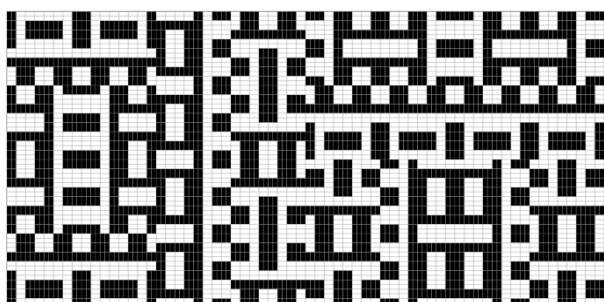
1023



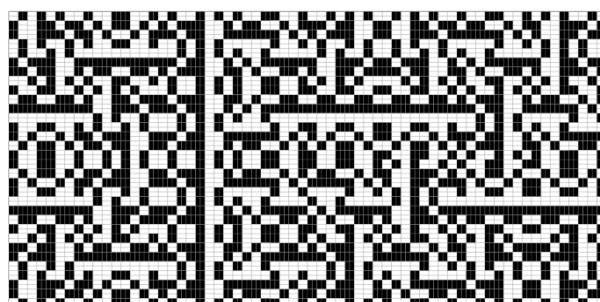
1030



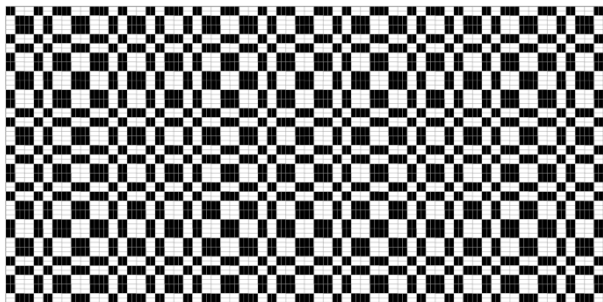
1031



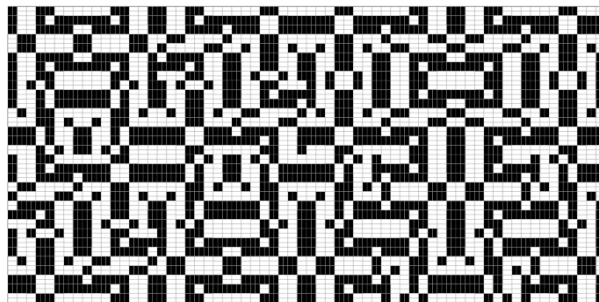
1032



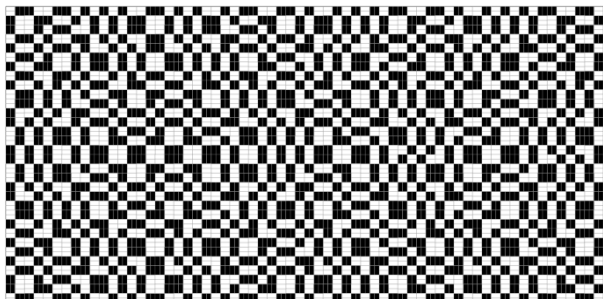
1033



1100



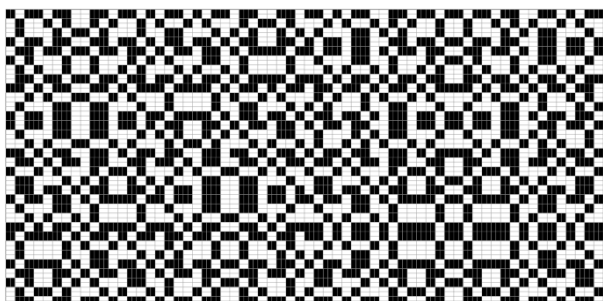
1101



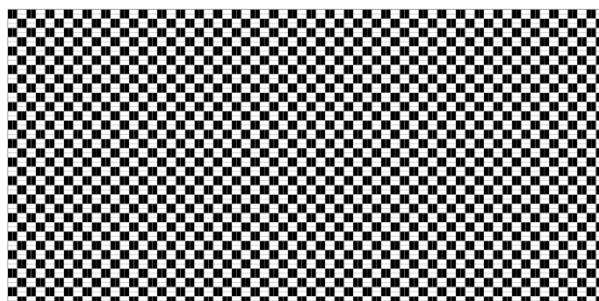
1102



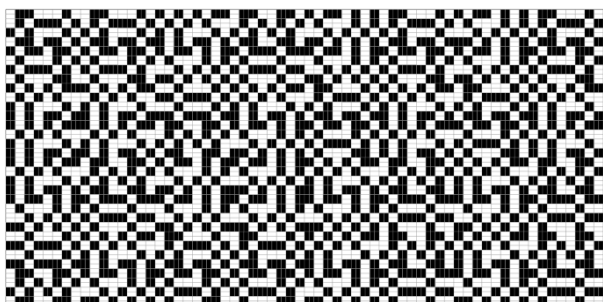
1103



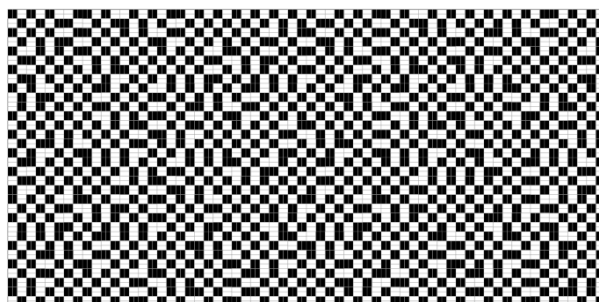
1110



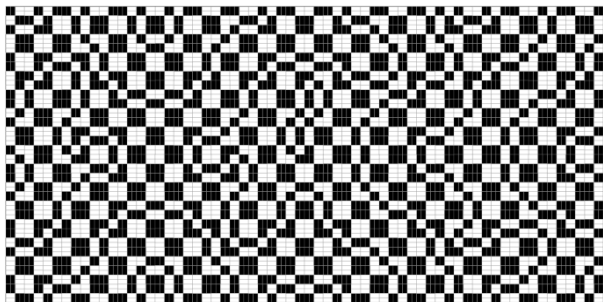
1111



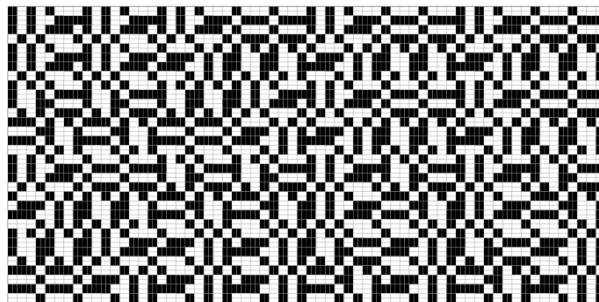
1112



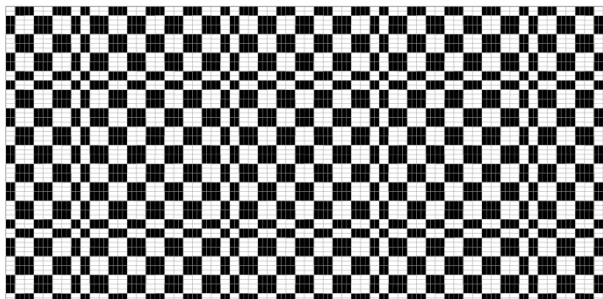
1113



1120



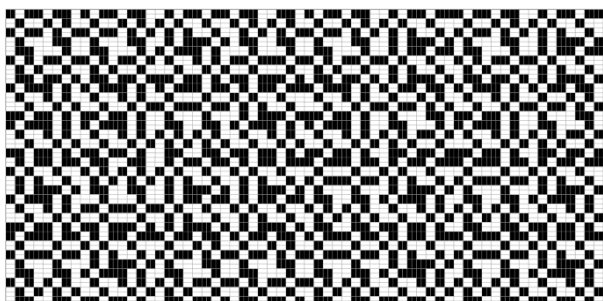
1121



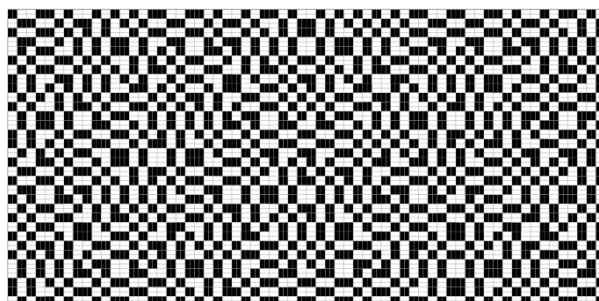
1122



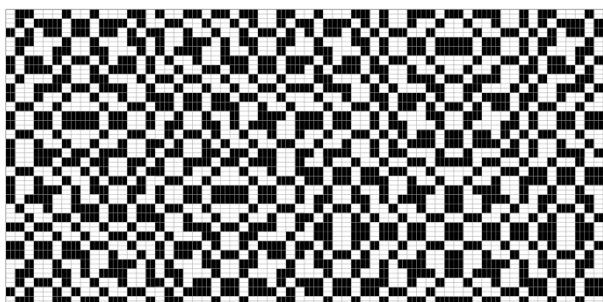
1123



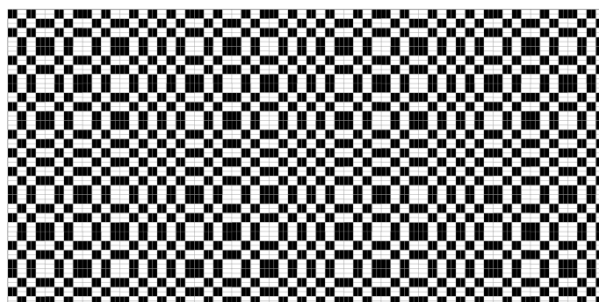
1130



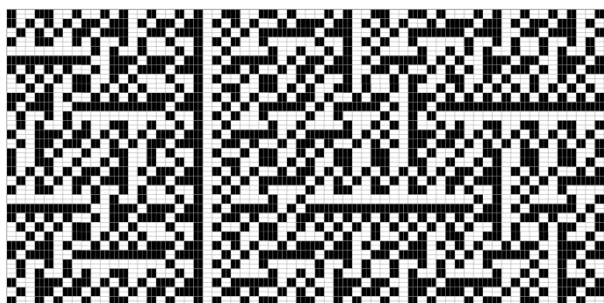
1131



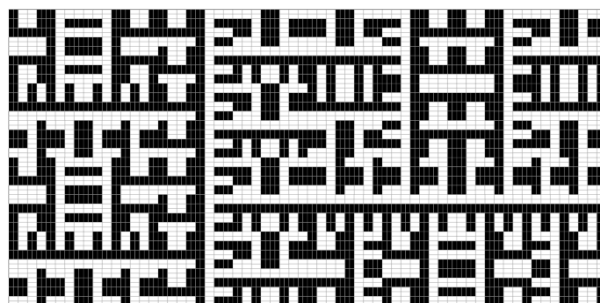
1132



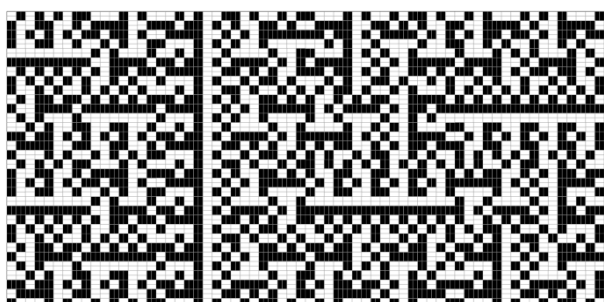
1133



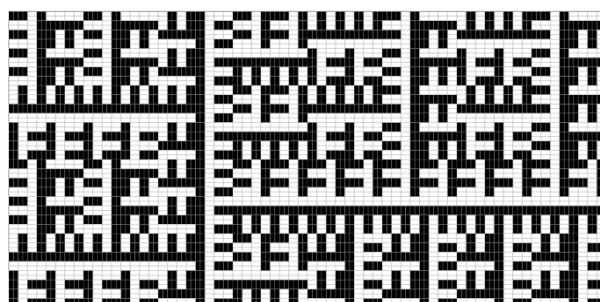
1200



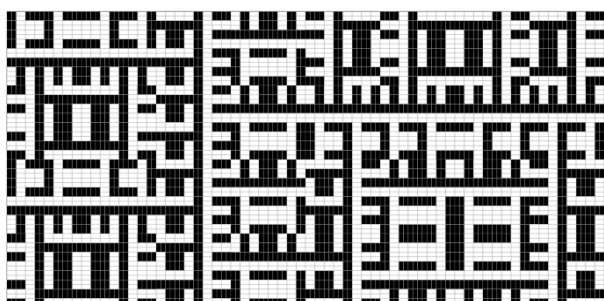
1201



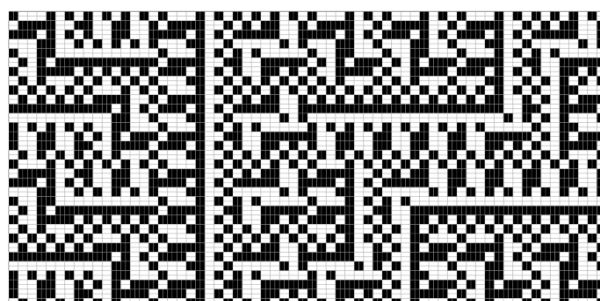
1202



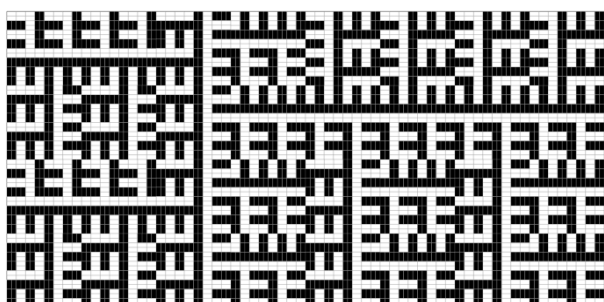
1203



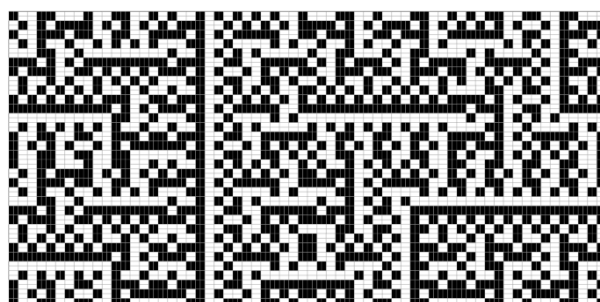
1210



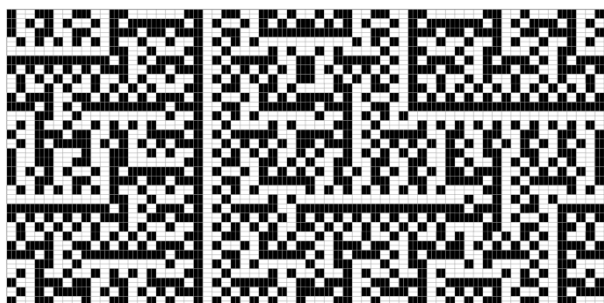
1211



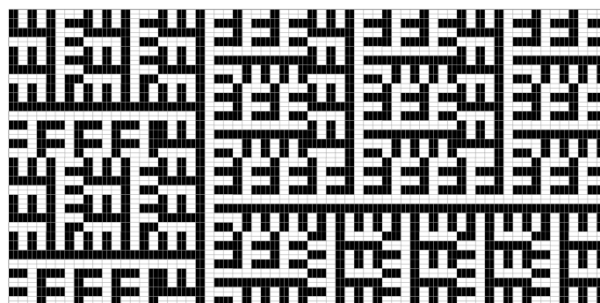
1212



1213



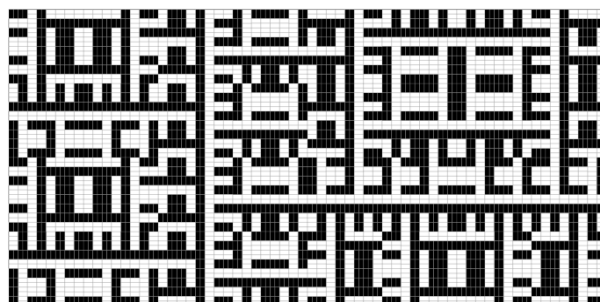
1220



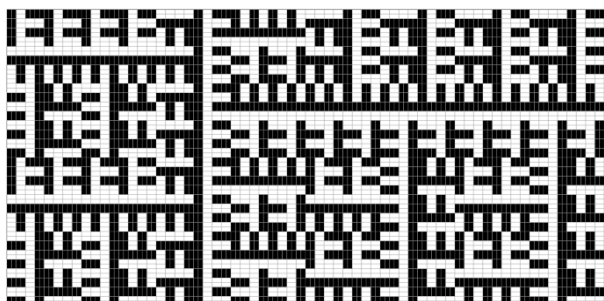
1221



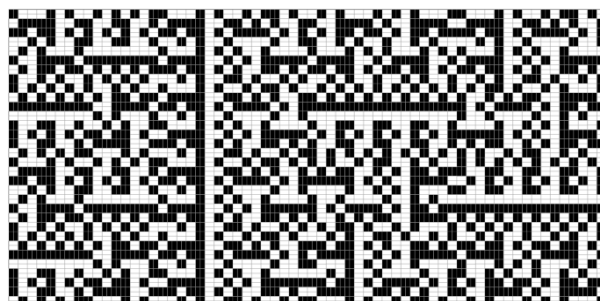
1222



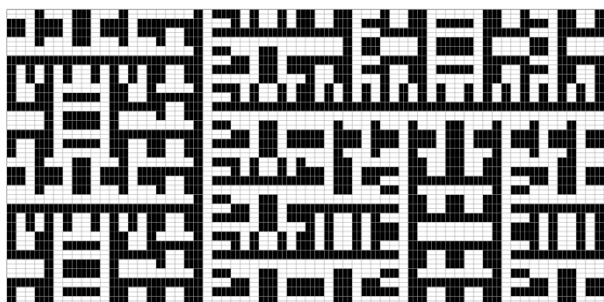
1223



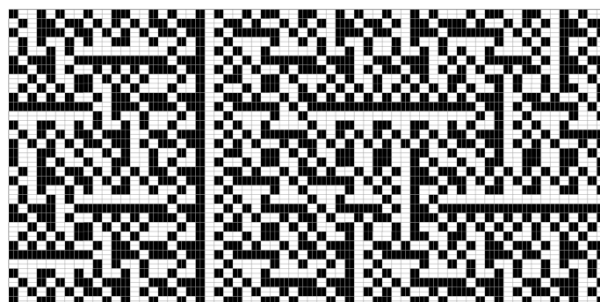
1230



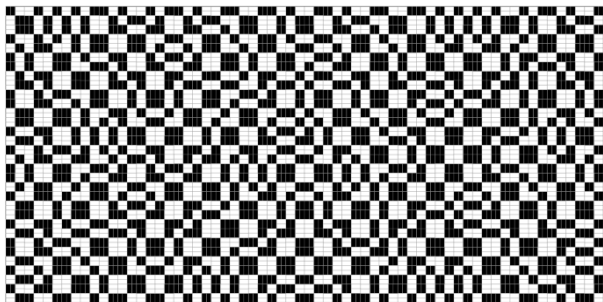
1231



1232



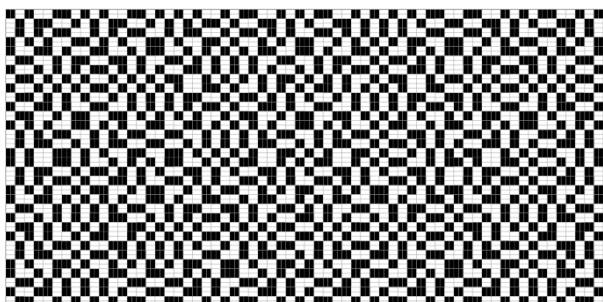
1233



1300



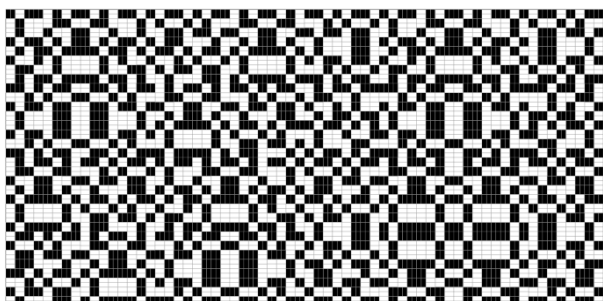
1301



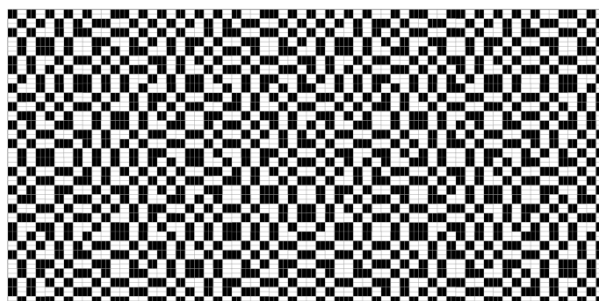
1302



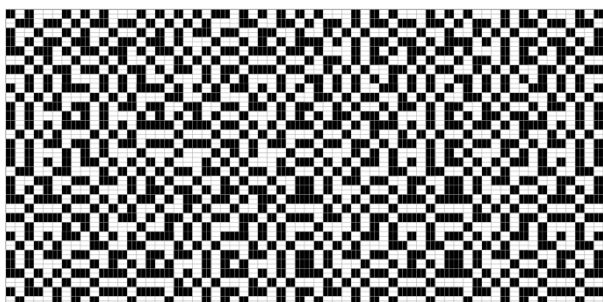
1303



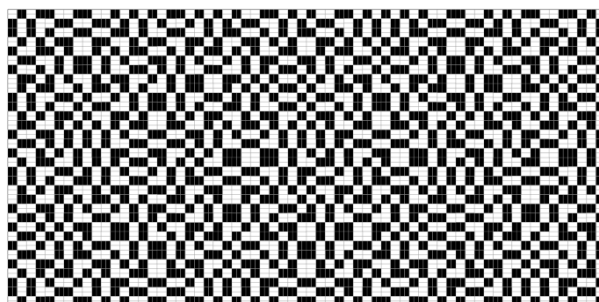
1310



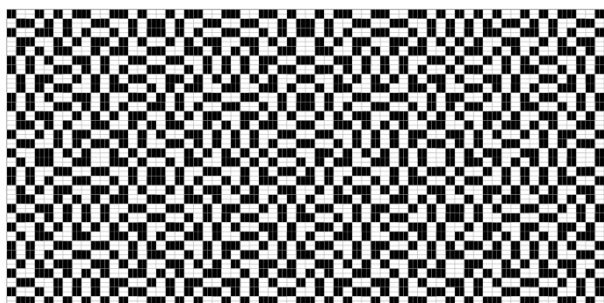
1311



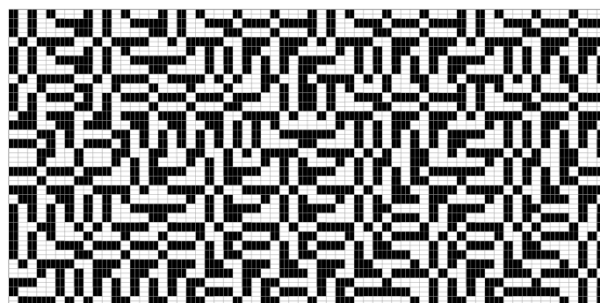
1312



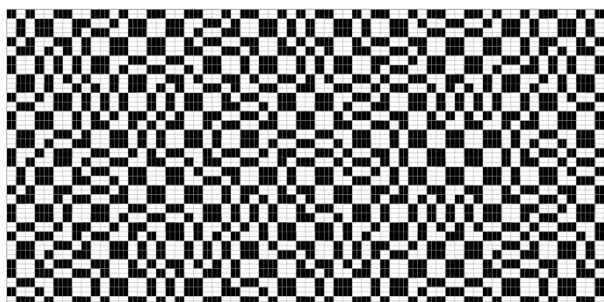
1313



1320



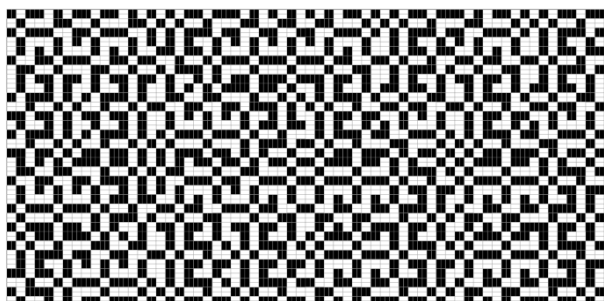
1321



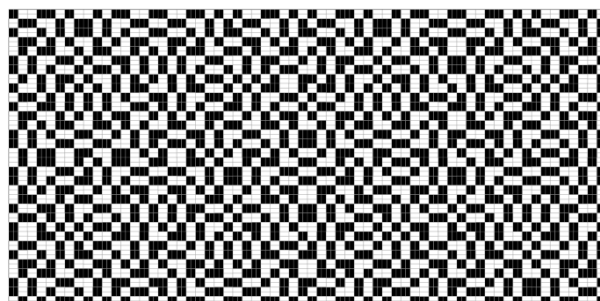
1322



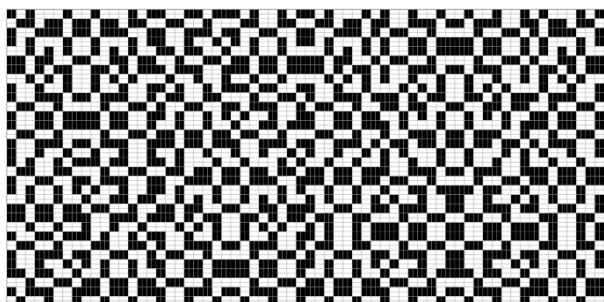
1323



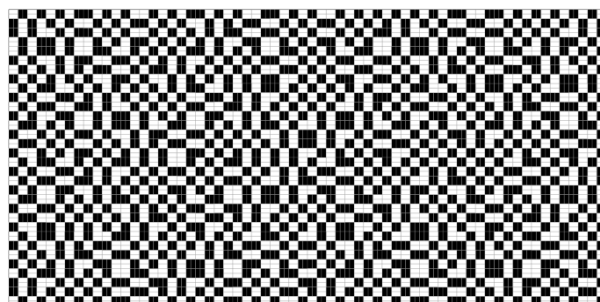
1330



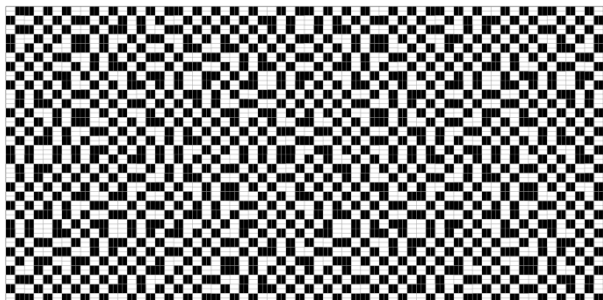
1331



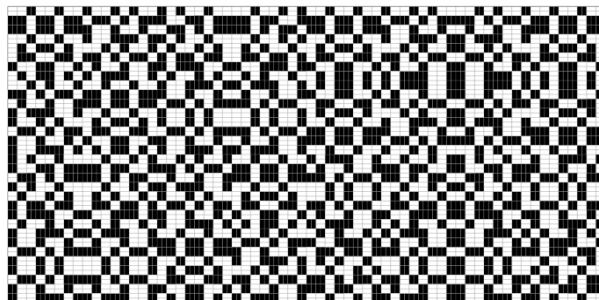
1332



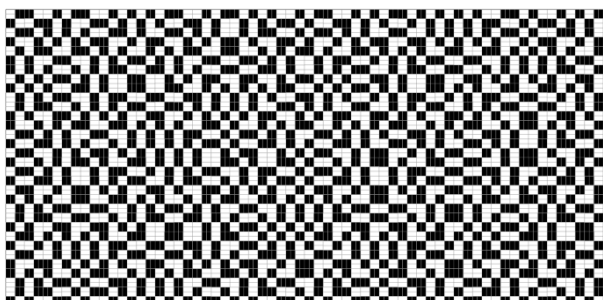
1333



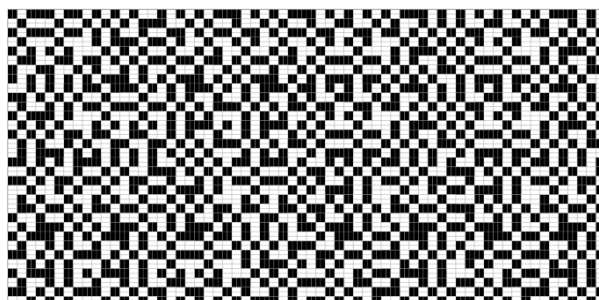
2000



2001



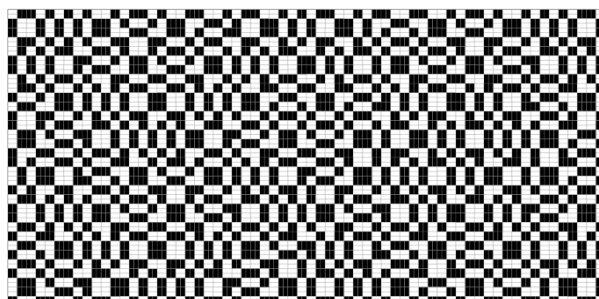
2002



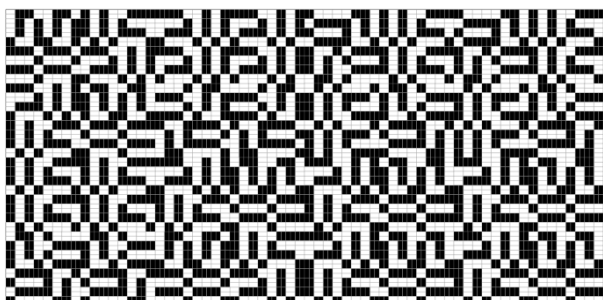
2003



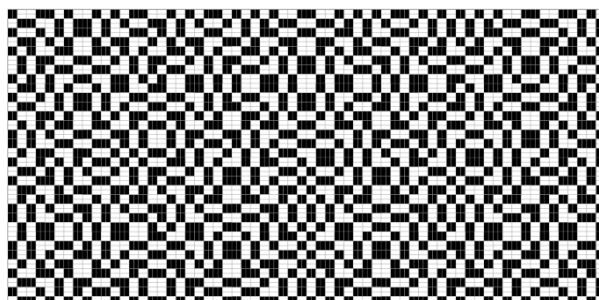
2010



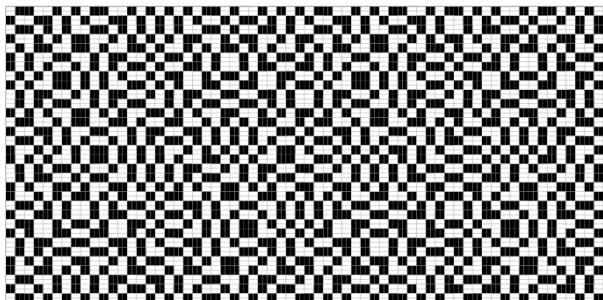
2011



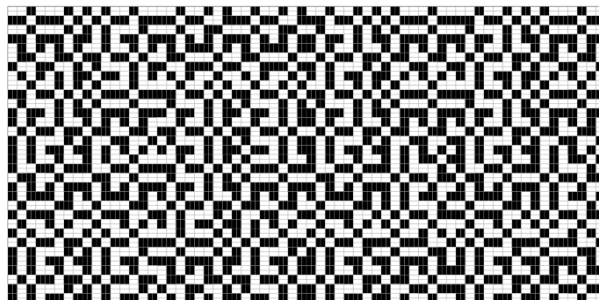
2012



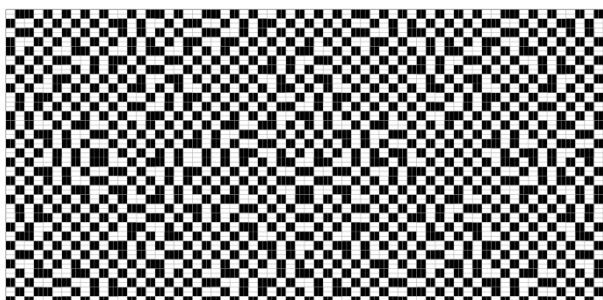
2013



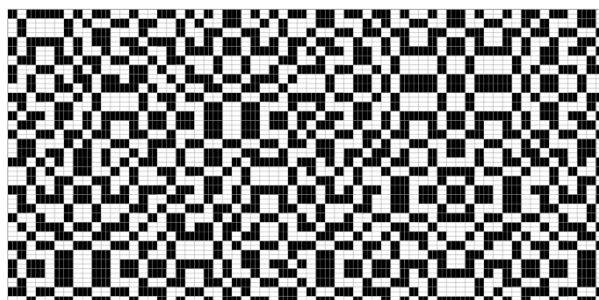
2020



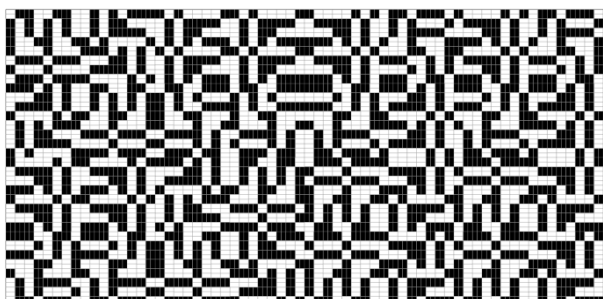
2021



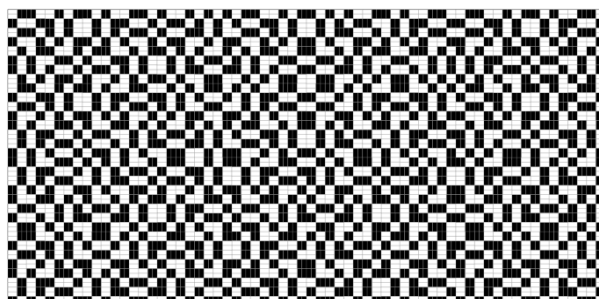
2022



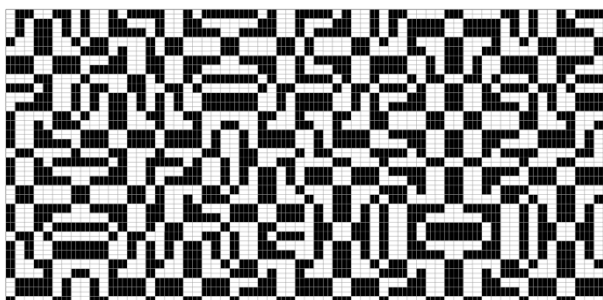
2023



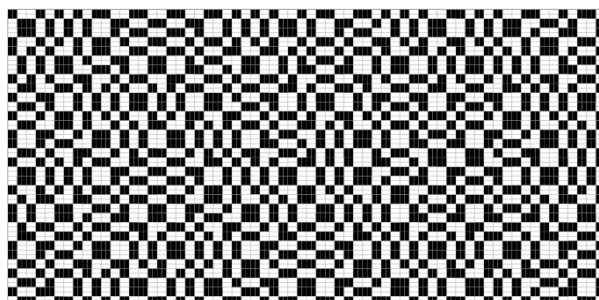
2030



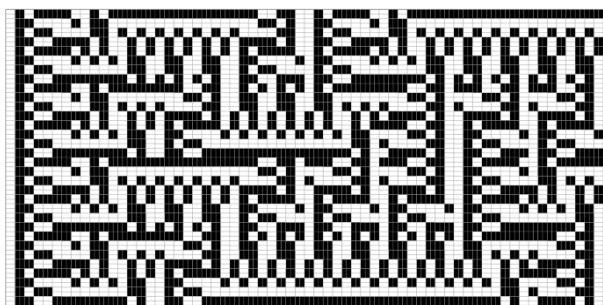
2031



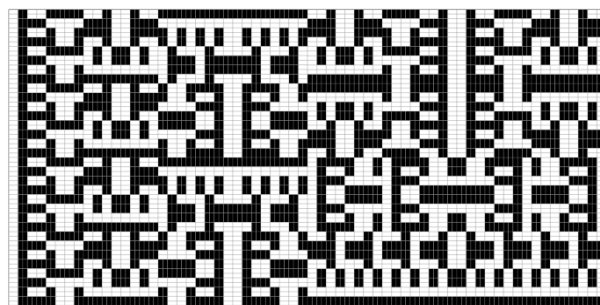
2032



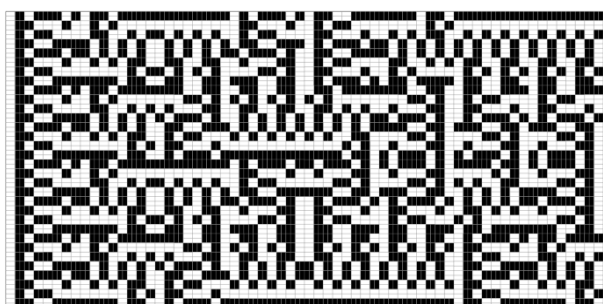
2033



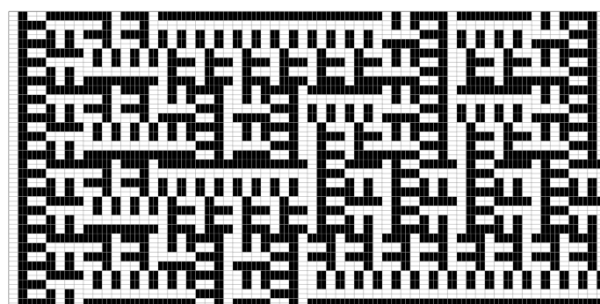
2100



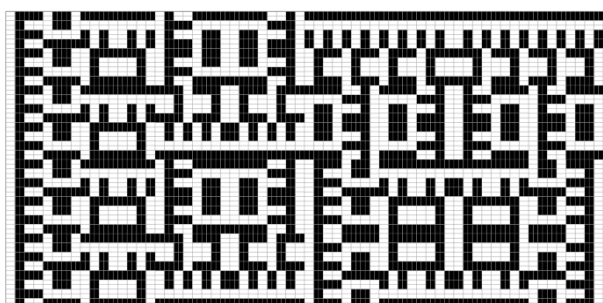
2101



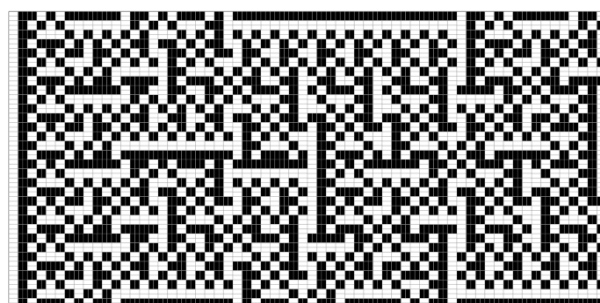
2102



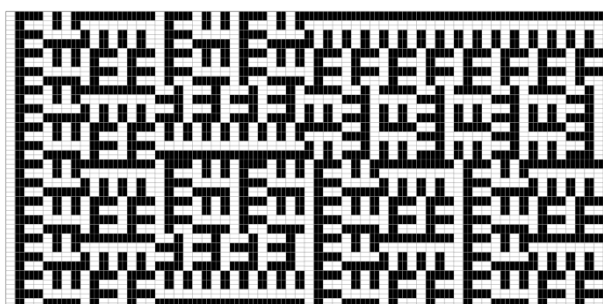
2103



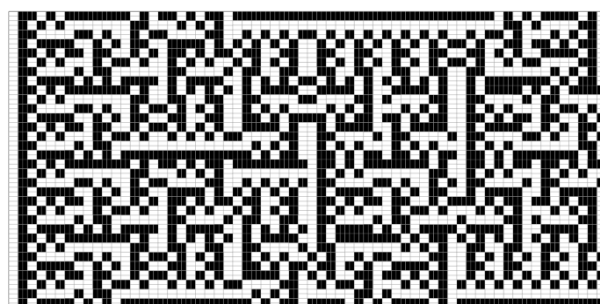
2110



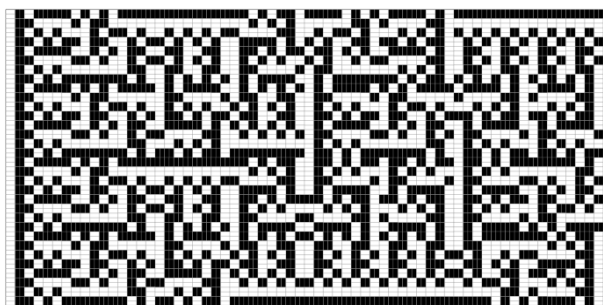
2111



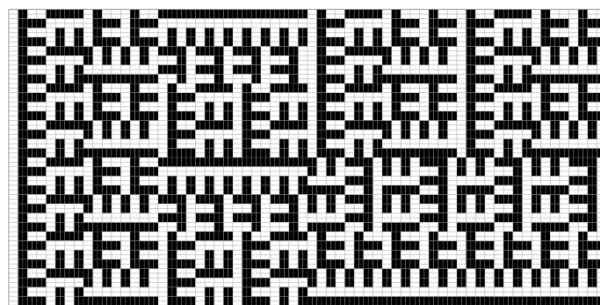
2112



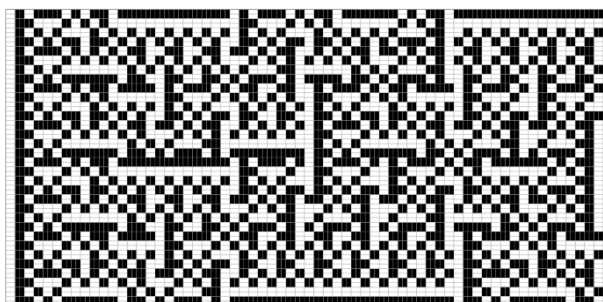
2113



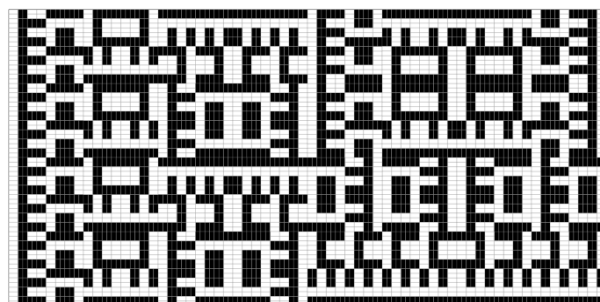
2120



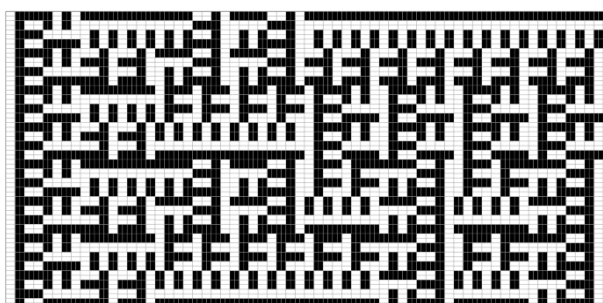
2121



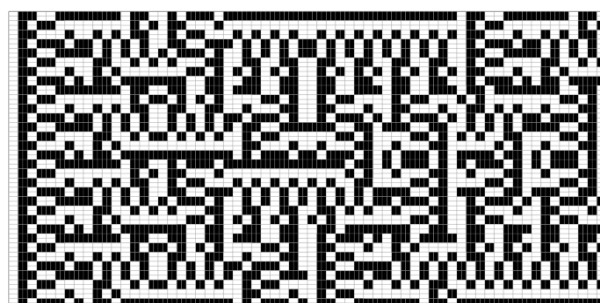
2122



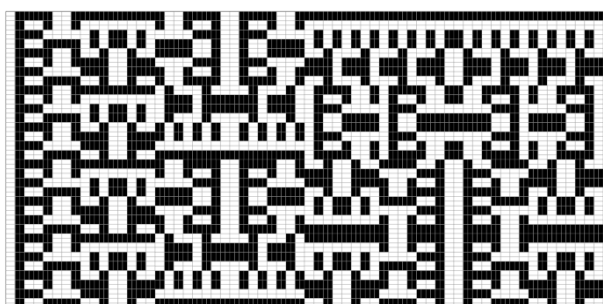
2123



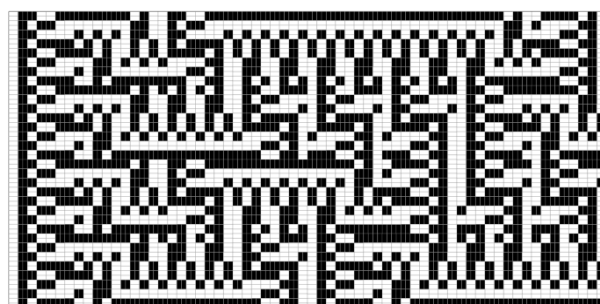
2130



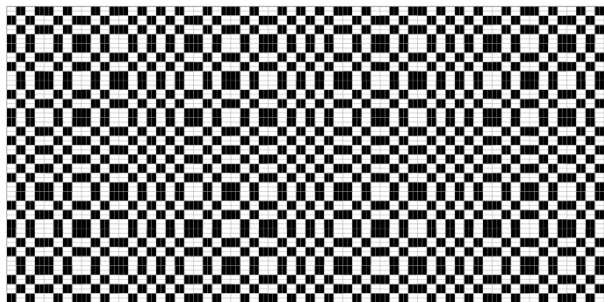
2131



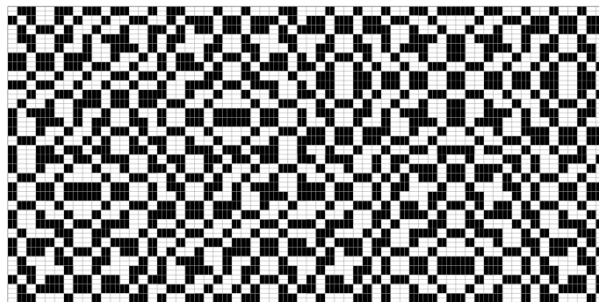
2132



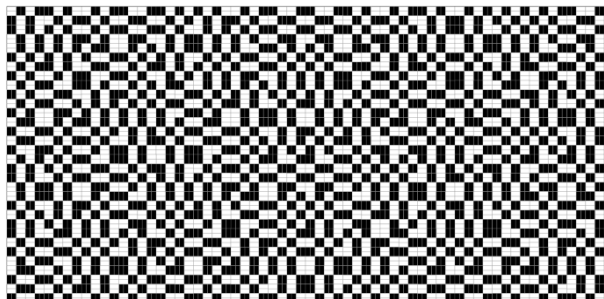
2133



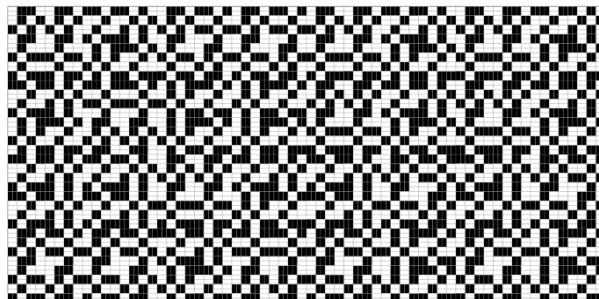
2200



2201



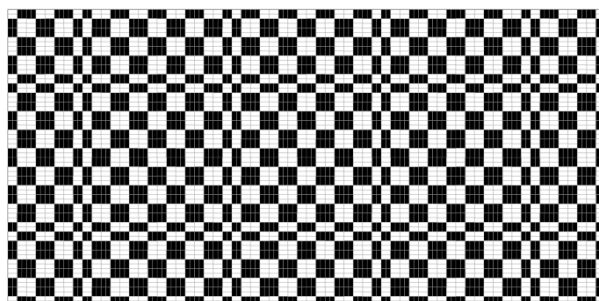
2202



2203



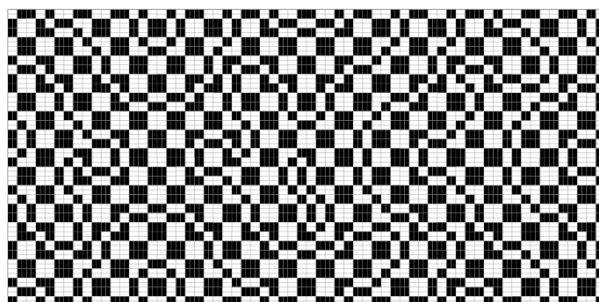
2210



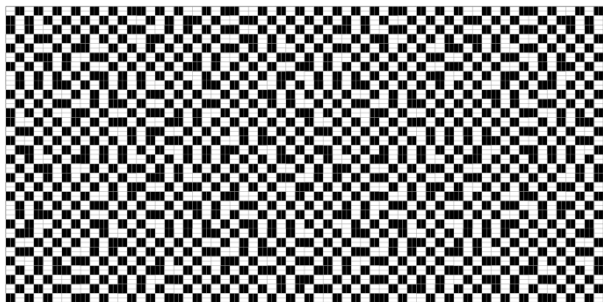
2211



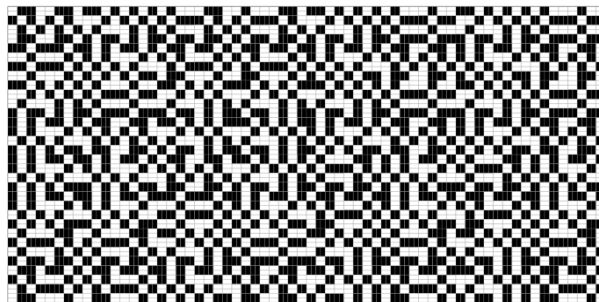
2212



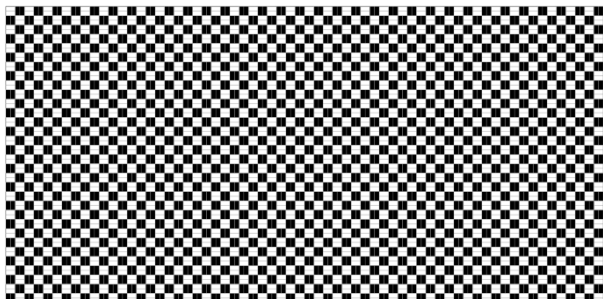
2213



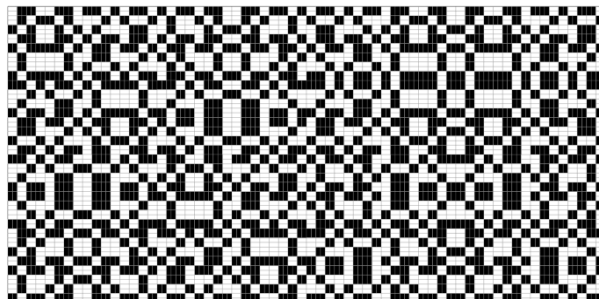
2220



2221



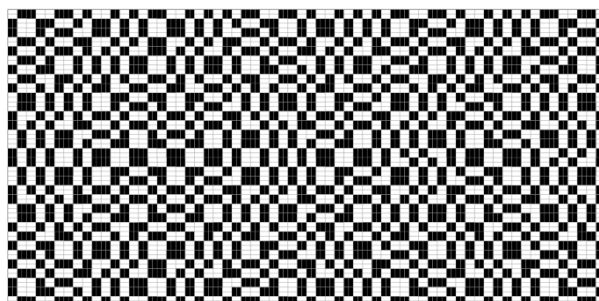
2222



2223



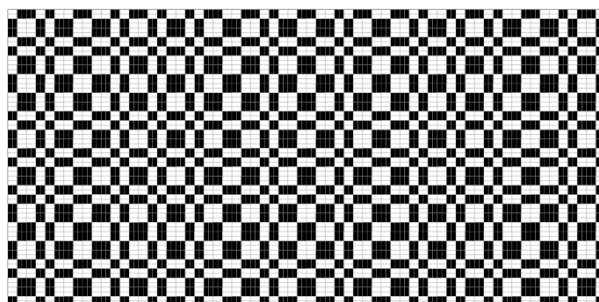
2230



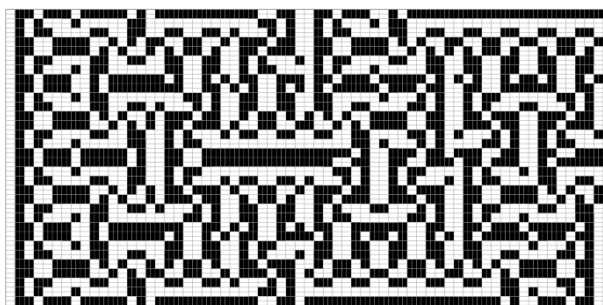
2231



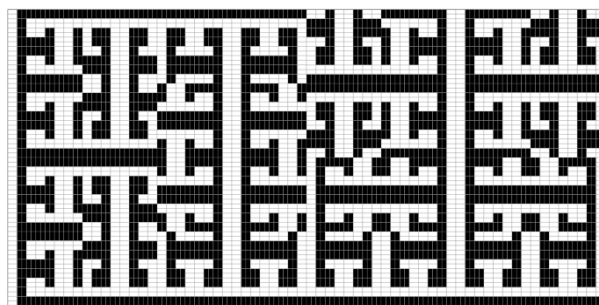
2232



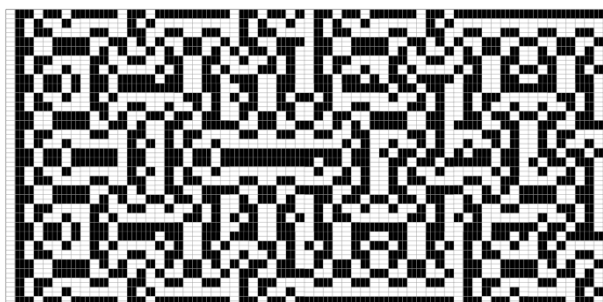
2233



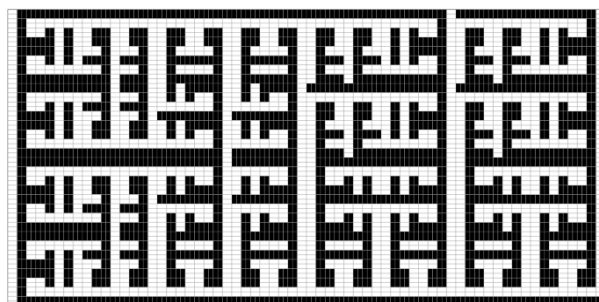
2300



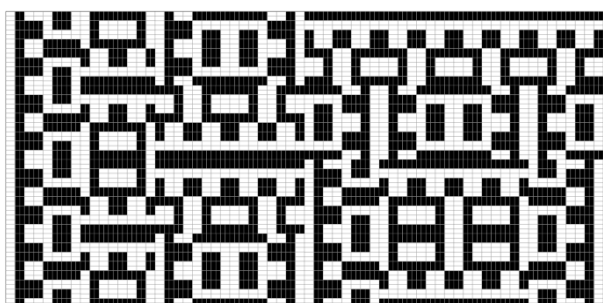
2301



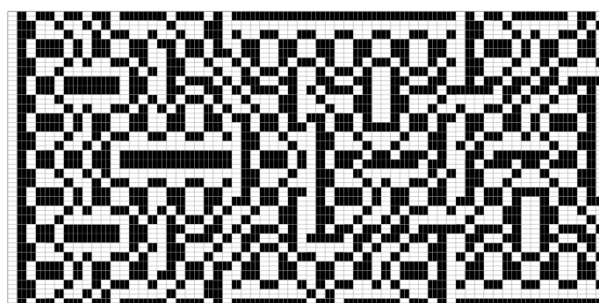
2302



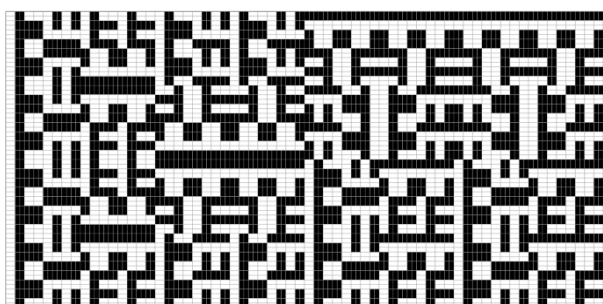
2303



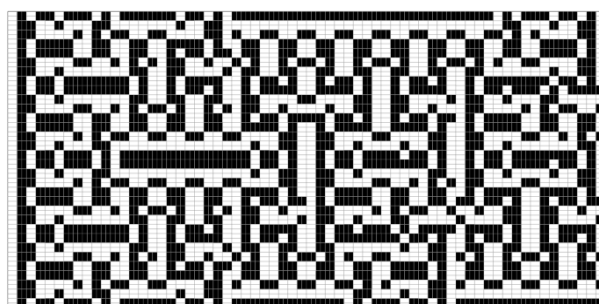
2310



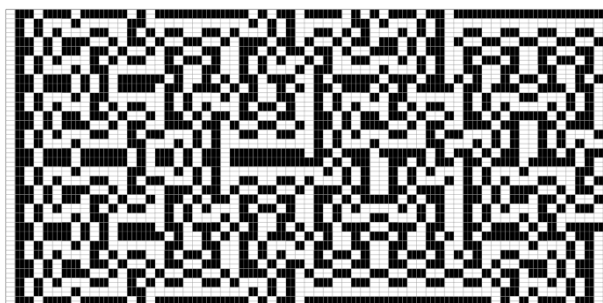
2311



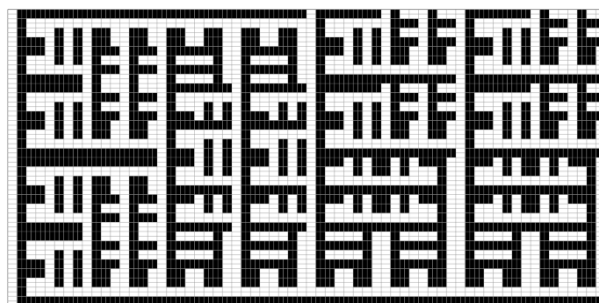
2312



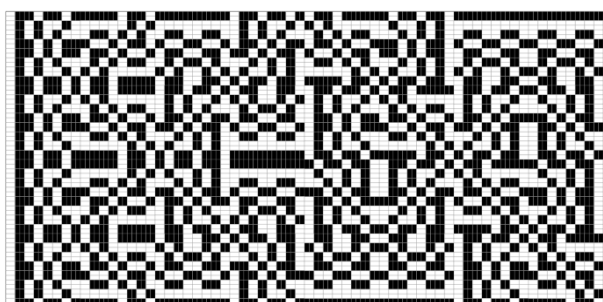
2313



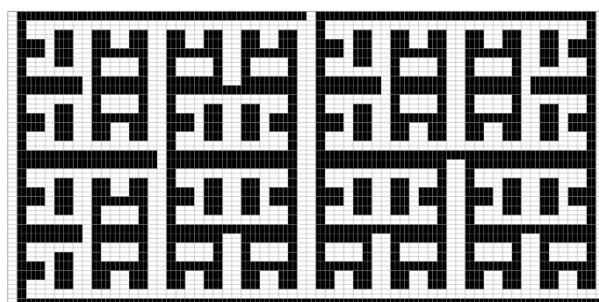
2320



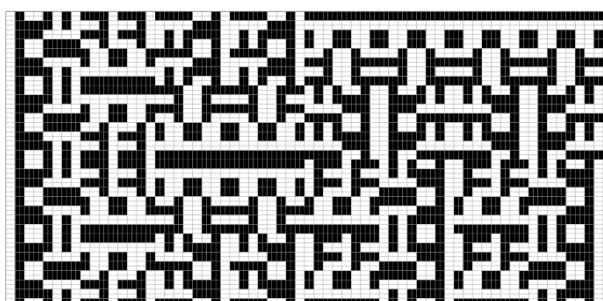
2321



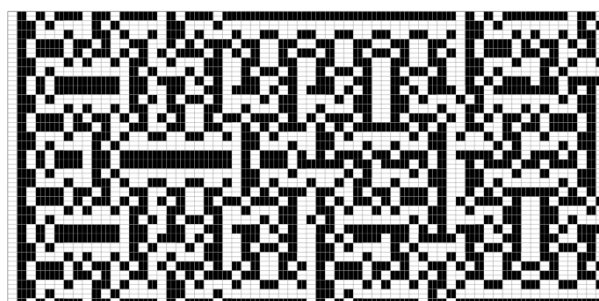
2322



2323



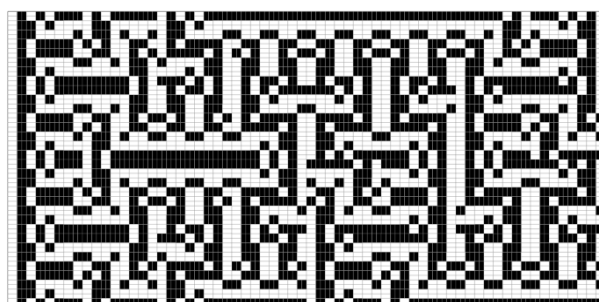
2330



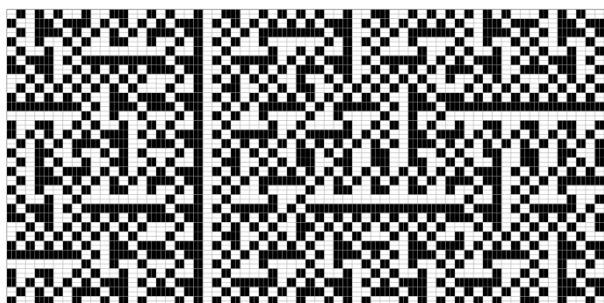
2331



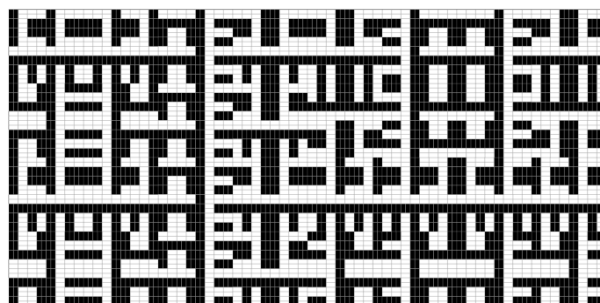
2332



2333



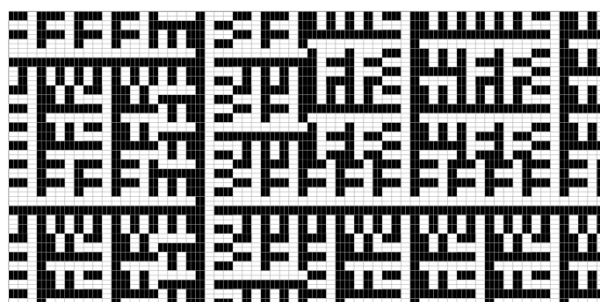
3000



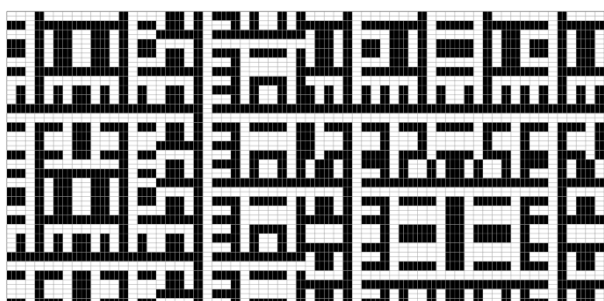
3001



3002



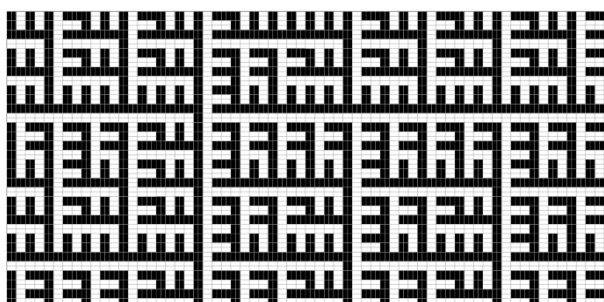
3003



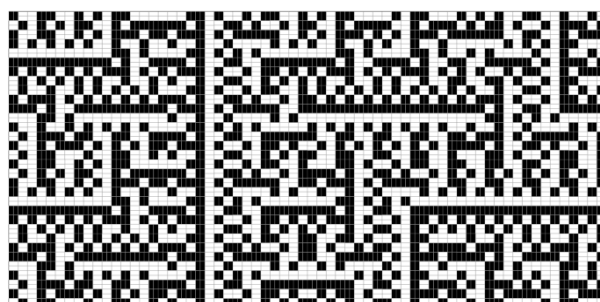
3010



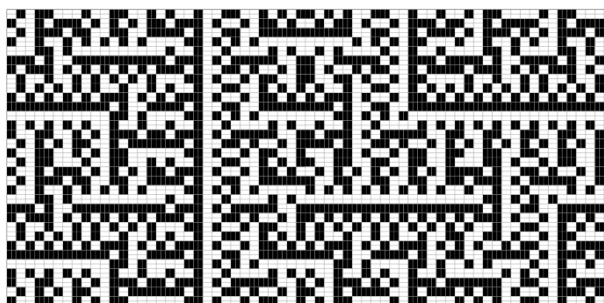
3011



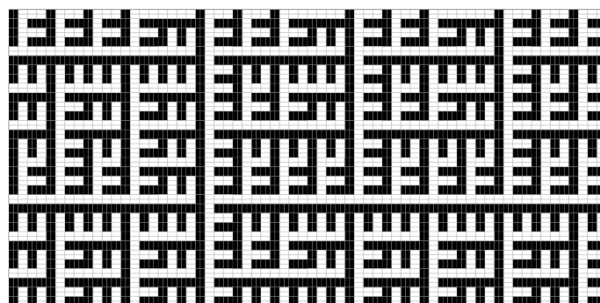
3012



3013



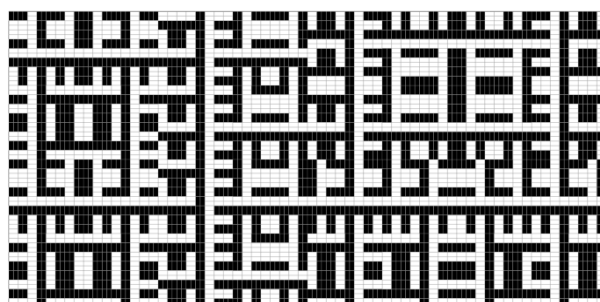
3020



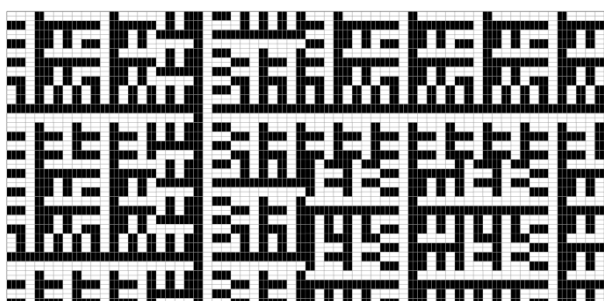
3021



3022



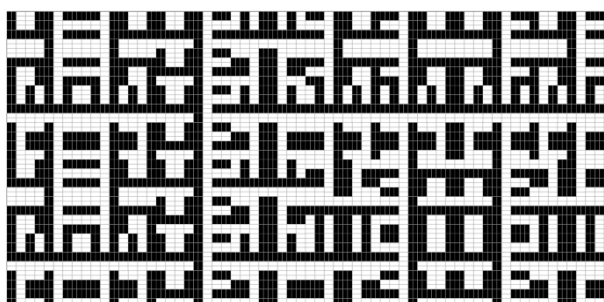
3023



3030



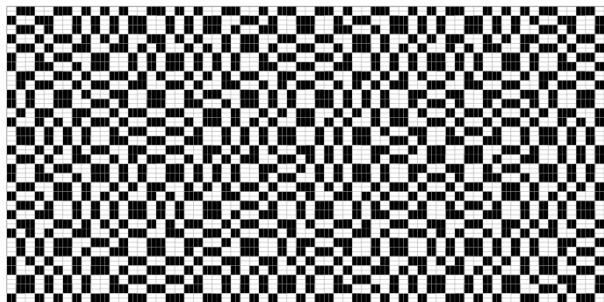
3031



3032



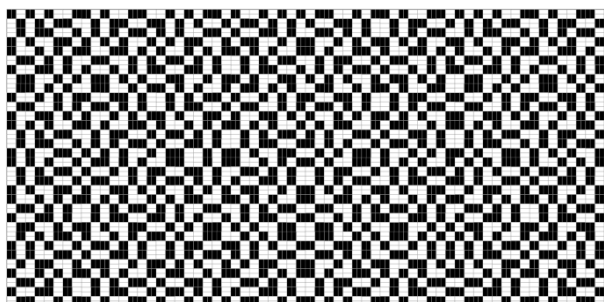
3033



3100



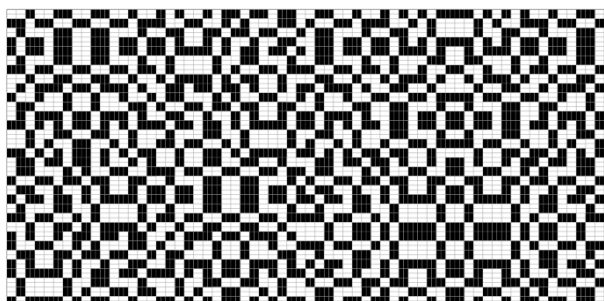
3101



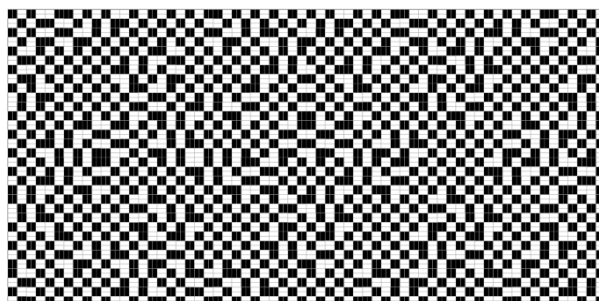
3102



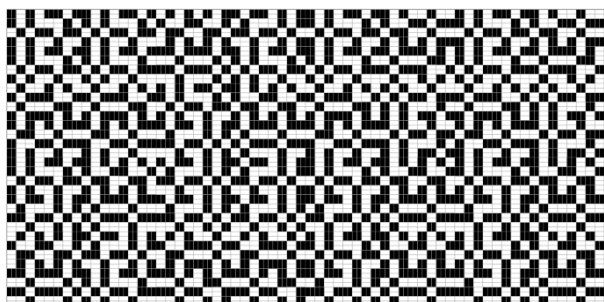
3103



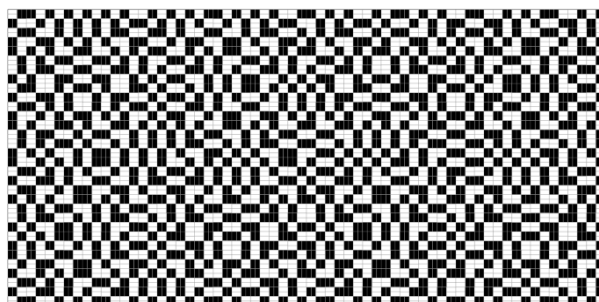
3110



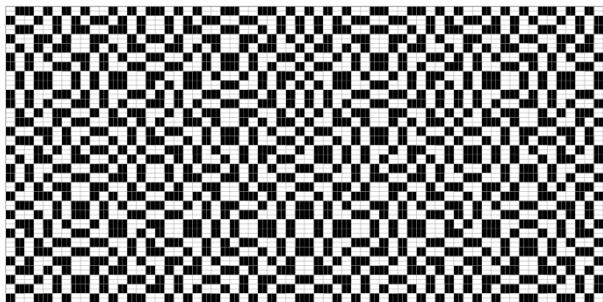
3111



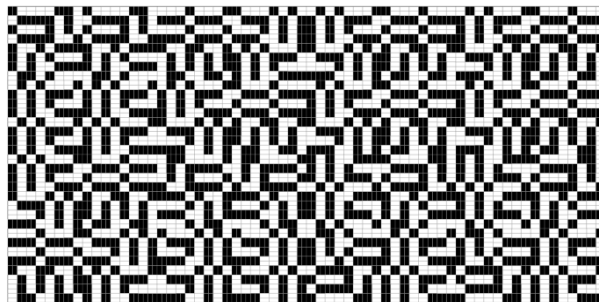
3112



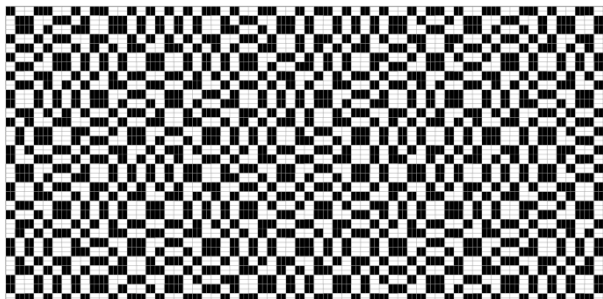
3113



3120



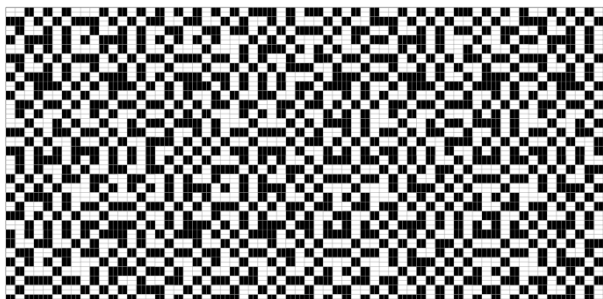
3121



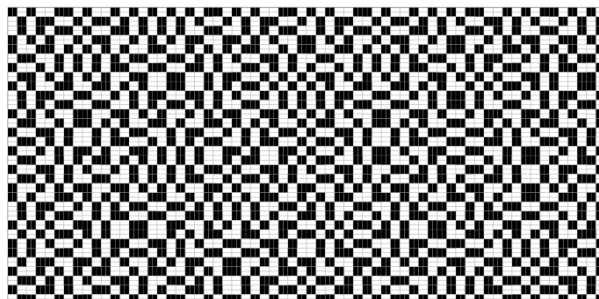
3122



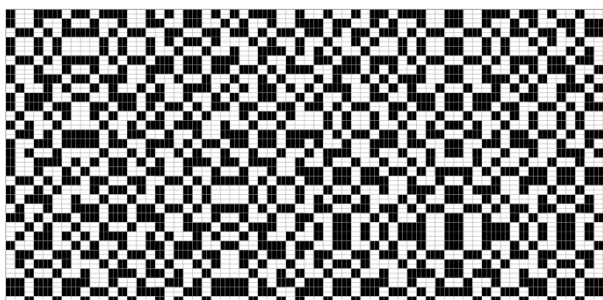
3123



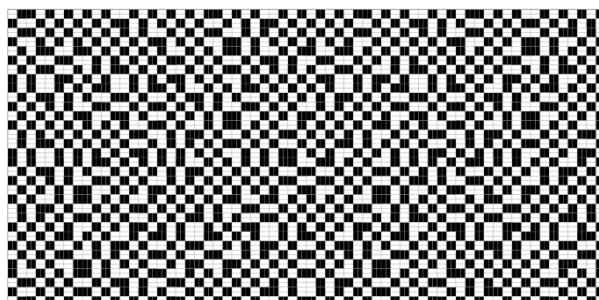
3130



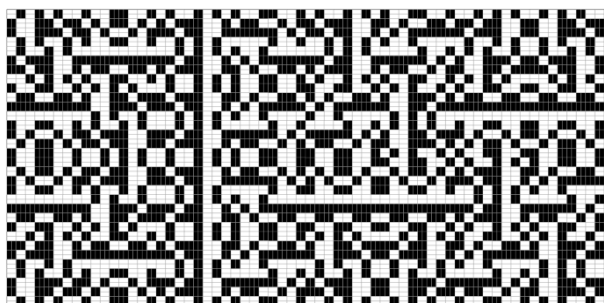
3131



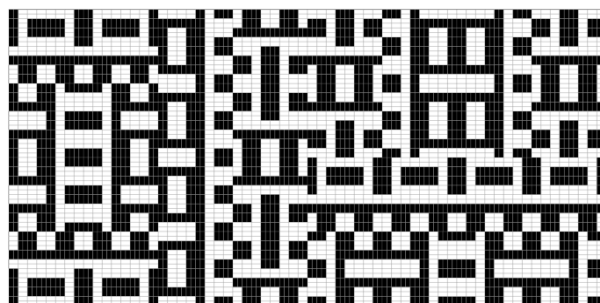
3132



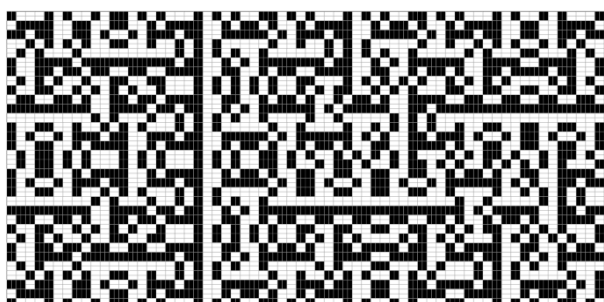
3133



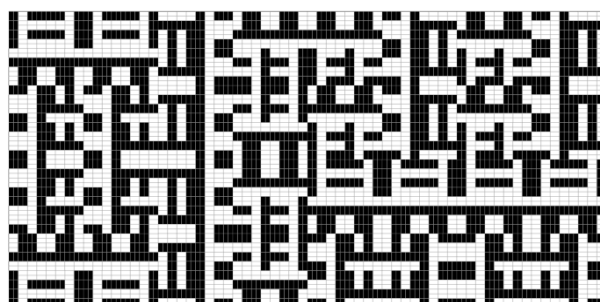
3200



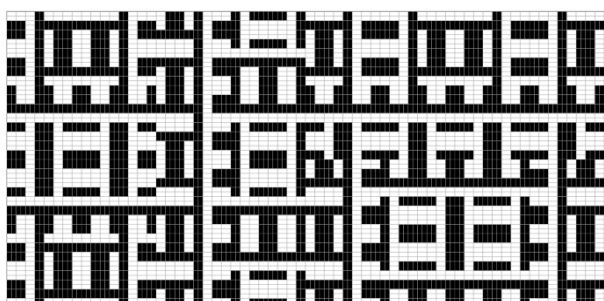
3201



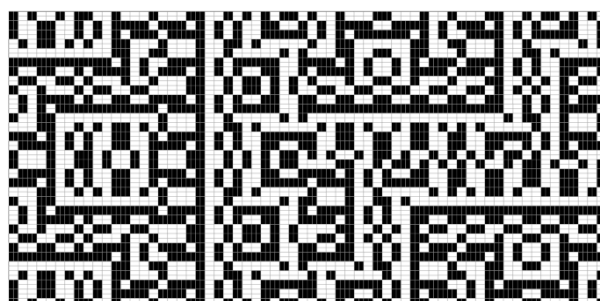
3202



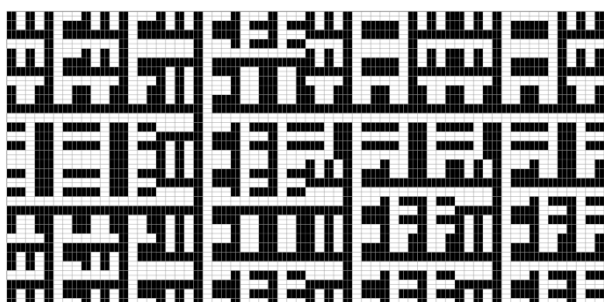
3203



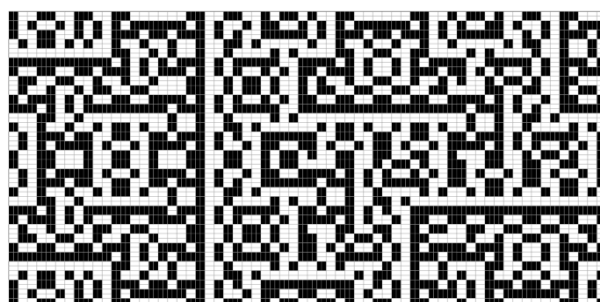
3210



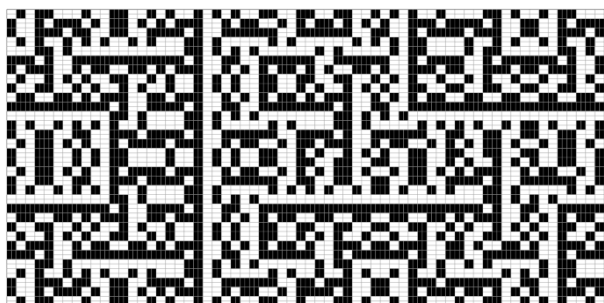
3211



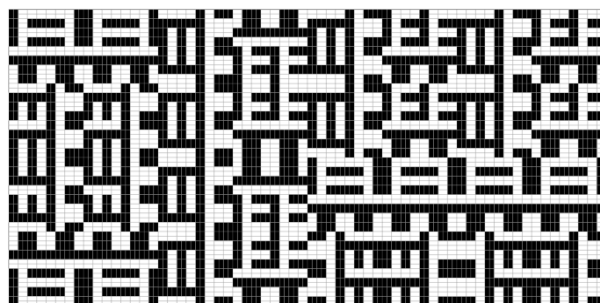
3212



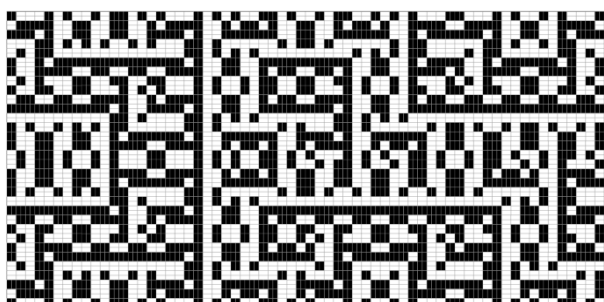
3213



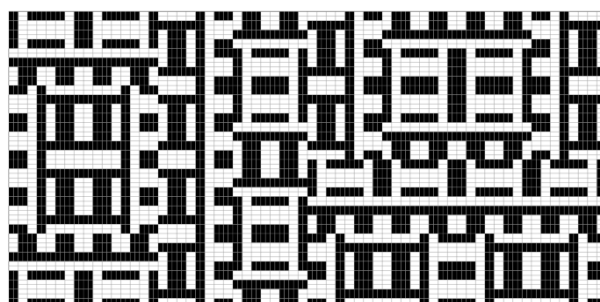
3220



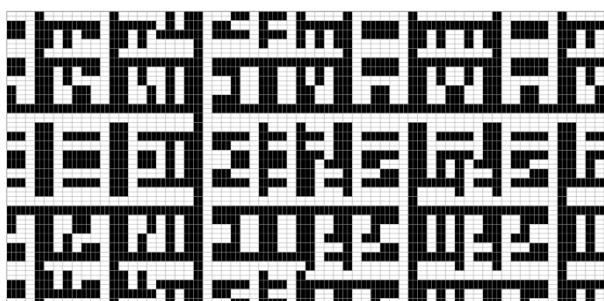
3221



3222



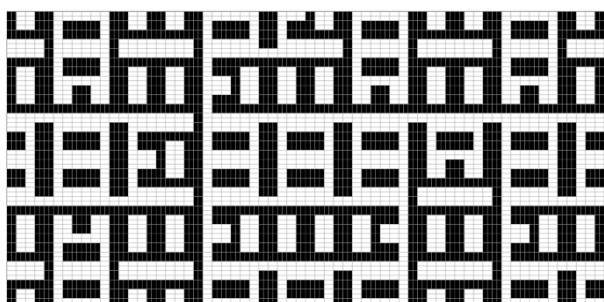
3223



3230



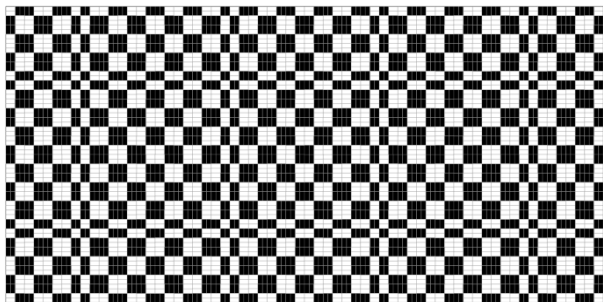
3231



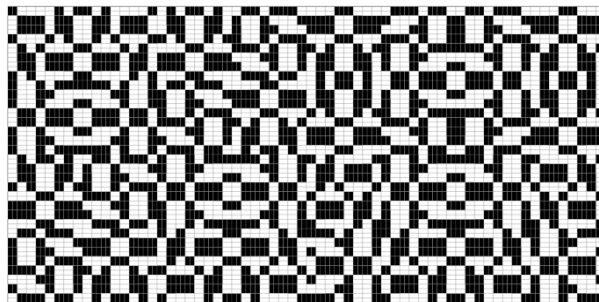
3232



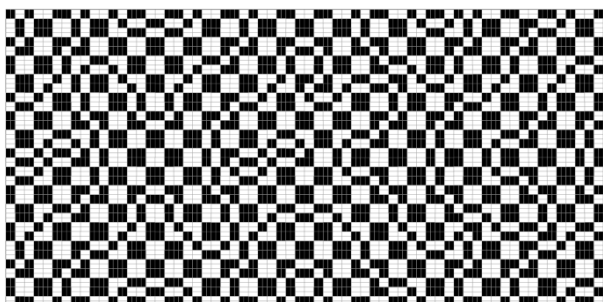
3233



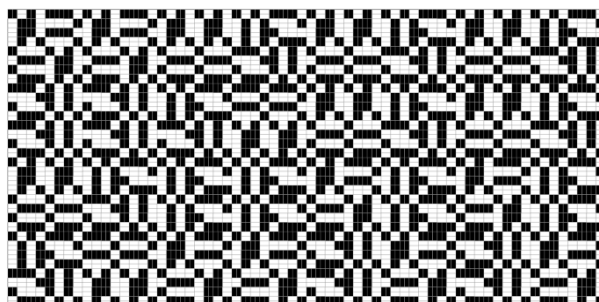
3300



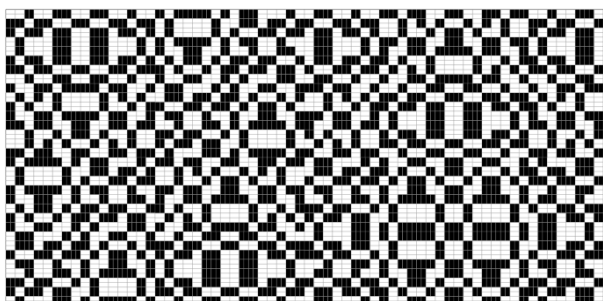
3301



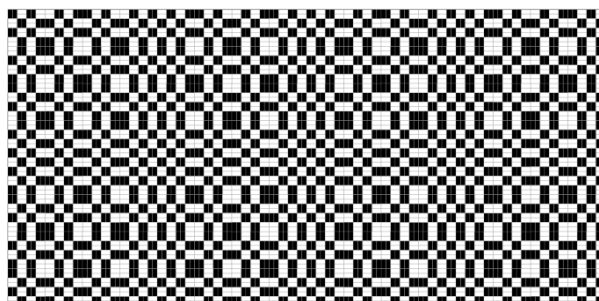
3302



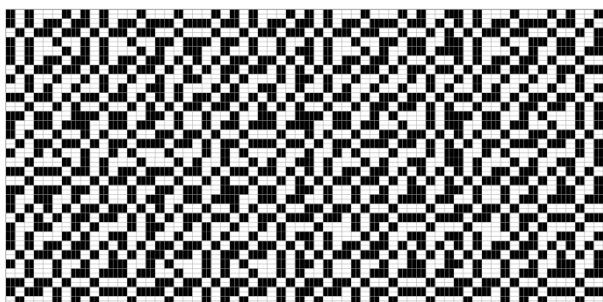
3303



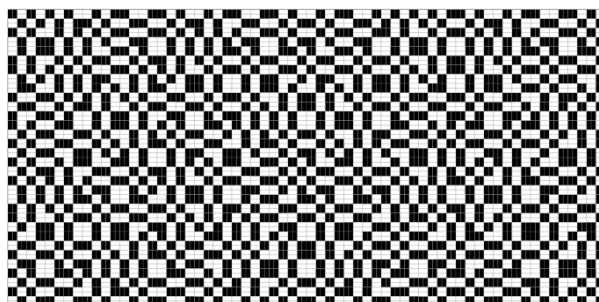
3310



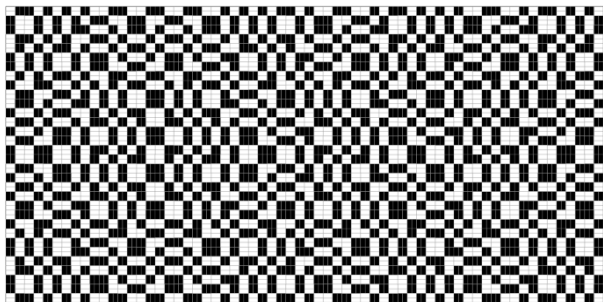
3311



3312



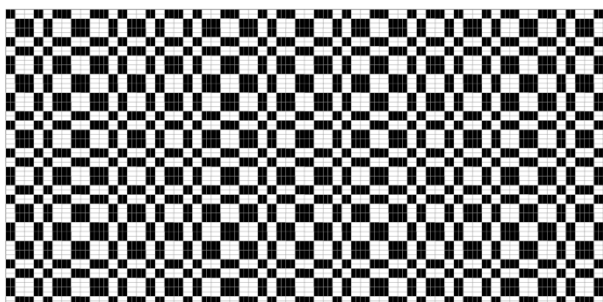
3313



3320



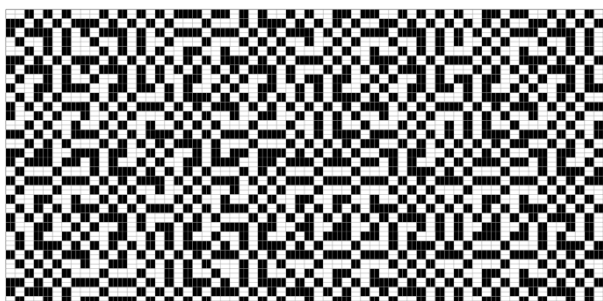
3321



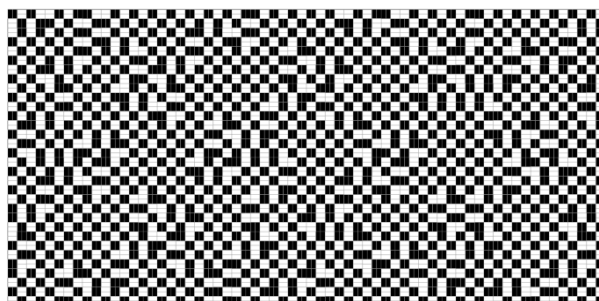
3322



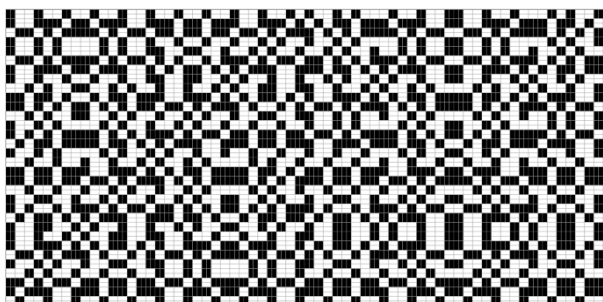
3323



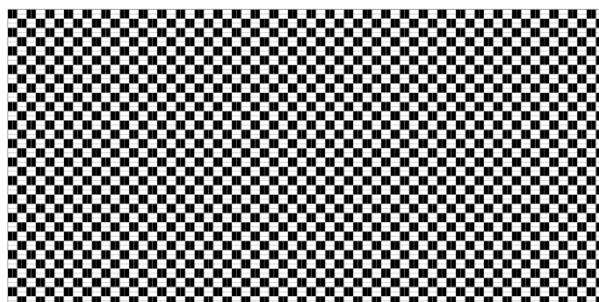
3330



3331



3332



3333