lpgyori/innerreliefs

Los endoX Application Suite

Background and Concepts

This document describes the functionalities
implemented in the latest releases (December 2025)
of the programs within Los endoX app suite.

The information contained herein is provided
for reference only and may be subject to change
without prior notice.

Contents

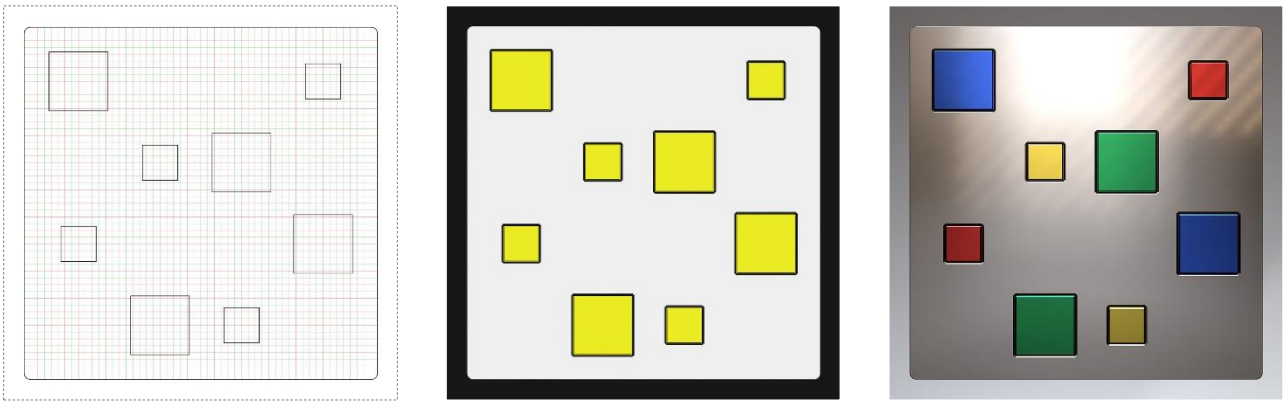
innerreliefs | Background and Concepts

0. Brief Introduction

My current work focuses on the *automated* exploration of a Euclidean geometry of elementary shapes, and on the interplay between *chance* and *control* in digital art. It also brings together certain concerns from industrial design and the subsequent *co*-creation with the user in an *interactive three-dimensional* environment.

In this context, the digital artworks I present are called ***innerreliefs***. They consist of 3D models whose front faces display a geometric composition generated *automatically* through *random* processes, within a specific configuration set in advance by the author.

These *innerreliefs* unfold in stages: beginning with generative *two-dimensional* geometric art, followed by automated *three-dimensional* realization and its corresponding visual output, and culminating in an *interactive* phase.



endograph#17553 | *its conformation* | *sample* innerrelief#000

Once the project is completed—still in its initial production stage and gradually developing—it will present a final catalog of **576 unique** individual artworks, arranged in a 24 × 24 cell grid. These are digital artworks whose visual themes are not yet fully defined, but they are expected to be formed from elementary geometry and a high degree of abstraction—if not absolute non-representativity—as is still the case today. *Tomorrow never knows*.

The *innerreliefs* are presented in series of *four* artworks belonging—let us say—to the same "family": sharing the same set of shapes and the same color palette. At least in the initial series, in which I undertake an exploration of various shape sets, each series displays this particularity: the allowable sizes fall into four specific ranges; the orientations of the figures

follow a characteristic alternation; and, similarly, the coloration adheres to a predetermined sequence. All of this gives the catalog of the entire body of artworks a certain "rhythm," thus forming a kind of more encompassing entity. (See the Appendix on Endographic Generation).

Unlike other generative projects—this must be emphasized—here the algorithmic generativity remains *reserved* for the author, while the viewer accesses an artwork that has already been conceived—not necessarily completed! There are reasons for this; let us consider them…

First of all, we should note the steps involved in producing an *innerrelief*. These include generating its two-dimensional composition, applying color, forming it three-dimensionally, engraving the surface of its front plate along with an informational panel on its back, applying materials to the now engraved artwork, exporting it in a format suitable for WebGL visualization in web browsers, and ensuring that, once acquired, it can be experienced *offline*, in keeping with the concept of *decentralization*.

Given the complexity of the production process, the system was automated using Autodesk 3ds Max and its MAXScript language, providing capabilities that no web framework can match. It offers advanced modeling with a modifier stack, versatile geometry editing, parametrization, and various optimizations—functions difficult to replicate in tools like p5.js, Three.js, or Processing, which excel at interactive and generative visualization but are limited in 3D modeling and integration with full production workflows. Due to these characteristics, the generative process is reserved for the author, leaving the user to engage with an already established artwork.

Another crucial point regarding the author's exclusive control over the generative process lies in his—still non-transferable—role as *selector* and *validator* among the myriad of massively generated formulations, of which only a small fraction—admittedly— holds what might be called genuine aesthetic value. This selective control ensures that the final artwork reflects the author's vision and maintains the intended aesthetic and conceptual integrity, rather than being a mere output of algorithms.

The software solution developed here operates as a *massive producer* of compositions that I have called ***endographs***—*endo* referring to interior or endogenous, since the drawings and their transformations are generated internally, *in silico*, according to the rules of each program in the suite. To clarify:

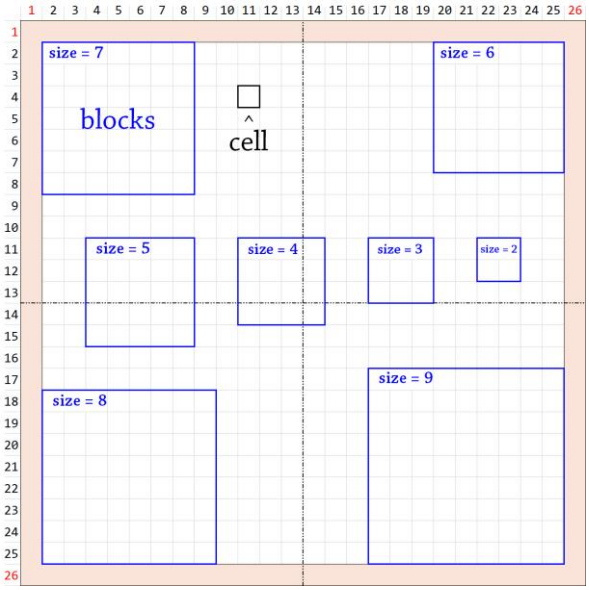
From the *two-dimensional* production arises an ***endograph***.

From its *three-dimensional* transformation arises an ***innerrelief***.

Initially, the system places a square figure with rounded corners, which serves as a containing enclosure or *base*, within which the series of figures that make up each drawing are arranged.

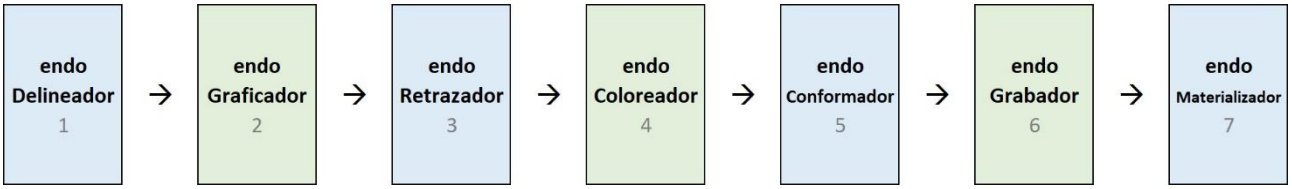
Before proceeding, and since both concepts will be immediately relevant, I define: *cell* as the *smallest* identifiable division within the compartmentalization of this base, measuring

15 dimensionless units; and *block* as any *square* grouping of contiguous cells, each serving as a container for *a single individual figure*. See the following illustration:



Structurally, this enclosure—measuring 390 units per side—features a subdivision 26 × 26 cell *grid*, with a one-cell-wide border representing its *forbidden* zone. It follows that the remaining usable area consists of 24 × 24 cells, that is, 576 cells in total. The *smallest* figure occupies a block measuring 2 cells per side (30 units), while the *largest* occupies 9 cells (135 units). *Overlapping* of figures within the drawing is *not* allowed, meaning that blocks may be *contiguous* at most.

With these initial specifications, I began programming my project in August 2022.
The software suite, named **Los endoX**, now consists of *seven* independent modules, whose names are in Spanish, each accompanied by its respective function, as detailed below:



1. **endoDelineador** (Delineator): generation of micro-images for delineating *endographs*.
2. **endoGraficador** (Grapher): generation of *endographs*.
3. **endoRetrazador** (Retracer): repetition of the tracing of an *endograph*, and its storage.
4. **endoColoreador** (Colorizer): coloring of an *endograph*.
5. **endoConformador** (Former): three-dimensional forming of an *endograph*.
6. **endoGrabador** (Engraver): surface engraving of the front plate and rear cover in an already formed *endograph*.
7. **endoMaterializador** (Materializer): application of virtual materials to an already engraved artwork.

2. endoGraficador (Grapher)

There is no pagination error here! I deliberately begin with the second program. Why? Because it represents the true starting point—the first piece of the app suite to be coded. Although, in functional terms, it is preceded by **endoDelineador**, that module is actually derived from **endoGraficador**, since it reuses much of its code.

As my programs have no graphical interface, I do not have at hand a user-friendly environment to interact with; this was intentional in the original implementation, as it is not a commercial venture nor one meant for distribution. It is simply a personal and highly customized software solution! Well, "simple" might be a bit of an overstatement... The entire app suite kept me programming for over a year, writing more than 10,000 lines of code (including comments), followed by at least another two years of improvements, expansions, and debugging—adding no fewer than 4,000 additional lines; an effort that, in the end, became as much a part of the creative process as the artworks themselves!

A. General Definition of Arrays

This program, useful for the automatic generation of *endographs*, includes a wide range of definitions that determine the general characteristics of each productive session.

The repertoire of figures and their attributes, used to compose a drawing, is defined through five *arrays* established at the outset: *density*, *shape* class, *spacing*, *size*, and *orientation*.

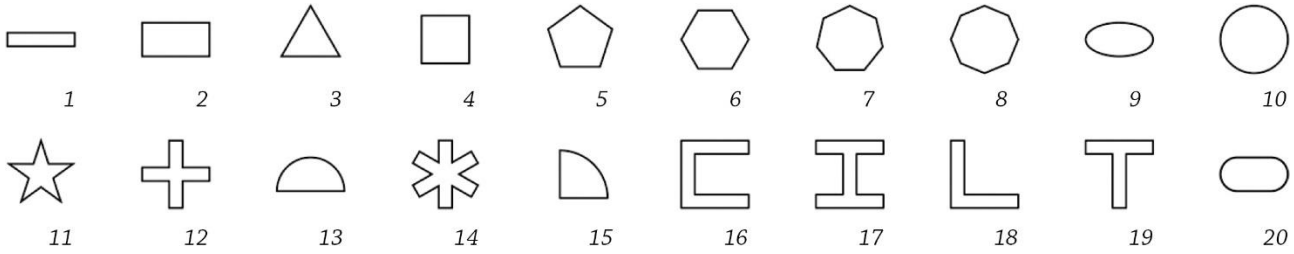
Essentially, each array stores *relative* values of the *expected occurrence* for each characteristic it represents. Consequently, it sets up the framework for characteristics with *weighted* or *unequal probabilities*—in other words, where the probabilities of any given attribute may *differ* from one another, if that is desired. Some concepts associated with the definition of these five arrays are detailed below.

1. Density

Determination of figure *density*. This variable controls how *dense* or populated the drawing will be—that is, whether all the cells, addressed by column (x-axis) and row (y-axis), will carry a figure or a "portion" assigned to them. The first element of the array represents the weighted value given to *non*-assignment, while the second corresponds to actual assignment.

2. Shape Class

Determination of the occurrence rate of each distinct *shape* class. This second array stores the *relative* occurrence rates of each of the twenty distinct classes of elementary geometric shapes implemented to date: line segment, rectangle, triangle, square, pentagon, hexagon, heptagon, octagon, ellipse, circle, regular five-pointed star (pentagram), cross, semicircle, six-pointed asterisk, circular sector, C-profile, I-profile, L-profile, T-profile, and stadium.

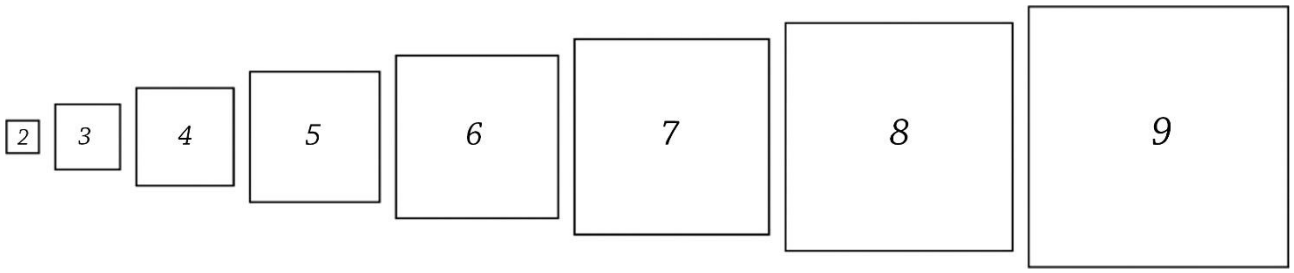


3. Spacing

Two *spacing* levels are defined, representing the amount of *free* space around the figures: zero and one unit, implying figures that are "touching" or, conversely, separated by a distance equal to one empty cell. This third array stores the *relative* occurrence rates of both spacing levels.

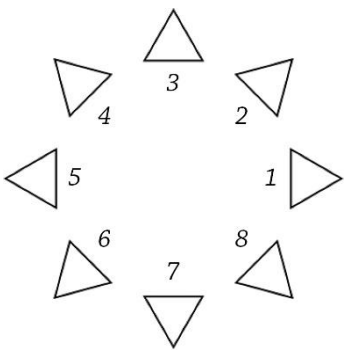
4. Size

Eight possible *sizes* are defined for the figures. The *smallest* figure—as noted above—occupies a square block of 2 cells per side and is therefore referred to as having size 2; the *largest*, with 9 cells per side, is assigned size 9. *No figures of size 1 exist*. This fourth array stores the *relative* occurrence rates of each possible figure size.



5. Orientation

Eight possible *orientation* values are defined for the figures: 1 (east), 2 (northeast), 3 (north), 4 (northwest), 5 (west), 6 (southwest), 7 (south), and 8 (southeast). Odd numbers correspond to the cardinal points, while even numbers indicate the intermediate directions. This fifth and final array stores their *relative* occurrence rates.



6. Weighting Recalculation

This procedure was implemented to eventually *modify* the occurrence rates of each characteristic during program execution. A constant for adjusting individual occurrence is set, which can either *reinforce* the future probability of the characteristic—*biasing it*—or *weaken it*, as if performing a *random selection without replacement*. This recalculation can be enabled independently for shape classes, figure sizes, and orientations.

7. Preexisting Figures

Although the **endoGraficador** was designed as a program for automatic generation, it also provides the optional capability to draw, at the start of its execution, a series of *predetermined* figures if required. The program can then either complete the loop immediately by limiting itself to this series or continue afterward with the randomized drawing process.

B. Definition of Additional Variables

The program includes an extensive set of variables associated with the numerous attributes and initial configurations. Some of them are as follows:

1. Length of the Productive Session

The *number* of drawings to be generated in each program execution must be defined.

2. Delineation Map

This program uses so-called *delineation maps* as a guiding framework for the structure of each drawing. A delineation map can specify both the *density* levels within the area available for drawing and the approximate positions of graphical elements. Although it works *in conjunction* with the *density* array, the drawing options defined by the map always take precedence.

3. Operating Modes

The **endoGraficador** is a program for automatic generation of *randomized* drawings, operating under a pseudo-random number generator algorithm. Its randomness, in the first instance, applies *solely* to the *selection* of each figure's attributes, carried out over the arrays whose contents were *determined* at the start—a mode that remains enabled by default. However, this *randomness* can be extended even to the *definition* or loading of the arrays themselves, consequently expanding the uncertainty from the individual occurrence of figures to the very composition of the complete characteristic repertoire, resulting in an *unpredictable* generation system.

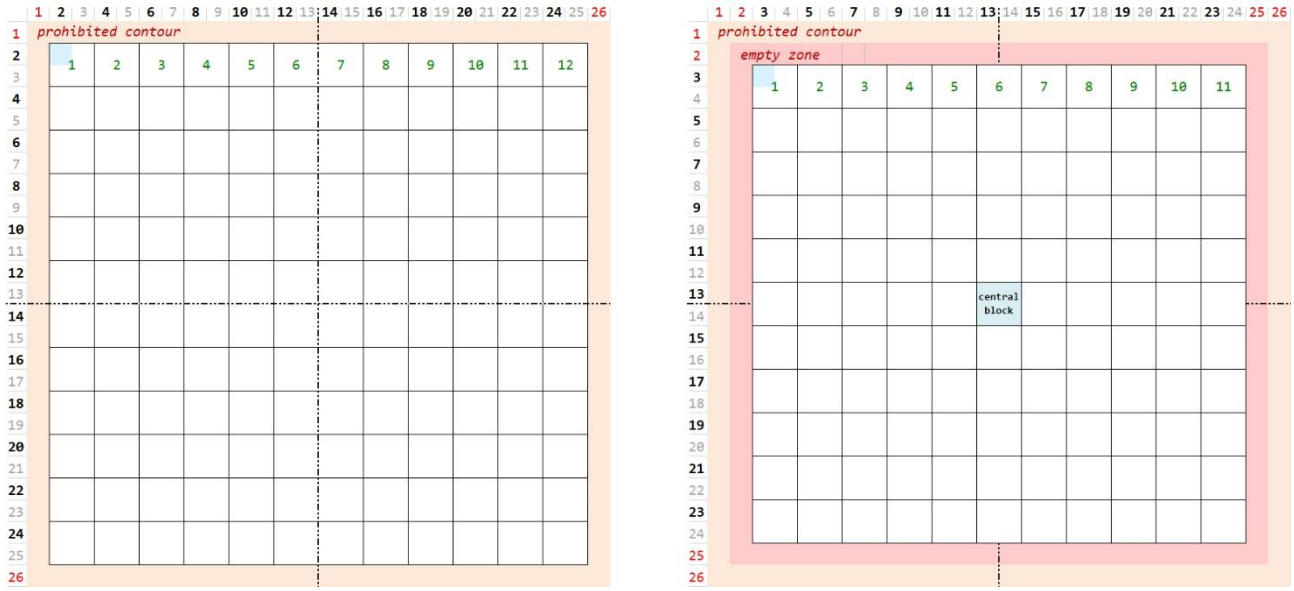
4. Unordered vs. Ordered (Even- and Odd-Restriction)

The *position* of a figure is managed through a *row-column* pair. Precisely because this is a random process, both values can assume any of the admissible numbers: from 2 to 25 (recall that positions 1 and 26 are prohibited as they belong to the empty contour). If the *unrestricted* determination of row and column is left to chance, they will, of course, assume both even and odd values.

The **endoGraficador** includes the optional activation of a second design methodology, a *restrictive* one, which allows the composition of a drawing whose elements now follow an "ordered" arrangement.

It is necessary to clarify that, as the reference for specifying the exact *position* of each figure, the *upper-left cell* of the block containing it was adopted instead of its center.

This second methodology offers two options. *Even-restriction* places the figure—specifically its upper-left cell—into any cell whose row and column values are both *even* numbers.



even-restriction and odd-restriction

Complementarily, *odd-restriction* places the upper-left cell into any cell whose row and column values are both *odd* numbers. Both row 25 and column 25, although odd-numbered, must remain excluded, since not even a figure of minimum size can be placed there.

5. Numerical Data Files

As a record of the production of each drawing and the quantification of its composition, the creation of a TXT *data* file can be requested. This file contains: figure number; row; column; shape class; size; and orientation; along with two additional values useful for calculating the probability associated with the resulting size: the minimum and maximum

available spaces at the moment of its determination. These numerical data will be used in the other programs of the suite.

6. Output Images

After completing each drawing, a *rendered* JPG image file can be created.

7. Informative Labels

It is possible to include a set of *informative labels* with the drawing image: *upper* labels referring to the drawing's *data*, and *lower* labels referring to the *results*.

8. Mass Production File

Finally, all the information displayed in both sets of labels, in addition to appearing on the image, can be exported to a TXT *production* file.

C. Drawing Production

A main *loop*, covering the entire productive session, ensures that the process is *repeated* according to the requested number of drawings. This graphic production process runs in parallel with a *verification* procedure using a specific *matrix* (an ordered two-dimensional array of numerical data for our purposes). Figures are determined and then drawn *one at a time*, until either the requested maximum number is reached or the available grid is filled.

1. Limitation on the Occurrence Proportion of Size-2 Figures

Size 2 was defined as the *minimum available size*. It will remain the smallest possible size *as long as* the number of size-2 figures has *not* exceeded the *maximum allowable proportion*. If this limit is reached, size 2 is restricted, making size 3 the new minimum.

2. Secondary Drawing Loop

Within the main loop, there is a *nested loop* that governs the generation of each individual drawing, randomly selecting a position, after which it is determined whether or not a figure is assigned to it.

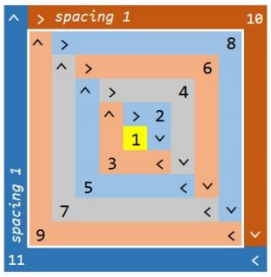
If required, *preexisting figures* are drawn first.

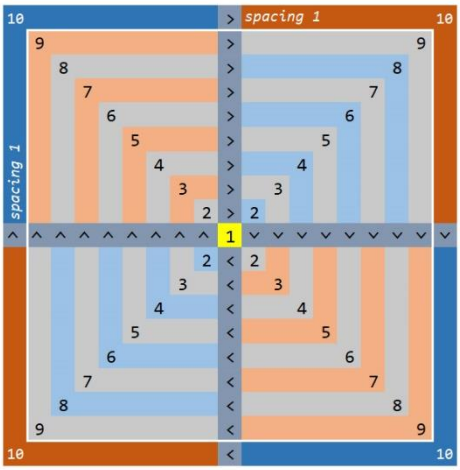
3. Initial Attribute Determination

The first step consists of determining the *row* and *column*, either according to the corresponding arrays when preexisting figures are present, or randomly. The same differentiated procedure is then applied to determine *density*, *shape* class, *spacing*, and *orientation*.

4. Evaluation of Available Space

Determining the size requires an evaluation of the *available space*, except for preexisting figures, for which this is unnecessary. The program uses two methods to determine available space, namely the *two-alternating-quadrants method* and the *four-quadrants method*, as shown in the illustrations that follow:





Both methods operate *randomly*, meaning the sequence of space evaluation follows *no* predetermined rule, ensuring that *no* areas are intentionally favored or disadvantaged regarding subsequent figure placement. Considering the advantages and disadvantages of each method—which do not need to be detailed here—a compromise solution is adopted: the most favorable option is chosen, providing the widest margin, that is, the *largest available maximum size*.

5. Determination of a Possible Limitation for Size-2 Figures

Evaluation of whether adding a new figure would *exceed* the maximum allowable proportion of size-2 figures relative to the current quantity.

6. Second Attribute Determination

First, the *size* of the figure to be drawn is determined randomly, or read from the corresponding array in the case of a preexisting figure.

Next, the *accumulated area*—or total number of cells occupied by figures—is calculated, for the subsequent computation of the *occupancy index*.

Then, the upper-left cell for drawing the figure is determined according to the resulting method and size.

Finally, the position of the *center* of the element to be drawn is established.

Once all required variables are in hand, the corresponding drawing *function* is called based on the *shape* class.

D. Subsequent Tasks: Verification and Probabilistic Calculation

1. Loading the Verification Matrix

The verification matrix is filled, categorizing positions as: not yet evaluated (0); forbidden, as part of the contour (1); affected by spacing (2); or assigned to the figure (3).

```
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 0 2 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 2 0 0 1
1 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 2 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2 2 2 2 0 0 1
1 0 0 0 0 0 0 0 2 2 2 2 2 2 0 0 0 2 3 3 3 3 2 0 0 1
1 0 0 3 3 3 3 0 2 3 3 3 3 2 3 3 3 3 3 3 3 3 2 0 0 1
1 0 0 3 3 3 3 0 2 3 3 3 3 2 3 3 3 3 3 3 3 3 2 0 0 1
1 0 0 3 3 3 3 0 2 3 3 3 3 2 3 3 3 3 3 3 3 3 2 0 0 1
1 0 0 3 3 3 3 0 2 3 3 3 3 2 3 3 3 3 2 2 2 2 2 0 0 1
1 0 0 2 0 0 0 0 2 2 2 2 2 2 2 2 0 2 3 3 3 3 2 0 0 1
1 0 0 0 0 0 0 0 0 2 3 3 3 3 3 2 0 2 3 3 3 3 2 0 0 1
1 0 0 2 2 2 2 2 2 2 3 3 3 3 3 2 0 2 3 3 3 3 2 0 0 1
1 2 0 2 3 3 3 3 2 2 3 3 3 3 3 2 0 2 3 3 3 3 2 0 0 1
1 0 2 2 3 3 3 3 2 2 3 3 3 3 3 2 0 2 2 2 2 2 2 2 0 1
1 0 0 2 3 3 3 3 2 2 3 3 3 3 3 2 0 0 0 0 0 0 0 0 0 1
1 0 0 2 3 3 3 3 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 1
1 0 0 2 2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 2 3 3 3 3 2 0 0 0 0 0 0 0 0 0 0 2 1
1 0 0 0 0 0 0 0 2 3 3 3 3 2 0 0 0 2 2 2 2 2 2 0 0 1
1 0 0 0 2 0 2 2 3 3 3 3 2 0 0 0 2 3 3 3 3 2 0 0 0 1
1 0 0 2 0 0 0 2 3 3 3 3 2 0 0 0 2 3 3 3 3 2 0 0 0 1
1 0 0 0 0 0 0 2 2 2 2 2 2 0 0 0 2 3 3 3 3 2 0 0 0 1
1 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 2 3 3 3 3 2 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 2 2 2 2 2 2 0 0 0 1
1 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

This example shows a typical verification matrix during the drawing generation process. Note the isolated positions categorized with a "2", corresponding to drawing events that were randomly cancelled by the action of the *density* array.

2. Writing the Numerical Data File

The defining data of the most recently drawn figure are written to this output file.

```
|num|fi|co|cl|ta|or|mi|ma
1001101101104126101100100 < the first row of numbers corresponds to the square base
1002102102104106101102109
```

3. Calculation of the Figure's Attributes Probability and Information

The calculation of the occurrence probability for the set of figures composing a drawing takes into account, first, the probabilities associated with the spatial availability for *placing* each figure; then, the probability of assigning or not assigning figures, considering the delineation map or the *density* array; and, finally, those derived from the figure's *shape* class, *size*, and *orientation*.

A *sequential conditional* method was implemented for this calculation—i.e., a technique that follows a series of ordered steps, where each step depends on the result of the previous one.

As the first stage of the method, the *probabilities* and *information* (as defined in Claude Shannon's information theory) contributed *individually* by these *attributes* of the most recently drawn figure are calculated, along with the concomitant values that determined its position on the drawing plane.

Note: A detailed discussion of this topic, both theoretical and practical, can be found, at the end of this document, in the Appendix on Probability and Information.

4. Calculation of the Figure's Total Probability and Information

The occurrence *probability* of a figure equals the *product* of the *five* probabilities corresponding to its positioning, actual assignment, shape class, size, and orientation.

The *information* contributed by a figure equals the *sum* of the *five* information values derived from its positioning, actual assignment, shape class, size, and orientation.

5. Calculation of the Drawing's Cumulative Probability and Information

The occurrence *probability* of a drawing—that is, of the set of *all* figures composing it—equals the successive *product* of the individual probabilities of each figure.

Then, the drawing's cumulative *information* in *bits* is calculated as the successive *sum* of the individual information contributed by each figure.

6. Weighting Recalculation

If its Boolean variables allow it, the *weighting recalculation* of shape class, size, or orientation is carried out.

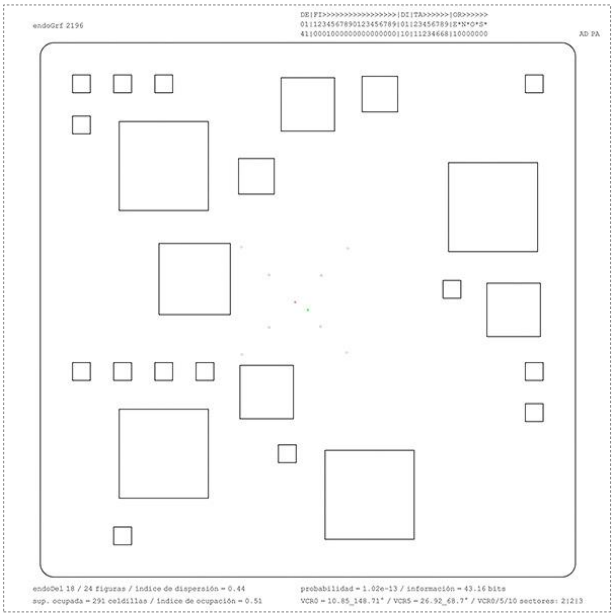
E. Generation of Drawing Rating Values and Image Informational Captions

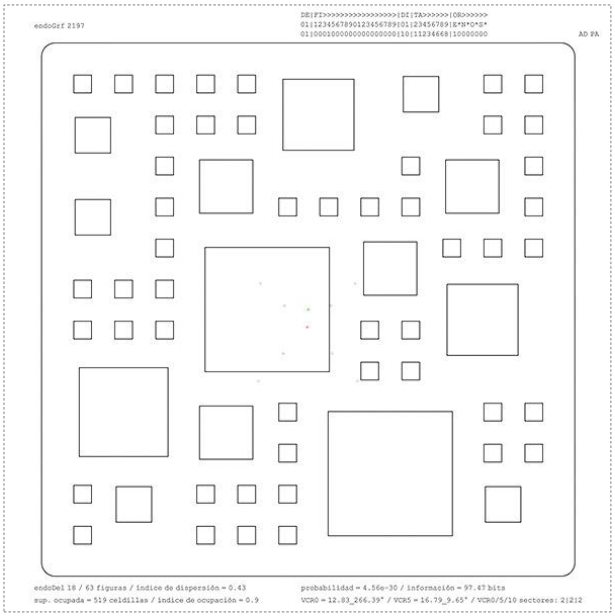
1. Upper Informational Captions

The upper informational captions of the image are generated, displaying the *data* from which the drawing was derived.

2. Occupancy Index

Calculated by dividing the number of occupied cells by the total available cells (always 576).





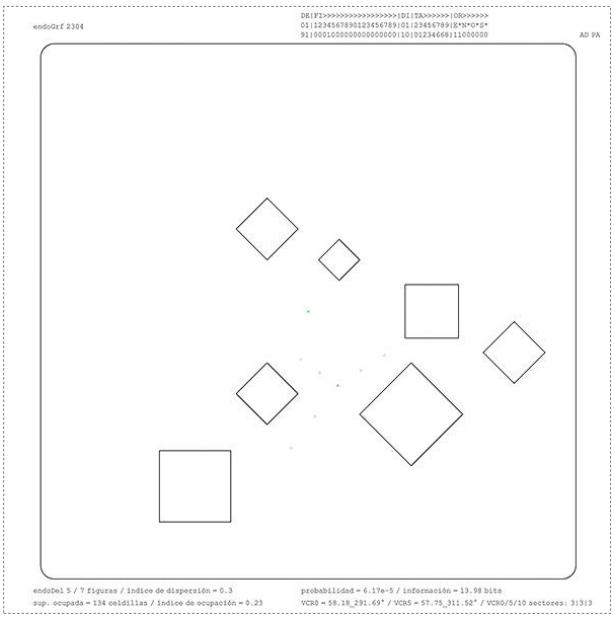
The drawing on the left shows a medium occupancy index, while the one on the right, an almost maximum one.

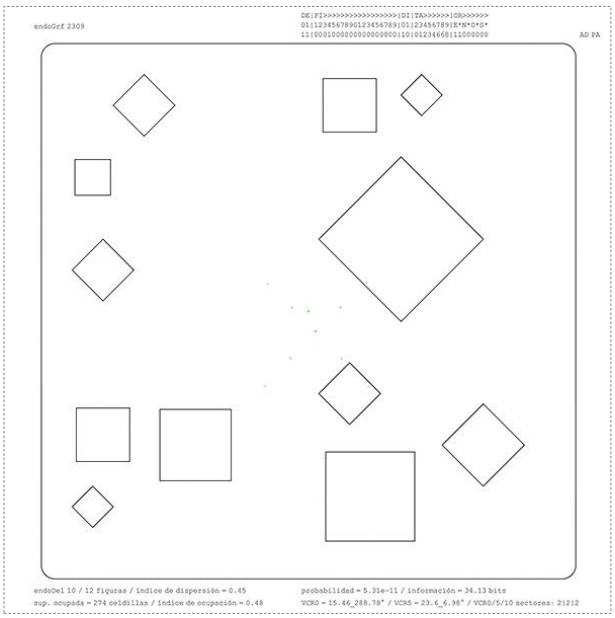
3. Dispersion Index

The distances between the centers of all figures must be calculated, followed by their average.

The *maximum absolute distance* is defined as the distance between two size-2 figures placed at opposite corners of the base: lower-left and upper-right, respectively.

The *dispersion index* is then calculated by dividing the average of the computed distances by the maximum absolute distance. Its value provides an indication of the general *spacing* of the figures, always within the limits of the base, relative to the arbitrarily chosen reference measure.





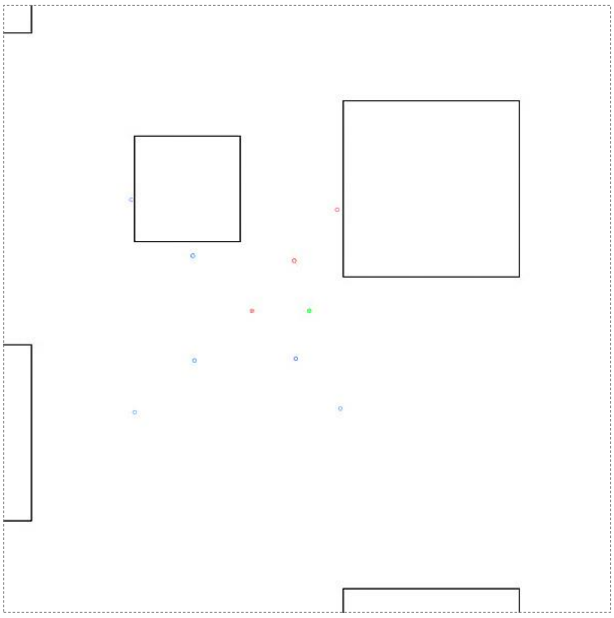
The drawing on the left, with its figures clustered closely together, shows a low dispersion index, while the one on the right, with its figures more widely spaced, shows a high one.

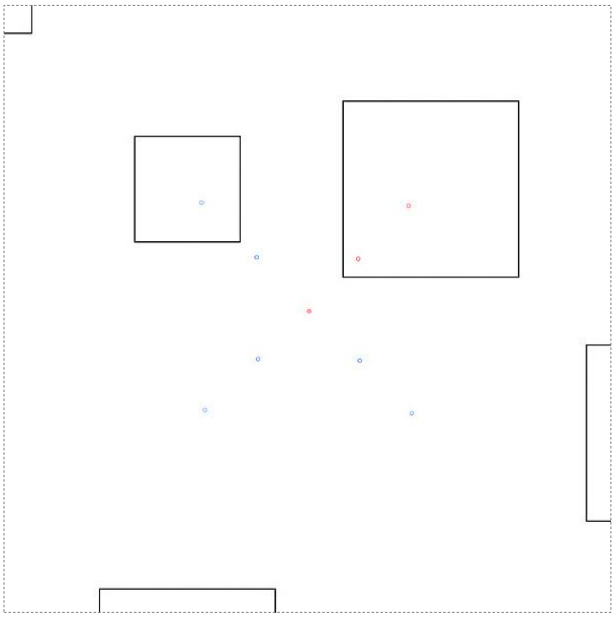
4. Reporting the Drawing's Cumulative Probability and Information

The associated probability is reported using scientific notation (a coefficient and an integer exponent), along with the total information. It should be noted that this is a *discrete memoryless source*, since the events (occurrence of a figure's shape class, etc.) are *independent*.

5. Evaluation of the Drawing's Balance According to Arnheim's Factors

At this point, I will say that the idea was to provide myself with a tool that offers an *additional* conceptual element to help me *decide* the *acceptability* of a given composition. This tool is based on an arbitrary *quantification* of the factors determining the visual weight of a figure, and of a drawing as a whole, derived from the dependencies observed by the German researcher Rudolf Arnheim in his *Art and Visual Perception*: namely, *size*, *distance* from the *center* of the artwork, *elevation* relative to the level considered zero, *lateralization*, etc. The convenient processing of all these factors, combined into a *center deviation* vector, provides me with immediate and intuitive visual indicators, overlaid on the rendered images of each drawing. The tip of the aforementioned vector is marked by a small red circle, while another small greenish circle indicates the center of the base. A quick glance is sufficient to perceive the degree of displacement of the former relative to the latter. The operation is then repeated using the same factors, but now arbitrarily weighted at 5% and 10%, each time marking another small circle in a lighter red. Subsequently, three additional arrangements are analyzed, with the artwork rotated 90° counterclockwise in each case. The tips of the vectors resulting from these rotated arrangements are displayed in light blue.





The drawing on the left shows an off-center, that is, unbalanced composition, while the one on the right is perfectly centered, or perfectly balanced.

6. Lower Informational Captions

The lower informational captions of the image are generated, displaying the *results* derived from the drawing's final configuration.

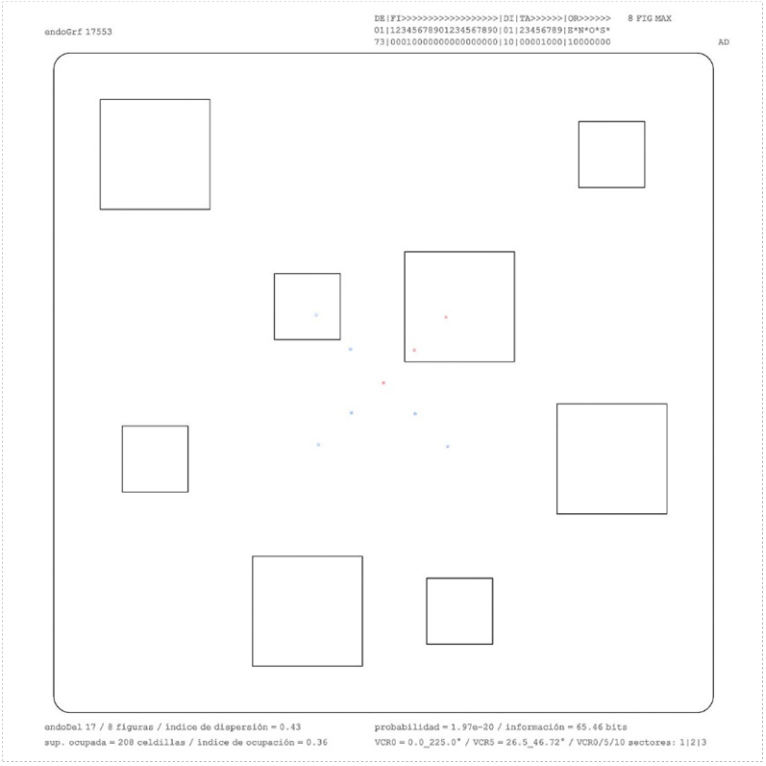
```
endoDel 17 / 8 figuras / índice de dispersión = 0.43          probabilidad = 1.97e-20 / información = 65.46 bits
sup. ocupada = 208 celdillas / índice de ocupación = 0.36     VCR0 = 0.0_225.0° / VCR5 = 26.5_46.72° / VCR0/5/10 sectores: 1|2|3
```

7. Writing the Mass Production File

All data and numerical results of the current drawing are written to the output file.

endoGrf#	DE	FI>>>>>>>>>>>>>>>>>>>	DI	TA>>>>>>	OR>>>>>>	5V	RP	fig	mapaDl#	iDis	sup	iOcu	manP	oMP	informa	modVE0	angVE0	modVE5	angVE5	s	s	s
00010530	73	0000000001000000000	10	11000000	10000000	AD	00	042	0000018	0.42	218	0.38	2.80	104	0343.99	027.77	207.72	010.00	162.64	2	2	2
00010531	73	0000000001000000000	10	11000000	10000000	AD	00	043	0000018	0.42	247	0.43	4.22	104	0343.40	000.76	206.57	023.68	044.05	1	2	2
00010532	73	0000000001000000000	10	11000000	10000000	AD	00	043	0000018	0.43	222	0.39	2.66	106	0350.71	010.02	151.86	025.76	070.77	2	2	3
00010533	73	0000000001000000000	10	11000000	10000000	AD	00	039	0000018	0.41	211	0.37	2.66	097	0320.81	004.43	138.81	025.49	056.47	1	2	3
00010534	73	0000000001000000000	10	11000000	10000000	AD	00	032	0000018	0.41	178	0.31	4.96	082	0270.09	015.87	232.43	009.65	030.06	2	2	2
00010535	41	0000000001000000000	10	11000000	10000000	AD	00	027	0000018	0.37	153	0.27	2.66	070	0231.12	015.50	284.53	021.49	004.61	2	2	2
00010536	41	0000000001000000000	10	11000000	10000000	AD	00	021	0000018	0.43	109	0.19	4.90	055	0180.41	028.73	330.46	046.94	006.61	2	2	3
00010537	11	0000000001000000000	10	11000000	10000000	AD	00	053	0000018	0.43	262	0.45	3.70	128	0423.32	012.53	084.21	038.09	057.81	2	2	3
00010538	73	0000000001000000000	10	11000000	10000000	AD	00	039	0000018	0.41	211	0.37	1.36	095	0315.14	008.09	304.51	026.09	027.88	2	2	3

Finally, a rendered JPG image is created for each generated drawing.



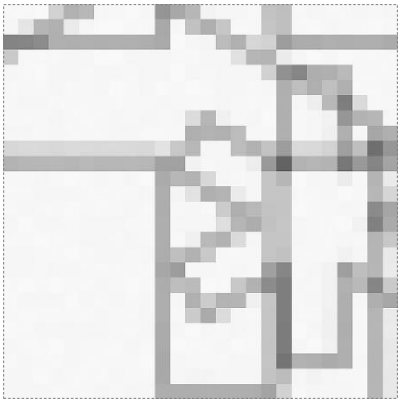
Boost Edition

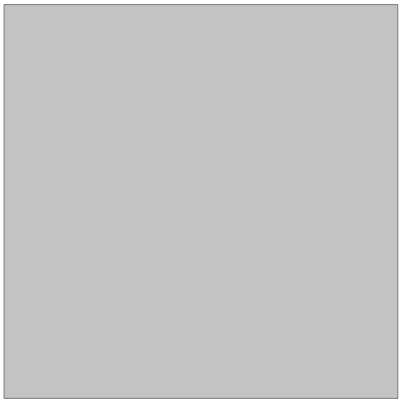
This is an "augmented" edition of the **endoGraficador**, which *automates* the execution of *multiple* productive sessions. Each session includes a certain number of drawings to produce and the configuration of its arrays, which are automatically adjusted based on a *planning* file that progressively details these features.

1. endoDelineador (Delineator)

This program, which occupies the first position in the functional scheme of the production cycle, is a direct derivative of the **endoGraficador**. The role of the **endoDelineador** is to generate small square geometric compositions, which later serve the **endoGraficador** as guiding lines for positioning the figures on the drawing plane.

Only the *shape*-class array is used here, along with a reduced *size* array (containing sizes 1 to 3 and excluding larger ones) and an *orientation* array, all equally probable. This program allows both figure *overlap* and placement *beyond* the boundaries of the contour. As a result, a delineation map typically consists of 3 figures, which may fall entirely or partially within the drawing plane, itself only 26 pixels wide.





The map on the left shows a typical composition. The central one is the gray map used to enforce the allocation a figure to any single cell; while the one on the right, an all-light gray map, lets chance "decide" its placement.

3. endoRetrazador (Retracer)

Each drawing produced by the **endoGraficador** undergoes my strict supervision, to discern any aesthetic quality it might contain (and that could convince me); and not many actually pass this test! So, what would be the point of maintaining an overwhelmingly vast collection of drawings—literally tens of thousands—most of them flawed, weak, or lacking even a minimal aesthetic quality? None at all. For this reason, I implemented the third program in my app suite. The **endoRetrazador** takes on the task of retracing the requested drawing, based on its data file number, and then—finally—saving it as a two-dimensional scene file.

4. endoColoreador (Colorizer)

This program determines the automatic *coloring* of any drawing produced by the **endoGraficador**, based on its numerical data file and the scene file previously generated by the **endoRetrazador**.

A. Initial Definitions

These four categories involved in the coloring processes must be specified: 1) colors; 2) palettes; 3) schemes; and 4) criteria.

1. Colors

The repertoire of *colors* available for subsequent palette formation is defined using the RGB triad. An *undefined* color may also be included, which can then be *randomly* determined during execution.

2. Palettes

Within the scope of this program, a *palette* is a set of four ordered colors; in programming terms, it takes the form of a 4-element array.

3. Schemes

Again, within the scope of this program, a *scheme* determines how the four colors of a palette are distributed.

There are two categories: 1) schemes that directly assign color *numbers*, used for *figure*-based coloring; and 2) schemes that assign *relative* or *weighted* values between colors, used for *area*-based coloring; where each array defines the *relative weight* of each color involved.

Color-*number* schemes can further be of 3 types, depending on whether the assignment is based on *shape* class, *size*, or *orientation*.

4. Criteria

A drawing is colored according to a specific distribution *criterion* applied to the up-to-4 colors.

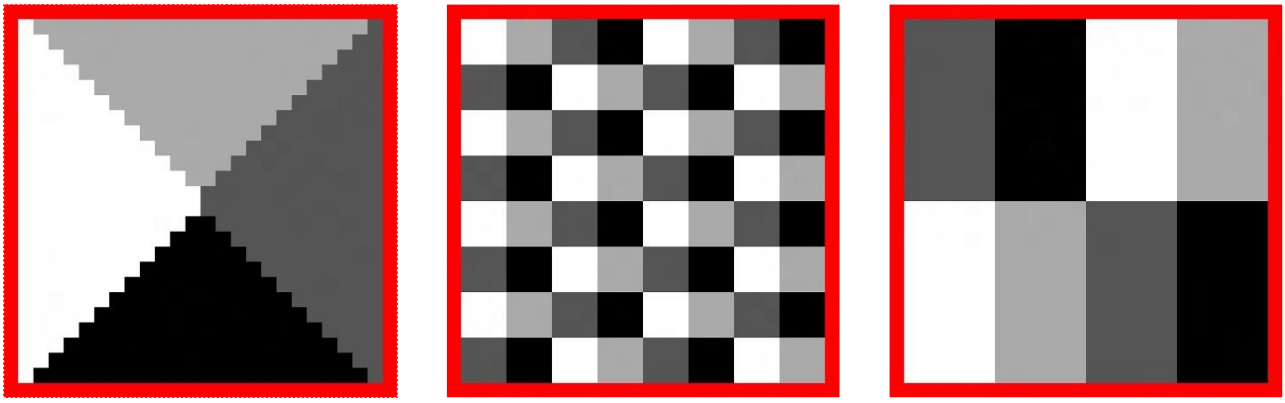
The coloring criteria implemented in the **endoColoreador** fall into these two categories: 1) by *areas* or 2) by *figures*.

B. Coloring Processes

1. Area-Based Coloring

In *area*-based coloring, the color assigned to a figure depends on the area of the drawing

that the figure belongs to. It is necessary to determine which of the predefined areas contains most—or, in some cases, all—of the figure. Once this area is identified, its corresponding color *number* is assigned to the figure. Below are three sample maps:

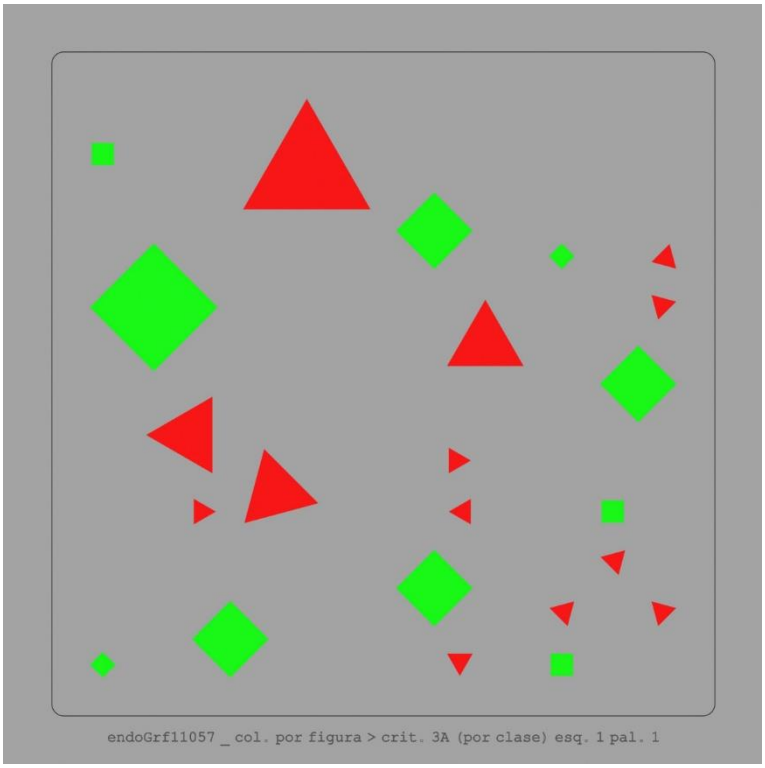


2. *Figure*-Based Coloring

Figure-based coloring operates according to five possibilities: 1) shape classes; 2) sizes; 3) orientations; 4) the four-color theorem; and 5) areas.

The general functioning of any of these criteria is relatively straightforward, as it relies on an array—called the *coloring* array—that stores the color distribution assigned to each figure.

In the particular case of the last criterion, based on approximate *equality of areas*, the algorithm manages a gradual color-assignment process aimed at achieving, as closely as possible, the area relationships dictated by the corresponding scheme.

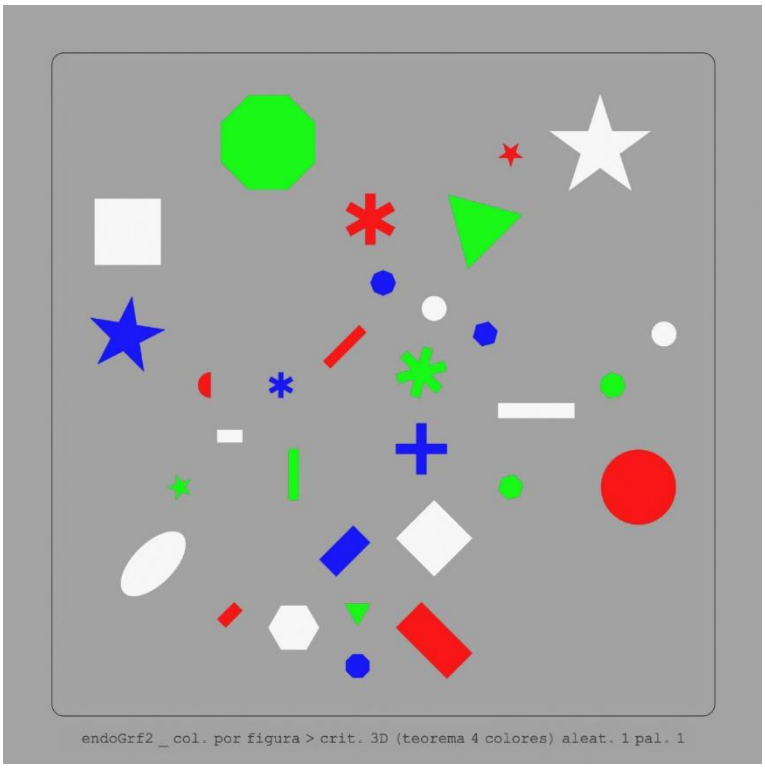


example of the application of the shape classes-based coloring criterion

3. Four-Color Theorem

The much-cited *four-color theorem*, deriving from *graph theory*, when applied to the present problem, solves the coloring of a drawing so that no adjacent figures share the same color.

The algorithm operates on an *adjacency matrix* derived from the data corresponding to the drawing under consideration. This matrix stores information on whether or not each figure is *adjacent* to every other figure. The computation proceeds starting with the figures that have the highest number of verified adjacencies. No further details are provided here, as doing so would require presenting somewhat intricate logical and computational arguments, likely obscure to most prospective readers—who, as intended, come from the visual arts.



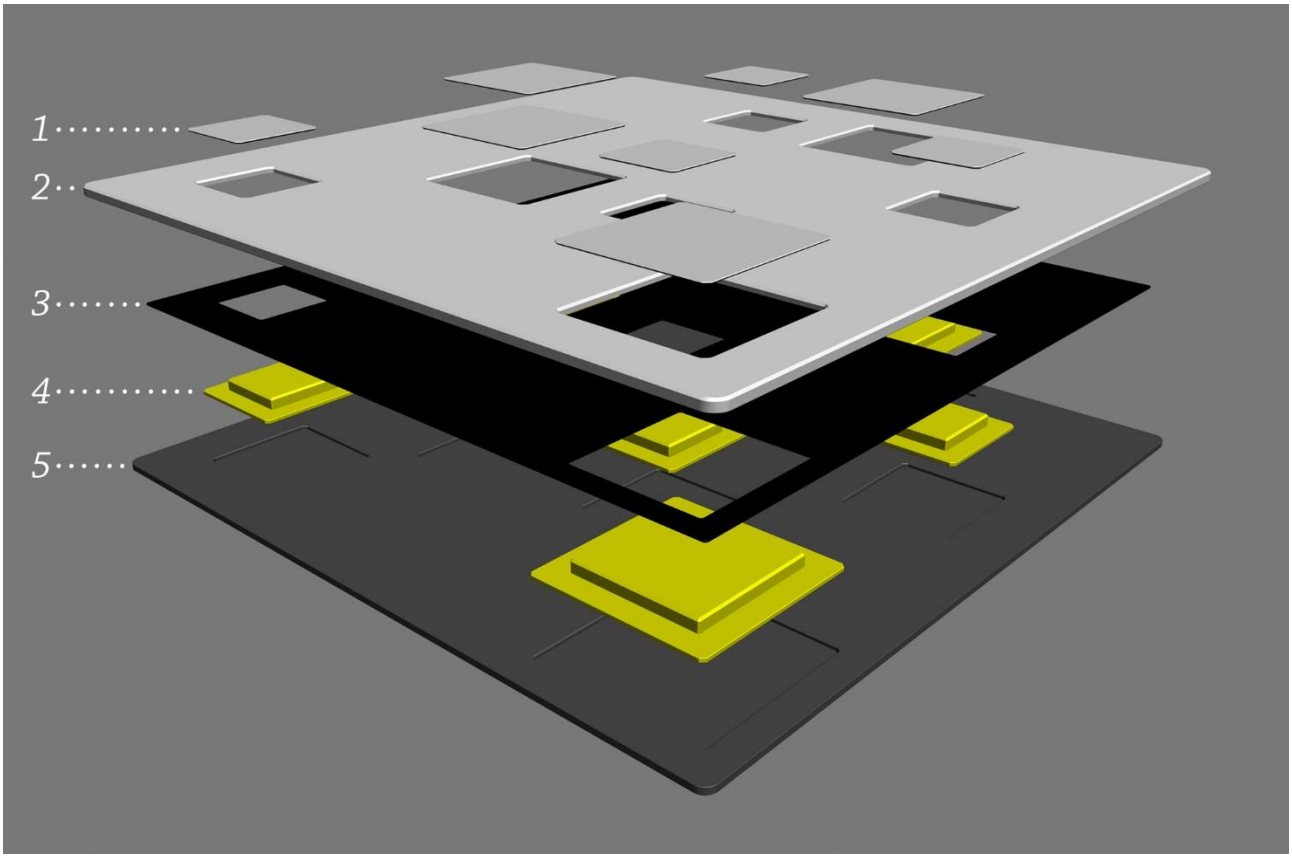
example of the application of the four-color theorem

To conclude, regardless of the processes requested, the results of each coloring produced are saved in a general TXT file created for this purpose; and a rendered JPG image is created for each coloring, featuring a lower caption displaying the results obtained.

5. endoConformador (Former)

The ultimate purpose of the *two-dimensional* designs generated by the **endoGraficador** lies in their subsequent conversion into a virtual *three-dimensional* artwork, either as a 3D model or as a simple still image providing a certain—let's say—plan view.

What does the finally formed artwork consist of? Five main components, arranged according to their spatial disposition or display, from front to back:

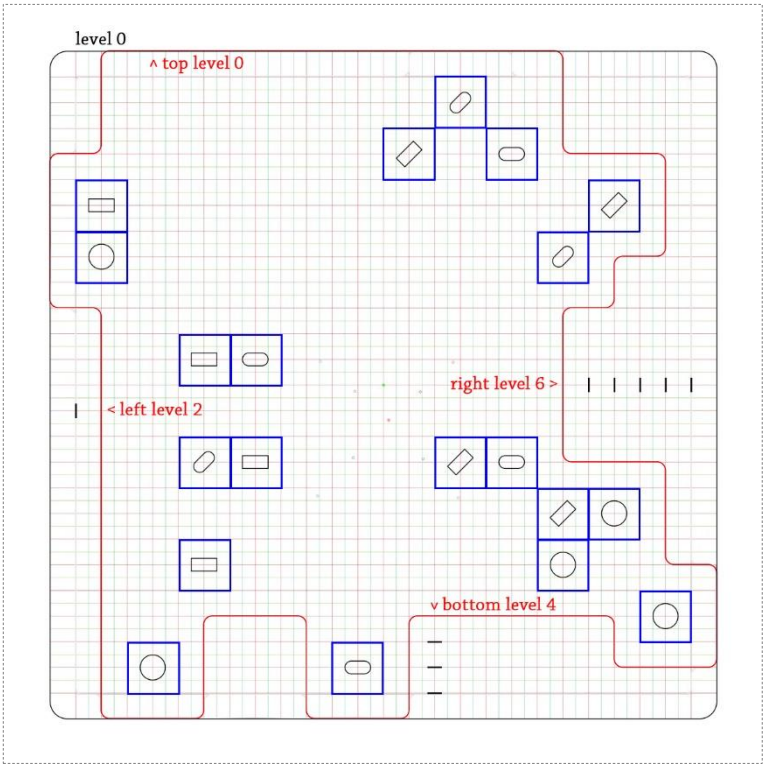


exploded-view of a formed endograph

1. A set of individual piece *covers*, initially concealed; that is, additional flat components to be used in the interactive artworks accessed through web browsers, intended to eventually cap the openings through which the individual pieces protrude from the front plate. This set of *covers* facilitates the so-called *reconfiguration* of the *innerrelief*.
2. A perforated *front plate*, through which the original figures of the drawing—now three-dimensionalized—protrude.
3. A barely visible *central sheet*, also perforated, providing a narrow *outline* to the individual pieces, acting as an intermediary between the front plate and them.
4. A set of *individual pieces*, into which the aforementioned figures are transformed through their three-dimensionalization.
5. A *rear cover*, on top of which the individual pieces, the central sheet, and the front plate are placed, functioning as the base of the artwork.

Accordingly, the **endoConformador** is the program which, within the **Los endoX** app suite, manages the automatic *three-dimensional forming* of a drawing produced with the **endoGraficador**, based on its numerical data file together with the scene file generated by the **endoRetrazador**. In short: a flat drawing goes in, a set of solids comes out.

Although this feature has not been employed to date, it should be noted that the program includes an eventual modification applicable to the *endograph* as it forms: its possible *irregular shaping*, or *madification*—as I chose to call it, in homage to the Madí group, who practiced it systematically.



heterogeneous external madification

6. endoGrabador (Engraver)

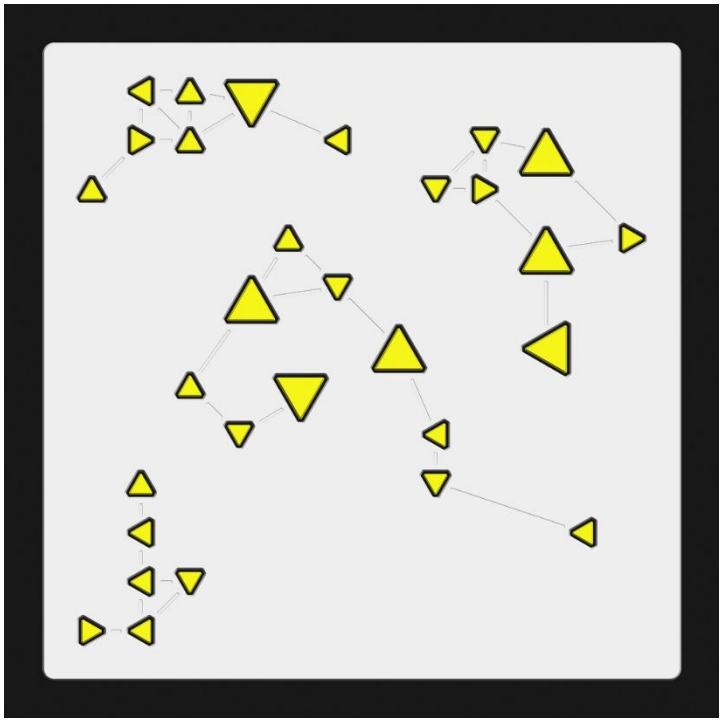
Once the artwork has been formed, its front plate and rear cover are *engraved*—acting as a *supplement* to the main composition, so as to provide not only a mid-range plane of appreciation defined by the layout of its individual pieces, but also a near-range plane that adds to the composition the incisions made on its front.

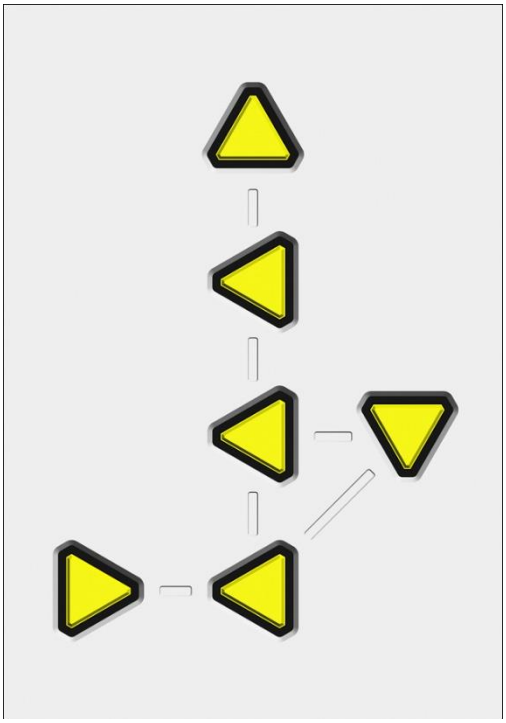
Engravings

The **endoGrabador** is precisely responsible for this automatic surface engraving, carried out across six specific sections:

1. Interconnection Lines

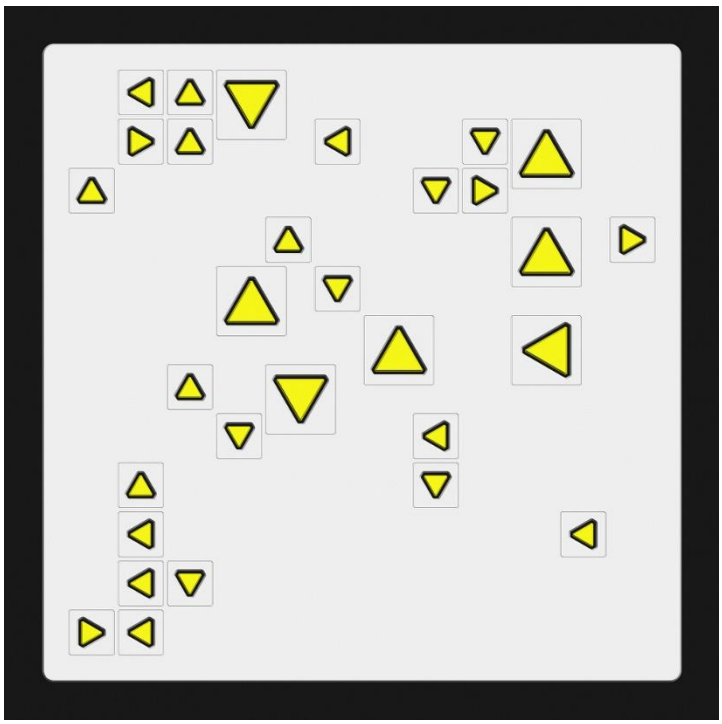
The engraving of a series of "lines" or channels—resembling an *asterism*—interconnecting the individual pieces is carried out as follows: first, the distances between the centers of all pieces are determined; next, the minimum distances are computed; these are then converted into interconnection rectangles, which are Boolean-cut through the front plate. Finally, after extruding the remaining material, a Boolean subtraction is performed on the front plate, thus obtaining its engraving.

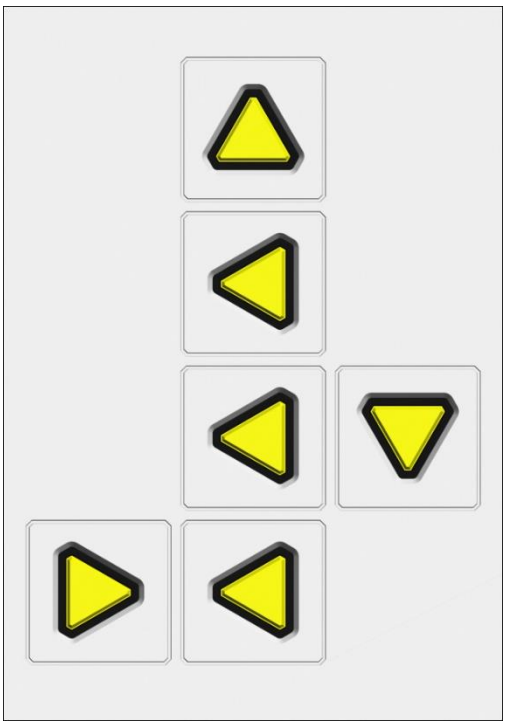




2. Demarcating Squares

Simply put, for each existing individual piece, two squares must be drawn corresponding to the size of the block occupied by the piece: an outer square and an inner square, which preserves the width of the groove to be engraved, useful for delimiting a *square ring*.

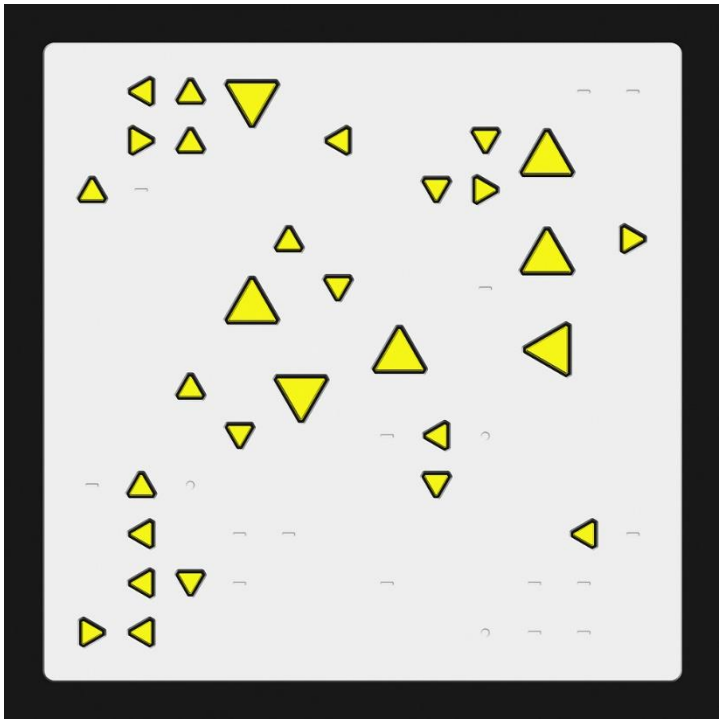


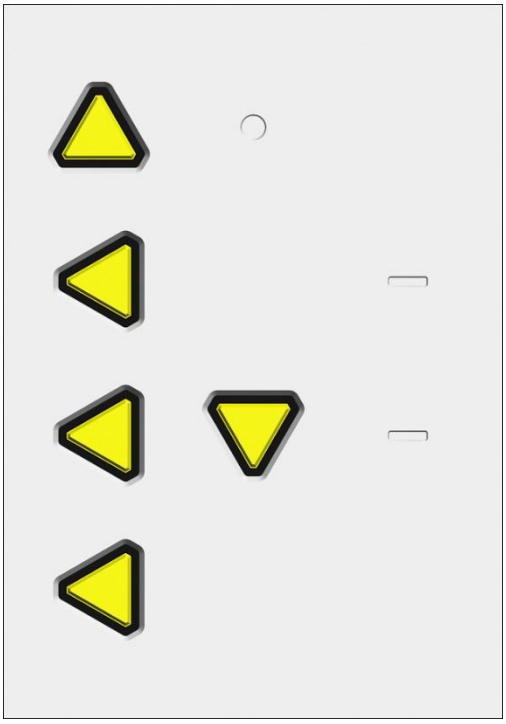


3. Constellation of Elementary Geometric Concavities

The engraving of a *constellation* of small elementary geometric shapes is performed on a 12 × 12 cell grid. By setup, either the *random* or the *deterministic* method is applied.

The first method *randomly* determines the number and shapes of concavities according to the total number of positions available for their placement, until the requested occupancy percentage is reached.





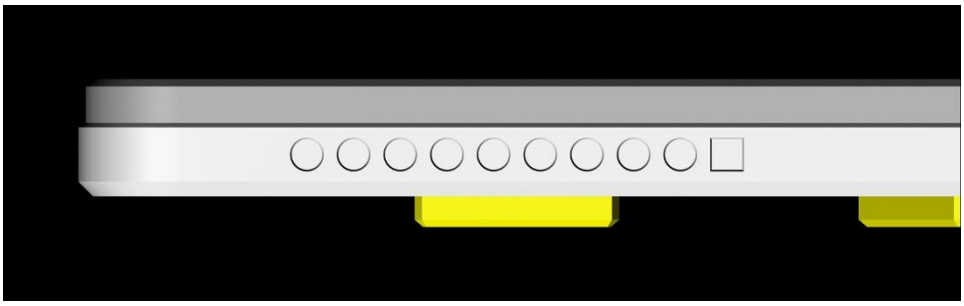
The *deterministic* method, which takes into account the distribution of individual pieces, allows for two versions: a *major* one and a *minor* one.

Major version of the deterministic constellation. To put it simply: first, a "map" (computationally, a matrix) is built representing the areas occupied by all individual pieces. Then, a second "map" is derived, obtained by rotating the first one 180°. Next, both are superimposed. Those areas of the second map that do *not* overlap with the first become part of a third, final "map"—the one used to place the concavities.

Minor version. This one, for the construction of the first "map," considers *only* the *upper-left cell* of each piece, thus reducing its area to the single block (of the 12 × 12 grid) it occupies. The result is a constellation populated by a limited number of concavities, and this version is the one commonly used.

4. innerrelief *Number in Binary Code*

The original front plate, which carries no engraving on its face, receives on its left edge a small engraving of the corresponding *innerrelief* number, represented in a 10-digit binary code where "0" is replaced by a *circle* and "1" by a *square*.



5. Triple Engraving of the Front Plate

The computation flow is redirected to the beginning of the program, where sections 1, 2, and 3 described above are processed again, in order to produce a front plate that partially incorporates interconnection lines, demarcating squares, and a reduced constellation of concavities—that is, a *mixed* version with all *three* types of engraving.

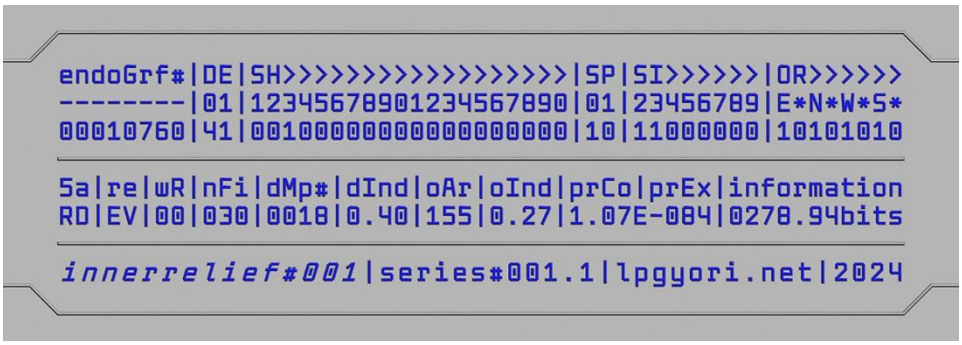
6. Informational Panel on the Rear Cover

On the visible side of the rear cover, a panel is engraved containing *data* and *results* associated with the *endograph* from which the current *innerrelief* derives, along with framing lines. The composition displayed by the individual pieces shows their numerical counterpart via this panel, which may be of interest to anyone wishing to delve into the preliminary settings and *quantitative* outcomes of the process that produced the artwork before them.

What is recorded in this panel?

On its first line: the number of the involved *endograph* and the 5 arrays used.

On the second line: the loading mode of these 5 array (DeTermined by the author / RAndom / MIxed [alternating DT and RA] / RD [Random, but applied only to the Density array]); if restriction of upper-left cells was applied during the drawing process (UNrestricted / EVen-restricted / ODd-restricted); if weighting recalculation was applied during the drawing process, and in that case, its type; the resulting number of figures; the delineation map used; the dispersion index; the occupied area measured in cells; the occupancy index; the drawing's associated probability (expressed as coefficient and exponent); and the information contributed by it, expressed in bits. For these last three, *recalculated* values replace the originals from the production file—those reflecting the evolution of the author's uncertainty *during* the drawing's generation rather than that of its current recipient.



Its final line displays: the artwork's title, its serial number, the author's signature, and the year of production.

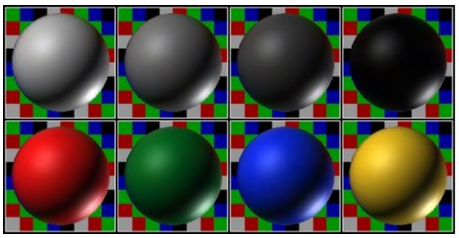
7. endoMaterializador (Materializer)

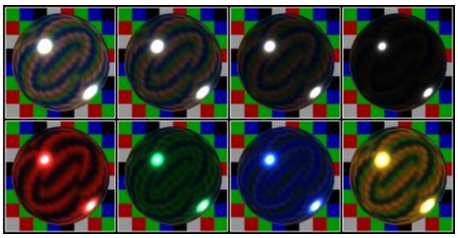
To complete the production cycle, it is required to assign *virtual materials* to the intervening objects, in a way that both takes advantage of the photorealistic qualities of the Chaos V-Ray rendering plug-in and enables real-time 3D rendering through WebGL in web browsers. The **endoMaterializador** automates this entire process.

Within the exclusive scope of the present work, *materialization* shall be understood as the process of applying, to all objects that constitute an *innerrelief*, the corresponding materials.

In a first execution of the program, materialization for V-Ray is enabled, while in a second one—by means of recursion—materialization is enabled for the subsequent export to glTF (GL Transmission Format) of the 3D model intended for the web.

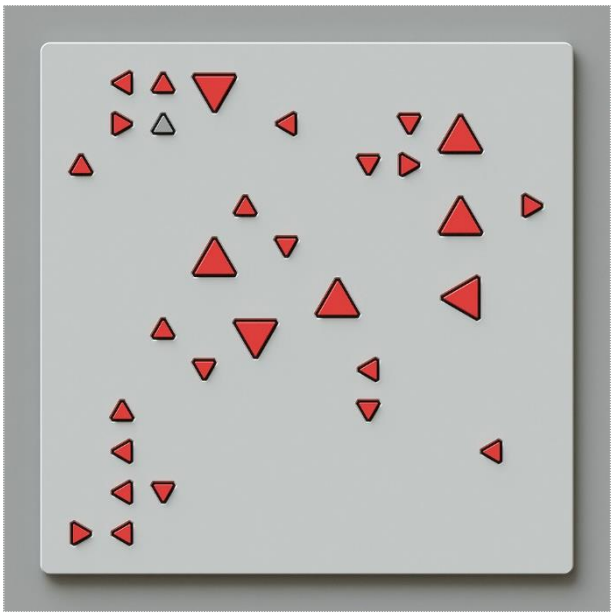
The definition of the various materials to be applied to the objects comes from two pre-established *libraries* created for this purpose: one with materials specific to V-Ray, and another with so-called *physical materials* intended for WebGL.

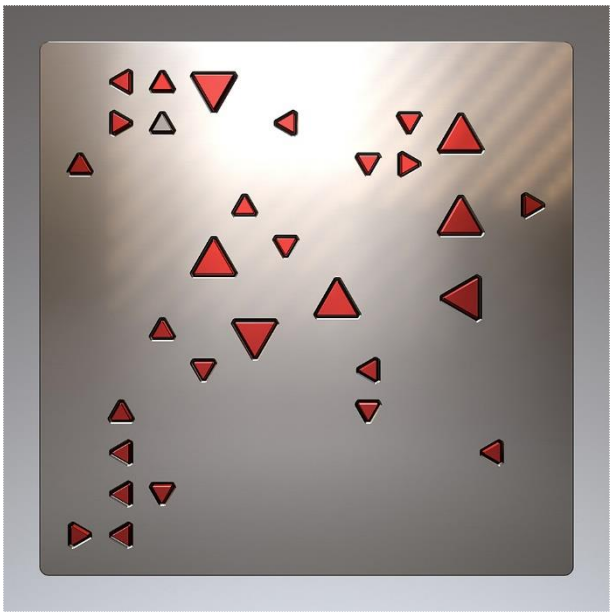




First, an auxiliary scene—lighting and camera—is loaded, according to the specific materialization being carried out.

The program applies the chosen coloration from the data file created by the **endoColoreador**.



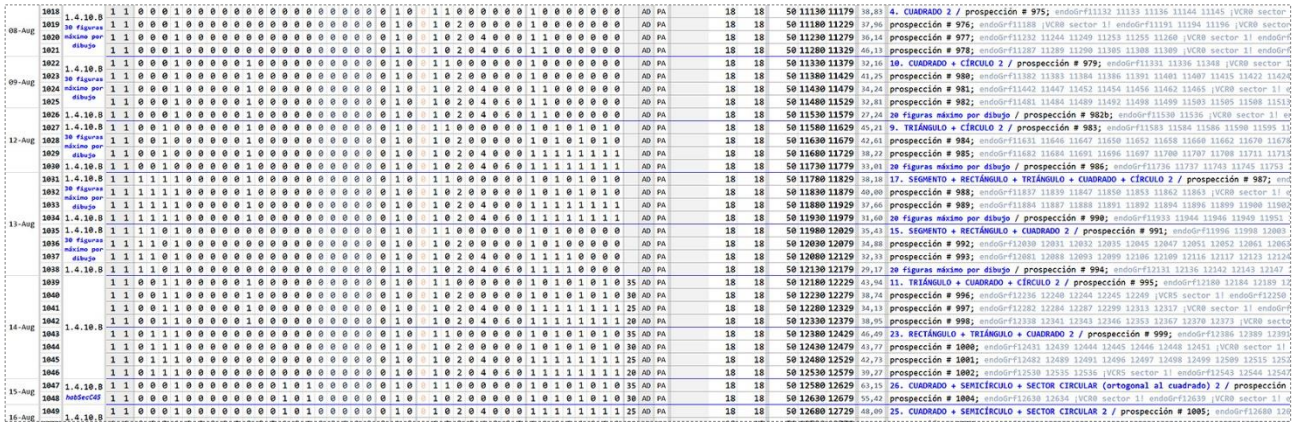


Based on the numerical series of colors assigned to the individual pieces, as provided by the coloration data file, the corresponding materials are applied to the objects, along with a texture mapping. The same procedure is carried out for the front plates present in the scene, the central sheet, the rear cover, and other objects.

Lastly, a V-Ray rendered JPG image of the materialized work is created. For the final preparation of the scene for web display, the covers are first grouped, followed by all objects composing the *innerrelief*. The 4 alternative front plates are hidden, and a pre-programmed animation is then loaded.

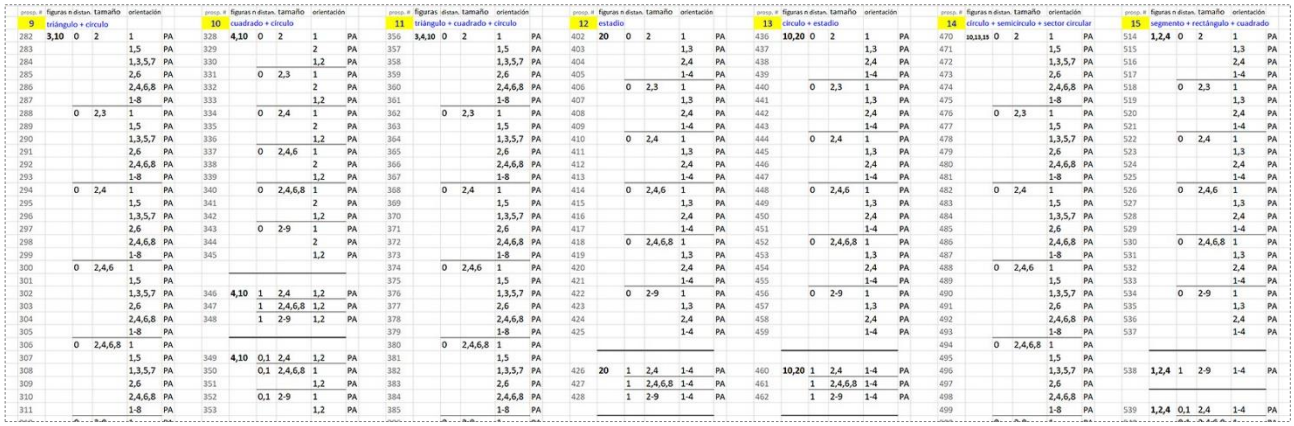
Appendix on Endographic Generation

Using the **Boost** edition of the **endoGraficador** and its planning files, during 2024 more than a thousand *prospective* sessions were carried out, based on a similar number of diverse configurations of the intervening arrays. *Prospecting* is used to denote the exploration of aesthetically valid possibilities that might emerge during each generative session.



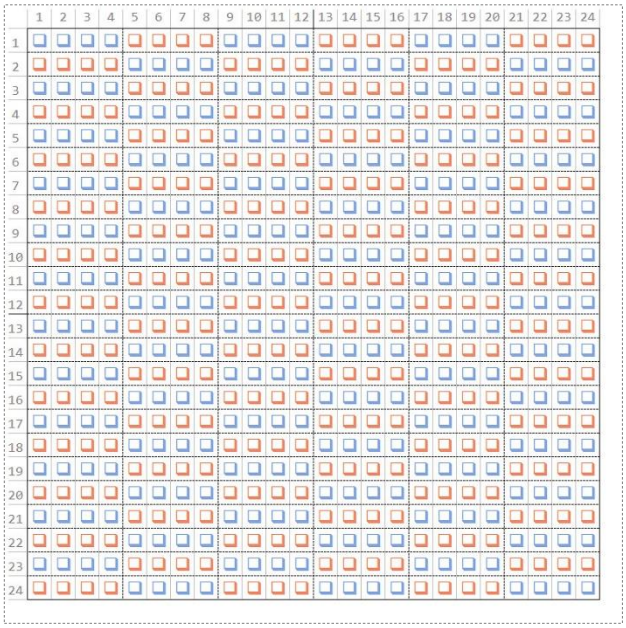
detail of the endographic production plan

The sequencing of these sessions followed an *endographic prospecting guide*, through which successive series of *endographs* were generated, mainly using 24 sets of geometric figures (ranging from a single class to five). Furthermore, several additional conditions were imposed for all of them: randomization of the *density* array, deactivation of weighting recalculation, use of a *probable* drawing-area delineation map, exclusion of preexisting figures, and even-restriction.



detail of the endographic prospecting guide

The general catalog—as already mentioned in the introduction—consists of a grid of 24 rows by 24 columns for the placement of, consequently, 576 *innerreliefs*.



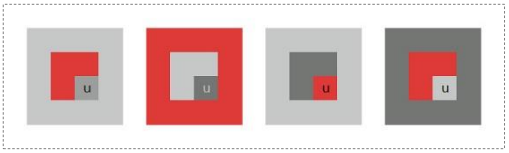
structural model of the catalog

The *innerreliefs* are distributed here into 144 *quadriptych* series, each containing four works conceptually linked by their set of shape classes and color palette. For instance, the *first* row of the catalog, with its half-dozen series, features triangles, squares, and circles, associated with the three additive colors—namely red, green, and blue—along with aluminum and dark gray, thus establishing the following chromatic combinatorial sequence:

chromatic distribution of the first row of the catalog

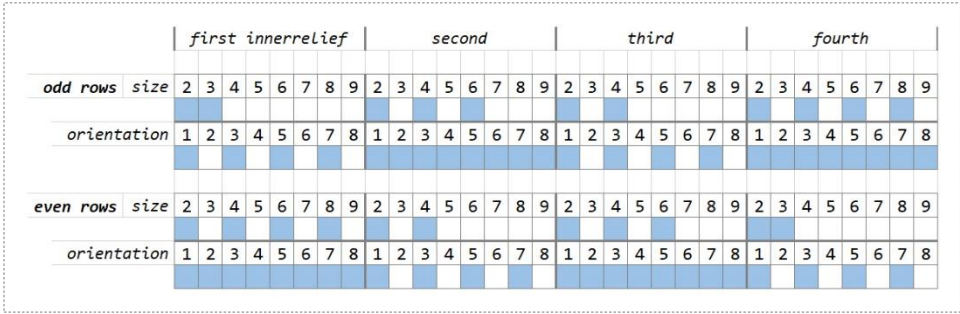
Where the color of the larger square indicates that of the front plate, and the smaller shape(s) indicate the color(s) of the individual pieces.

An additional detail is a small piece in an alternative color that serves as a *marker* for identifying the *ultimate* (i.e., last) minor figure located on the plane—that is, in the automatic generation of its *endograph*, the final position reached by the process—highlighting the culmination of its random flow and, by convention, the completion of the composition, even though in rare cases a larger piece may subsequently be added.



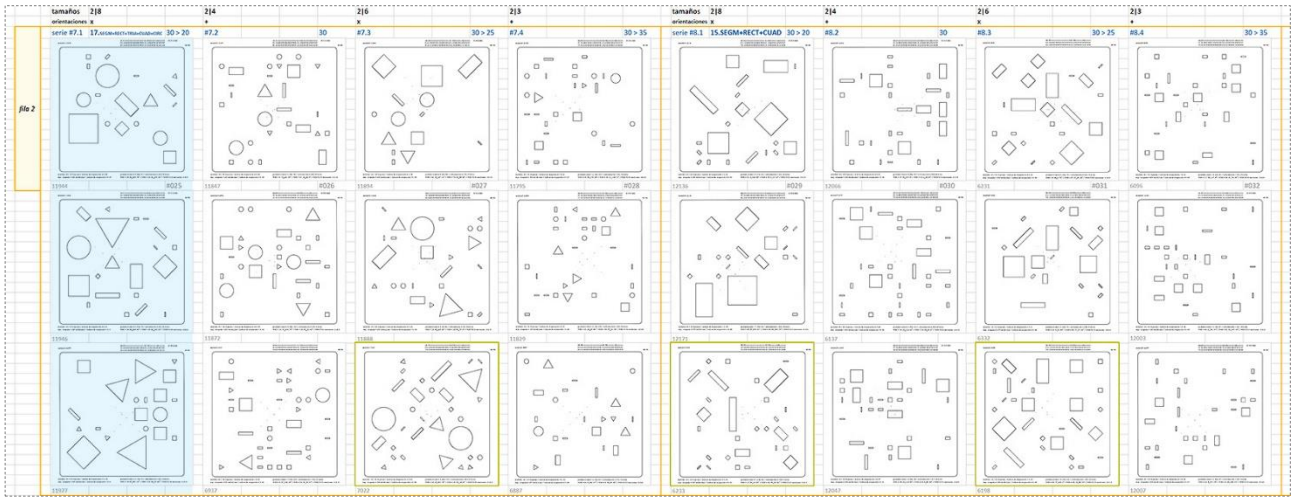
chromatic distribution of the first series

Each series also applies an arbitrary yet consistent ordering of the admissible sizes and orientations, as shown in the following illustration. Even-numbered rows display the inverted scheme of the odd-numbered ones.



distribution of sizes and orientations in odd-row and even-row series

After a preselection of the *endographs* massively obtained during the generative prospecting sessions, a *shortlist* emerges for each slot in the catalog, from which the final selection of the drawing representing that option is made.



detail of the general endographic catalog (a shortlist highlighted in light blue)

Once the *endograph* representing the corresponding position in the catalog has been selected, it enters the three-dimensionalization cycle provided by the rest of the **Los endoX** suite.

Appendix on Probability and Information

Within the specific scope of this work, the probabilistic calculation is carried out *with respect to the present author*, with respect to the person operating the program, in relation to *my* degree of uncertainty experienced as the entire generative drawing process unfolds.

So then… I run the **endoGraficador**. My productive session begins. What do I *expect* to happen during it? My *lack of knowledge* about what the screen will display… what questions will that prompt me to ask myself?

First, what do I have at my disposal? What *certainties* do I possess? Clearly, the configuration I established, concerning the five vectors where the repertoires of density, shape class, spacing, size, and orientation of the potential figures are stored, as well as all the primary variables and initial settings that define the repetition of the drawing process, the delineation map, and the activation of modes and services. In other words, I know in which direction things will move and which attributes will *approximately* characterize them.

And second, what *uncertainties* arise regarding the production that is about to take place? They are those reflected, for each figure the machine decides to draw—or decides *not* to draw—, in these five questions that I keep asking myself recursively as the session progresses, until it comes to an end:

1) Where will it position the figure?

2) Will it actually place it, or skip it?

3) If it does place it, which shape class will it select from the available repertoire?

4) What size will it have?

5) And finally, what orientation?

An additional issue—in this case, the sixth—would concern *spacing*, that is, the possible free cell around the figure. However, since this factor only affects the composition indirectly—because, if it occurs, it merely results in a reduction of the available space for the next figure-generation cycle, thereby decreasing the number of eligible cells—its impact will be reflected in the first probability calculation: the one associated with positioning.

Note on Color

My calculation of the probability of occurrence of the set of figures that make up a composition could be described as incomplete, since the computation of *color* is absent. And it is absent because there is no "consensus" (my own decision!) regarding the extent of the general palette… and there isn't because: how many colors should be included in that palette—three, six, ten, or…? Which colors should be chosen—only primary… and maybe secondary? Should black and white be included? And what about the grayscale range? What would its final gradation be? Or, perhaps, should I simply settle for the additive RGB triad? The list of possibilities could go on indefinitely.

In any case, the **endoGraficador**—where the probabilistic calculation takes place—does not handle colors, so I must resign myself: it makes a certain sense that they are not part of it!

Moreover, the focus of this probabilistic analysis was deliberately placed on the *formal* composition, providing an evaluation based *exclusively* on the geometric figures involved.

A. Calculation Concerning the Individual Attributes of Each Figure

Before proceeding, I wish to note the existence of two methods for calculating the total *Shannon information* contributed by a drawing composed of multiple figures:

a) By *probabilities*. The probability associated with each attribute of each figure is calculated and then *multiplied* cumulatively, after which the total information is determined.

b) By *information values*. The information contributed by each attribute of each figure is calculated and then *added* cumulatively until reaching that same total information.

I favor the second method, as it is mathematically more straightforward.

With this in mind, the first stage will involve calculating the *probabilities* and the *information* contributed *individually* by the *attributes* of the figure that has just been drawn, along with the concomitant values that determined its position on the drawing area.

1. Regarding Its Positioning

A first issue arises from calculating the probability associated with the spatial availability in which each figure is to be *positioned*. To do this, the number of *eligible* cells at the start of each assignment cycle is determined. An eligible cell is any cell *available* for placing a figure's upper-left cell. Since, at the beginning of the figure-assignment loop, all potentially available space will be accessible—not only that explicitly enabled by the delineation map or by the alternative even- and odd-restriction processes, but also the space determined by the minimum size set in the size array—and because, for this reason, uncertainty will be greater, the first figure to be assigned will have the lowest probability value, meaning it will contribute the greatest information associated with its positioning. As the drawing

becomes increasingly populated, reducing the available space and, consequently, the possible assignment positions, this probability will begin to rise, and, conversely, the information will decrease. It is evident: if there are many options for positioning, uncertainty is higher; if there are few, it is lower.

The *probability* of the *positioning* is inversely proportional to the number of eligible cells:

$$p_p = 1/\text{eligible\_cells}$$

The *information* contributed by the *positioning* is given by:

$$I_p = \log_2(1/p_p) = -\log_2 p_p \text{ [bits]}$$

2. Regarding Its Actual Assignment

Once the position is determined and available, will a figure actually be assigned to it or not? The second issue arises from calculating the probability associated with the degree of uncertainty regarding whether a figure is assigned. Logically, if the decision is imposed by the delineation map—either to enforce the assignment (*mandatory* drawing area) or to enforce *non*-assignment (*forbidden* drawing area)—and since this map is known to this author, there is no uncertainty; the associated probability will be equal to 1, and the information contributed, to 0.

Conversely, for any other *probable* drawing area on the delineation map, where *chance* decides whether a figure is assigned or not, the associated probability must be calculated based on the *density* vector—and its two values corresponding to the only possible states: assignment or non-assignment.

Therefore, the *probability* of the actual *assignment* of a figure can be expressed as the ratio between the value corresponding to the actual assignment and the sum of the two values (assignment and non-assignment), both derived from the *density* vector:

$$p_a = \text{assignment}/(\text{non-assignment} + \text{assignment})$$

The *information* contributed by the actual *assignment* is given by:

$$I_a = \log_2(1/p_a) = -\log_2 p_a \text{ [bits]}$$

At this point, some additional concepts need to be considered... Beyond the limitations imposed by the respective arrays, and since I intend to understand what influence the available space may have on the figure's *shape* class, *size*, and *orientation* attributes, I will ask myself three more questions:

3. Regarding Its Shape Class

Is there any limitation, in terms of available space, on the figure's *shape* class? There is not! The available space has no impact on the shape class. Therefore, the calculation of the

probability associated with it will be limited to computing the values contained in the corresponding array.

Consequently, the *probability* of the resulting *shape* class is calculated as the ratio between its relative value and the sum of all relative values stored in the *shape* class array:

$$p_c = \text{resulting\_shape\_class}/\text{sum\_of\_all\_shape\_classes}$$

If it is a preexisting figure, its probability will be assigned a value equal to 1, since it is an element imposed in advance with *no uncertainty*, given that I myself requested its inclusion in the drawing.

Subsequently, the *information* contributed by the *shape* class is calculated:

$$I_c = \log_2(1/p_c) = -\log_2 p_c \text{ [bits]}$$

4. Regarding Its Size

Is there any limitation, in terms of available space, on the figure's *size*? Yes, there is! The available space does influence the figure's size. Therefore, to calculate this probability, the values of the minimum and maximum available space—determined when evaluating the available space—will be used to establish the valid range among all the sizes contained in the *size* array.

Consequently, the *probability* of the resulting *size* is calculated as the ratio between its relative value and the sum of all relative values stored in the *size* array that belong to the valid range:

$$p_s = \text{resulting\_size}/\text{sum\_of\_all\_valid\_sizes}$$

Likewise, if it is a preexisting figure, its probability will be assigned a value equal to 1, since it is an element imposed in advance.

Subsequently, the *information* contributed by the *size* is calculated:

$$I_s = \log_2(1/p_s) = -\log_2 p_s \text{ [bits]}$$

5. Regarding Its Orientation

Finally, is there any limitation, in terms of available space, on the figure's *orientation*? There is not! The available space has no impact on the orientation.

However, the type of "behavior" that each figure's shape class exhibits in relation to orientation—or rotation—does affect the calculation of its probability.

When calculating the *probability* associated with orientation, the following premise must be taken into account: different shape classes of geometric figures behave *differently* under rotation. A few examples illustrate this point. The circle is not affected by rotation. The square is not either... provided it undergoes rotations that are multiples of a right angle.

The triangle, on the other hand, shows a clear difference each time it is rotated, as it has an arrow-like feature.

The various implemented shape classes, depending on their behavior with respect to orientation (and its eight possible alternatives), can be classified into four types:

Type 1. Shows 8 different dispositions; *no* coinciding dispositions under rotation. Examples: triangle, pentagon, heptagon, pentagram, semicircle, circular sector, C-profile, L-profile, and T-profile.

Type 2. Shows 4 different dispositions. The east orientation coincides with the west; northeast with southwest; north with south; and northwest with southeast. Examples: line segment, rectangle, hexagon, ellipse, six-pointed asterisk, I-profile, and stadium.

Type 3. Shows 2 different dispositions. East, north, west, and south orientations coincide; likewise, northeast, northwest, southwest, and southeast coincide. Examples: square and cross.

Type 4. Shows 1 single disposition; all 8 possible orientations coincide. Examples: octagon and circle.

As can be seen, each shape class displays a *different* effective total number of possible orientations, and this number directly affects the calculation of the associated probability.

Once this distinction is made, the *probability* of the resulting *orientation* is calculated as the ratio between the sum of the relative values of those orientations that produce a disposition of the figure similar to that of the resulting orientation, and the sum of all relative values stored in the *orientation* array:

$$p_o = \text{sum\_of\_similar\_orientations}/\text{sum\_of\_all\_orientations}$$

The *information* contributed by the *orientation* is then:

$$I_o = \log_2(1/p_o) = -\log_2 p_o \text{ [bits]}$$

For example, if the orientation array were: (2, 1, 2, 1, 2, 1, 2, 1), which gives double weight to the four orthogonal orientations compared to the oblique ones, and the resulting shape class were the square—a type-3 figure—, then, since its east, north, west and south orientations coincide, the relative value for orthogonality will be the sum of the corresponding relative values, namely 8. Similarly, the northeast, northwest, southwest and southeast orientations also coincide; therefore, the relative value for obliquity will be the sum of the other corresponding relative values, namely 4. Ultimately, the array is reduced to the ordered pair (8, 4), yielding a probability of 2/3 for any orthogonal orientation of the square, and a probability of 1/3 for any oblique orientation.

B. Subsequent Calculations

1. Calculation of the Figure's Probability and Information

Since the *probability* of occurrence of a *figure* is equal to the *product* of the five previously calculated probabilities:

$$p_{fig} = p_p \times p_a \times p_c \times p_s \times p_o$$

Similarly, the *information* in bits contributed by a *figure* is equal to the *sum* of the five previously calculated information values:

$$I_{fig} = I_p + I_a + I_c + I_s + I_o$$

2. Calculation of the Drawing's Probability and Information

Since the *probability* of occurrence of a *drawing* is equal to the *product* of the individual probabilities of the *n* figures that compose it:

$$p_{draw} = p_{fig1} \times p_{fig2} \times \ldots \times p_{fig\mathit{n}}$$

Finally, the *information* in bits contributed by the *drawing* is calculated as the *sum* of the individual information values contributed by the *n* figures:

$$I_{draw} = I_{fig1} + I_{fig2} + \ldots + I_{fig\mathit{n}}$$

3. On Total Information

One final theoretical observation. The calculation of information described above considered only the *generative* process of the artwork—the series of successive choices (positioning, shape class, size, orientation) carried out by the relevant program, which resulted in its original composition. However, once generated, the artwork became capable of accessing a second level—let us call it—of *variability*, provided by the interactive 3D system. This allows us to add, to the already-completed stage of its genesis, another, *performative*—so to speak—layer, from which it is also possible to deduce attributes that contribute additional information.

As mentioned in the last section of the ***Installation and User Guide***, all *innerreliefs* allow for the *co-creation* of a very large number of visual *variants* through four resources: adjustment of their orientation (which offers eight alternatives), selection of their front plate (five alternatives), reconfiguration of their *n* internal components, and modification of their coloration (*m* alternatives). The associated probability is therefore equal to:

$$p_{per} = 1/(8 \times 5 \times 2^{n} \times m)$$

Each of the aforementioned resources introduces new discrete alternatives which, from the perspective of Shannon's information theory, correspond to independent choices and therefore contribute additional information:

$$I_{per} = \log_2(1/p_{per}) = -\log_2 p_{per} \text{ [bits]}$$

To the *generative* information (I_{draw}), we now add this latest, *performative* component, in order to quantify the *total* informational potential that each *innerrelief* carries throughout its interactive digital existence. In conclusion, the artwork conveys not only the information associated with its elementary digital existence, but also that which arises from the interactive process of its display:

$$I_{total} = I_{draw} + I_{per}$$

C. An Example

The method implemented in the **endoGraficador** is—as noted previously—both *sequential* and *conditional*. Sequential, because the process advances step by step, placing one figure after another in a defined order. Therefore, the probability of the entire set is not calculated all at once, but *progressively*, as the drawing develops. Conditional, because each step *depends* on the current state of the drawing—the figures already placed, the remaining available space, the positions still eligible, etc. The probability of the next figure is calculated *conditionally* with respect to what has already been drawn.

The following will present this procedure for calculating probability and information for a pair of figures in a sample drawing. To do so, we start from their initial configuration and then explain the entire sequence for obtaining each of the involved quantities.

The chosen initial configuration of the five arrays is as follows:

Density array = (1, 1). The probability of both non-assignment and actual assignment is 50%.

Shape class array = (0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0). The probabilities of obtaining a rectangle, triangle, square, and circle are all 25%.

Spacing array = (1, 1). The probability of null or unit spacing is 50%.

Size array = (0, 0, 0, 2, 0, 3, 0, 5, 0). The probability of obtaining size 4 is 2/(2 + 3 + 5), i.e., 20%; size 6 is 30%; and size 8 is 50%.

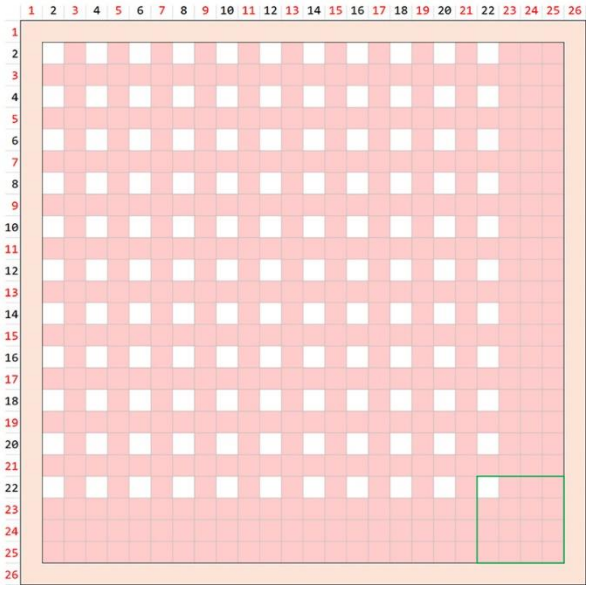
Orientation array = (1, 0, 1, 0, 1, 0, 1, 0). The probability of obtaining any of the four cardinal-point orientations is 25%.

For the production of the drawing, the delineation map used consists of a single *probable* drawing area (known internally as #18), that is, one in which the positioning of the figure

is left entirely to *chance*—in other words, one that ensures there is no predefined delineation known in advance.

Even-restriction was requested, meaning that the upper-left cell of each figure must be placed in cells whose row and column values are *even* numbers only.

As an important point, it should be noted that the previous configuration imposes position *restrictions* on the drawing area. The following illustration shows the eligible cells (in white), with all other cells forbidden for placing a figure's upper-left cell.



Given the size limitation established by the corresponding array (with the smallest size being 4), the last admissible row and column are both 22. Note that, consequently, the last admissible figure (green box), of size 4, whose upper-left cell would be positioned there, would extend from rows 22 to 25 and from columns 22 to 25 as well.

Due to the even-restriction, only 9 additional rows remain eligible between rows 2 and 22, giving a total of 11 rows eligible for placing a figure's upper-left cell; the same applies to the columns. At the start of the drawing process, the total number of eligible cells for assigning a figure's upper-left cell amounts to 11 × 11 = 121 cells.

What follows is a step-by-step calculation, as executed by the **endoGraficador**:

Cycle #1

The random attribute generation produced the following results:

Upper-left cell: row = 4; column = 10

Density = 1, meaning the figure is assigned

Shape class = 4, i.e., a *square*

Spacing = 1 cell

Available space: minimum = 2 cells; maximum = 9 cells

Size = 4

Orientation = 3, i.e., *north*

Consequently…

Since this is the first figure to be placed on an empty area, and given the even-restriction, the 121 eligible cells calculated above were available, the *probability* of the *positioning* is:

$$p_{p1} = 1/\text{eligible\_cells} = 1/121 = 0{,}00826$$

The *information* contributed by the *positioning* is:

$$I_{p1} = -\log_2 p_{p1} = -\log_2 0{,}00826 = 6{,}919 \text{ bits}$$

Now, since this appendix aims to illustrate a generative process that *has already taken place*—an automatic drawing generation process in which a series of geometric figures *has already been determined*, using a sequential method as detailed here—this reproduction *contains no non-assignments*; in other words, each cycle *always* results in the effective assignment of a figure. For this reason, as there is no longer any uncertainty for the reader regarding figure assignment, unlike the uncertainty I faced during the actual random generation process with the program running, the *probability* of effective figure *assignment* is and will remain equal to 1:

$$p_{a1} = 1$$

Consequently, the *information* contributed by the effective figure *assignment* is zero:

$$I_{a1} = -\log_2 p_{a1} = -\log_2 1 = 0 \text{ bit}$$

The *probability* of the resulting *shape* class is:

$$p_{c1} = \text{resulting\_shape\_class}/\text{sum\_of\_all\_shape\_classes} = 1/(1 + 1 + 1 + 1) = 1/4 = 0{,}25$$

The *information* contributed by the *shape* class is:

$$I_{c1} = -\log_2 p_{c1} = -\log_2 0{,}25 = 2 \text{ bits}$$

The *probability* of the resulting *size* is:

$$p_{s1} = \text{resulting\_size}/\text{sum\_of\_all\_valid\_sizes} = 2/(2 + 3 + 5) = 2/10 = 0{,}2$$

The *information* contributed by the *size* is:

$$I_{s1} = -\log_2 p_{s1} = -\log_2 0{,}2 = 2{,}322 \text{ bits}$$

Probability of the resulting *orientation*. Since the four available orientations do not affect the disposition of the square—always appearing in its orthogonal disposition—all four available orientations can be considered equivalent for it:

$$p_{o1} = \text{sum\_of\_similar/sum\_of\_all} = (1 + 1 + 1 + 1)/(1 + 1 + 1 + 1) = 4/4 = 1$$

The *information* contributed by the *orientation* is zero:

$$I_{o1} = -\log_2 p_{o1} = -\log_2 1 = 0 \text{ bits}$$

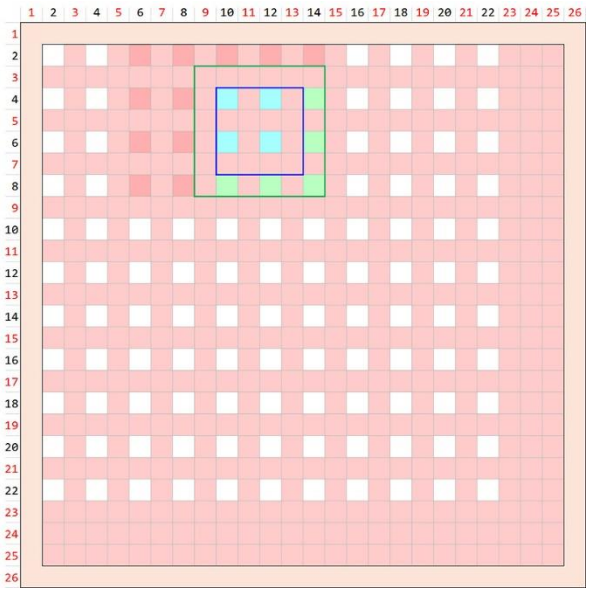
Consequently, the *probability* of occurrence of this first figure is equal to the *product* of its five previously calculated probabilities:

$$p_{fig1} = p_{p1} \times p_{a1} \times p_{c1} \times p_{s1} \times p_{o1} = 0{,}00826 \times 1 \times 0{,}25 \times 0{,}2 \times 1 = 0{,}000413$$

Similarly, the *information* contributed by it is equal to the *sum* of its five previously calculated information values:

$$I_{fig1} = I_{p1} + I_{a1} + I_{c1} + I_{s1} + I_{o1} = 6{,}919 \text{ bits} + 0 \text{ bit} + 2 \text{ bits} + 2{,}322 \text{ bits} + 0 \text{ bits} = 11{,}241 \text{ bits}$$

It should be noted that, for the next cycle, the number of eligible cells has been reduced. As shown in the illustration below, the new figure (blue box) occupies 4 eligible cells (light blue). In addition, because the spacing resulted in 1 cell, a total of 5 eligible cells (light green) are now allocated to it. Since the smallest figure size is 4, 11 additional cells are excluded (light red), leaving the number of eligible cells as: 121 – 4 – 5 – 11 = 101.



Cycle #2

The random attribute generation produced the following results:

Upper-left cell: row = 18; column = 18

Density = 1, meaning the figure is assigned

Shape class = 3, i.e., a *triangle*

Spacing = 0 cells

Available space: minimum = 2 cells; maximum = 9 cells

Size = 8

Orientation = 7, i.e., *south*

Consequently…

The *probability* of the *positioning* is:

$$p_{p2} = 1/\text{eligible\_cells} = 1/101 = 0{,}0099$$

The *information* contributed by the *positioning* is:

$$I_{p2} = -\log_2 p_{p2} = -\log_2 0{,}0099 = 6{,}658 \text{ bits}$$

The *probability* of effective figure *assignment* is:

$$p_{a2} = 1$$

The *information* contributed by the effective figure *assignment* is:

$$I_{a2} = -\log_2 p_{a2} = -\log_2 1 = 0 \text{ bit}$$

The *probability* of the resulting *shape* class is:

$$p_{c2} = \text{resulting\_shape\_class/sum\_of\_all\_shape\_classes} = 1/(1 + 1 + 1 + 1) = 1/4 = 0{,}25$$

The *information* contributed by the *shape* class is:

$$I_{c2} = -\log_2 p_{c2} = -\log_2 0{,}25 = 2 \text{ bits}$$

The *probability* of the resulting *size* is:

$$p_{s2} = \text{resulting\_size/sum\_of\_all\_valid\_sizes} = 5/(2 + 3 + 5) = 5/10 = 0{,}5$$

The *information* contributed by the *size* is:

$$I_{s2} = -\log_2 p_{s2} = -\log_2 0{,}5 = 1 \text{ bit}$$

Probability of the resulting *orientation*. The dispositions of the triangle are indeed affected by its orientation; therefore, the available orientations cannot be considered equivalent:

$$p_{o2} = \text{sum\_of\_similar/sum\_of\_all} = 1/(1 + 1 + 1 + 1) = 1/4 = 0{,}25$$

The *information* contributed by the *orientation* is:

$$I_{o2} = -\log_2 p_{o2} = -\log_2 0{,}25 = 2 \text{ bits}$$

Consequently, the *probability* of occurrence of this second figure is equal to the *product* of its five previously calculated probabilities:

$$p_{fig2} = p_{p2} \times p_{a2} \times p_{c2} \times p_{s2} \times p_{o2} = 0{,}0099 \times 1 \times 0{,}25 \times 0{,}5 \times 0{,}25 = 0{,}000309$$

Similarly, the *information* contributed by it is equal to the *sum* of its five previously calculated information values:

$$I_{fig2} = I_{p2} + I_{a2} + I_{c2} + I_{s2} + I_{o2} = 6{,}658 \text{ bits} + 0 \text{ bit} + 2 \text{ bits} + 1 \text{ bit} + 2 \text{ bits} = 11{,}658 \text{ bits}$$

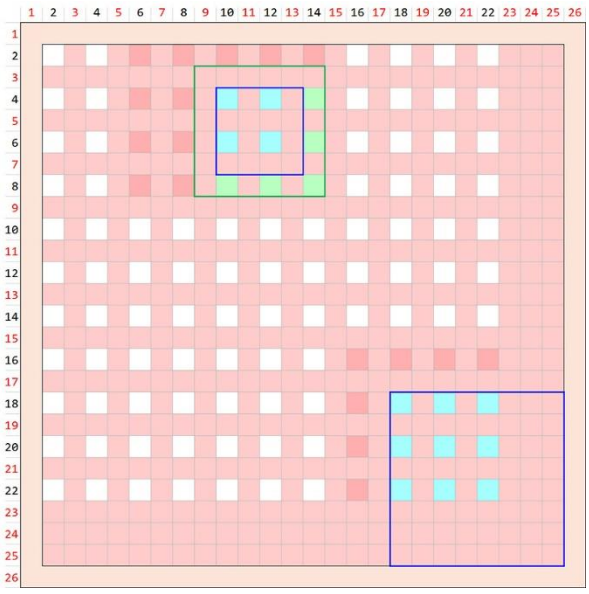
Next, the *accumulated* probability and information of the *drawing* can be computed. The *probability* of occurrence of our drawing composed of the two figures above is:

$$p_{draw} = p_{fig1} \times p_{fig2} = 0{,}000413 \times 0{,}000309 = 1{,}27617 \times 10^{-7}$$

Finally, the *information* contributed by the drawing is the sum of the individual values contributed by its two figures:

$$I_{draw} = I_{fig1} + I_{fig2} = 11{,}241 \text{ bits} + 11{,}658 \text{ bits} = 22{,}899 \text{ bits}$$

If this procedure were to continue, a third cycle would have an even smaller number of free cells eligible to accommodate its figure. Once again, as shown in the illustration below, the new figure occupies 9 eligible cells. Since the spacing turned out to be zero, no cells are allocated to it. Since the smallest figure size is always 4, 7 additional cells are excluded, leaving the number of eligible cells as: 101 – 9 – 7 = 85.



The probability of the positioning will increase as the generation of the drawing advances, because with each cycle the available space decreases, thereby reducing the uncertainty regarding the position that the next figure may occupy.

* * *

document version 1 2026-03-05