EUROGRAPHICS 2021
H. Rushmeier and K. Bühler
(Guest Editors)

*Volume 40 (2021), Number 2*
*STAR – State of The Art Report*

# WT: A Survey of Control Mechanisms for Creative Pattern Generation

Name[1,2] and Name[2]

[1]Place
[2]Place

## 1. Analysis of the State of the Art

The contribution of this survey is twofold. On the one hand, its analysis the state of the art with regard to control mechanisms, translates this categorization to overall control paradigms and discusses their means for creative control. On the other hand, it selects the work not by its underlying algorithms and creation techniques but by its design goal, namely creative pattern generation. This type of pattern generation includes a variation of representative creation challenges, such as combining ordered fillings and repetitive structures with individual frames or lining and highlighting components that might break with an underlying repetitive structure. The focus on the visual output instead of specific underlying algorithms allows for a novel and unifying discussion of techniques and merges a discussion of work that is traditionally studied separately. Thus this focus furthers a common understanding of creative control mechanisms.

The analysis of the control mechanisms is directly taken from the authors' descriptions. However, the following discussion of the creative means has an interpretative nature to it. Despite our best efforts to give a clear reasoning, we also agree that this is not always unambiguously manageable. Thus, this discussion should be viewed as a step toward an objective discussion about terms such as artist-usable and creatively controllable as well as a more realistic usage of these terms.

### 1.1. Commonly Used Control Mechanisms

**Minor:** Add intro

#### 1.1.1. Example-Based Control

On of the most prominently investigated techniques for texturing are example-based approaches. Example-based control mechanisms compute a separate output based on a given example and provide a *goal-oriented control*. The motivation behind using these techniques is mainly to generate a specific and predictable output as efficiently as possible. Example-based and inverse approaches have a long history in the control of procedural representations. They remain a dominant research field and are relevant for any discussion about controlling procedural models.

In regard to creative control, example-based approaches detach the design task to a data-driven image generation techniques, such as taking a photograph or designing a sample in an application such as Adobe Photoshop or Illustrator.

Relevant factors for differentiating example-based techniques are the size of the design space, hence their expressiveness, performance and initialization requirements. The following investigation is roughly sorted by increasing expressiveness.

#### 1.1.2. Shapes and Masks

The most basic control requirement is to define an area to be filled. Methods that focus on the development of novel underlying pattern systems with no acknowledgment of control mechanisms, also need to define the space to fill and a relationship of the system to that space. Many approaches take the idea of simply outlining a space further and carefully design growth constraints, offer masking and the sketching of areas to be filled, thus leading to complex designs.

#### 1.1.3. Vector Fields

Fields constitute a powerful tool for combining an automatic procedural filling by individually designing regions on the canvas. In the context of two-dimensional creative pattern generation, these fields are usually vector fields. The streamlines of a field can create curves as part of the pattern that fill and structure a space. The design of a vector field requires less manual work than the manual creation of curves. Other global design choices, such as an overall growth direction or the alignment of elements, are simple to translate from a vector field to procedural generation rules.

#### 1.1.4. Curves, Sketches and Painting

Curves and hand-drawn paths give an artist more direct control than the previously discussed methods to fill a space. In addition to the visual output being further constrained, the control is put onto the actual canvas. Curves are needed for tasks such as creating an decorative frame or structuring the space. Some techniques consider the whole curve before computing the ornament, optimizing the filling of the curve based on certain design goals, enabling a form of global planning.

Painting-tool-like methods create output along curves but do so

directly without taking an a priori completed curve into consideration, as if using a spray can or a brush. Painting techniques usually include a brush diameter, hence the size of the area to be filled along the curve.

Besides the value of the indirect use of curves as a control tool, their direct employment as a visual element is also relevant. Formed curves, such as circles, spirals or hearts, are essential components for many pattern designs such as ornamentation.

### 1.1.5. Element Placement

The placement of single elements onto the canvas maximizes artist control and is on its own a trivial data-driven control principle. However, in combination with procedural modeling, this mechanism becomes interesting. Separately placed elements that do not follow any rules should be integrated and processed to remain part of the underlying global scene structure. Even though this functionality can be compared to using the tip of a brush, paint-like procedural modeling techniques often have a more spray-can-like quality [MM12] and do not include this option.

For creative pattern generation, this type of variation is needed for preventing a monotonous, texture-like output. The placement of single elements as highlights should visually break the underlying order of repetition, while still being a homogeneous part of the global layout. The following work presents steps toward this demanding goal by integrating the placement of single elements into a global control mechanism.

**Minor:** Lead over to the next section

## 1.2. Design Features

**Minor:** Add intro

### 1.2.1. Distribution and Repetition

The generation of repetitive pattern designs and an overall distribution of elements are understood as texturing methods. Texture generation mainly focuses on creating a repetitive and homogeneous pattern as automatically as possible. These methods usually provide only parts of the design space and controllability needed for creative pattern generation. They are applicable for the subparts with a texture-like quality to it, such as background regions and fillings. But as the investigation of procedural and data-driven texturing has been the driving force behind the development of procedural representations in general, it produced manifold approaches and noteworthy control mechanisms, which we analyze in the following.

**1.2.1.1. Stochastic Pattern** For procedural texture generation, stochastic textures have been the foundation of both research investigations and many complex models. Stochastic textures are generated with noise functions, and LAGAE et al. [LLC*10] present the state of the art for work before the year 2010. In terms of the controllability of the textures, the authors identify three main approaches. First, the indirect access to the noise through the control of the power spectrum. Second, the direct access to its appearance through function parameters, and third, example-based techniques. Also, in the context of stochastic pattern, the control of a model often directly derives from a new model definition, and the focus of the related work is usually the latter.

**1.2.1.1.1. Example-Based Control** In addition to performance and input requirements as common characteristics, the expressiveness of stochastic textures can be split into representations that approximate a Gaussian texture and textures including global structures [GLM17; LLC*10].

In summary, example-based stochastic texturing techniques with no further artist input are presented by the following techniques. LAGAE et al. [LVLD10] match noise bandwidths for isotropic multi-resolution noise with the performance described as “rapid”, given by GILET et al. [GDG12a] as a few milliseconds. GALERNE et al. [GLM17] introduced an efficient sparse convolution noise based on textons. The example match takes a couple of seconds.

GALERNE et al. [GLLD12] present a bandwidth-quantized Gabor noise matched by estimating the power spectrum of the exemplar through its decomposition into a sparse sum of Gaussians. Their fitting performance is about 2 minutes per texture with no input in addition to the exemplar. The noise can be further adjusted with an interactive visual editor in which the power spectrum of the noise is represented by individually modifiable sets of Gaussians. Layers can be rotated, scaled, translated and cloned. Due to the abstract nature of the visual features of a power spectrum (which is used in the editor) and the for artists not directly intuitive connection between a power spectrum and the visual features of the noise, the editor has a strong explorative nature to it. However, as the editing itself is interactive and visually appealing, it is inviting to do so.

By introducing a noise that permits for the approximation of arbitrary spectral energy distributions, GILET et al. [GDG12a] increase the expressiveness of their model toward more structural texture designs. A straightforward noise by example computation takes up to 20 seconds, depending on the number of artist-defined convolution noises. For greater control and expressiveness, a perturbation function and a multi-layer approach are presented. The perturbation, as well as positions for different texture configurations can be additional artist-defined image maps. Further pursuing the topic of greater expressiveness and a more structured noise, GILET et al. [GSV*14] introduced a local random phase noise. Artist-controlled parameters control the visual quality of the noise and the amount of structure in comparison to noise. The authors do not report performance times for the matching step. PAVIE et al. [PGDG16] also focus on extending the expressiveness of noise-based representations and argue for control mechanisms being more intuitive in the spatial domain instead of the commonly used editing of the power spectrum and aligning local random phase noise on a regular grid with a spot noise model based on a random distribution of structured kernels. The artist has interactive control of the spatial structures by modifying the spot functions and their distribution, thus increasing the range of possible designs.

GUINGO et al. [GSDC17] base their work on an underlying novel noise model and a separate handling of structures with a

bilayered setup. Their method improves spatial variation and visual quality in comparison to other methods. Artists need to adjust two parameters, the number of different random patterns in the input and the size of the local spectra weighting faithfulness to spatial variability of the exemplar. The performance of matching a $512 \times 512$ input image can take up to 1 hour (with the current implementation not parallelized). KANG et al. [KH17] decompose the power spectrum of an input image into so-called "feature" and "non-feature" parts. Non-features are obtained by a noise-by-example method. Feature parts, such as edges, can be edited in the feature image and are combined with the noise based on a artist-controlled ratio. For the procedural representation of the feature parts, the authors employ data-driven tiling. The feature extraction for a $257 \times 257$ input image, and therefore the texture matching, ranges from few seconds to 2 minutes. GILET et al. [GD10] apply a more general optimization strategy for choosing the parameters of a noise-based procedure. With the help of the artist estimating the light source direction in the input, GILET et al. [GD10] can create displacement map textures, with the parameter computation taking from 1 to 3 hours. With a given rough approximation of the geometry and choosing a representative pattern patch in the input, even volumetric representations can be created from the exemplar.

#### 1.2.1.2. Regular to Irregular Pattern

**1.2.1.2.1. Example-Based Control** All the above discussed noise-based methods control a single stochastic procedural model. Even though recent advances greatly increase their expressiveness, the design space of noise-based models is too limited for creative pattern generation. In addition to methods dealing with generalizable stochastic models, methods employ procedural textures optimized for specific design goals. These textures can potentially be visually more complex to meet the requirements of their intended task. For brick and wood textures, the early work of LEFEBVRE et al. [LP00] presents an example-based control by transferring specific measured properties of an input to corresponding parameters for the procedural representation. The algorithm takes a suitable reference image, a binary mask, and the texture class as input and produces results for these two structural texture types. The authors describe the matching performance from a few minutes up to an hour. [GDG12b] focus on the interactive creation of procedural semi-structured texture models and also handle the control of visual features. With an improved point distribution function that can consider hierarchical spatial relationships, random variations of statistical shape models are generated from artist input. In order to do so, an artist needs to give multiple exemplary object distributions.

BOURQUE et al. [BD04] allow for the whole procedural texture spectrum with their parameter retrieval technique. In so doing, they employ two types of similarity metrics, and two types of optimization strategies. As input, an artist needs to individually select the distance metric and optimization strategy for each fitting task. As initialization for the optimization, the authors propose "on the order of 200" pre-computed random choices to choose from. The authors report an average optimization time of 12 minutes, not specifying for how many parameters. GILET et al. [GDG12a] report more than an hour for the performance times. For such a search-based approach, the parameter count is highly influential on the performance for both visual quality and computation time. With a higher number of parameters the current form of the approach quickly becomes unfeasible.

Todo: Add [GKHF14; TWY*20; GAD*20]

Drawing Tiles:

Todo: Add [BWL18; GAD*20; DSJ19b]

### 1.2.2. Rule-based and Design-Specific Pattern

**Minor:** Add intro

**1.2.2.1. Shapes and Masks** The following section discusses techniques that consider global shapes and masks.

WONG et al. [WZS98] introduced a programmable procedural system that employs a greedy rule-based strategy to generate floral ornaments. A procedural model is created with artist-defined elements and with a set of growth rules that handle the selection, appearance and connections of elements. The process iterates, finding tentative places for elements by testing them against constraints in the procedural model and, where suitable, placing elements in the found spaces, optionally and connecting them to existing elements. Possible ornament designs are technically restricted only by this iterative creation logic. All adjustments to the design and layout of an ornament have to be done by writing code, with the exception that a “region specification” for the filling can be given. The authors do not report any performance times.

SANTONI et al. [SP16] present the procedural generation of tangles, which are repetitive black-and-white hand-drawn patterns made from dots, straight lines, simple curves and circles. Tangle elements usually align to the shape they fill, for example, by outlining it. A stochastic group grammar with grouping, geometric and decorative operators composites recursive patterns at different scales, filling two-dimensional shapes as well as handling holes. A tangle generation usually takes a few seconds, with a complex example taking about 3 minutes. The authors demonstrate the applicability of their method with an interactive system based on a parameterized artist interface, including history navigation, rule re-expansion and sketch-based operator modification. A user study evaluates the system as accurate, controllable and easy to use after a reasonable training time.

LOI et al. [LHVT17] present a procedural framework for a large variety of element texture designs. The authors aim for designs that are unrelated to their spatial location and the space they fill, calling it stationary. Their programmable method is developed for technical artists and requires programming expertise. Generating pattern scripts are built with partitioning, mapping and merging operators. These operators enable both global and local design control and the composition of designs. The operator-based technique would enable a node-based interface design, which is not explicitly demonstrated in the article. The execution time for most designs is a few seconds, with some examples taking more than 1 minute. A user study with technical artists carefully evaluates the system’s scripting interface, concluding positive results overall.

**1.2.2.1.1. Probabilistic Interference** Other systems provide global outlining shape control on procedural processes by interpreting the modeling task as a probabilistic inference problem.

TALTON et al. [TLL*11] present for grammar-based procedural models, as example for their flexible analytic objective functions, non code-based global controls through image and volume matching. The authors stress that in principle any control mechanism can be matched with any grammar through their decoupling of the growth control from the grammar itself. The authors discuss that to come to the desired design goal, some experimentation might be needed, making the approach less transparent. Performance depends on the complexity of the grammar and the number of optimization steps needed. The authors report performance times ranging from a few seconds to several hours. For their examples, the authors manually terminated the optimization iteration.

RITCHIE et al. [RMGH15] controlled rule-based hierarchical and iterative procedural models similar to TALTON et al. [TLL*11] with image-based matching and target volumes. The authors present a sequential Monte Carlo variant that is able to score incomplete model states, thus improving convergence behavior and final scores. The reported performances range from around 3 seconds to 12 minutes, and the authors show that the number of included primitives scales reasonably. RITCHIE et al. [RTHG16] make use of machine learning to improve the performance of the image-matching grammar-based models of RITCHIE et al. [RMGH15]. The updated system increases performance up to 10 times by integrating a neural network and sampling a learned constraint-satisfying approximation. Reported performances are overall below 3 seconds. Interactive performance is the foundation of all creative control means and hence of great importance.

**1.2.2.1.2. Grammar Generation** Grammars are a classical procedural representation. Grammar-based output can be designed through the generating grammar, the included visual elements and often custom-made parameters for visual features. To translate a desired visual output to an abstract grammar is a daunting task for most artists, and first efforts have been made to provide an example-based technique for the grammar generation itself.

ŠT'AVA et al. [ŠBM*10] present a context-free L-System that is able to recreate a given two-dimensional vector image consisting of groups of line segments. The algorithm creates similarity groups of these basic elements, computes spatial relationship clusters and iteratively translates these into rules. An artist is required to define a similarity threshold and significance weights for the different clusters, such as element distance or similarity, for example, thus guiding their representation according to the L-system rules. The time needed for the inverse step, depending on the number of elements in the input, is reported to range from a few seconds up to 20 minutes. TALTON et al. [TYK*12] further generalize the idea of inverse grammar generation and interpret it as a probabilistic interference problem. Their system induces a probabilistic formal grammar from a hierarchy of labeled components with a Bayesian model merging technique.

**1.2.2.2. Vector Fields** LI et al. [LBZ*11] present a shape grammar that is guided by either a vector or tensor field. The field can influence the grammar's translation command, potentially leading to globally pronounced structures. The field can furthermore guide rotation, scaling, and color parameters. The artist can specify a priori field constraints, such as regular and singular field elements, on the surface to be filled. Once the field is computed, local Laplacian smoothing can be applied. The authors report a synthesis performance for geometric surfaces from less than a second up to 3 minutes.

**1.2.2.3. Sketch-based Control** Curves and hand-drawn paths give an artist more direct control than the previously discussed methods to fill a space. In addition to the visual output being further constrained, the control is put onto the actual canvas. Curves are needed for tasks such as creating an decorative frame or structuring the space. Some techniques consider the whole curve before computing the ornament, optimizing the filling of the curve based on certain design goals, enabling a form of global planning.

Painting-tool-like methods create output along curves but do so directly without taking an a priori completed curve into consideration, as if using a spray can or a brush. Painting techniques usually include a brush diameter, hence the size of the area to be filled along the curve.

**1.2.2.3.1. Procedural Generation** MĚCH et al. [MM12] present examples of painting methods for different aspects of generating procedural models, from painting growth constraints, such as masks, to having a pattern grow along the strokes. This discussion only refers to the actual examples given by the authors. However, these are only selective examples for the flexible *Deco* procedural engine. The engine opens up and generalizes environments for interactive control mechanisms for various types of procedural models. For the programming of decorative pattern models within the engine, helpful functionalities, such as symmetry objects and control guides, are predefined. All artist control mechanisms have an interactive performance. Overall performance mainly depends on the pattern generation scripts. The engine offers to load pattern codes as a dynamic library, optimizing performance. In theory, the Deco engine could allow for the editing both of single elements and their connections. This is crucial for decorative patterns, for example, in setting visual highlights. In MĚCH et al. [MM12] however, no examples for this feature are given.

JACOBS et al. [JBMR18] developed the programming and drawing environment *Dynamic Brushes*, in which an artist can create individual procedural brushes for a stylus pen. General programming logic and relevant mathematical functions for creating patterns are translated into a visual programming interface. The evaluation of the system by two professional artists shows that once initial struggles to learn the system were mastered, the artists were able capture their personal analog styles with the procedural brushes. Overall, the authors and the artists open many valuable questions about the usage of current tools and about alternative approaches that seek to seamlessly blend manual and procedural creation processes.

More painting-like methods can be found, for example, in procedural botanical modeling [APS08; CNX*08; PHL*09], procedural landscape generation [EVC*15], as part of a procedural water color engine [DKMI13] or for dynamic effects [XKG*16].

Todo: Add [GALF17]

**1.2.2.3.2. Data-driven Generation** In order to create an pattern along a sketch, LU et al. [LBW*14] present a data-driven approach. Given vector pattern exemplars are placed and deformed

along a artist-given curve. Boundaries between element segments and visual soundness are optimized through graph cut and hierarchical texture synthesis. For the exemplars, an artist has to define the start and end point of their spines. If needed, the whole spine can be sketched as an input. The artist can refine results with add and erase constraints that are drawn on the pattern. The authors report a synthesizing performance from 1 to 8 seconds. A related data-driven approach for synthesizing example-based vector patterns along a curve was presented by ZHOU et al. [ZJL14] in the same year. In this work, the authors focus on ensuring a structurally sound output pattern and an extension to fill a surface. Topology descriptors and artist-given topological constraints are included in the element assembling optimization process. Additionally, local pattern orientations and a variation value can be defined by an artist. Once a pattern is generated, an artist can interactively adjust the underlying curve, with the pattern being updated accordingly. Generation performances are reported to be around a few seconds, with complex models a little more than 2 minutes.

KAZI et al. [KIZD12] present a multifaceted tool to create textures from pen-and-ink drawings with sketch-based control mechanisms, mixing data-driven and procedural modeling. Basis drawings can be repeated along paths, used for brushes, fill regions, optionally consider perspective and propagate modifications of the drawing to all repeated elements. A user study confirms the system's usefulness to efficiently create repetitive textures while maintaining the natural workflow and artistic control of an artist. XING et al. [XCW14] build upon that work by automatically detecting and suggesting possible repetitions to the artist, aiming for a less regular, more painting-like quality. The presented system also offers various brush options and navigation tools in order to combine automation with artist control.

Similar approaches have also been investigated in the context of texture painting [LFB*13], hand-drawn animations [XWSY15], creating mosaics [Iga10; AGYS14] and data visualization [XHC*18]. These ideas cohere to the needed control principles for creative creation while focusing on their specific design tasks.

**1.2.2.4. Feature Exploration** Even though not a generating technique in itself, exploration is an important characteristic of a creative process. TODI et al. [TWO16] present a tool for exploring sketches and the automatic optimization of common layout types. With the method of CHEN et al. [CFA16], an artist can browse a collection of texture images by sketching highly abstracted pattern features. The represented structural features of reflection, rotation, and translation symmetries adhere to important design principles for visually pleasing patterns. One could imagine a similar intuitive approach for exploring the parameter space of an ornamental procedural representation, for example.

Todo: Add [TGY*09]

### 1.2.3. Element Arrangements

Element arrangements have individual and unconnected visual entities as smallest unit. Arrangements are often function-based distributions and hence can be considered rule-based procedural models, while the elements themselves usually come from input data, such as vector files. Because many visually complex pattern contain areas of formal arranged elements, this is a relevant sub-goal for a creative pattern generation task.

**Minor:** Also mention shapes and masks.

**1.2.3.1. Example-Based Control** An example-based control can be used to arrange the elements. From an example arrangements, relationships between elements are extracted, and results are reproduced for the synthesis.

BARLA et al. [BBT*06] and HURTUT et al. [HLT*09] focus on example-based element arrangements of stroke-based vector elements. BARLA et al. [BBT*06] map vector data to an intermediate representation based on proximity and continuation, which the authors call clusters of strokes. To synthesize a similar arrangement, elements are transferred by local neighborhood matching to a global seed distribution computed by Lloyd relaxation. Computing arrangements takes up to 10 seconds, and artist-input is used in addition to the stroke patterns. A choice between two modes for processing strokes and the amount of variation added is a post-processing step. HURTUT et al. [HLT*09] extend that work by categorizing elements as appearance units and transferring their spatial statistical interactions to new arrangements in the order of seconds, also being able to capture non-uniform distributions. As a possible artist input, one exemplary shape input and density map are shown, and other input options are discussed in principle. The authors clearly state their focus to be on automation.

IJIRI et al. [IMIM08] analyze a given element distribution by local neighborhood comparisons and synthesize output with interactive performance with incremental rule-based local growth. Hence, the technique combines data-driven texture synthesis with procedural generation. Element attributes that go beyond the positions of the elements and orientation cannot be controlled. Artists can choose between three element orientation modes, and as a global design constraint, artists can use an interactive spray tool to define areas to grow in, a flow field tool to define overall alignments and a boundary tool. Moreover, the reconstructed topology can manually be adjusted. The combination of tools that allow the artist to work on the canvas support the immersion in the creative tasks because an artist can think less about abstract setups and instead focus on the actual output.

The technique of MA et al. [MWT11] is based on a sample of a discrete element distribution and an output shape to fill both in two and three dimensions. The exemplar has to contain the actual elements in their domain and cannot be basic pixel data. In its broadest sense, this underlying distribution model can be seen as a procedural model. Even though there are no generative rules, characteristics of the discrete elements and their distribution can be parametrized, and changes can be automatically processed and reproduced in the output. In order to fill the output shape with elements, an energy optimization is processed with a novel neighborhood similarity metric. In addition to element positions, the metric includes variable features referring to orientation, geometry, appearance and type, for example. Hence, the metric is capable of reproducing global aggregate distributions that go beyond local element placements. The authors also extended their work to the spatial-temporal domain [MWLT13]. In regard to the available control mechanisms for

artists, necessary inputs are the exemplary element distribution, the neighborhood size to consider and the output shape. Further distribution constraints based on element attributes are optional. Examples for the inclusion of a vector field and element drag and drop are given. The authors report seconds to minutes for performance times with a non-optimized implementation.

Todo: Sort and add [AKA13; LGH13; DSJ19a; PH19; CXL19; LBMH19]

**1.2.3.2. Vector Fields** SAPUTRA et al. [SKAM17] optimize a flow-based ornamental packing of elements into a two-dimensional outline. For each element, a predefined spine controls the element's deformation. The artist defines direction guides and optionally fixed elements that control the computation of evenly placed streamlines. Elements are placed and deformed along streamlines. An iterative refinement step optimizes for a dense and balanced filling. First, streamlines are slightly shifted to cohere to the space available. Second, elements a re-placed with rotational adjustments and possible overlaps into free space of neighboring elements, reducing negative space. An average packing takes about an hour.

Todo: Add [SKA18; HWYZ18; DSJ19a; HWYZ20]

**1.2.3.3. Data-Driven Approach** PHAN et al. [PLA*16] offer a data-driven recommendation system for circular ornamentation, employing a learned style and composition feature vector. Based on a custom ring-based layout system that represents, for example, plates, vases and a first decorative element chosen by the artist, the system completes a design. The artist can also chose to incrementally add elements manually, while the system accompanies this by suggesting suitable elements and placements. This work indicates the promising direction of using learned characteristics to further stimulating tools, which, for example, generate meaningful design suggestions.

### 1.3. Frames and Hierarchies

BENEŠ et al. [BŠMM11] offer a complex shape-filling and masking system for procedural open L-system models by dividing a target space into artist editable guide shapes. Seeds for the L-system are interactively given by an artist as a position and orientation. The guide shapes determine what types of patterns grow in different areas. The connections between the shapes are manually specified by the artist and in turn guide the connections between elements. Based on a mass-spring system, the guides can be intuitively edited as a whole. The authors report on pattern generation performance for most scenarios as less than a second, with up to 45 seconds for only one complex scenario.

Todo: Add [AMM19]

### 1.4. Curves, Lines and Sketches

The following section discusses work that enables the direct control of curves as pattern elements.

ANDERSON et al. [AW08] adapted the design principles for ornamentation discussed by WONG et al. [WZS98] as well as their core growth mechanism of "finding the largest space to fill next." ANDERSON et al. [AW08]s' technique places discrete elements on the sides of an artist-given curve, while not filling the curve itself. The artist can input masks not to be filled, proxies controlling the size and type of elements to be placed and to equal the sum of radii on both sides of the curve. Two input interfaces exist, the interactive view and the buffer view. The authors do not report a user study or specific performance times but call their system interactive.

Also incorporating an artist-defined curve as the spine of a pattern, CHEN et al. [CSC12] use an interactive L-system to attach decorative spiral designs to the curve given by an artist. XU et al. [XM09] use the space-filling algorithm of WONG et al. [WZS98] in combination with particle tracing in simulated magnetic forces for the generation of decorative curves. The physical properties of the charges, the magnetic field and the initialization of the particles are the parameters for designing the curves. The computation takes less than 5 seconds. The authors acknowledge the non-intuitive parameterization of the system and give an example timing of 2 minutes for finding the parameters of a specific example. MERRELL et al. [MM10] generated a set of curves in the same style of a given parametric example curve. A style is defined by local properties, such as tangents and curvatures that are derived from a local shape analysis. The new curves are computed with a rule-based system that allows artists to interactively edit the result. Interactivity is somewhat diminished by computation times of a few minutes for a curve set. ZEHNDER et al. [ZCT16] provide artists with a tool to directly assemble structurally sound curve networks on a three-dimensional surface. The components of the network are spline curves defined by the artist. Components can be placed manually or are repeated semi-automatically. The curves can be moved on the surface while having an elastic quality to them. To prevent structural weaknesses, the system indicates problematic areas and suggests improvements, seamlessly combining the design task with engineering requirements.

Todo: Add [PS06]

In section **??** 1.2.2.3.2 we already discussed data-driven techniques for creating repetitive structures with sketching. [KIZD12; XCW14; XWSY15] can be similarly used to sketch parts of a pattern directly.

### 1.5. Connections, Branches and Directionality

- Example-based branching: Todo: Add [GJB*20], as inspiration?: [GDG*17
- Data-driven directionality: Todo: Add [XKG*16; HXF*19]
- Street-Modeling: Todo: Add [CLA*19], Plants: Todo: Add [HBDP17]

#### 1.5.1. Vector Fields

In section TODO: about element arrangements discussed in detail, IJIRI et al. [IMIM08] employ vector fields to define the overall growth direction and alignment of elements within an example-guided arrangement. SAPUTRA et al. [SKAM17] optimize a flow-based ornamental packing of elements into a two-dimensional outline.

In section TODO: about rule-based pattern discussed in detail, LI et al. [LBZ*11] present a shape grammar that is guided by either a vector or tensor field.

Todo: Add [GALF17]

Vector fields are further employed in various other specific procedural modeling contexts. For example for procedural street modeling [CEW*08], micrography [MBS*11] or botanical models [XM15].

#### 1.5.2. Creative Means Discussion

The discussed work shows that fields allow for greater visual variation by opening the design space and transparent control for filling a space automatically. When designing a vector field, artists do not work with the pattern directly, but fields are intuitive to understand. Their abstraction translates to the model in a straightforward manner. Thus, using flow within a vector field to design is a suitable control mechanism, especially for designs that aims to align their elements to the space.

### 1.6. Single Accents

By detecting symmetries and curvilinear element arrangements in a given vector pattern, YEH et al. [YM09] extend the manual data-driven design processes with procedural-modeling-like editing options. Based on the detected element groups, an artist can adjust the spacing, location and scale of one element directly and propagate that change to the all other elements in the group. The authors also offer a brush that recreates recognized element groups.

The technique of GUERRERO et al. [GBLM16] offers suitable design variations of the vector pattern an artist is working on. An artist can select and continue with one of the offered alternatives. The system constantly re-selects from an exponential number of relevant variations based on the artist's modifications. The user interface is carefully laid out in order to offer design variations in an intuitive and efficient manner while at the same time not hindering an artist's own workflow. The authors thoroughly evaluate their system quantitatively and qualitatively - for example, with a user study. Overall, participants agreed on the usefulness of technique.

#### 1.6.1. Creative Means Discussion

TODO (still applicable?): The discussion of this section is closely related to the data-driven sketch-based techniques, and it shows a further promising approach for integrating procedural modeling functionalities into a data-driven process. GUERRERO et al. [GBLM16] present a overall transparently navigable and stimulating control mechanism. With a carefully designed workflow, it further fosters an artist stimulation by offering novel but suitable design variations.

### 1.7. Combination of Design Features

Todo: Add [MM12; GALF17; JGMB17; JBMR18; LBM*20]

### 1.8. Discussion of the Creative Means

**Minor:** Add intro

#### 1.8.1. Example-Based Control

The investigation of example-based techniques shows valuable achievements for goal-oriented control and for increasing design spaces within specific contexts. With regard to creative control, in addition to the gain in *variability* being a crucial step, the presented work also improves *navigability* through interactive performances.

Element arrangements potentially enable greater visual variation because they do not need to adhere to any rule formulation. At the same time, the generation of a sample arrangement, usually done in an external application, is potentially tedious. A sample that is too small might lead to uniform results. Moreover, elements are often carefully connected in creative pattern designs, and might include a hierarchy of structures, such as for ornaments. Hence, element arrangements can only provide a subset - albeit an important one - of the design space needed for creative pattern generation.

The related work is overall uniform in working toward the classical requirements of finding the most efficient goal-oriented control . With the exception of IJIRI et al. [IMIM08] and GALERNE et al. [GLLD12] little effort has been made towards improving visual control for an artist. MA et al. [MWLT13] present various powerful control functionalities but do not show their capabilities within an artist usable scenario. GILET et al. [GDG12b] also offer comparatively more mechanisms but as required configuration for their computation not necessarily as variable controllability.

Even if they are example-based, many techniques still require considerable non-creative effort for an artist, such as working with a power spectrum or predicting how changes in the exemplar, such as element arrangements, affect the output. The potential of these methods for creative control lies in furthering interactive performance, reducing initialization requirements and experimenting with the spatial influence of controls. The presented work only focuses on global designs, such as the whole canvas and repeating regions. Methods for which regions could be defined, models layered or the placement of single elements integrated constitute valuable directions for creative control mechanisms.

#### 1.8.2. Shapes and Masks

Procedural generation techniques offer novel systems that decouple control mechanisms from the implementation of individual models. This enables more possible results for one specific technique, thus improving the size of design spaces. Sophisticated masks and growth constraints lead to visually interesting and complex designs. However, it is not directly predictable how a space will be filled exactly. Because most of the presented methods only offer quite limited interactive performance, even a basic trail and error exploration is hardly feasible; hence, the navigation of the design space becomes cumbersome, and stimulation becomes hindered. The one technique [SP16] that offers the means for a transparent navigation is also the one with the most restricted design space. SANTONI et al. [SP16]s' consideration of a navigation history stands out from all related work in this survey. In terms of stimuli, the mass-spring system for editing control guides offered by BENEŠ et al. [BŠMM11] is a promising direction because it is intuitive, enjoyable to use and encourages exploration.

In terms of control mechanisms for a more complex design goal,

these techniques do not permit hierarchical or element-level local controls or the control of element connections needed by artists who want to generate patterns creatively without having to write code.

#### 1.8.3. Vector Fields

The discussed work shows that fields allow for greater visual variation by opening the design space and transparent control for filling a space automatically. When designing a vector field, artists do not work with the pattern directly, but fields are intuitive to understand. Their abstraction translates to the model in a straightforward manner. Thus, using flow within a vector field to design is a suitable control mechanism, especially for designs that aims to align their elements to the space.

#### 1.8.4. Curves, Sketches and Painting

Curves and sketch-like methods offer a well communicated, hence transparent navigation. The discussed techniques are mostly interactive, artists are familiar with their functionality from the real world and they work directly on the canvas. The ease and directness of usage also constitute a foundation for possible immersive flow of work. Using painting-like methods can allow for smoother navigation by integrating brush settings and increasing the quantity of controls.

However, because creation techniques and design spaces are open, it could lead to manual and tedious creation requirements for patterns, such as when filling a background. Here, the incorporation of procedural creation principles for automatic fillings into a data-driven process by KAZI et al. [KIZD12] and XING et al. [XCW14] is a promising direction. Instead of fostering a free painting-like quality, design principles for pattern designs could be added to create a more organized output.

#### 1.8.5. Element Placement

Element placement control mechanisms are closely related to the data-driven sketch-based techniques, and it shows a further promising approach for integrating procedural modeling functionalities into a data-driven process. GUERRERO et al. [GBLM16] present a overall transparently navigable and stimulating control mechanism. With a carefully designed workflow, it further fosters an artist stimulation by offering novel but suitable design variations.

### 1.9. Outlook

The review of the state of the art shows that there are various limitations and possibilities for future work within the specific contexts of the work. However, there are also novel paradigms for creative control and their underlying algorithms that uniquely add to the state of the art. The most prominent development is the integration of machine learning techniques and so-called mixed-initiative algorithms and possibilities of the usage of semantic attributes.

In addition collaboration is a valuable future line of investigation for enabling creative work. With regard to technology, more and more aspects of common tools either are fully browser-based or are in some way connected to the cloud-based storage of assets, settings and results; therefore, they function online and are easily shared. Collaboration is closely connected to the previously discussed issue of navigation histories. This is not only relevant for individual work processes but also for more general production pipelines in a commercial context. In this regard, the sharing and collaborative work on iterations, which involves multiple persons referencing different versions, is essential. Some work has been done (e.g., [TGY*09; SSTP15; OWL*18]) but further investigations of collaboration for creative control are called for.

As discussed in **??**, control techniques are closely inter-related with the representation of the underlying models. Therefore, a more unified development of models across research communities would be beneficial. Expert knowledge of usability should especially be considered. However, further automation for the creation of complex decorative models also poses interesting challenges, such as abstraction [NSX*11], symmetry computation [CO11] and design space variations.

#### 1.9.1. Mixed Initiative Interfaces

The procedural content generation (PCG) for games community is pushing the general integration of artificial intelligence (AI) into a procedural creation process. A new paradigm of *mixed-initiative creative interfaces* is rising and is actively fostered, as an ACM Conference on Human Factors in Computing Systems (CHI) workshop under the same name in 2017 shows [DHF*17]. As the workshop summary states, it is the goal to "put human and computer in a tight interactive loop where each suggests, produces, evaluates, modifies, and selects creative outputs in response to the other." In order to achieve this, AI enables computer agency, and novel interfaces enable collaboration between computers and human users. The workshop brought PCG and interaction design researchers together, stressing the importance of bridging disciplines. Similarly to games, the context of creative pattern generation also constitutes a challenging but fruitful testing ground for the investigation of mixed-initiative creative interfaces and for the task of balancing artist control and automation, as this survey shows. In general, the involvement of the computer graphics community with its various topics and rich algorithmic knowledge would be promising.

Todo: Add [Hee19]

#### 1.9.2. Semantic Attributes

The usage of semantic attributes presents a highly intuitive navigation technique, which so far has been successfully applied in the context of shape modifications, for example by YUMER et al. [YCHK15].

In the context of complex patterns, procedural textures constitute the most related field of investigation. For the control of procedural textures, methods are based on the analysis and description of texture in regard to human perception, which has a long research tradition. In his influential work, JULESZ [Jul81] defines textons as the basic units of pre-attentive human texture perception. Since then, this line of research has continued, and texture descriptions with perceptual [LDC*15] and semantic [MNN13; CMK*14] attributes have been investigated. DONG et al. [DWLS17] and LIU et al. [LGD*18] employed such features in first experiments for the navigation of a procedural texture space and for the generation

of suitable textures by given features. However, the results of such studies are still limited and of varying quality - and the authors themselves [LGD*18] call their results experimental.

Nonetheless, these works present an interesting approach that is worth further investigation. Because many pattern designs are structured and follow an internal logic, it seems feasible to come up with a collection of suitable attributes. For example an ornamental design space is much smaller in comparison to all "textures in the wild" [CMK*14], and ornamentation could constitute a valuable context for further investigations into the incorporation of semantic attributes into a creative creation process.

Todo: Add [JGG*19]

## References


[AGYS14] ABDRASHITOV, RINAT, GUY, EMILIE, YAO, JIAXIAN, and SINGH, KARAN. "Mosaic: Sketch-based Interface for Creating Digital Decorative Mosaics". *Proceedings of the Joint Symposium on Computational Aesthetics, Non-Photorealistic Animation and Rendering, and Sketch-Based Interfaces and Modeling*. ACM, 2014, 5–10 5.

[AKA13] ALMERAJ, ZAINAB, KAPLAN, CRAIG S., and ASENTE, PAUL. "Patch-Based Geometric Texture Synthesis". *Proceedings of the Symposium on Computational Aesthetics*. New York, NY, USA: Association for Computing Machinery, 2013, 15–19 6.

[AMM19] ALVAREZ, LUIS, MONZÓN, NELSON, and MOREL, JEAN-MICHEL. "Interactive design of random aesthetic abstract textures by composition principles". *Leonardo* 0 (2019), 1–11 6.

[APS08] ANASTACIO, F., PRUSINKIEWICZ, P., and SOUSA, M. C. "Sketch-based Parameterization of L-systems Using Illustration-inspired Construction Lines". *Proceedings of the Eurographics Conference on Sketch-Based Interfaces and Modeling*. Eurographics Association, 2008, 119–126. URL: http://dx.doi.org/10.2312/SBM/SBM08/119-126 4.

[AW08] ANDERSON, DUSTIN and WOOD, ZOË. "User driven two-dimensional computer-generated ornamentation". *Advances in Visual Computing*. Springer, 2008, 604–613 6.

[BBT*06] BARLA, PASCAL, BRESLAV, SIMON, THOLLOT, JOËLLE, et al. "Stroke Pattern Analysis and Synthesis". *Computer Graphics Forum* 25.3 (2006), 663–671 5.

[BD04] BOURQUE, ERIC and DUDEK, GREGORY. "Procedural Texture Matching and Transformation". *Computer Graphics Forum* 23.3 (2004), 461–468 3.

[BŠMM11] BENEŠ, B., ŠT'AVA, O., MĚCH, R., and MILLER, G. "Guided Procedural Modeling". *Computer Graphics Forum* 30.2 (2011), 325–334 6, 7.

[BWL18] BIAN, XIAOJUN, WEI, LI-YI, and LEFEBVRE, SYLVAIN. "Tile-Based Pattern Design with Topology Control". *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 1.1 (2018) 3.

[CEW*08] CHEN, GUONING, ESCH, GREGORY, WONKA, PETER, et al. "Interactive Procedural Street Modeling". *ACM Transactions on Graphics* 27.3 (2008), 103:1–103:10 7.

[CFA16] CHEN, YILAN, FU, HONGBO, and AU, KIN CHUNG. "A Multi-level Sketch-based Interface for Decorative Pattern Exploration". *SIGGRAPH Asia Technical Briefs*. ACM, 2016, 26:1–26:4 5.

[CLA*19] CHU, HANG, LI, DAIQING, ACUNA, DAVID, et al. "Neural Turtle Graphics for Modeling City Road Layouts". *ICCV*. 2019 6.

[CMK*14] CIMPOI, M., MAJI, S., KOKKINOS, I., et al. "Describing Textures in the Wild". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014 8, 9.

[CNX*08] CHEN, XUEJIN, NEUBERT, BORIS, XU, YING-QING, et al. "Sketch-based Tree Modeling Using Markov Random Field". *ACM Transactions on Graphics* 27.5 (2008), 109:1–109:9 4.

[CO11] CULLEN, B. and O'SULLIVAN, C. "Symmetry Hybrids". *Proceedings of the International Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging*. ACM, 2011, 33–38 8.

[CSC12] CHEN, YU-SHENG, SHIE, JIE, and CHEN, LIEU-HEN. "A NPR System for Generating Floral Patterns based on L-System". *Bulletin of Networking, Computing, Systems, and Software* 1.1 (2012) 6.

[CXL19] CHEN, MINGHAI, XU, FAN, and LU, LIN. "Manufacturable pattern collage along a boundary". *Computational Visual Media* 5 (2019) 6.

[DHF*17] DETERDING, SEBASTIAN, HOOK, JONATHAN, FIEBRINK, REBECCA, et al. "Mixed-Initiative Creative Interfaces". *Proceedings of the CHI Conference Extended Abstracts on Human Factors in Computing Systems*. ACM, 2017, 628–635 8.

[DKMI13] DIVERDI, S., KRISHNASWAMY, A., MĚCH, R., and ITO, D. "Painting with Polygons: A Procedural Watercolor Engine". *IEEE Transactions on Visualization and Computer Graphics* 19.5 (2013), 723–735 4.

[DSJ19a] DAVISON, TIMOTHY, SAMAVATI, FARAMARZ, and JACOB, CHRISTIAN. "Interactive example-palettes for discrete element texture synthesis". *Computers & Graphics* 78 (2019), 23–36 6.

[DSJ19b] DEROUET-JOURDAN, ALEXANDRE, SALVATI, MARC, and JONCHIER, THÉO. "Generating Stochastic Wall Patterns On-the-fly with Wang Tiles". *Computer Graphics Forum* 38.2 (2019), 255–264 3.

[DWLS17] DONG, JUNYU, WANG, LINA, LIU, JUN, and SUN, XIN. "A Procedural Texture Generation Framework Based on Semantic Descriptions". (2017). arXiv:1704.04141 [cs.CV] 8.

[EVC*15] EMILIEN, ARNAUD, VIMONT, ULYSSE, CANI, MARIE-PAULE, et al. "WorldBrush: Interactive Example-based Synthesis of Procedural Virtual Worlds". *ACM Transactions on Graphics* 34.4 (2015), 106:1–106:11 4.

[GAD*20] GUEHL, PASCAL, ALLÈGRE, REMI, DISCHLER, JEAN-MICHEL, et al. "Semi-Procedural Textures Using Point Process Texture Basis Functions". *Computer Graphics Forum* (2020) 3.

[GALF17] GIESEKE, LENA, ASENTE, PAUL, LU, JINGWAN, and FUCHS, MARTIN. "Organized Order in Ornamentation". *Proceedings of the Symposium on Computational Aesthetics*. ACM, 2017, 4:1–4:9 4, 7.

[GBLM16] GUERRERO, PAUL, BERNSTEIN, GILBERT, LI, WILMOT, and MITRA, NILOY J. "PATEX: Exploring Pattern Variations". *ACM Transactions on Graphics* 35.4 (2016), 48:1–48:13 7, 8.

[GD10] GILET, G. and DISCHLER, J-M. "An Image-Based Approach for Stochastic Volumetric and Procedural Details". *Computer Graphics Forum* 29.4 (2010), 1411–1419 3.

[GDG*17] GUÉRIN, ÉRIC, DIGNE, JULIE, GALIN, ÉRIC, et al. "Interactive Example-Based Terrain Authoring with Conditional Generative Adversarial Networks". *ACM Transactions on Graphics* 36.6 (2017) 6.

[GDG12a] GILET, G., DISCHLER, J-M., and GHAZANFARPOUR, D. "Multiple kernels noise for improved procedural texturing". *The Visual Computer* 28.6 (2012), 679–689 2, 3.

[GDG12b] GILET, G., DISCHLER, J-M., and GHAZANFARPOUR, D. "Multi-scale Assemblage for Procedural Texturing". *Computer Graphics Forum* 31.7 (2012), 2117–2126 3, 7.

[GJB*20] GUO, JIANWEI, JIANG, HAIYONG, BENES, BEDRICH, et al. "Inverse Procedural Modeling of Branching Structures by Inferring L-Systems". *ACM Transactions on Graphics* 39.5 (2020) 6.

[GKHF14] GIESEKE, L., KOCH, S., HAHN, J.-U., and FUCHS, M. "Interactive Parameter Retrieval for Two-Tone Procedural Textures". *Computer Graphics Forum* 33.4 (2014), 71–79 3.

[GLLD12] GALERNE, BRUNO, LAGAE, ARES, LEFEBVRE, SYLVAIN, and DRETTAKIS, GEORGE. "Gabor Noise by Example". *ACM Transactions on Graphics* 31.4 (2012), 73:1–73:9 2, 7.

[GLM17] GALERNE, B., LECLAIRE, A., and MOISAN, L. "Texton Noise". *Computer Graphics Forum* (2017) 2.

[GSDC17] GUINGO, GEOFFREY, SAUVAGE, BASILE, DISCHLER, JEAN-MICHEL, and CANI, MARIE-PAULE. "Bi-Layer textures: a Model for Synthesis and Deformation of Composite Textures". *Computer Graphics Forum* 36.4 (2017), 111–122 2.

[GSV*14] GILET, GUILLAUME, SAUVAGE, BASILE, VANHOEY, KENNETH, et al. "Local Random-phase Noise for Procedural Texturing". *ACM Transactions on Graphics* 33.6 (2014), 195:1–195:11 2.

[HBDP17] HÄDRICH, TORSTEN, BENES, BEDRICH, DEUSSEN, OLIVER, and PIRK, SÖREN. "Interactive Modeling and Authoring of Climbing Plants". *Computer Graphics Forum* 36.2 (2017), 49–61 6.

[Hee19] HEER, JEFFREY. "Agency plus automation: Designing artificial intelligence into interactive systems". *Proceedings of the National Academy of Sciences* 116.6 (2019), 1844–1850 8.

[HLT*09] HURTUT, T., LANDES, P.-E., THOLLOT, J., et al. "Appearance-guided Synthesis of Element Arrangements by Example". *Proceedings of the International Symposium on Non-Photorealistic Animation and Rendering*. ACM, 2009, 51–60 5.

[HWYZ18] HSU, CHEN-YUAN, WEI, LI-YI, YOU, LIHUA, and ZHANG, JIAN JUN. "Brushing Element Fields". *SIGGRAPH Asia 2018 Technical Briefs*. Association for Computing Machinery, 2018 6.

[HWYZ20] HSU, CHEN-YUAN, WEI, LI-YI, YOU, LIHUA, and ZHANG, JIAN JUN. "Autocomplete Element Fields". *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, 2020, 1–13 6.

[HXF*19] HU, ZHONGYUAN, XIE, HAORAN, FUKUSATO, TSUKASA, et al. "Sketch2VF: Sketch-based flow design with conditional generative adversarial network". *Computer Animation and Virtual Worlds* 30.3-4 (2019), e1889 6.

[Iga10] IGARASHI, YUKI. "DECO: A Designing Editor for Line Stone Decoration". *ACM SIGGRAPH Posters*. ACM, 2010, 34:1–34:1 5.

[IMIM08] IJIRI, TAKASHI, MĚCH, RADOMÍR, IGARASHI, TAKEO, and MILLER, GAVIN. "An Example-based Procedural System for Element Arrangement". *Computer Graphics Forum* 27.2 (2008), 429–436 5–7.

[JBMR18] JACOBS, JENNIFER, BRANDT, JOEL R., MĚCH, RADOMÍR, and RESNICK, MITCHEL. "Dynamic Brushes: Extending Manual Drawing Practices with Artist-Centric Programming Tools". *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*. ACM, 2018, D316:1–D316:4 4, 7.

[JGG*19] JOPPI, CHRISTIAN, GODI, MARCO, GIACHETTI, ANDREA, et al. "Texture Retrieval in the Wild Through Detection-Based Attributes". *Image Analysis and Processing – ICIAP 2019*. Ed. by RICCI, ELISA, ROTA BULÒ, SAMUEL, SNOEK, CEES, et al. Springer International Publishing, 2019, 522–533 9.

[JGMB17] JACOBS, JENNIFER, GOGIA, SUMIT, MUNDEFINEDCH, RADOMÍR, and BRANDT, JOEL R. "Supporting Expressive Procedural Art Creation through Direct Manipulation". *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 2017, 6330–6341 7.

[Jul81] JULESZ, BÉLA. "Textons, the elements of texture perception, and their interactions". *Nature* 290 (1981), 91–97 8.

[KH17] KANG, HYEONGYEOP and HAN, JUNGHYUN. "Feature-preserving Procedural Texture". *The Visual Computer* 33.6-8 (2017), 761–768 3.

[KIZD12] KAZI, RUBAIAT HABIB, IGARASHI, TAKEO, ZHAO, SHENGDONG, and DAVIS, RICHARD. "Vignette: Interactive Texture Design and Manipulation with Freeform Gestures for Pen-and-ink Illustration". *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2012, 1727–1736 5, 6, 8.

[LBM*20] LI, JINGYI, BRANDT, JOEL, MECH, RADOMÍR, et al. "Supporting Visual Artists in Programming through Direct Inspection and Control of Program Execution". *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, 2020, 1–12 7.

[LBMH19] LI, YIFEI, BREEN, DAVID E., MCCANN, JAMES, and HODGINS, JESSICA. "Algorithmic Quilting Pattern Generation for Pieced Quilts". *Proceedings of Graphics Interface 2019*. Canadian Information Processing Society, 2019 6.

[LBW*14] LU, JINGWAN, BARNES, CONNELLY, WAN, CONNIE, et al. "DecoBrush: Drawing Structured Decorative Patterns by Example". *ACM Transactions on Graphics* 33.4 (2014), 90:1–90:9 4.

[LBZ*11] LI, YUANYUAN, BAO, FAN, ZHANG, EUGENE, et al. "Geometry Synthesis on Surfaces Using Field-Guided Shape Grammars". *IEEE Transactions on Visualization and Computer Graphics* 17.2 (2011), 231–243 4, 6.

[LDC*15] LIU, JUN, DONG, JUNYU, CAI, XIAOXU, et al. "Visual Perception of Procedural Textures: Identifying Perceptual Dimensions and Predicting Generation Models". (2015). PLoS ONE 10(6): e0130335 8.

[LFB*13] LUKÁČ, MICHAL, FIŠER, JAKUB, BAZIN, JEAN-CHARLES, et al. "Painting by Feature: Texture Boundaries for Example-based Image Creation". *ACM Transactions on Graphics* 32.4 (2013), 116:1–116:8 5.

[LGD*18] LIU, JUN, GAN, YANHAI, DONG, JUNYU, et al. "Perception-driven procedural texture generation from examples". *Neurocomputing* 291 (2018), 21–34 8, 9.

[LGH13] LANDES, PIERRE-EDOUARD, GALERNE, BRUNO, and HURTUT, THOMAS. "A Shape-Aware Model for Discrete Texture Synthesis". *Proceedings of the Eurographics Symposium on Rendering*. Eurographics Association, 2013, 67–76 6.

[LHVT17] LOI, HUGO, HURTUT, THOMAS, VERGNE, ROMAIN, and THOLLOT, JOELLE. "Programmable 2D Arrangements for Element Texture Design". *ACM Transactions on Graphics* 36.3 (2017), 27:1–27:17 3.

[LLC*10] LAGAE, A., LEFEBVRE, S., COOK, R., et al. "State of the Art in Procedural Noise Functions". *Proceedings of the Conference of the European Association for Computer Graphics*. The Eurographics Association, 2010 2.

[LP00] LEFEBVRE, LAURENT and POULIN, PIERRE. "Analysis and Synthesis of Structural Textures". *Proceedings of the Graphics Interface Conference*. 2000, 77–86 3.

[LVLD10] LAGAE, ARES, VANGORP, PETER, LENAERTS, TOON, and DUTRÉ, PHILIP. "Procedural Isotropic Stochastic Textures by Example". *Computers and Graphics* 34.4 (2010), 312–321 2.

[MBS*11] MAHARIK, RON, BESSMELTSEV, MIKHAIL, SHEFFER, ALLA, et al. "Digital Micrography". *ACM Transactions on Graphics* 30.4 (2011), 100:1–100:12 7.

[MM10] MERRELL, PAUL and MANOCHA, DINESH. "Example-based Curve Synthesis". *Computers and Graphics* 34.4 (2010), 304–311 6.

[MM12] MĚCH, RADOMÍR and MILLER, GAVIN. "The Deco framework for interactive procedural modeling". *Journal of Computer Graphics Techniques* 1.1 (2012), 43–99. (Visited on 12/03/2015) 2, 4, 7.

[MNN13] MATTHEWS, T., NIXON, M. S., and NIRANJAN, M. "Enriching Texture Analysis with Semantic Data". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, 1248–1255 8.

[MWLT13] MA, CHONGYANG, WEI, LI-YI, LEFEBVRE, SYLVAIN, and TONG, XIN. "Dynamic Element Textures". *ACM Transactions on Graphics* 32.4 (2013), 90:1–90:10 5, 7.

[MWT11] MA, CHONGYANG, WEI, LI-YI, and TONG, XIN. "Discrete Element Textures". *ACM Transactions on Graphics* 30.4 (2011), 62:1–62:10 5.

[NSX*11] NAN, LIANGLIANG, SHARF, ANDREI, XIE, KE, et al. "Conjoining Gestalt Rules for Abstraction of Architectural Drawings". *ACM Transactions on Graphics* 30.6 (2011), 185:1–185:10 8.

[OWL*18] O'LEARY, JASPER, WINNEMÖLLER, HOLGER, LI, WILMOT, et al. "Charrette: Supporting In-Person Discussions Around Iterations in User Interface Design". *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2018, 535:1–535:11 8.

[PGDG16] PAVIE, NICOLAS, GILET, GUILLAUME, DISCHLER, JEAN-MICHEL, and GHAZANFARPOUR, DJAMCHID. "Procedural texture synthesis by locally controlled spot noise". *Proceedings of International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*. 2016 2.

[PH19] PEIHAN TU, DANI LISCHINSKI and HUANG, HUI. "Point Pattern Synthesis via Irregular Convolution". *Computer Graphics Forum (Proceedings of SGP 2019)* 38.5 (2019), 109–122 6.

[PHL*09] PALUBICKI, WOJCIECH, HOREL, KIPP, LONGAY, STEVEN, et al. "Self-organizing Tree Models for Image Synthesis". *ACM Transactions on Graphics* 28.3 (2009), 58:1–58:10 4.

[PLA*16] PHAN, H. Q., LU, J., ASENTE, P., et al. "Patternista: Learning Element Style Compatibility and Spatial Composition for Ring-based Layout Decoration". *Proceedings of the Joint Symposium on Computational Aesthetics and Sketch Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering*. Eurographics Association, 2016, 79–88 6.

[PS06] PEDERSEN, HANS and SINGH, KARAN. "Organic Labyrinths and Mazes". *Proceedings of the 4th International Symposium on Non-Photorealistic Animation and Rendering*. Association for Computing Machinery, 2006, 79–86 6.

[RMGH15] RITCHIE, DANIEL, MILDENHALL, BEN, GOODMAN, NOAH D., and HANRAHAN, PAT. "Controlling Procedural Modeling Programs with Stochastically-ordered Sequential Monte Carlo". *ACM Transactions on Graphics* 34.4 (2015), 105:1–105:11 4.

[RTHG16] RITCHIE, DANIEL, THOMAS, ANNA, HANRAHAN, PAT, and GOODMAN, NOAH D. "Neurally-guided Procedural Models: Amortized Inference for Procedural Graphics Programs Using Neural Networks". *Proceedings of the International Conference on Neural Information Processing Systems*. Curran Associates Inc., 2016, 622–630 4.

[ŠBM*10] ŠT'AVA, O., BENEŠ, B., MĚCH, R., et al. "Inverse Procedural Modeling by Automatic Generation of L-systems". *Computer Graphics Forum* 29.2 (2010), 665–674 4.

[SKA18] SAPUTRA, REZA ADHITYA, KAPLAN, CRAIG S., and ASENTE, PAUL. "RepulsionPak: Deformation-Driven Element Packing with Repulsion Forces". *Proceedings of the 44th Graphics Interface Conference*. Canadian Human-Computer Communications Society, 2018, 10–17 6.

[SKAM17] SAPUTRA, REZA ADHITYA, KAPLAN, CRAIG S., ASENTE, PAUL, and MĚCH, RADOMÍR. "FLOWPAK: Flow-based Ornamental Element Packing". *Proceedings of the Graphics Interface Conference*. Canadian Human-Computer Communications Society, 2017, 8–15 6.

[SP16] SANTONI, CHRISTIAN and PELLACINI, FABIO. "gTangle: A Grammar for the Procedural Generation of Tangle Patterns". *ACM Transactions on Graphics* 35.6 (2016), 182:1–182:11 3, 7.

[SSTP15] SALVATI, GABRIELE, SANTONI, CHRISTIAN, TIBALDO, VALENTINA, and PELLACINI, FABIO. "MeshHisto: Collaborative Modeling by Sharing and Retargeting Editing Histories". *ACM Transactions on Graphics* 34.6 (2015), 205:1–205:10 8.

[TGY*09] TALTON, JERRY O., GIBSON, DANIEL, YANG, LINGFENG, et al. "Exploratory Modeling with Collaborative Design Spaces". *ACM Transactions on Graphics* 28.5 (2009), 1–10 5, 8.

[TLL*11] TALTON, JERRY O., LOU, YU, LESSER, STEVE, et al. "Metropolis procedural modeling". *ACM Transactions on Graphics* 30.2 (2011), 1–14 4.

[TWO16] TODI, KASHYAP, WEIR, DARYL, and OULASVIRTA, ANTTI. "Sketchplore: Sketch and Explore with a Layout Optimiser". *Proceedings of the ACM Conference on Designing Interactive Systems*. ACM, 2016, 543–555 5.

[TWY*20] TU, PEIHAN, WEI, LI-YI, YATANI, KOJI, et al. "Continuous Curve Textures". *ACM Transactions on Graphics* 39.6 (2020) 3.

[TYK*12] TALTON, JERRY, YANG, LINGFENG, KUMAR, RANJITHA, et al. "Learning Design Patterns with Bayesian Grammar Induction". *Proceedings of the ACM Symposium on User Interface Software and Technology*. ACM, 2012, 63–74 4.

[WZS98] WONG, MICHAEL T., ZONGKER, DOUGLAS E., and SALESIN, DAVID H. "Computer-generated Floral Ornament". *Proceedings of the Conference on Computer Graphics and Interactive Techniques*. ACM, 1998, 423–434 3, 6.

[XCW14] XING, JUN, CHEN, HSIANG-TING, and WEI, LI-YI. "Autocomplete Painting Repetitions". *ACM Transactions on Graphics* 33.6 (2014), 172:1–172:11 5, 6, 8.

[XHC*18] XIA, HAIJUN, HENRY RICHE, NATHALIE, CHEVALIER, FANNY, et al. "DataInk: Direct and Creative Data-Oriented Drawing". *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2018, 223:1–223:13 5.

[XKG*16] XING, JUN, KAZI, RUBAIAT HABIB, GROSSMAN, TOVI, et al. "Energy-Brushes: Interactive Tools for Illustrating Stylized Elemental Dynamics". *Proceedings of the Symposium on User Interface Software and Technology*. ACM, 2016, 755–766 4, 6.

[XM09] XU, LING and MOULD, DAVID. "Magnetic Curves: Curvature-Controlled Aesthetic Curves Using Magnetic Fields". *Proceedings of the Eurographics conference on Computational Aesthetics in Graphics, Visualization and Imaging*. The Eurographics Association, 2009 6.

[XM15] XU, LING and MOULD, DAVID. "Procedural Tree Modeling with Guiding Vectors". *Computer Graphics Forum* 34.7 (2015), 47–56 7.

[XWSY15] XING, JUN, WEI, LI-YI, SHIRATORI, TAKAAKI, and YATANI, KOJI. "Autocomplete Hand-Drawn Animations". *ACM Transactions on Graphics* 34.6 (2015) 5, 6.

[YCHK15] YUMER, MEHMET ERSIN, CHAUDHURI, SIDDHARTHA, HODGINS, JESSICA K., and KARA, LEVENT BURAK. "Semantic Shape Editing Using Deformation Handles". *ACM Transactions on Graphics* 34.4 (2015), 86:1–86:12 8.

[YM09] YEH, YI-TING and MĚCH, RADOMÍR. "Detecting Symmetries and Curvilinear Arrangements in Vector Art". *Computer Graphics Forum* 28.2 (2009), 707–716. (Visited on 06/10/2015) 7.

[ZCT16] ZEHNDER, JONAS, COROS, STELIAN, and THOMASZEWSKI, BERNHARD. "Designing Structurally-sound Ornamental Curve Networks". *ACM Transactions on Graphics* 35.4 (2016), 99:1–99:10 6.

[ZJL14] ZHOU, SHIZHE, JIANG, CHANGYUN, and LEFEBVRE, SYLVAIN. "Topology-constrained Synthesis of Vector Patterns". *ACM Transactions on Graphics* 33.6 (2014), 215:1–215:11 5.