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

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

A Survey of Control Mechanisms for Creative Pattern Generation

Name[1,2] and Name[2]

[1]Place
[2]Place

Abstract

Note: *This is currently a summary of the STAR for communication purposes:*

Supporting artists with meaningful digital tools for creative creation is an ongoing research challenge and spans over various disciplines. On an algorithmic level, most solutions focus on adding singular features and control mechanisms. Little attention is paid to novel methods that can complement each other as part of a cohesive pipeline and creative workflow. Towards the goal of enabling a creative workflow, we survey current methods in the scope of creative pattern generation and their control mechanisms. We classify techniques and discuss their potential to support creative creation with newly developed criteria.

The specific goal of two-dimensional and potentially aesthetically pleasing pattern generation, offers a complex design challenge. On the one hand, the repetitive nature of a pattern is well represented with algorithmic creation, especially with procedural representations. On the other hand, the design space of possible pattern designs ranges from realistic, over abstracted to artistic. Here, considerations about a creative workflow and the need to go beyond automation rise and meaningful control needs to be given to artists. Typically however, with the power of procedural generation comes difficult controllability. Combining the algorithmic nature of procedural generation with novel forms of controllability has great capabilities and potential to offer truly novel benefits to traditional creation processes.

For analyzing the state of the art and a better understanding of creative creation with digital tools, we dissect a creation process into to the methodologies of how, what, where and when. To relate control mechanisms on algorithmic level to the stages of a creation process, we classify specific control mechanisms into exemplars, parameterization, handling, filling, guiding and placing interactions. For making the domain of creative creation more manageable, the creative means of navigation, transparency, variation and stimulation are defined and linked to the control mechanisms.

With these criteria, we survey the most recent algorithmic representations for creative pattern generation. For this chosen design goal, the analysis bridges between various techniques such as procedural and data-driven approaches. We identify open issues and sketch out promising research directions towards the unification of different building blocks to a coherent pipeline. All in all, we hope to inspire innovation for artist-centered creation processes on a grander scheme.
(see https://www.acm.org/publications/class-2012)

CCS Concepts
• ***Computing methodologies*** → *Collision detection;* • ***Hardware*** → *Sensors and actuators; PCB design and layout;*

Important: All references from 2018 on are still missing!

1. Introduction

Digital tools for creative processes with various forms of output are indispensable for most artists. Even designs that have a distinct hand-made quality to them, such as Disney's animation Paperman for example, were computer-generated [Wal12], employing novel software solutions that are fully controllable by artists. Furthermore, more than three decades of research in academia have produced various control mechanisms for creation. These are often said to be artist-controllable but are less often proven to be so.

Most research has been executed without direct and continuous collaboration with artists. Moreover, large-scale user studies with suitable participants are usually impractical. Due to these obstacles, there is little common understanding in the research community regarding what artist needs actually are and how mechanisms are validated. Furthermore, research has typically focused on solving

one specific aspect while accepting significant trade-offs for other steps in a creation process.

For example, the automatic targeted control of a large design space results in long computation times and non-intuitive configuration requirements (e.g., [BD04; WZS98]). Equally, flexible creation pipelines that enable both automation and manual controllability often suffer from a restricted design space (e.g., [SP16]).

It is an important challenge to investigate creation processes from a more artist-centered perspective and to consider all tasks and efforts as a whole, from initial configuration requirements to local edits. As a further challenge, control mechanisms have to be linked to an expressive design space, which is the basis for all meaningful creative creation. In order to analyze and validate novel creation algorithms, artist feedback also has to be evaluated. However, such interdisciplinary studies still suffer at times from undefined language and vague discussions. It is an open challenge to complement user studies with a more precise and determinative terminology that is also commonly applicable and understandable by artists.

The previous statement naturally leads to general questions about artist-centered full controllability in combination with distinctive output spaces and a meaningful and applicable evaluation of digital creation tools. These considerations are based on decades of research in multiple disciplines. However, there is little common ground on what is needed as theoretical basis for an evaluation of creative control mechanisms with various open questions. For example, what are relevant characteristics of a creation process with digital tools? What are specific control mechanisms, ranging from global to local and from automatic to manual, with varying levels of abstraction for their handling? How do these mechanisms relate to the stages of a creation process? What are the requirements for creative creation, and how can these be customized to the context of digital creation tools?

To bring further insights, this survey investigates those problems in the scope of creative two-dimensional visual pattern generation, tackling a limited but highly challenging and representative creation process and design space. In particular, in this report we present the state of the art in control mechanisms for the creative generation of visual patterns with the following goals:

- The identification of relevant characteristics of a creation process with digital tools.
- The classification of control mechanisms as theoretical grounding, ranging from global to local and from automatic to manual, with varying levels of abstraction for their handling and the relation of these mechanisms to the stages of a creation process.
- An interlinking analysis of the state of the art, bridging between procedural and data-driven solutions and joint classification results.
- A discussion of the capabilities of a specific mechanism in a creative process and their potential for creative creation within a coherent pipeline.

2. Terminology

Note: This is far too long.

The following clarifies the usage of terms that are relevant for an analysis of control mechanisms. Some aspects are discussed in detail in different sections of this report but are included here for an overview of terms.

Pattern: Constitutes a generic term for any type of repeated, often regular, arrangement [Oxf17].

Texture: In the context of computer graphics, *texturing* is commonly understood as modeling a surface's color (i.e., their color texture) with no implications for a design, while designing a surface' s interaction with light is understood as *shading*.

Texture refers in its traditional meaning to the character of a woven fabric [Oxf17] with properties such as *fine* or *coarse*. This work understands texture similarly with regard to potentially repetitive structures. LIN et al. [LHW*06] define a spectrum for such structures. The spectrum ranges from regular deterministic textures with distinguishable texture elements recurrently placed to irregular placements to purely stochastic textures.

Decor: Refers to elements that generally embellish and beautify without implying any specific design rules in itself.

Ornament: Constitutes a specific type of decor adhering to certain design rules, such as order, hierarchal structures, space adaptation and visual contrast and accents (**?? ??**).

Artistic: Refers to a task with an outcome that potentially has meaning and value beyond aesthetics and practicality. In addition to formal skills that depend on a given domain, an artistic task usually requires creative thinking as well as intuition, emotion and sensual considerations, for example.

Creative: Refers to a task that intentionally produces a novel, non-standard outcome. Please note that this work refers to the academic usage of the term. In common language, a creative task is often misunderstood as one that produces a visual product.

Design space: Refers loosely to all visual results a technique can create. For example, a simple Perlin noise has a rather restricted design space of noise images, only differing, for example, in their frequency. Drawing with a pen can result in many different designs, thus resulting in a larger design space.

Expressiveness: Refers in this work to the size, the variability and the openness of a design space (these terms are in detail discussed in **??**). It is important to note that expressiveness is commonly used in the context of creative controls - however, usually without a clear understanding of its meaning.

Goal-oriented control: Refers to an clearly targeted design task and stands in contrast to exploration. For example, reference images might be given, or an artist may have a clear mindset about how the output of the creation process should look.

User Interface (UI): Refers in this work to a space that is separate from the canvas where an artist controls the system through abstracted representations, such as buttons and sliders or custom-made visual controls. In terms of controllability it makes a difference for an artist to be able to work directly on a canvas or being required to do so in a separated and often abstracted UI and hence this explicit distinction is needed in the context of this work.

Interactive: Refers in this work to systems with which an artist

can interact (e.g., through a UI with reasonable response performance). In terms of control mechanisms, we evaluate interactive systems as a whole and also qualitatively. An one-time investment for an initial computation of 10 seconds, for example, is still acceptable, while a 5-second delay at each click is not.

Canvas: Constitutes the area in which the output is generated, similar to a canvas in a painting context.

Shape: Refers to the external boundary or outline on the canvas or of an object without any restrictions on the form.

Curves: Refers in this work as general term for arbitrarily shaped curves or lines without any implications for the formal representation they are based on. Curves can be computed or be derived from drawn strokes, for example. The specifics of these inherently different formal representations are not relevant for the following discussion about control mechanisms.

Procedural: Refers to the production of output by evaluating an algorithm or a rule-based system.

Data-driven: Refers to the production of output based on given, and usually limited, data.

Parameterized: Refers in its original meaning to a system that is based on an implicit equation. However, in regard to control mechanisms and for this work, it simply means that a system offers separated, individually controllable characteristics. Parameterization commonly does not imply a procedural representation but can be part of any technique, including data-driven ones.

3. Designs and Models

3.1. Design Goals

[Description of pattern designs we consider.]

3.2. Models

In the context of computer graphics, generation techniques are differentiated into procedural and data-driven approaches.

3.2.1. Procedural

Ebert et al. [2002] describe procedural techniques as algorithms and mathematical functions that synthesize a model or an effect.

[...]

We further categorizes procedural models as stochastic, function and rule based, grammar based, simulation based, and artificial intelligence based.

[...]

3.3. Data-Driven Models

In contrast to procedural techniques, data-driven methods can be used in two ways in the context of ornamentation. First, they describe the processing of input pixel data, such as a photograph. Second, they refer to the output of a method, which is again pixel data. Data-driven models traditionally do not include underlying design models, as procedural representations do. Consequently, data-driven approaches are flexible in terms of possible designs and can achieve photorealism by processing real photographs.

At the same time, photographs bring the disadvantage of potentially including visual features, such as illumination effects, which are unwanted and difficult to remove. Moreover, further down a production pipeline, pixel data is usually not editable anymore. Working with data such as high-resolution images leads to high memory requirements, and without additional algorithms, data is fixed to its given resolution and scale.

Addressing the issue of resolution example-based synthesis is a well established field of research and aims to create infinite amounts of pixel data based on a given exemplar. The pyramid-based texture synthesis of HEEGER et al. [HB95] is an early famous example. WEI et al. [WLKT09] present a comprehensive summary of such example-based texture synthesis techniques, discussing statistical feature matching, neighborhood matching, patch-based and optimization methods. Overall, example-based methods for texture synthesis have achieved similar results in data size, random accessibility and editing and resolution options as procedural textures âĂŞ but only within specialized contexts and not in an unified manner. Procedural textures offer these capabilities as inherent and combined characteristics.

Data-driven models are numerous and diverse because they can use and produce any input and output data without an underlying procedural model. Their classification is out of scope of this work. However, we do include in the following various techniques that offer further meaningful control mechanisms in the context of ornamentation. These techniques include the tiling and distribution of elements and drawing and brush mechanisms.

4. Taxonomy of Control Mechanisms

We dissect a creation process into overall control paradigms and classify specific control mechanisms by their interaction types. The taxonomy shows the capabilities of the different control mechanisms and potential trade-offs between approaches.

4.1. Control Paradigms

A creation process can be described by answering the questions of *how*, *what*, *where*, *when* and *who*. These paradigms can be discussed in various creation contexts and could even be translated to traditional media such as aquarell on paper.

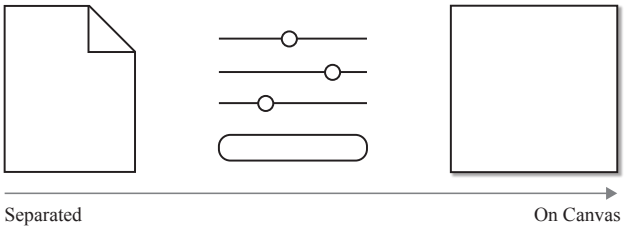
4.1.1. How

How is a control executed or an input given by an artist? How far is it from the visual result on the canvas?

File: The control is externally given, such as with code or a configuration file.

UI: A separate UI is given through which an artist gives input and activates states. User interfaces are often in close proximity to the canvas, carefully designed and easily usable. However, because they detach the work from the actual output, UIs still have an abstract nature. An artist must actively translate his or her interaction with the UI to the resulting output on the canvas.

On canvas: Controls are executed directly on the output canvas. Most of these controls require an activation or selection of a tool in a separate UI, such as selecting a pen for drawing on a canvas. In this case we consider the pen primarily as a control mechanism. There are cases where controls cannot clearly be classified as either UI or on canvas. A pen, for example, can have different characteristics that an artist needs to set in the UI. Ideally, the adjustment of settings should be as seamlessly integrated into an on-canvas tool as possible (e.g., with selection choices appearing as tool tips).



4.1.2. What

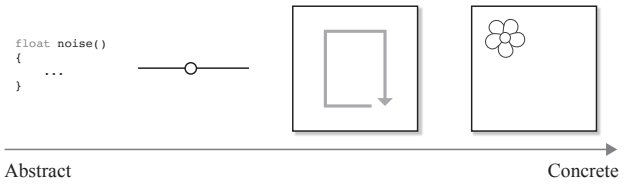
What does an artist give as input? What is the level of abstraction of the content that an artist works with?

Code: Input is a syntactically structured and formal language.

Value: The input is a single value, chosen from a range âĂŞ for example, with a slider.

Intermediate: The input is visual but still of an abstract nature, such as controlling sketches for a mask or arrows for directionality. Again, artists have to interpret how these inputs affect the result.

Element: The input constitutes a component of the resulting pattern.



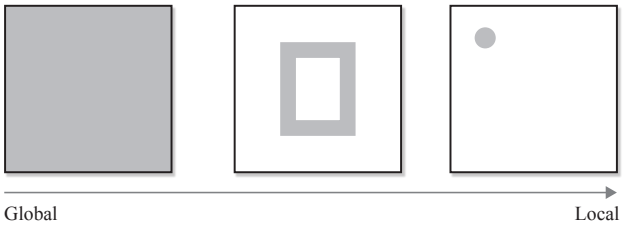
4.1.3. Where

Where does the input have an effect spatially and what is its area of influence?

Global: The input has global influence (e.g., by filling the whole space or by adjusting all elements on the canvas).

Region: The input has an effect in a region of the canvas (e.g., on a drawn curve).

Local: The input has an effect on one specific element.



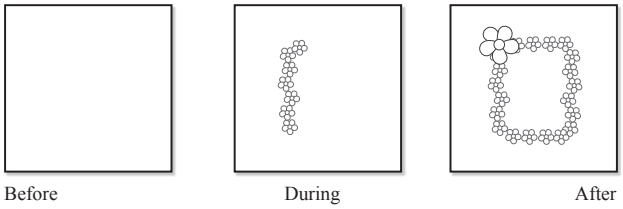
4.1.4. When

When can input be given and at what time in the creation process is the control executed?

Before: Input is given before the actual creation process.

During: Input is given during the creation process, when parts of the results are already visible. This is typically a painting mechanism. However, some processes can also be paused and adjusted.

After: Input is given after the creation process. The result is visible to the artist and can be adjusted retrospectively.



4.1.5. Who

Who can give the input in regard to the type of skill set needed? This category can be in part derived from the above characteristics of *how* and *what*. In most general terms this category can be classified as the following.

Programmer: To give input with dissecting analytical-formal and logical thinking and the ability to abstract.

Artist: To give input with comprehensive intuitive-visual and spatial thinking and the ability to create (e.g., by drawing).

The who category is listed here for completeness. However, to fully answer the questions of needed competencies, skill- and mindsets, including the accompanying psychological and artistic aspects, is out of the scope of this thesis and requires knowledge in fields other than computer science. The following discussions are rooted in computer graphics research and aim for an assessment of algorithmic controllability. Hence, the classification of who specifically is most suitable to use a tool, is put aside.

4.2. Control Mechanisms

For a meaningful analysis, the above classification must be further broken down into the specific control mechanisms.

The following low-level characteristics categorize input modes and their primary effect. Because this survey focuses on interfacing algorithms, UI specifics, such as the layout of buttons, are not considered. Once the specific control mechanisms are analyzed, they constitute a method in combination with the above control paradigms.

4.2.1. Initialization

System Configuration: Required overall setup of the system, such as computing caches or training a model. This is usually a one-time investment.

Task Initialization: A non-creative task that has to be executed each time in order to produce an output, such as selecting the specific optimization algorithm.

4.2.2. Exemplars

Image: An example image that should be matched in its entirety. Examples are usually pixel data.

Element Arrangement: An example element arrangement that should be matched in its entirety. Elements are usually separate shapes and might carry additional data.

Element: One specific asset that becomes in the result part of a whole. Elements can be shapes or pixel data.

4.2.3. Parameterization

Visual Output: Parameters that can adjust visual features directly in the output.

System/Generation: Parameters that influence the output indirectly, such as parameters for an optimization algorithm or constraints.

4.2.4. Handling

Visualization: Any type of visual interface that goes beyond the standard UI elements, such as sliders and buttons.

Image-Based: Images as indirect control input, such as pixel data masks.

Sketch-Based: Sketches and curves directly put on the canvas âĂŞ for example, the drawing of a mask with a pen tool.

4.2.5. Filling

Shapes: A space to fill (e.g., a specific shape).

Masking: Areas within the shape to fill that should remain unaffected.

Curves to Fill: A one-dimensional curve or path to be filled. The curve is given as a whole before the filling starts.

			HOW			WHAT				WHERE			WHEN		
		Total (out of 40)	File	UI	Canvas	Code	Value	Intermediate	Element	Global	Region	Local	Before	During	After
	Setup														
	Configuration	9	9			8	1			9	1		9		
	Initalization	14	12	2		3	1	5	1	13	2		14		
⟵ *Decreasing Automation*	*Exemplars*														
	Image	1	9	1				1	1	1	2		1		
	Arrangement	9	8	1	1	4		3	6	9	2		9		
	Element	9	6	2	1			5	4	7	2		9		
	Parameterization														
	Visual Output	31	23	8		5	25	2		29	2		17	1	16
	System	16	1	7			15	1		17	4		16		
	Handling														
	Visual UI	6		3	3			5	2	4	5		3	1	4
	Image	9	9			2		7		9	5		9		
	Sketch	7		1	6			8	3	3	7	2	6		2
	Filling														
	Shapes	32	25		8	21		11		32	12		32	1	
	Masking	9	4		6	2		7		6	9		9	1	
	Curve	8			8			6	2	1	8		7	1	
	Guiding														
	Painting	6			6			1	5		6			6	
	Directions	7	1		6			7		7	7		7		
	Placing														
	Element	7			7				7			7	7		
	Drag&Drop	5			5				5	1	4	4		4	1

Table 1: *Prevalence of control mechanisms in the literature: In total, 40 publications are included (the discussed state of the art work). Please note, that the totals of each step (how, what, where, when) can exceed the total of that category as it can be implemented within multiple usage scenarios.*

4.2.6. Guiding

Painting/Strokes to Follow: A curve, usually created by mouse movements or with a stylus pen, that is filled with output elements while the curve is generated âĂŞ often understood as brushing.

Directions: Visual elements such as intermediate curves, arrows or output components that define directions for the design to follow (e.g., with an underlying vector field).

4.2.7. Placing

Element Placement: The direct placement of components on the canvas as part of the final result.

Element Drag & Drop: Drag and drop of components on the canvas within the existing result.

4.2.8. Control Stages of the Mechanisms

For interrelating the control mechanisms to the control paradigms, we considered the publications that are investigated in **?? ??**. Due to the diversity of the underlying methods and the different design

goals of the considered body of work, we believe this to be a representative summarization.

Table 1 shows that global, hence automatic, control is usually enabled through intermediate representations, such as an example image, while on the other end of the spectrum, the placement of elements as part of the actual output is local, and automation is lost.

Parameterization and the different types of handling also require abstracted input from an artist, such as the use of a slider. Sketch-based controls, such as an eraser, move the interaction onto the canvas and can make small-scale adjustments. The definition of a space or a curve to fill and masking areas is also usually done directly on the canvas but only influence the output indirectly.

A painting mechanism simultaneously creates the output directly on the canvas but can only do so in a limited region depending on the brush size. All other inputs are typically given before or after the generation of the output.

This classification underlines that a focus on one control type, as is usual in computer graphics research, leads to the common trade-off between global automation and local manual manufacturing. In order to support creative work, control mechanisms need to be combined in a novel and unified manner.

5. Analysis of the State of the Art

On the one hand, patterns include repetitive and ordered structures that are often considered as *textures*, thus demanding automatic and procedural creation. On the other hand, visually pleasing patterns often require a global layout, adapt to the space they are filling and include visual hierarchies and highlights that are singularly placed with creative intent. This artistic challenge either requires computational creativity, or an artistâĂŹs creativity must be supported with meaningful digital tools.

Procedural representations are notoriously difficult to control [BD04; LVLD10; GD10; BŠMM11; LLD12b; LLD12a], and much effort has gone into investigating control mechanisms within specific contexts. Botanical and architectural procedural modeling, for example, are popular fields of research. There have been summarizing surveys for procedural noise [LLC*10], landscapes [STBB14] and urban spaces [VAW*09] as well as in the context of games [HMVI13; TYSB11], including even a short summary of models for ornamentation [Whi10]. However, the overall investigation of generating and designing decorative patterns and ornamentation is less prominent. This could be credited to ornamentation being an ill-defined domain due to it involving creative-artistic considerations. Nonetheless, its diverse design aspects make ornamentation a rich and compelling topic.

Creative pattern generation is also an interesting testing ground for addressing the delicate balance between giving artists as much control as is needed without burdening them with unwanted details. Procedural representations are well suited for expressing repetitive and ordered structures and enable parametric control. However, traditional procedural modeling approaches are in need of novel control mechanisms that are intuitively navigable, flexible and engaging.

The contribution of this survey is twofold. On the one hand, its categorizes the state of the art with regard to control mechanisms and translates this categorization to overall control paradigms. 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. Pattern generation includes a variation of representative creation challenges, such as combining ordered fillings and repetitive structures with individual global layouts and highlighting components that might break that underlying order.

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.

Note: The following categories for the subsections are up for discussion.

Todo: Check for correct author names: replace all heeger_1995_pbt occurences.

5.1. Texturing Methods

Texturing methods focus on creating a repetitive and homogeneous pattern as automatically as possible. These methods provide only a fraction of the controllability needed for ornamentation. They are solely applicable for the subparts of an ornament with a texture-like quality to it, such as background regions and fillings.

But as the investigation of procedural texturing has been the driving force behind the development of procedural representations in general, it produced manifold approaches and noteworthy control mechanisms. Even though texturing methods are not one-to-one transferrable to ornamentation, their rich research history and solutions must be included when investigating procedural ornamentation and might inspire ornamentation specific controllability.

5.1.1. Example-Based Control

Example-based approaches 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 this context, the control of a model often directly derives from a new model definition, and the focus of the related work is usually the latter. 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.

5.1.1.1. Stochastic Textures

Todo: Condense and shorten this section.

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.

The first two approaches are based on specific function characteristics and are hardly generalizable for decorative pattern design. In the context of ornamentation, noise functions are seldom used in their initial state but more often as a basis for pattern design. We do not investigate the specific noise function parameters further but only include the overall example-based control mechanism. 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].

For Gaussian-like textures, the input analysis and function parameter derivation methods are specific to the targeted noise functions. The method of LAGAE et al. [LVLD10] matches noise bandwidths for isotropic multi-resolution noise with the performance described as "rapid", given by GILET et al. [GDG12a] as a few milliseconds. In addition to the cropped exemplar, no artist input is required. 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. Recently, the same authors [GLM17] introduced an efficient sparse convolution noise based on textons. A texton, which is a bilinearly interpolated function, summarizes the power spectrum of a given texture exemplar, and the final noise is generated by placing and summing up textons. The example match takes a couple of seconds, and no further artist input is required.

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. For a straightforward noise by example computation, the method of GILET et al. [GDG12a] successively decomposes noise frequencies to match the power spectrum of a multiple kernels noise, and, depending on the number of artist-defined convolution noises, it takes up to 20 seconds. For greater control and expressiveness, a perturbation function and a multi-layer approach are presented. The perturbation can be an additional artist-defined image map of the desired pattern, showing how the pattern should be repeated, breaking the regularity and possibly creating a more structural pattern. How the magnitude of the perturbation is defined is not described by the authors, but it could easily be an input parameter. Furthermore, GILET et al. [GDG12a] interpret texture design as hierarchical composition and employ a layer function that assigns positions to different textures. For this function a artist-defined map can be used.

Further pursuing the topic of greater expressiveness and a more structured noise, GILET et al. [GSV*14] introduced a local random phase noise. The key aspect of their approach is the separation of structure and noise. The noise function itself blends a sum of cosines with random phase, locally centered on a regular spatial grid. A artist-controlled parameter relates to the number of cosines and the visual quality of the noise. Examples of Gaussian patterns can be given by a spectrum or a discrete noise image. For structured designs, specific phases in the power spectrum are fixed independently from the spatial domain. The amount of structure in comparison to noise is controlled with a parameter by the artist. 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. The authors argue for control mechanisms being more intuitive in the spatial domain instead of the commonly used editing of the power spectrum. Local random phase noise [GSV*14] is extended by aligning the 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. A spatially varying Gaussian noise can be matched to a suitable exemplar by computing a structure layer, extracted by filtering the exemplar, blending masks derived from clustering the spectral domain and different spectra from an auto-correlation technique. In order to procedurally represent the discrete structure and mask layers, a method based on tiling is applied. In order to control the synthesis, the artist needs 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 512×512 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 âĂIJfeatureâĂİ and âĂIJnon-featureâĂİ parts. Non-features are obtained by a noise-by-example method. The authors do not mention whether the noise can be further adjusted. 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 257×257 input image, and therefore the texture matching, ranges from few seconds to 2 minutes, depending on an additional frequency clustering exploiting spatial coherences in the input. GILET et al. [GD10] apply a more general optimization strategy for choosing the parameters of a noise-based procedure. They minimize an image distance met-

ric computed with a multi-resolution Gabor filter bank and a windowed Fourier transform with gradient descent. 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.

5.1.1.2. Unrestricted Texture Designs 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 ornamental patterns. 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. We include their work in this discussion because it also handles 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, one based on the Fourier transform of the images and the other one utilizing histograms of the Laplace pyramid of the images. For optimization, they apply the Nelder-Mead and the gradient descent method. 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 âĂIJon the order of 200âĂİ 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.

5.1.1.3. Element Arrangements An example-based control can be used to arrange elements, which refer to individual visual entities that are the smallest unit for these techniques. From an example arrangements, relationships between elements are extracted, and results are reproduced for the synthesis. 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 ornaments contain areas of formal arranged elements, this is a relevant sub-goal for designing an ornament. 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.

5.1.2. 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.

5.1.3. Summary

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 ornamentation, and ornaments include a hierarchy of structures. Hence, element arrangements can only provide a subset âĂŞ albeit an important one âĂŞ of the design space needed for ornamentation.

The related work is overall uniform in working toward the classical requirements of finding the most efficient goal-oriented control as **??** shows. 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 of ornamentation 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 ornamentation.

5.2. Conclusions

Towards the goal of supporting artists in their creative work with innovative and meaningful tools, first a well defined and interdisciplinary understanding of creation processes and creativity is needed.

We dissect a creation process into overall characteristics and classify specific control mechanisms by their interaction types. The analysis of the state of the art shows the capabilities of the different control mechanisms and potential trade-offs between approaches. For handling the ill-defined topic of creativity, we follow the definition of creativity as intentionally producing a novel and surprising product. We establish means for creative control and relate specific control mechanisms to creative processes. By this we further a more objective judging of the ability of a technique to support creativity and a detailed comparison of methods.

However, some aspects of the analysis framework still leave room for interpretation. Knowledge from other disciplines, for example in regard to the perception of visual features, might be able to contribute with valuable insights. We hope that our results inspire such research towards a quantifiable analysis of creative control.

References

[BBT*06] BARLA, PASCAL, BRESLAV, SIMON, THOLLOT, JOËLLE, et al. "Stroke Pattern Analysis and Synthesis". *Computer Graphics Forum* 25.3 (2006), 663–671 8.

[BD04] BOURQUE, ERIC and DUDEK, GREGORY. "Procedural Texture Matching and Transformation". *Computer Graphics Forum* 23.3 (2004), 461–468 2, 6, 8.

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

[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 6–8.

[GDG12a] GILET, G., DISCHLER, J-M., and GHAZANFARPOUR, D. "Multiple kernels noise for improved procedural texturing". *The Visual Computer* 28.6 (2012), 679–689 7, 8.

[GDG12b] GILET, G., DISCHLER, J-M., and GHAZANFARPOUR, D. "Multi-scale Assemblage for Procedural Texturing". *Computer Graphics Forum* 31.7 (2012), 2117–2126 8, 9.

[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 7, 9.

[GLM17] GALERNE, B., LECLAIRE, A., and MOISAN, L. "Texton Noise". *Computer Graphics Forum* (2017) 7.

[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 7.

[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 7.

[HB95] HEEGER, DAVID J. and BERGEN, JAMES R. "Pyramid-based Texture Analysis/Synthesis". *Proceedings of the SIGGRAPH Conference on Computer Graphics and Interactive Techniques*. ACM, 1995, 229–238 3.

[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 8.

[HMVI13] HENDRIKX, MARK, MEIJER, SEBASTIAAN, VAN DER VELDEN, JOERI, and IOSUP, ALEXANDRU. "Procedural Content Generation for Games: A Survey". *ACM Transactions on Multimedia Computing, Communications, and Applications* 9.1 (2013), 1:1–1:22 6.

[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 8, 9.

[KH17] KANG, HYEONGYEOP and HAN, JUNGHYUN. "Feature-preserving Procedural Texture". *The Visual Computer* 33.6-8 (2017), 761–768 7.

[LHW*06] LIN, WEN-CHIEH, HAYS, J., WU, CHENYU, et al. "Quantitative Evaluation of Near Regular Texture Synthesis Algorithms". *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2006, 427–434 2.

[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 6, 7.

[LLD12a] LASRAM, ANASS, LEFEBVRE, SYLVAIN, and DAMEZ, CYRILLE. "Procedural Texture Preview". *Computer Graphics Forum* 31.2 (2012), 413–420 6.

[LLD12b] LASRAM, ANASS, LEFEBVRE, SYLVAIN, and DAMEZ, CYRILLE. "Scented Sliders for Procedural Textures". *Proceedings of the Conference of the European Association for Computer Graphics*. The Eurographics Association, 2012 6.

[LP00] LEFEBVRE, LAURENT and POULIN, PIERRE. "Analysis and Synthesis of Structural Textures". *Proceedings of the Graphics Interface Conference*. 2000, 77–86 8.

[LVLD10] LAGAE, ARES, VANGORP, PETER, LENAERTS, TOON, and DUTRÉ, PHILIP. "Procedural Isotropic Stochastic Textures by Example". *Computers and Graphics* 34.4 (2010), 312–321 6, 7.

[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 8, 9.

[MWT11] MA, CHONGYANG, WEI, LI-YI, and TONG, XIN. "Discrete Element Textures". *ACM Transactions on Graphics* 30.4 (2011), 62:1–62:10 8.

[Oxf17] OXFORD ENGLISH DICTIONARY ONLINE. http://www.oed.com. Accessed August 20, 2017. 2017 2.

[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 7.

[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. URL: http://onlinelibrary.wiley.com/doi/10.1111/j.1467-8659.2009.01636.x/abstract (visited on 06/09/2015) 9.

[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 2.

[STBB14] SMELIK, RUBEN M., TUTENEL, TIM, BIDARRA, RAFAEL, and BENES, BEDRICH. "A Survey on Procedural Modeling for Virtual Worlds". *Computer Graphics Forum* (2014) 6.

[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 9.

[TYSB11] TOGELIUS, J., YANNAKAKIS, G. N., STANLEY, K. O., and BROWNE, C. "Search-Based Procedural Content Generation: A Taxonomy and Survey". *IEEE Transactions on Computational Intelligence and AI in Games* 3.3 (2011), 172–186 6.

[VAW*09] VANEGAS, CARLOS A., ALIAGA, DANIEL G., WONKA, PETER, et al. "Modeling the Appearance and Behavior of Urban Spaces". *Proceedings of the Conference of the European Association for Computer Graphics*. The Eurographics Association, 2009 6.

[Wal12] WALT DISNEY ANIMATION STUDIOS. *Paperman*. https://www.disneyanimation.com/studio/our-films. Accessed August 20, 2018. 2012 1.

[Whi10] WHITEHEAD, JIM. "Toward Proccedural Decorative Ornamentation in Games". *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*. ACM, 2010, 9:1–9:4 6.

[WLKT09] WEI, LI-YI, LEFEBVRE, SYLVAIN, KWATRA, VIVEK, and TURK, GREG. "State of the art in example-based texture synthesis". *Proceedings of the Conference of the European Association for Computer Graphics*. The Eurographics Association, 2009, 93–117 3.

[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 2.