MNPR: A Framework for Real-Time Expressive Non-Photorealistic Rendering of 3D Computer Graphics

Santiago E. Montesdeoca
Nanyang Technological University
Interdisciplinary Graduate School, MAGIC

Hock Soon Seah
Nanyang Technological University, School of Computer Science and Engineering

Amir Semmo
Hasso Plattner Institute for Digital Engineering, University of Potsdam

Pierre Bénard
LaBRI (UMR 5800, CNRS, Univ. Bordeaux), Inria

Romain Vergne
Joëlle Thollot
Univ. Grenoble Alpes
CNRS, Inria, Grenoble INP, LJK

Davide Benvenuti
Nanyang Technological University
School of Art, Design and Media

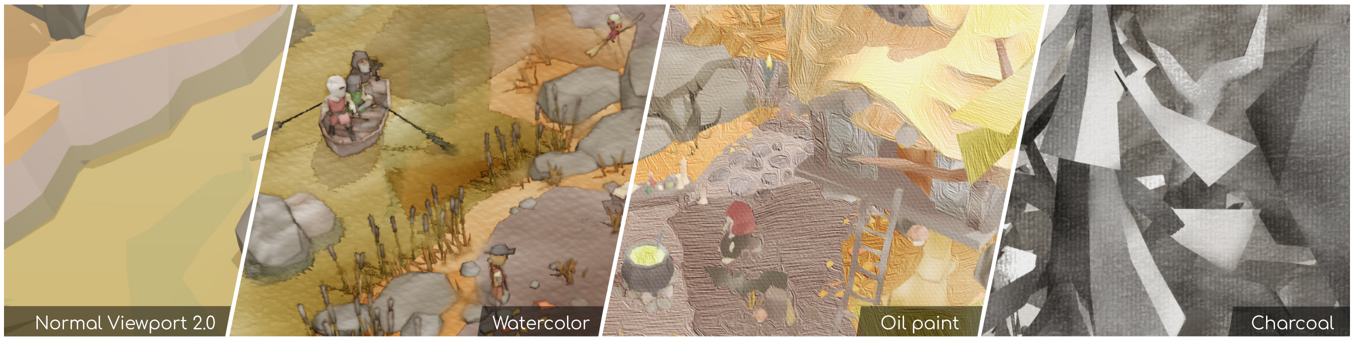


Figure 1: A scene rendered through MNPR in different styles. Baba Yaga's hut model, 🅭🅯 Inuciian.

ABSTRACT

We propose a framework for expressive non-photorealistic rendering of 3D computer graphics: MNPR. Our work focuses on enabling stylization pipelines with a wide range of control, thereby covering the interaction spectrum with real-time feedback. In addition, we introduce control semantics that allow cross-stylistic art-direction, which is demonstrated through our implemented watercolor, oil and charcoal stylizations. Our generalized control semantics and their style-specific mappings are designed to be extrapolated to other styles, by adhering to the same control scheme. We then share our implementation details by breaking down our framework and elaborating on its inner workings. Finally, we evaluate the usefulness of each level of control through a user study involving 20 experienced artists and engineers in the industry, who have collectively spent over 245 hours using our system. MNPR is implemented in Autodesk® Maya® and open-sourced through this publication, to facilitate adoption by artists and further development by the expressive research and development community.

CCS CONCEPTS

• **Computing methodologies → Non-photorealistic rendering**;

KEYWORDS

Expressive NPR, object-space stylization, filtering, levels of control, real-time, interaction, framework.

ACM Reference Format:
Santiago E. Montesdeoca, Hock Soon Seah, Amir Semmo, Pierre Bénard, Romain Vergne, Joëlle Thollot, and Davide Benvenuti. 2018. MNPR: A Framework for Real-Time Expressive Non-Photorealistic Rendering of 3D Computer Graphics. In *Expressive '18: The Joint Symposium on Computational Aesthetics and Sketch Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering, August 17–19, 2018, Victoria, BC, Canada.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3229147.3229162

1 INTRODUCTION

Non-photorealistic rendering (NPR) has been studied since the emergence of computer graphics—either directly, or indirectly as a side-product in the pursuit of photorealism. As such, NPR presents a significant gamut of research, introducing models, algorithms and systems, which have found practical applications in fields that rely on computer graphics [Winnemöller 2013]. However, the application of non-photorealistic rendering in 3D computer graphics (e.g., games, animated films) is limited, especially when compared to photorealistic rendering.

With the plethora of 3D content coming out each year, it has become increasingly relevant for productions to differentiate themselves from others. We argue that expressive NPR approaches are a promising alternative for this purpose, which will be of increasing relevance in the years to come. Therefore, the main goal of this paper is to propose and validate a framework towards expressive NPR of 3D computer graphics in a real production software. This framework provides the tools required for art-direction at different levels of control and covering the interaction spectrum [Isenberg 2016] in multiple styles. In the context of this paper, expressive rendering refers to the ability given to the artist to art-direct the rendered imagery that is produced from 3D data, enabling an individual vision in a specific style.

Rendering, be it photorealistic or non-photorealistic, is not expressive by itself. Photorealistic rendering has pushed towards expressiveness since its inception, offering tools to let artists control the lighting in the scene, the material attributes [Schmidt et al. 2016] and tweak the overall photoreal look at the compositing stage. These tools have shown considerable success, to the extent where skilled artists often achieve expressive results that simulate a custom reality, indistinguishable from a photograph.

Non-photorealistic rendering also takes advantage of photorealistic tools and provide a further alternative by releasing them from any physical laws. Additionally, depending on the style of the render, special effects need to be included that help emulate a specific look. Correspondingly, these effects require custom tools that enable different levels of control—depending on the intended audience and use. We cannot generalize tools for all NPR approaches found in 3D computer graphics (i.e., filter-based, example-based, mark-based). However, considering our intended real-time application in games and animation, we focus on providing tools for filter-based NPR approaches that use image-processing to generate the stylized outcome.

Recent research that focused on art-directed filter-based approaches in 3D-space has shown promising results [Harvill 2007; Montesdeoca et al. 2017a,b]. However, these existing approaches have only been applied to a single style and present only a narrow interaction spectrum, limiting and slowing down their adoption in the industry. There is still extensive work to be done for expressive non-photorealistic rendering to address and enable the range of stylized looks that artists envision.

The research presented in this paper is motivated by the artistic need for an intuitive and generalized approach towards expressive NPR in 3D computer graphics. For this purpose, we have developed MNPR, an expressive rendering framework in a real production software which focuses on enabling art-direction throughout the interaction spectrum—to maximize its adoption by any skill level and optimize the workflow for each use case. MNPR empowers our stylization pipelines in watercolor, oil and charcoal, but can easily be extended by developers due to its cross-stylistic and open-source nature. To summarize, we make the following contributions:

(1) an art-directed approach towards covering the interaction spectrum in expressive NPR of 3D computer graphics (Section 3.1);
(2) generalized control semantics to guide cross-stylistic art-direction (Section 3.2), which is exemplified for three styles;
(3) the implementation insights and source code of MNPR (Section 3.3) as part of its public release;
(4) a user study with experienced artists and engineers to evaluate the usefulness of each level of control (Section 4).

The remainder of this paper consists of related work (Section 2), the proposed levels of control and generalized control semantics with implementation details (Section 3), evaluations (Section 4), results (Section 5), and drawn conclusions with potential future work (Section 6).

2 RELATED WORK

In the footsteps of Salesin [2002] and Gooch et al. [2010], Isenberg [2016] argued in his recent meta paper that we, NPAR researchers, “should focus—in addition to working on algorithmic contributions—on how to design the interaction with the algorithmic support”. Our work can be seen as the first practical answer to Isenberg’s call, in the specific context of interactive filter-based stylization of 3D computer graphics.

We present the design of a stylization framework that is generic enough to incorporate different styles, and that allows the programmer to expose the respective expressive functionality to the artist, throughout the interaction spectrum. Our system has proven its merit in the hands of hundreds of artists who successfully produced art-directed results.

Although interaction might have drawn insufficient attention by the NPAR community, a number of computer systems have been developed along the years, both in the academia and the industry, to give artists some control over the stylization process of 3D assets.

Stroke-based stylization methods offer the most natural, but lowest-level, interaction metaphor, most often achieved by on-screen painting with virtual brushes. Its extension to 3D [Hanrahan and Haeberli 1990] led to painting systems such as Disney’s *Deep Canvas* [Daniels 1999] and *OverCoat* [Schmid et al. 2011]. Similar low-level interaction techniques have been proposed for line rendering from 3D models [Cardona and Saito 2015; Kalnins et al. 2002; Whited et al. 2012]. This resulted in systems that facilitate the creation, combination and application of non-photorealistic techniques in 3D such as *Jot* [Kalnins et al. 2003], *OpenNPAR* [Halper et al. 2003, 2002], *RenderBots* [Schlechtweg et al. 2005] and more recently, Disney’s *Meander*. Unfortunately, these systems see little adoption in the industry as they either remain proprietary or are implemented as standalone software that is not production-ready. A notable exception is *Freestyle* [Grabli et al. 2010], which allows a higher-level, programmable control. However this kind of control is so technical that it was eventually supplemented with dials and knobs when integrated into *Blender*.

At the opposite end of the interaction spectrum, most generic example-based methods (e.g., [Bénard et al. 2013; Fišer et al. 2016]) lack intuitive local control of the style transfer, as discussed by Semmo et al. [2017]. Only a few methods targeting a specific task or style, such as hatching illustration [Gerl and Isenberg 2013], offer tools that let the user guide the underlying synthesis algorithm.

Closest to our work, but limited to 2D content, the filter-based image stylization framework of Semmo et al. [2016] offers three levels of control—default presets, global parameters, and local adjustments through a painting interface—that drive a variety of stylization algorithms (cartoon, watercolor, oil paint, pencil hatching). Each style is independently parameterized, which makes the framework fully modular but prevents cross-stylistic art-direction.

For 3D graphics, the local parameter adjustments can be painted onto the surface of the 3D objects [Harvill 2007; Montesdeoca et al. 2017a,b], instead of a 2D canvas, to ensure spatial coherence in motion. Once rasterized into G-buffers, the obtained 2D parameter maps can then be used to control image-space processing. Going a step further towards a production-ready system raised two questions: (1) how to build higher levels of control on top of the locally painted parameters to cover the full interaction spectrum?; (2) how to design sufficiently generic, yet semantically meaningful parameters to allow cross-stylistic art-direction? We present our answers to those questions in the next section.

3 EXPRESSIVE NPR

Art-direction is key towards expressive rendering, however, there is arguably no single solution or tool that will satisfy the expressive requirements of every artist. Each artist has a unique vision, approach and workflow when creating artwork, requiring a varied set of tools. 3D artists have the same requirements, but are often constrained by the available algorithms and digital tools. Developers put much effort in addressing these demands, but the task becomes increasingly complex when considering all the different styles and technical approaches inherent to NPR. Our research focuses on proposing a generalized framework for filter-based expressive NPR that provides a wide range of tools for artists and facilitates the further development of art-directed styles for developers.

In Section 3.1, we approach our generalized framework by augmenting the interaction spectrum for expressive rendering, granting a wide range of control to the artist. By doing so, we provide expressive tools for multiple levels of control that will facilitate their workflow and the art-direction of the rendered images.

In Section 3.2, we present our stylistic control semantics, which help developers create tools that work predictably with other styles, allowing cross-stylization with the same parameters. Thus, creating stylizations that generalize upon common phenomena and enabling the artist to visualize their expressive results in different styles.

In Section 3.3, we share our thought process and implementation details of this endeavor, which is supplemented by an open-source release to enable any future researcher and developer to incorporate their non-photorealistic pipelines in an expressive rendering framework within Autodesk® Maya®: MNPR.

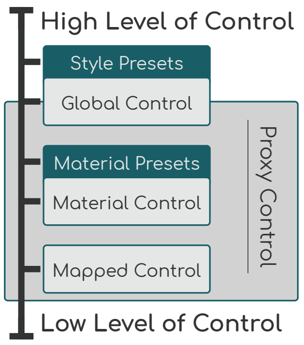


Figure 2: Schematic overview of the different levels of control that we address in MNPR.

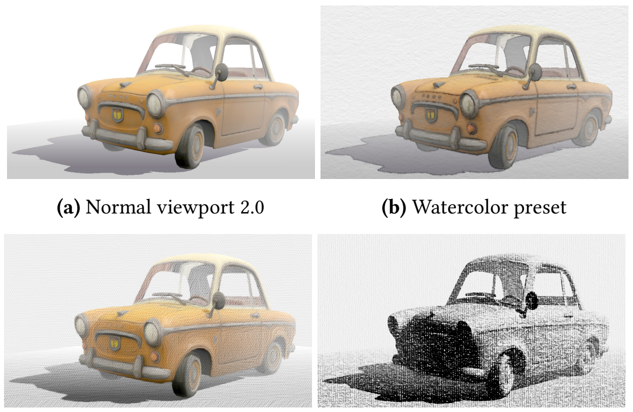
3.1 Levels of control

Research in NPR commonly focuses on algorithms that attain a specific effect or style, but offer limited tools for artists to interact with these effects and modify the look of the stylized imagery. This often restraints the artistic workflow and potential of the algorithm itself. To avoid this, ideally a set of art-directed tools that address different levels of control in real-time are required.

Art-directed tools take time to develop, but the rewards justify the effort. Practical interactive tools can increase the creative involvement of artists who can validate the quality of the algorithm and highlight potential intrinsic problems. An ideal workflow should enable 3D artists to quickly set up a general style using high levels of control, offer tools at the mid-level of control to accelerate the stylization process and also provide the ability to meticulously tweak localized parameters—to achieve their particular vision within the style. The tools to cover this range of control are addressed in this section, as outlined in Figure 2.

Style presets and global control. At the highest level of control, style presets allow the artist to save or load predefined global control parameters—interactively changing between different stylization pipelines (e.g., oil, watercolor, charcoal) and assigning the global control parameters in the entire scene, as illustrated in Figure 3. Style presets provide a fast rough start towards a specific style, which is refined by modifying each global control. Global control parameters act as an abstraction layer of effect variables within the image-processing algorithms applied in image-space (e.g., effect amounts, substrate attributes, atmospheric perspective).

Material presets and material control. At the mid-level of control, material presets allow the artist to save or load predefined material parameters of 3D objects. A material preset modifies the underlying shader code and attributes of each material and enable the creation of material libraries with varying painterly shading properties (e.g., dilution, specularity, transparency). Additionally, the painterly materials are embedded with the ability to drive different image-processing effects using procedural noise parameters N assigned in material-space, as shown in Figure 4. These allow natural-looking fluctuations in effects that might require them.



(a) Normal viewport 2.0 **(b)** Watercolor preset

(c) Oil paint preset **(d)** Charcoal preset

Figure 3: Different stylization applied through style presets to the Pony Cartoon model, 🅭🅯 Slava Zhuravlev.

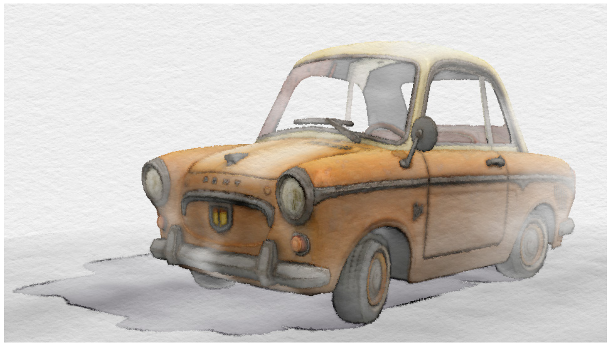


Figure 4: Material-space procedural effects applied after the watercolor preset of Figure 3b.

Mapped control. At the lowest level of control, parameters assigned through mapped control M allow the artist to locally assign specific effects in object-space. This level of control provides the most refined control for the detail oriented artist, but it can also be the most time consuming. However, under real-time interactivity, these tools can be designed to assimilate a painting workflow, becoming immediately intuitive and fun for the artist to use [Montesdeoca et al. 2016]. Mapped control possesses significant uses by itself, but a well thought-out combination of effects can also create targeted stylization concepts (e.g., watercolor turbulence using different pigment densities, as seen in Figure 5)—going beyond the specific effect that they were designed to produce.

Proxy control. In parallel to most levels of control is proxy control, which comprises invisible 3D elements that only contribute to the render with localized stylization parameters P. However, by taking advantage of the 3D space they operate in, proxies encompass the assignment of stylization parameters from a high level of control—affecting the whole image by being placed in front of the camera, resembling NPR lenses [Neumann et al. 2007]—, to a low level of control—affecting a specific part in 3D space—, as visualized in Figure 6. This versatility is unique to proxies, since these operate as standalone elements in the scene, which can form arbitrary representations in object-space. Depending on their design, control proxies can support the previously mentioned procedural parameters in material-space and mapped parameters in object-space.

All these levels of control work in unison to produce the final stylized render. First, the 3D-space driven parameters, which include the procedural noise effect parameters N, the mapped effect parameters M and the proxy effect parameters P contribute to the

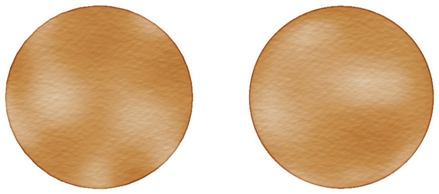


Figure 5: Left: Procedural pigment density (turbulence <5 sec.). Right: Mapped pigment density (turbulence 40+ sec.).

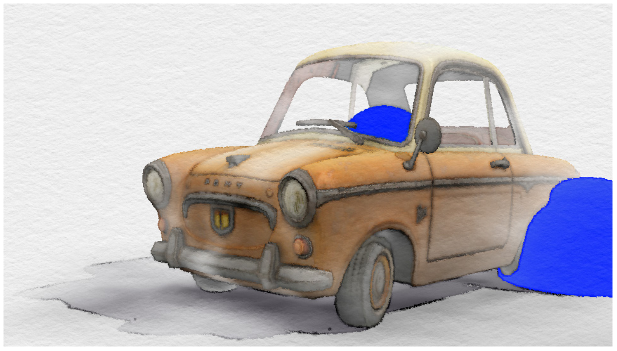


Figure 6: Stylized result after applying the mapped control; proxies are visualized in blue.

image-space stylization maps M_{f_x}. These positive or negative parameters are combined at the rasterization stage $M_{f_x} = N + M + P$, as seen in Figure 7.

Once the rasterization stage is completed, the stylization pipeline modulates the effect parameters found in the stylization maps with the global effect parameters of the style during runtime. These parameters control the image-processing stages, modifying the effect algorithms that are applied to the original rendered image, to finally create the stylized imagery, as shown in Figure 8.

3.2 Generalizing control semantics

Artistic visions can be represented in different styles, as seen in Figure 9. These artworks are an excellent example of two paintings using different media, where the art-direction remains mostly the same. Both masterfully portray a dynamic piece with the same subject in a mystical African setting.

The cross-stylistic art-direction found in traditional paintings is desirable in expressive NPR, as well. However, control semantics are necessary and these need to follow a defined control scheme for the expressive styles to work predictably.

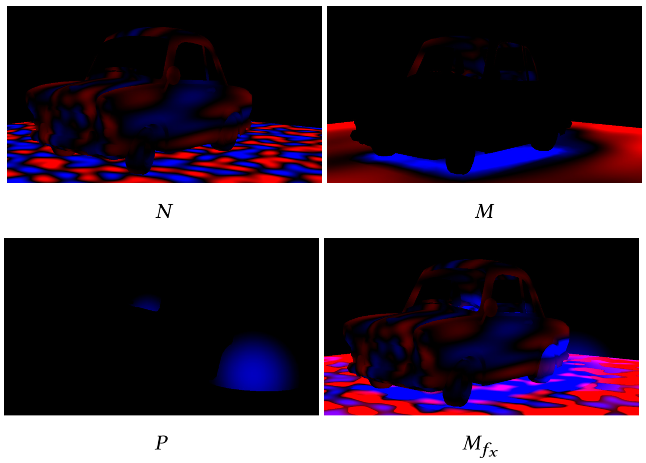


Figure 7: Effect parameters within stylization map M_{f_x}. Positive values are visualized in blue, negative values in red.

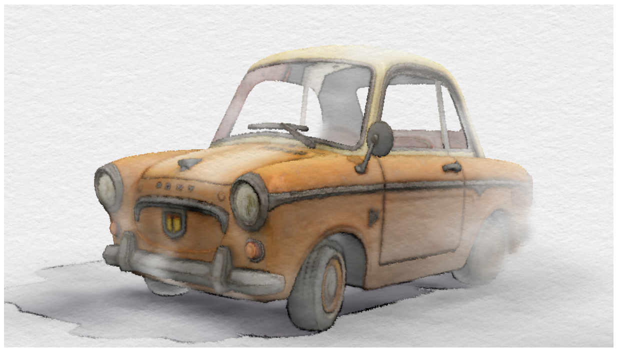


Figure 8: Final watercolor rendering leveraging all levels of control.

Based on our observations and discussions working with artists, we argue that stylization control parameters can be generalized into four distinct categories:

- Pigment-based effects
- Substrate-based effects
- Edge-based effects
- Abstraction-based effects

These four groups can be used to directly correlate the stylization control parameters between different styles.

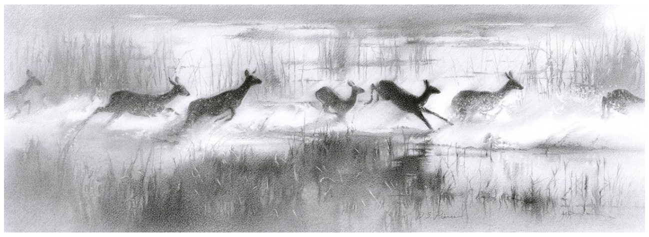
We have the majority of art-directed parameters in the stylization maps M_{f_x}, of which each can manage an effect category. Within each effect category, there are semantics for the type of effect that can be modified. These semantics need to be sufficiently generic, yet semantically meaningful to be adapted to different styles and accepted by the NPR development community. Additionally, these effects need to adhere to a control scheme (summarized in Table 1), which defines what semantics goes to which channel in the stylization map—so that these can be interpreted by other styles. That way, stylization pipelines supporting the same effect categories, respecting the semantics and following the control scheme, would enable e.g., to map an art-directed rendering in a watercolor style to an oil and charcoal style (Figure 10).

We have defined our stylization semantics based on the experience and observations that we have gathered while researching and implementing the styles of watercolor [Montesdeoca et al. 2017a,b], oil paint [Semmo et al. 2016] and charcoal. Based on these different stylizations, we defined the following effects semantics:

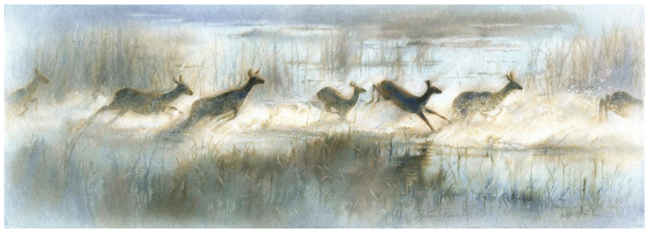
Table 1: Stylization control scheme.

Channel	Pigment M_{f_x}	Substrate M_{f_x}
R	Pigment variation	Substrate distortion
G	Pigment application	U-inclination
B	Pigment density	V-inclination

Channel	Edge M_{f_x}	Abstraction M_{f_x}
R	Edge intensity	Detail
G	Edge width	Shape
B	Edge transition	Blending



Life-Spring, Graphite 4x12". © Dylan Scott Pierce.



Life Spring, Watercolor 8x22". © Dylan Scott Pierce.

Figure 9: Paintings showing cross-stylization with different types of media.

3.2.1 *Pigment-based effects.*

This category contains parameters that direct the way the pigmentation is rendered within a specific style:

- *Pigment variation.* Controls the degree at which the reflected color of a compound pigment varies towards one or another color. E.g., green pigmentation that deviates to a more blue or yellow color in certain parts.
- *Pigment application.* Controls how the pigment is placed over a substrate. This can be interpreted as the amount or pressure at which pigment is applied to achieve an effect. E.g., dry-brush application, thick application.
- *Pigment density.* Controls the concentration of the pigment placed over a substrate. This is especially relevant to transparent and translucent media (i.e., watercolor, ink, colored pencils), but can also influence opaque media. E.g., dilution, lightness, saturation.

3.2.2 *Substrate-based effects.*

This category contains parameters that direct the way the substrate (i.e., canvas, paper, wall) affects a specific style:

- *Substrate distortion.* Controls the distortion caused by the substrate roughness on the rendered image. This is especially relevant for fluid media (i.e., watercolor, graffiti).

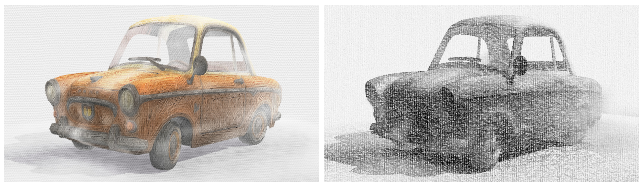


Figure 10: Art-directed Pony Cartoon model visualized in oil paint (left) and charcoal (right) styles.

- *U- & V-inclinations.* Control the inclination of the substrate, which generally affects the direction at which patterns or marks from fluid media evolve. However, generalizing upon this, these parameters are used to define the offset of existing patterns or marks in a horizontal or vertical direction. E.g., bleeding direction, cross-hatching direction, stroke direction.

3.2.3 *Edge-based effects.*

This category contains parameters that direct the way the edges of the subject are rendered within a specific style:

- *Edge intensity.* Controls the edge strength/intensity within the stylized render. E.g., linework darkness, edge darkening.
- *Edge width.* Controls the edge thickness of the stylized render. E.g., linework width, edge darkening width.
- *Edge transition.* Controls the edge transition of the subject in relation to neighboring elements. E.g., edge softening, gaps and overlaps.

3.2.4 *Abstraction-based effects.*

This category contains parameters that direct the abstraction of the rendered subject within different styles.

- *Detail.* Controls the amount of detail at different parts of the stylized render within the subject. E.g., detail blur.
- *Shape.* Controls the amount of shape abstraction/distortion of the subjects. E.g., hand tremors.
- *Blending.* Controls the color blending at different parts of the stylized render. E.g., smudges, color bleeding.

By adhering to these semantics throughout the stylization pipeline, art-directed scenes can predictably change style and, for the most part, keep the intended effects and look of the expressive render. While these semantics are neither final, nor applicable to all styles and effects, they provide a starting point to address cross-stylization paradigms in expressive rendering.

3.3 Implementation

To maximize the use of MNPR by artists and developers alike, we implemented our framework in one of the widely adopted digital content creation packages used in the animation and game industries, Autodesk® Maya®. Relying on this software allowed us to easily give access to our technology to hundreds of artists worldwide (over 500 registered users to date), some of which are already using our framework for their own productions.

3.3.1 *Tools for different levels of control.*

Within MNPR, broken down in the schematic of Figure 11, the 3D scene interaction is entirely managed by Maya. However, enabling the different levels of control for our stylization pipeline required us to develop multiple custom tools, which were developed using Python and PySide2.

Proxy control. In 3D space, we handle the creation of proxy controls from any mesh object, by first disabling the casting and receiving of shadows so that these are not present in the shadow map calculation. We then assign a custom proxy material, which will only render the proxy object to the control maps, once the application data is rasterized. A proxy material preset, loaded from our custom material presets tool, automatically performs these changes, so that the user is only concerned with the art-direction of the "invisible" object.

Mapped control. Any mesh object in the 3D scene can be embedded with mapped control, even proxies. We handle mapped control using vertex color sets, which are created and managed automatically using our custom *PaintFX* tool. The *PaintFX* tool modifies three RGBA vertex color sets that render out the controls for the four effect categories that we previously defined. By using vertex colors (instead of textures, for instance), we have a sparse control representation that can take advantage of Maya's animation capabilities to keyframe effect parameters.

Material control. Mapped parameters can only be rendered from the application data if the material supports them. Our custom real-time ShaderFX material renders the 3D application data into the different render targets required by our stylization pipeline: the albedo—which includes some painterly shading properties—, diffuse and specular targets, any necessary G-buffers and the four stylization control maps. Maya's ShaderFX is a real-time, node-based shader editor that gives us the flexibility to recompile the GPU shader on demand, depending on the operating system and the material properties we require. This mechanism saves unnecessary computation time of optional shading requirements—such as specularity, transparency and procedural noise parameters.

The material control of effects is directed by procedural noise parameters that are also embedded into the ShaderFX material. Our custom *NoiseFX* tool allows the artist to control noise functions from an abstracted interface, avoiding the navigation through each shader graph towards the specific noise parameter in a node. Our materials accumulate the control contributions generated by these noise functions and the previously mapped parameters, found in meshes and proxies, into the stylization control maps—following the scheme found in Table 1.

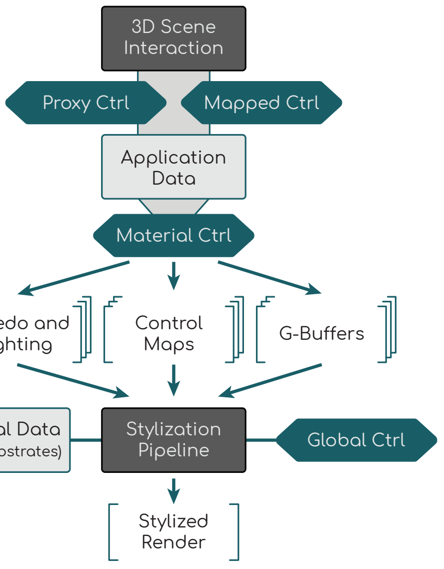


Figure 11: Schematic of MNPR with its different interaction stages within the 3D pipeline.

3.3.2 *Stylization pipeline.*

Once all the data is rendered from the 3D scene, a series of image processing operations, driven by shaders (HLSL or GLSL), are defined and run. These operations conform to the stylization pipeline, which is further driven by the global control parameters and any other external data (e.g., substrate). While the exact implementation of each stylization pipeline is under constant evolution and out of the scope of this paper, we encourage the interested reader to check the code available at the project's Git repository (see Section 6).

The stylization pipeline, powered by our custom plugin, uses the Maya C++ API and takes advantage of the Viewport 2.0 API. However, we extended some API classes and methods to design and abstract a modular framework. This facilitates the creation of stylization pipelines for people who are not used to working with the Maya API, enabling a simple adaptation of art-directed NPR stylization pipelines in three steps:

- define and create attributes for global effect parameters;
- outline the custom stylization pipeline;
- define *NoiseFX* and *PaintFX* controls (Python).

Additionally, we have embedded debugging utilities into MNPR to help the developer debug passes, change color spaces and visualize individual channels and their negative values.

3.3.3 *Lessons learned.*

While working with Maya has propelled the set of features and the pool of artists that we can access, developing for this closed-source software has its caveats. We would like to share a brief list of lessons learned, for any developers or researchers looking into using and expanding MNPR.

- Documentation, example code and user-base of the Viewport 2.0 API is limited.
- Maya will not return certain shader errors when working with GLSL shaders. This led us to develop in HLSL first and then port to GLSL afterwards.
- Keep hardware limitations in mind when rendering to multiple render targets (8 max).
- Packing control parameters into 32 bit floating point targets will not support a simple accumulation of parameters from different levels of control.
- Maya's internal texture painting tool will not work on custom materials, which drove us to map to vertex colors instead.

4 USER EVALUATION

To evaluate MNPR and its usability, we concentrate on two aspects: (1) how effectively developers can implement different styles and (2) how useful artists find the levels of control we provide.

4.1 Developers' evaluation

We adapted the watercolor stylization of Montesdeoca et al. [2017a, 2016, 2017b] while developing our framework, but we needed other styles in order to evaluate it. Therefore, we implemented the oil stylization of Semmo et al. [2015, 2016] (without color quantization) and invited an inexperienced computer science undergraduate student to develop and implement a charcoal stylization as part of a final year project.

Porting over Semmo et al.'s oil paint stylization work was quite straightforward and took approximately 2 weeks, which included re-writing the compute shaders to HLSL and redefining the order of the image processing operations. However, this time did not include the different development challenges that arose from porting a 2D stylization technique into 3D. A particular challenge was porting over the procedural impasto marks in a coherent way under animation. In this pursuit, we extended MNPR to include a way to extract the motion of each vertex by calculating the deltas with their previous position (after deformations) and made use of coherent noise [Kass and Pesare 2011] to advect the procedural noise along the object's motion. While the results are promising, calculating per vertex motion of fully deformable objects is slow and only enabled when real-time interaction is not required.

The charcoal stylization took understandably longer to develop, as it was not a full-time effort, there was no prior shader code and it involved a significant learning journey into image processing and shader development from the developer. Nonetheless, our framework managed to abstract the development process of a graphics engine, allowing to concentrate on the image processing operations and the interplay between them.

MNPR successfully supported both development efforts through the example code/documentation to embed the art-directed parameters within the image-processing operations, following the control scheme at Table 1. Once these parameters were incorporated, the expressive control tools were managed by our framework by simply defining the code to manipulate them. Both of these additional stylization pipelines are featured in our results, showcasing the successful adaptation of styles into MNPR, as well as the cross-stylistic support of art-directed effects. In general, the feedback from these development efforts was quite positive and allowed us to iterate on our framework to improve upon it and extend its functionality.

4.2 Artists' evaluation

We aimed to grant artists the widest range of control in object-space and, thereby, evaluate their preferences and uses at each level of control. For this purpose, we conducted a user study involving experienced artists and engineers to evaluate the usefulness of each control and gather feedback on how to improve them.

A total of 65 users registered their interest to take part in our study and received MNPR with a watercolor stylization pipeline. To bring everyone to the same level of proficiency using the framework

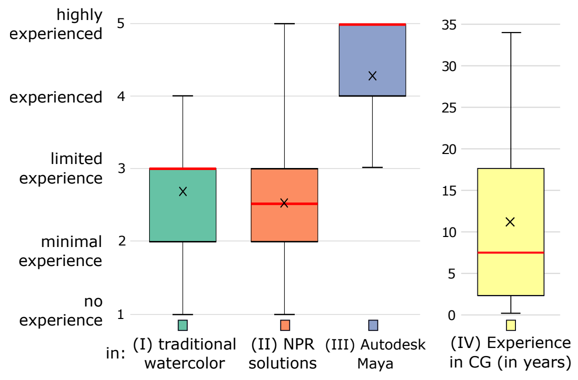
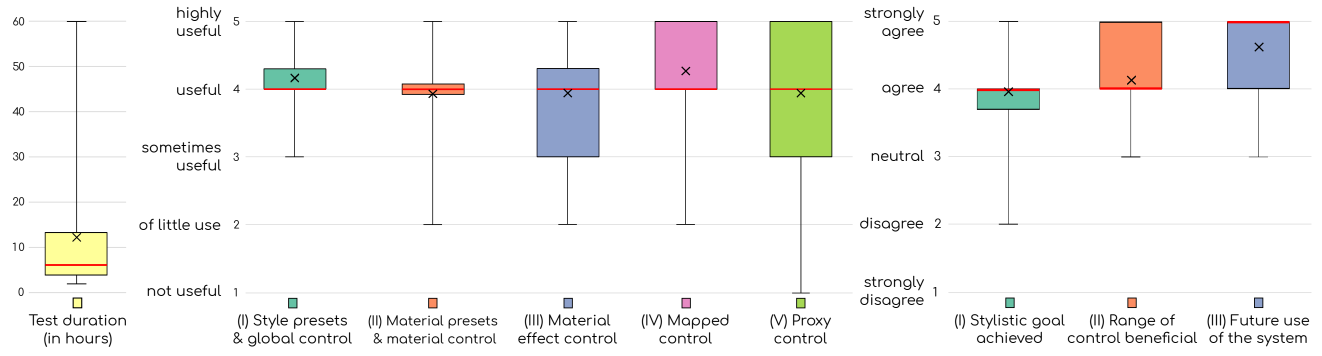


Figure 12: Experience level of users.



(a) Duration **(b)** Levels of control rating **(c)** Overall satisfaction

Figure 13: Results of the user experiment

and its tools, we included tutorials for each level of control and how to use them effectively. After a period of two weeks, 20 participants answered an online questionnaire that helped us gather their usage data and analyze their feedback. The individual responses are available within the supplementary materials of this paper, of which we have consolidated the following.

4.2.1 *Prior experience.*

To assess their responses, we were initially interested in their prior experience (Figure 12) using traditional watercolor, other NPR solutions and Autodesk® Maya®—together with their overall years of experience in a computer graphics related industry. The majority of users had limited to minimal experience using traditional watercolors (I), which is expected as we were targeting 3D artists and engineers. These results resembled their prior experience with other NPR solutions (II), probably reflecting on the scarce dedicated commercial NPR solutions available to date. Nonetheless, our users were mostly experienced with Maya (III) and within a wide spectrum of experience in computer graphics related industries—involving students and veterans alike (IV).

4.2.2 *User study.*

In this study, the participants were asked to commit to at least two hours when testing MNPR and its distinct levels of control. However, their dedication exceeded our expectations, as most participants reported spending substantially more time testing our framework, as observed in Figure 13a. It allowed them to familiarize themselves better with each level of control.

Levels of control (Figure 13b). Style presets and global control (I) were generally perceived as useful, tending towards highly useful in some cases. This result was expected, as these represent the main control over the stylized render and the starting point towards a stylized look. There was a general consensus that the material presets and material controls (II) were useful, with little deviation. We believe this result represents the general awareness of this level of control, especially considering existing photorealistic workflows that rely heavily on material attributes to define the look of objects. Material effect controls (III), with their procedural noise parameters—providing mid-level control for effects—were generally perceived useful. This level of control took the participants the longest to understand, but most found its usefulness after experimenting with it. Mapped control (IV) was considered the most useful level of control by the majority of the participants. This is probably due to its localized nature and intuitive application through painting tools. Finally, proxy control (V), while being completely detached of the rendered subject, was generally found useful. This decoupled approach to art-direct the stylization received the most varied result, with some participants enthusiastically welcoming the addition (see Figure 14), while others not seeing their usefulness.

Overall satisfaction (Figure 13c). Overall, the participants tended to agree that they achieved their stylistic goal using our framework (I). We also asked the participants if they previously used other NPR solutions, for their current workflow; and if they would have achieved their desired stylization faster using our framework. While they were mostly using offline workflows—which involve several stages with different software—their answers were mostly neutral, but with a tendency towards agreeing. We believe these responses were positive, considering that stylistic goals may differ widely between users and that they were constrained to only one stylization pipeline within our framework, with limited learning time. We expect this to change when more stylization pipelines are developed and MNPR matures in the public domain.

Of special importance to our study and our research is that most agree and even strongly agree that the range of control benefited their stylization (II), reinforcing our work and encouraging us to continue our pursuit towards art-directed tools. However, as one participant pointed out, it is important to find a balance within control, as this might distract users from the "bigger picture". Finally, an overwhelming majority of the participants strongly agreed that they would consider using our framework for their future projects (III).

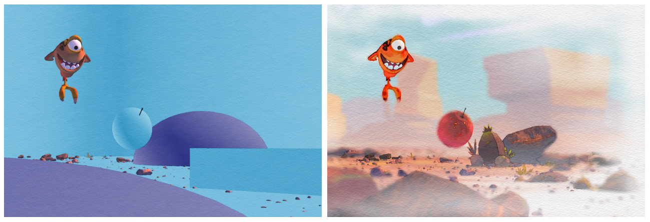


Figure 14: A screenshot of a scene by a test user where proxies (blue, left) are mostly used to drive the stylization (right).

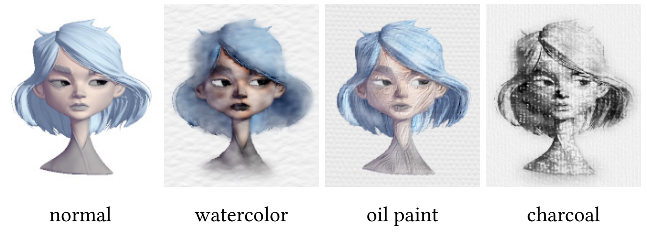


Figure 15: Art-directed cross-stylization character example. Miss Seagull model, © Julien Kaspar.

5 RESULTS AND DISCUSSION

Based on our experiments and reinforced by our user evaluation, we can conclude that the interaction spectrum, given by our different levels of control, was quite useful for artists—assisting the stylization process by enabling a faster and more versatile workflow. Additionally, since the interaction happens in real-time, the art-direction and rendering are performed concurrently, allowing the artists to see their changes immediately.

We tested all these levels of control, working in unison, using three distinct stylizations: watercolor, oil paint and charcoal. Our results are presented throughout Figure 16. Considering the experimental nature of these stylization pipelines, we have managed to produce satisfying results.

Finally, by generalizing our control semantics, our framework permitted the art-direction within one style to be visualized in others, as seen in Figure 1, Figure 15 and Figure 17. These stylization pipelines all run within MNPR in real-time and the user can interactively switch between them. The individual performances of each stylization pipeline have been measured in Table 2 with the following configuration: NVIDIA® GeForce GTX 1080, Intel® Xeon® E5-2609 @2.4Ghz and 16Gb of RAM using Maya 2017. Our framework, without any custom stylization pipeline, involves five different passes to render the 3D scene, adjust or load data, provide channel or color-space debugging utilities and perform anti-aliasing. Our most complex pipeline, with 27 different image-processing passes (including 6 separable filter passes) still manages to run at 60 frames per second when rendering Figure 15.

Table 2: Performance for the different styles of Figure 15, 288.800 triangles, screen resolution of 1920×1080 pixels.

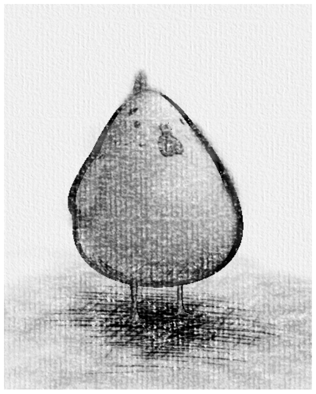
Style	Passes	Separable	FPS (avg)
Framework	5	0	250
Watercolor	13	2	100
Oil paint	27	6	60
Charcoal	17	6	70

6 CONCLUSION

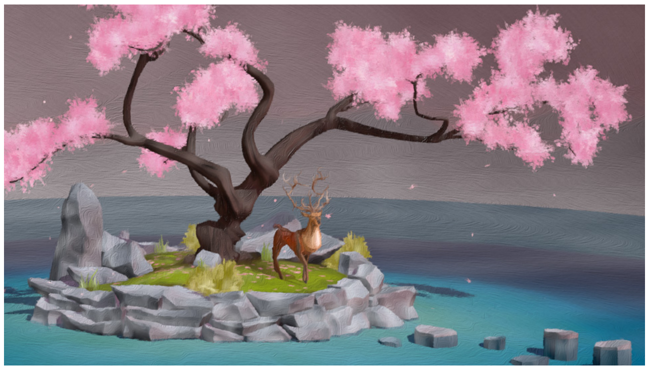
We have presented MNPR, a framework for expressive non-photorealistic rendering of 3D computer graphics, covering the interaction spectrum to optimize the workflow of users with different preferences and skill levels. We have validated this by conducting a user study with 20 experienced artists and engineers in the targeted industries, who shared their feedback and perceived usefulness of the developed tools for each level of control.

(a) Baba Yaga's hut model, 🅭🅯 Inuciian.

(b) Wiz Biz model, 🅭🅯 Tom Robinson.



(c) Chicken Friend ;o model, 🅭🅯 MahoSway.



(d) SIKA model, © Marine Quiviger.

Figure 16: Rendered frames using our framework. Please refer to the accompanying video for more results.

To enable art-direction in multiple styles, we also proposed sufficiently generic, yet meaningful semantics for parameterization, which allows users to interactively visualize their art-direction in manifold styles. This was tested and successfully showcased within our implemented stylization pipelines involving the watercolor, oil and charcoal mediums.

Finally, we have shared some insight towards our implementation in Autodesk® Maya® and are hereby releasing the source code to augment this contribution and motivate adoption in the *Expressive* community—as a practical answer to Isenberg's 2016 call towards designing the interaction with, or for algorithmic

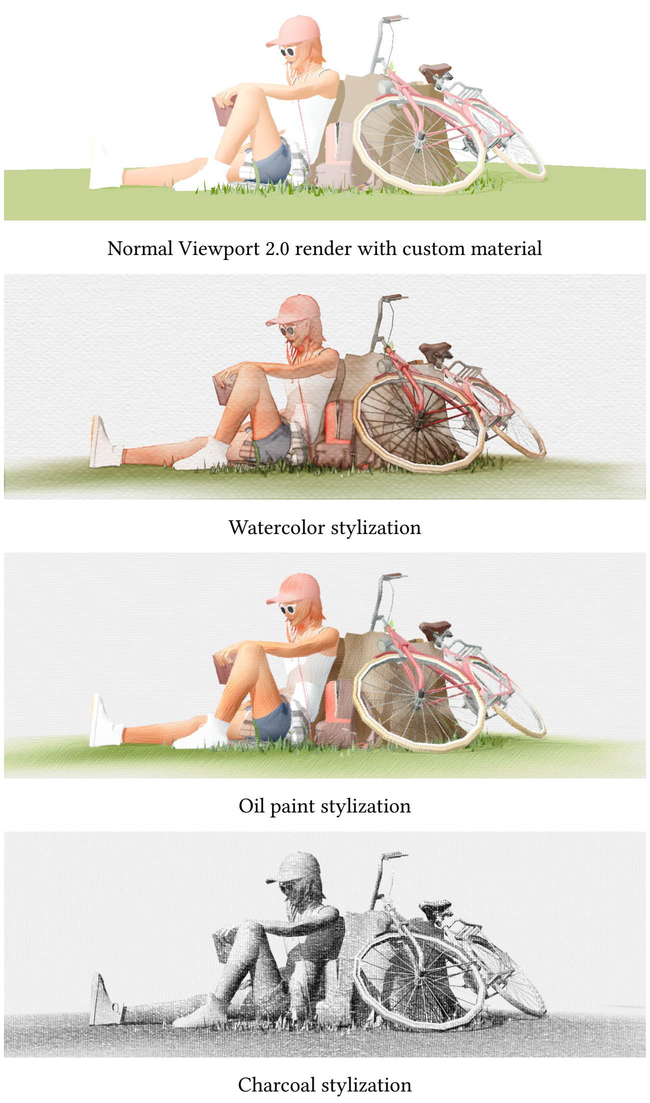


Figure 17: Art-directed cross-stylization example. Summer Afternoon model, © Stevie Brown.

support. The source code can be found at our institutional data repository [Montesdeoca 2018] and at the project's Git repository (http://github.com/semontesdeoca/MNPR). With this work, we hope to make Expressive research more accessible for the target users, validating our methods and inspiring new contributions from the research community.

Limitations and future work. While MNPR provides a useful starting point towards bringing NPR research to the hands of artists, it still remains to be tested and refined with other styles and in production. For example, we have not implemented a method to retain the motion coherence of substrate-based effects yet, as this is still an open problem that needs to be convincingly solved [Bousseau et al. 2006; Bénard et al. 2009, 2013; Hays and Essa 2004]. A production-ready solution would need to take this into account if temporal continuity is desired i.e., when alternating substrate patterns—which happen naturally with traditionally painted animation that present substrate-based effects—are not desired. We have also shared some limitations through the lessons learned during our implementation (please refer to Section 3.3). Some of these limitations pertain to the current available hardware, but others pertain to Autodesk Maya and its closed-source nature, from which we hope to spur support in the future and work together towards having more expressive NPR available to artists and productions alike.

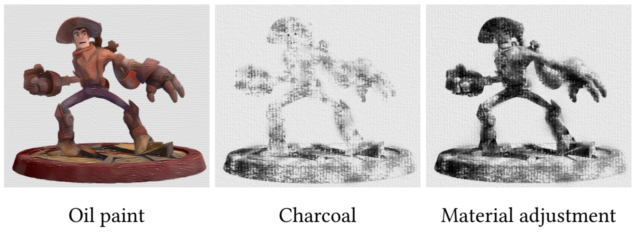


Figure 18: Cross-stylization: oil paint to charcoal, Steam Cowboy model © Black Spire Studio.

Our cross-stylistic semantics enable art-direction in multiple styles in a predictive manner. However, although we have tested these style control semantics with watercolor, oil paint and charcoal with relative success, their applicability still needs to be further proven in other styles e.g., hatching [Suarez et al. 2016] or even Cubism [Arpa et al. 2012]. Additionally, some styles require more than just a change in stylization pipelines and effects for the subject to acquire the intended look. A clear example is given when contrast in a specific style is achieved through color variations and another through lightness values, as is the case with watercolor/oil and charcoal. As illustrated in Figure 18, a simple conversion from an oil paint stylization to a charcoal stylization will lack the contrast that defines the form of the subject. Notwithstanding, in this case, an adjustment to the materials in the scene through an additional tool, can let us modify the diffuse behavior to accommodate the look and bring the necessary contrast back.

Finally, our tools to art-direct the stylization at different levels of control can be subject to further improvement and better alternatives. Based on the usefulness found by artists, we can especially argue that different ways to provide material and proxy control of effects could provide a promising area of future research, due to their rather unexplored nature and variance presented in their usefulness.

ACKNOWLEDGMENTS

Special thanks go to Yee Xin Chiew and all the participants in our user study for testing MNPR, sharing their feedback and enduring the hiccups during its development process. We also thank Hans-Martin Rall and Jürgen Döllner for their support during this endeavor. Our gratitude is also with our colleagues for their comments and suggestions, together with the anonymous reviewers for their constructive feedback. We would like to thank the Multi-plAtform Game and Innovation Centre (MAGIC) and the Hasso Plattner Institute for Digital Engineering (HPI) for supporting this international collaboration. This research is supported by the Interdisciplinary Graduate School (IGS) and the School of Computer

Science and Engineering (SCSE) at the Nanyang Technological University, Singapore, and was partly funded by the Federal Ministry of Education and Research (BMBF), Germany, for the AVA project 01IS15041B.

REFERENCES

Sami Arpa, Abdullah Bulbul, Tolga Capin, and Bulent Ozguc. 2012. Perceptual 3D rendering based on principles of analytical cubism. *Computers & Graphics* 36, 8 (2012), 991 – 1004. DOI:https://doi.org/10.1016/j.cag.2012.06.003 Graphics Interaction Virtual Environments and Applications 2012.

Adrien Bousseau, Matt Kaplan, Joëlle Thollot, and François X. Sillion. 2006. Interactive Watercolor Rendering with Temporal Coherence and Abstraction. In *Proceedings of International Symposium on Non-photorealistic Animation and Rendering*. ACM, 141–149. DOI:https://doi.org/10.1145/1124728.1124751

Pierre Bénard, Adrien Bousseau, and Joëlle Thollot. 2009. Dynamic Solid Textures for Real-Time Coherent Stylization. In *Proceedings of the Symposium on Interactive 3D graphics and games*. ACM, 121–127. DOI:https://doi.org/10.1145/1507149.1507169

Pierre Bénard, Forrester Cole, Michael Kass, Igor Mordatch, James Hegarty, Martin Sebastian Senn, Kurt Fleischer, Davide Pesare, and Katherine Breeden. 2013. Stylizing Animation By Example. *ACM Transactions on Graphics* 32, 4 (2013), 119. DOI:https://doi.org/10.1145/2461912.2461929

Luis Cardona and Suguru Saito. 2015. Hybrid-space Localized Stylization Method for View-dependent Lines Extracted from 3D Models. In *Proceedings of International Symposium on Non-photorealistic Animation and Rendering*. Eurographics Association, 79–89. DOI:https://doi.org/10.2312/exp.20151181

Eric Daniels. 1999. Deep Canvas in Disney's Tarzan. In *Proc. SIGGRAPH Conference Abstracts and Applications*. ACM, 200. DOI:https://doi.org/10.1145/311625.312010

Jakub Fišer, Ondřej Jamriška, Michal Lukáč, Eli Shechtman, Paul Asente, Jingwan Lu, and Daniel Sýkora. 2016. StyLit: Illumination-guided Example-based Stylization of 3D Renderings. *ACM Transactions on Graphics* 35, 4 (2016), 92:1–92:11. DOI:https://doi.org/10.1145/2897824.2925948

Moritz Gerl and Tobias Isenberg. 2013. Interactive Example-based Hatching. *Computers & Graphics* 37, 1–2 (Feb.–April 2013), 65–80. DOI:https://doi.org/10.1016/j.cag.2012.11.003

Amy A. Gooch, Jeremy Long, Li Ji, Anthony Estey, and Bruce S. Gooch. 2010. Viewing Progress in Non-photorealistic Rendering Through Heinlein's Lens. In *Proceedings of International Symposium on Non-photorealistic Animation and Rendering*. ACM, 165–171. DOI:https://doi.org/10.1145/1809939.1809959

Stéphane Grabli, Emmanuel Turquin, Frédo Durand, and François X. Sillion. 2010. Programmable Rendering of Line Drawing from 3D Scenes. http://freestyle.sourceforge.net/, *ACM Transactions on Graphics* 29, 2 (April 2010), 18:1–18:20. DOI:https://doi.org/10.1145/1731047.1731056

N. Halper, T. Isenberg, and F. Ritter. 2003. OPENNPAR: a system for developing, programming, and designing non-photorealistic animation and rendering. In *11th Pacific Conference on Computer Graphics and Applications, 2003. Proceedings.* 424–428. DOI:https://doi.org/10.1109/PCCGA.2003.1238288

Nick Halper, Stefan Schlechtweg, and Thomas Strothotte. 2002. Creating Non-photorealistic Images the Designer's Way. In *Proceedings of the 2Nd International Symposium on Non-photorealistic Animation and Rendering (NPAR '02)*. ACM, New York, NY, USA, 97–ff. DOI:https://doi.org/10.1145/508530.508548

Pat Hanrahan and Paul Haeberli. 1990. Direct WYSIWYG Painting and Texturing on 3D Shapes. *SIGGRAPH Comput. Graph.* 24, 4 (Sept. 1990), 215–223. DOI:https://doi.org/10.1145/97880.97903

Alex Harvill. 2007. *Effective Toon-Style Rendering Control Using Scalar Fields.* Memo. http://graphics.pixar.com/library/ToonRendering/index.html

James Hays and Irfan Essa. 2004. Image and Video Based Painterly Animation. In *Proceedings of the 3rd International Symposium on Non-photorealistic Animation and Rendering (NPAR '04)*. ACM, New York, NY, USA, 113–120. DOI:https://doi.org/10.1145/987657.987676

Tobias Isenberg. 2016. Interactive NPAR: What type of tools should we create?. In *Proceedings of International Symposium on Non-photorealistic Animation and Rendering*. The Eurographics Association, 89–96. DOI:https://doi.org/10.2312/exp.20161067

Robert D. Kalnins, Philip L. Davidson, and David M. Bourguignon. 2003. Jot. (2003). http://jot.cs.princeton.edu/

Robert D. Kalnins, Lee Markosian, Barbara J. Meier, Michael A. Kowalski, Joseph C. Lee, Philip L. Davidson, Matthew Webb, John F. Hughes, and Adam Finkelstein. 2002. WYSIWYG NPR: Drawing Strokes Directly on 3D Models. *ACM Transactions on Graphics* 21, 3 (2002), 755–762. DOI:https://doi.org/10.1145/566654.566648

Michael Kass and Davide Pesare. 2011. Coherent Noise for Non-photorealistic Rendering. *ACM Trans. Graph.* 30, 4, Article 30 (2011), 6 pages. DOI:https://doi.org/10.1145/2010324.1964925

Santiago E. Montesdeoca. 2018. MNPR. (2018). DOI:https://doi.org/10.21979/N9/KU4B6S

Santiago E. Montesdeoca, Hock Soon Seah, Pierre Bénard, Romain Vergne, Joëlle Thollot, Hans-Martin Rall, and Davide Benvenuti. 2017a. Edge- and substrate-based effects for watercolor stylization. In *Proceedings of International Symposium on Non-photorealistic Animation and Rendering*. ACM, New York, NY, USA, 2:1–2:10. DOI:https://doi.org/10.1145/3092919.3092928

Santiago E Montesdeoca, Hock Soon Seah, and Hans-Martin Rall. 2016. Art-directed Watercolor Rendered Animation. In *Proceedings of International Symposium on Non-photorealistic Animation and Rendering*. The Eurographics Association, 51–58. DOI:https://doi.org/10.2312/exp.20161063

Santiago E. Montesdeoca, Hock Soon Seah, Hans-Martin Rall, and Davide Benvenuti. 2017b. Art-directed watercolor stylization of 3D animations in real-time. *Computers & Graphics* 65 (2017), 60 – 72. DOI:https://doi.org/10.1016/j.cag.2017.03.002

Petra Neumann, Tobias Isenberg, and Sheelagh Carpendale. 2007. NPR Lenses: Interactive Tools for Non-photorealistic Line Drawings. In *Smart Graphics*, Andreas Butz, Brian Fisher, Antonio Krüger, Patrick Olivier, and Shigeru Owada (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 10–22. DOI:https://doi.org/10.1007/978-3-540-73214-3_2

David H Salesin. 2002. Non-photorealistic animation & rendering: 7 grand challenges. (June 2002). Keynote talk at NPAR.

Stefan Schlechtweg, Tobias Germer, and Thomas Strothotte. 2005. RenderBots—Multi-Agent Systems for Direct Image Generation. *Computer Graphics Forum* 24, 2 (2005), 137–148. DOI:https://doi.org/10.1111/j.1467-8659.2005.00838.x

Johannes Schmid, Martin Sebastian Senn, Markus Gross, and Robert W. Sumner. 2011. OverCoat: An Implicit Canvas for 3D Painting. *ACM Transactions on Graphics* 30, 4, Article 28 (July 2011), 10 pages. DOI:https://doi.org/10.1145/2010324.1964923

Thorsten-Walther Schmidt, Fabio Pellacini, Derek Nowrouzezahrai, Wojciech Jarosz, and Carsten Dachsbacher. 2016. State of the Art in Artistic Editing of Appearance, Lighting and Material. *Computer Graphics Forum* 35, 1 (2016), 216–233. DOI:https://doi.org/10.1111/cgf.12721

Amir Semmo, Tobias Dürschmid, Matthias Trapp, Mandy Klingbeil, Jürgen Döllner, and Sebastian Pasewaldt. 2016. Interactive Image Filtering with Multiple Levels-of-control on Mobile Devices. In *Proceedings SIGGRAPH ASIA Mobile Graphics and Interactive Applications*. ACM, New York, 2:1–2:8. DOI:https://doi.org/10.1145/2999508.2999521

Amir Semmo, Tobias Isenberg, and Jürgen Döllner. 2017. Neural Style Transfer: A Paradigm Shift for Image-based Artistic Rendering?. In *Proceedings of International Symposium on Non-photorealistic Animation and Rendering*. ACM, 5:1–5:13. DOI:https://doi.org/10.1145/3092919.3092920

Amir Semmo, Daniel Limberger, Jan Eric Kyprianidis, and Jürgen Döllner. 2015. Image Stylization by Oil Paint Filtering Using Color Palettes. In *Proceedings of the Workshop on Computational Aesthetics*. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 149–158. DOI:https://doi.org/10.2312/exp.20151188

Amir Semmo, Daniel Limberger, Jan Eric Kyprianidis, and Jürgen Döllner. 2016. Image stylization by interactive oil paint filtering. *Computers & Graphics* 55 (2016), 157 – 171. DOI:https://doi.org/10.1016/j.cag.2015.12.001

Jordane Suarez, Farès Belhadj, and Vincent Boyer. 2016. Real-time 3D rendering with hatching. *The Visual Computer* (2016), 1–16. DOI:https://doi.org/10.1007/s00371-016-1222-3

Brian Whited, Eric Daniels, Michael Kaschalk, Patrick Osborne, and Kyle Odermatt. 2012. Computer-assisted Animation of Line and Paint in Disney's Paperman. In *Proc. SIGGRAPH Talks*. ACM, New York, NY, USA, Article 19, 1 pages. DOI:https://doi.org/10.1145/2343045.2343071

Holger Winnemöller. 2013. NPR in the Wild. In *Image and Video-Based Artistic Stylisation*, Paul Rosin and John Collomosse (Eds.). Computational Imaging and Vision, Vol. 42. Springer-Verlag London, Book 17, 353–372. DOI:https://doi.org/10.1007/978-1-4471-4519-6