

CAPTURING AND EDITING THE VISUAL WORLD FOR ARTISTIC APPLICATIONS

A Ph.D. Thesis by Jose Ignacio Echevarria

Supervised by Dr. Diego Gutierrez



CAPTURING AND EDITING THE VISUAL WORLD FOR
ARTISTIC APPLICATIONS

JOSE I. ECHEVARRIA

SUPERVISOR: DIEGO GUTIERREZ

Tesis Doctoral - Ingeniería Informática
Departamento de Informática e Ingeniería de Sistemas
Universidad de Zaragoza

September 2016

To my parents, my sister, and Cris.

For all the good things to come.

ABSTRACT

In this Thesis we focus on providing practical solutions to different problems related with the capture and manipulation of data, and simulation of processes from the real world.

We dedicate one part of this work to explore novel use cases for tablet devices, leveraging the natural user interaction they enable to prototype more engaging image capture and editing tools. We present a novel framework for the simulation of the craftsmanship involved in analog photography techniques and other interesting optical manipulations.

A second part is devoted to image reconstruction algorithms that share the use of perceptual cues to compensate for the missing data. First, we introduce our SMAA anti aliasing filter for real time applications, where a comprehensive morphological analysis is performed to provide smooth but sharp results. Next, a simple procedure is described to capture extended dynamic range images with mobile devices, using computational photography techniques.

The last part deals with the capture of 3D data from the real world. We present a new depth from defocus algorithm for obtaining detailed depth maps of scenes. Finally, we describe the first system for stylized capture of hair, producing results suitable for 3D fabrication inspired by the abstraction process performed by sculptors.

MEASURABLE CONTRIBUTIONS

This Thesis has led to the following results, which can be found in detail in Section 1.5:

- 6 JCR-indexed journal publications (1 of them ACM Transactions on Graphics) [154, 153, 105, 61, 18, 58]
- 2 peer-reviewed conference publications [60, 59]
- 1 granted patent [206], 2 additional patent applications [25].
- 2 book chapters [107, 104]
- 2 research internships (6 months in total) at Adobe Systems
- 1 research internship (4 months) at Disney Research Zurich
- 2 research stays (2 months in total) at Porto Interactive Center
- 3 best paper awards
- 15+ invited talks
- Participation in 7 research projects
- Reviewer for 4 journals and 9 international conferences, and program committee member for 1 international conference
- 3 supervised TFGs and 1 more in progress

RESUMEN

En esta Tesis nos centramos en proporcionar soluciones prácticas a diferentes problemas relacionados con la captura y manipulación de datos, y simulación de procesos del mundo real.

Así pues, dedicamos la primera parte de este trabajo a explorar nuevos usos para dispositivos móviles, haciendo uso de la natural e intuitiva interacción de usuario que habilitan, para prototipar herramientas de captura y edición de imagen más atractivas para el usuario. Presentamos un novedoso *framework* para la simulación del trabajo manual involucrados en técnicas de fotografía analógica y otras interesantes manipulaciones ópticas.

Una segunda parte versa sobre algoritmos de reconstrucción de imágenes 2D que comparten el uso de mecanismos de la percepción humana para compensar la falta de información. Primero presentamos *SMAA*, un filtro de *antialiasing* para aplicaciones en tiempo real, donde un exhaustivo análisis morfológico es llevado a cabo para proporcionar resultados suaves pero nítidos. A continuación, describimos un simple proceso para capturar fotografías con rango dinámico extendido mediante dispositivos móviles, usando técnicas de fotografía computacional.

La última parte trata sobre la captura de datos 3D del mundo real. Presentamos un nuevo algoritmo de *depth from defocus* para obtener detallados mapas de profundidad de escenas cotidianas. Y finalmente, describimos el primer sistema para la captura estilizada de peinados, produciendo resultados óptimos para su impresión en 3D, inspirados por los procesos de abstracción que los escultores tradicionales llevan a cabo.

ACKNOWLEDGEMENTS

There have been lots of people that have had different roles and impact on me and my work during these years, here are some of the more meaningful ones.

Diego, for teaching me the way of the researcher. Before joining his group, I had no clue what was research about; nowadays it is an essential part of me.

Adolfo Muñoz for the smooth introduction to research, and *Jorge Jimenez* for the frantic days. *Jorge Lopez-Moreno* for all the (non)sensical talk.

Gregg Wilensky and *Aravind Krishnaswamy* for bringing me to Adobe for the first time, a dream come true. *Byungmoon Kim* for his continued support and trust in me through the following years, and all the things he keeps teaching me.

Past and present members of the *Graphics & Imaging Lab* for all the help, discussions, bakery, parties and laughs. Special mentions to *Elena Garces* and *Adrian Jarabo*: we started together and we (almost) finished together, supporting each other in so many ways. *Carlos Aliaga* and *Cristina Tirado*, for completing the crew. And *Susana Castillo*, always remembered.

Thabo Beeler and *Derek Bradley* for the opportunity at Disney Research Zurich and their trust: it was a lot of work, but I had a blast. *Alex Sorkine-Hornung* for being the starter. And the usual suspects at the "lunch and paper" sessions, it was great to see so many great minds at work.

Nathan Carr, for paving the way for my future. And my fellow summer interns of 2015 *Lena Gieseke*, *Jakub Fiser* and *Aron Monszpart*.

The students I have supervised: *Paz Hernando*, *Patrick Moosbrugger*, *Mariel Munilla*, *Alba Samanes* and *Stephen Bailey*. I learnt a lot from teaching them, and I hope they did too.

The anonymous reviewers for their feedback: while rejections always hurt, they made our work better. My collaborators and other people that helped in different ways in each paper. *Pilar Enguita* and *Ana Gimeno* for making sure the debts were always paid. And all the inspiring researchers in this amazing field.

My family, friends and *Viri*: although I believe they never understood all the misery, extra hours, and what were those ***** deadlines about; they were always by my side.

This research has been partially funded by the European Commission, Seventh Framework Programme, through the projects GOLEM (Marie Curie IAPP, grant agreement no.: 251415) and VERVE (Information and Communication Technologies, grant agreement no.: 288914); the Spanish Ministry of Science and Technology (TIN2010-21543); the Gobierno de Aragón (projects OTRI 2009/0411, CTPP05/09 and TAMA); Adobe Systems and Disney Research Zurich.

CONTENTS

I	INTRODUCTION & OVERVIEW	1
1	INTRODUCTION	3
1.1	Creative manipulation of digital photographs	3
1.2	Perceptual reconstruction of 2D images	4
1.3	3D Reconstruction and stylization	5
1.4	Goal & Overview	6
1.5	Contributions and Measurable Results	7
II	CREATIVE MANIPULATION OF DIGITAL PHOTOGRAPHS	13
2	ALTERNATIVE PHOTOGRAPHIC PROCESSES	15
2.1	Introduction	15
2.2	Related work	17
2.3	The analog photographic pipeline	18
2.4	Computational pipeline	19
2.5	Hybrid fluids simulation	25
2.6	Characteristic curves	27
2.7	Implementation	29
2.8	Results	30
2.9	Conclusions and Future Work	31
	Appendix 2.A Characteristic curves	35
	Appendix 2.B Results	41
3	LIQUID-ON-LENS SIMULATION	45
3.1	Introduction	45
3.2	Overview	45
3.3	Implementation details	46
3.4	Results	47
3.5	Conclusion	49
III	PERCEPTUAL RECONSTRUCTION OF 2D IMAGES	51
4	SMAA: SUBPIXEL MORPHOLOGICAL ANTIALIASING	53
4.1	Introduction	53
4.2	Related Work	54
4.3	Morphological Antialiasing	57
4.4	SMAA: Features and Algorithm	59
4.5	Results	68
4.6	Conclusions	70
	Appendix 4.A Supplementary material	72
5	MOBILE HDR IMAGING	79
5.1	Introduction	79
5.2	Previous Work	80
5.3	The Frankencamera architecture	81
5.4	HDR Acquisition	82
5.5	Implementation	83
5.6	Results and Discussion	85
5.7	Conclusions and Future Work	87
IV	3D RECONSTRUCTION AND STYLIZATION	89
6	FAST DEPTH FROM DEFOCUS FROM FOCAL STACKS	91
6.1	Introduction	91

6.2	Related Work	92
6.3	Background	94
6.4	Algorithm	95
6.5	Results	100
6.6	Conclusions	102
	Appendix 6.A Least Squares Function Analysis	105
7	CAPTURING AND STYLIZING HAIR FOR 3D FABRICATION	109
7.1	Introduction	109
7.2	Related Work	111
7.3	Method Overview	113
7.4	Data Acquisition	114
7.5	Color Initialization	115
7.6	Color Stylization	118
7.7	Geometry Stylization	123
7.8	Results	125
V	CONCLUSION	135
8	CONCLUSIONS AND FUTURE WORK	137
	BIBLIOGRAPHY	141

LIST OF FIGURES

Figure 1.1	Examples of captured BSSRDFs.	9
Figure 1.2	Results from our facial wrinkles simulations.	9
Figure 2.1	Results from our simulations of the wet plate collodion and cyanotype techniques.	16
Figure 2.2	Examples of real analog alternative processes.	17
Figure 2.3	Demonstration of different steps of the wet collodion process.	18
Figure 2.4	Overview of our simulation of the different steps of the analog pipeline.	20
Figure 2.5	Temporal evolution of the exposure, development, fixing and toning steps.	21
Figure 2.6	Spectral sensitivities of analog and digital sensors.	22
Figure 2.7	Spectral sensitivity variations based on the chemical mix.	23
Figure 2.8	Toning process.	24
Figure 2.9	Processing of real measured data to create our characteristic curve model.	28
Figure 2.10	Input images for our results.	30
Figure 2.11	Collodion simulations with subtle heterogeneities.	31
Figure 2.12	Wet plate collodion simulations with more visible modulations from the liquids.	33
Figure 2.13	Simulations of rubytypes, cyanotypes, multitoning and free-style techniques.	34
Figure 2.14	Examples of different toners.	34
Figure 2.15	Halide concentration curves.	36
Figure 2.16	Iodide to bromide ratio curves.	36
Figure 2.17	Cadmium iodide curves.	37
Figure 2.18	Setting time curves	37
Figure 2.19	Development time curves	38
Figure 2.20	Reference image for the exploration of the parameters of our curve model.	39
Figure 2.21	Iodide/bromide ratio exploration.	39
Figure 2.22	Cadmium iodide ratio exploration.	39
Figure 2.23	Halide concentration exploration.	40
Figure 2.24	Setting time exploration.	40
Figure 2.25	Development time exploration.	40
Figure 2.26	Results from the demonstration video.	43
Figure 3.1	Real examples of photos captured through refractive objects and liquids.	46
Figure 3.2	Overview of the system.	47
Figure 3.3	Refraction effects.	47
Figure 3.4	Screenshot of the prototype.	48
Figure 3.5	Examples from our liquid on lens simulation.	49
Figure 4.1	Results of SMAA integrated in the in the <i>Crysis 2</i> game.	55
Figure 4.2	Core concept of MLAA.	59
Figure 4.3	MLAA overview.	60
Figure 4.4	SMAA features.	60

Figure 4.5	Local contrast adaptation.	61
Figure 4.6	Sharp geometric features preservation.	62
Figure 4.7	Examples of aliased diagonals.	63
Figure 4.8	Diagonal patterns.	63
Figure 4.9	Accurate search of the crossing edges.	64
Figure 4.10	SMAA convergence to MSAA.	66
Figure 4.11	Correct subpixel offsets for coverage computation.	67
Figure 4.12	Temporal reprojection.	68
Figure 4.13	SMAA comparisons against MLAA and SSAA.	71
Figure 4.14	Comparison of features in different popular antialiasing techniques.	73
Figure 4.15	Comparison of features in different popular antialiasing techniques.	74
Figure 4.16	Comparison of features in SMAA against SSAA.	75
Figure 4.17	Comparison of features in SMAA against SSAA.	76
Figure 4.18	Comparison of SMAA against MSAA.	77
Figure 4.19	Pre-resolve filtering of multiple samples.	78
Figure 4.20	Post-resolve filtering of multiple samples.	78
Figure 5.1	Overview of the Frankencamera architecture.	82
Figure 5.2	Nokia N900.	82
Figure 5.3	Results from our implementation.	85
Figure 5.4	Comparison between our implementation and Nokia's one.	86
Figure 5.5	White balance application.	86
Figure 6.1	Circle of confusion plots for different focal distances.	92
Figure 6.2	Image formation model through a simple lens.	94
Figure 6.3	Example of the different steps of our algorithm.	98
Figure 6.4	Results obtained from synthetic datasets.	100
Figure 6.5	Relative local and global error estimations.	101
Figure 6.6	Our proposed global accuracy metric.	101
Figure 6.7	Results from real scenes.	103
Figure 6.8	Additional result from real scenes.	103
Figure 6.9	Comparison between accurate and approximated focus positions	104
Figure 6.10	Local maximizers and minimizers.	105
Figure 6.11	Global minimizer.	108
Figure 7.1	Examples of sculptures created by traditional and digital artists.	110
Figure 7.2	Example results from our method.	111
Figure 7.3	Method overview.	113
Figure 7.4	Capture setup.	114
Figure 7.5	Proxy reconstruction and hair region mask.	115
Figure 7.6	Color initialization.	116
Figure 7.7	Smoothing and shocking operations for hair stylization.	119
Figure 7.8	Sampling over the 3D mesh.	121
Figure 7.9	Combination of orientation tensors.	121
Figure 7.10	Color stylization pipeline.	123
Figure 7.11	Wisp profiles.	124
Figure 7.12	Results for furry objects.	125
Figure 7.13	Results for the same person with different hairstyles.	127
Figure 7.14	Results for the same person with different hairstyles.	128
Figure 7.15	Facial hair results.	129

Figure 7.16	Comparison against previous work.	130
Figure 7.17	Comparison with a different pipeline for 3D printable customized figurines.	130
Figure 7.18	Intermediate steps of our method.	131
Figure 7.19	Different stylization levels and combinations.	132
Figure 7.20	Actual color and monochrome printouts.	132
Figure 7.21	Front and back views from several printouts from our method.	133

LIST OF TABLES

Table 2.1	Parameters for the result shown.	41
Table 2.2	Parameters for the result shown.	42
Table 4.1	Features and computational cost of popular antialiasing techniques.	58
Table 5.1	Exposure times that maximize the dynamic range of the N900.	83

Part I

INTRODUCTION & OVERVIEW

INTRODUCTION

Computer graphics and digital imaging have come a long way when it comes to content generation, which can stay in the virtual world or end up in the real one by means of fabrication or printing. This process can start from zero, creating everything from scratch directly with a computing device; or it can start by capturing data from the real world. In any case, the user interacts with digital systems and follows processes that have been designed to be practical and efficient. Most of the times because of that, limitations appear in terms of data captured, the data generated, or the degrees of freedom available to the user while using such systems.

In this Thesis, we focus on some of these limitations in different contexts: creative manipulation of digital photographs, perceptual reconstruction of 2D images, and reconstruction of 3D models from the real world. In the following we describe them and the approaches followed by our solutions.

1.1 CREATIVE MANIPULATION OF DIGITAL PHOTOGRAPHS

Probably the biggest change suffered by traditional photography was the transition from analog to digital. With it, what previously was a process involving light, chemicals and the physical interactions to control it; turned into light, electrical signals and their processing. This paved the way to the creation of very powerful pipelines, with plenty of control during the capture process with the nowadays ubiquitous digital cameras, and even more control provided by processing software and digital tools, whom we see every year to perform actions previously only in our wishlists [78, 136, 210]. Some of these tools are more intuitive than others, but in general it could be argued that desktop image editing software is mostly used by professionals and enthusiasts, perhaps because of a deceiving perception of technical difficulty.

However, things changed abruptly some years ago with the introduction of mobile devices equipped with digital cameras, processing units with enough computing power to perform advanced image editing, big high resolution screens and the novelty of additional touch and gestures based input. This new scenario made the whole imaging pipeline more streamlined than ever: capture, processing and display of digital images is now performed on the same device; being available to any user of such devices. Maybe because of this streamlined and mobile nature, and the new user base that came with it, image editing on mobiles devices feels currently like a watered-down version of its desktop counterpart. This means, click and drag operations everywhere, with lots of templates to browse through with a finger flick.

Unexpectedly, filter *apps* (the ones that abuse this paradigm the most) have become hugely popular and relevant these days, all of them consisting in a limited set of pre-defined templates that change color attributes and apply texture overlays. However, being valid tools for quick content generation to share in social media, they fall short as artistic tools given the limited user experiences they provide. More interesting, though, is the popularity of filters that try achieve a rawer and more imperfect look and feel, usually inspired by analog photographic techniques.

All of this presents a great opportunity to revisit the real photographic processes that produce these popular looks, to explore their potential for creating novel digital experiences and image editing tools. While analog photography simulation is not something new [70], the fact that it usually focus on the final results rather than the process itself, leaves still room for exploration. In this line of research, we focus on studying photographic processes and manual manipulations that are frequent in analog photography, to create image editing tools with a newfound sense of craftsmanship and serendipity. The novel interaction metaphors resulting from there will require constant user interaction, which coupled with detailed chemical and physical simulations of their real counterparts, will lead naturally to intricate and unique results. Leveraging the natural user interaction provided by touch and gestures based interfaces, all this computation and procedural generation of content is transparent to the user, providing more engaging digital experiences.

1.2 PERCEPTUAL RECONSTRUCTION OF 2D IMAGES

Digital images usually consist of a 2D grid of pixels that can represent the projections of virtual or real worlds. During that process, multidimensional data passes through an optical system until it reaches a sensor that integrates everything into a 2D slice. Sometimes the time budget is too small to generate and process such data, while other times capture systems are not able to acquire data with the desired accuracy and resolution. However, previous research has shown that we can leverage the human perception and visual system to devise processes that overcome such limitations [147], producing results plausible or appealing enough for a human observer [158, 109, 146, 80].

In the case of computer generated imagery, this process is called *render*, and it simulates the light transport happening in a virtual world where light, objects and materials are defined mathematically. The more accurate those definitions and simulations, the more time it takes to compute their interactions. Thus, for a given amount of time and resolution of the final image, only a finite number of samples can be computed. For real time rendering (as in videogames and other interactive applications), that time budget usually varies from 33ms to 16ms per frame; while resolution nowadays starts typically at 1280x720. So typically when pushing for the highest overall image quality, one can only afford one sample per pixel. When that happens (specially for the lower range of resolutions), the image is undersampled showing *aliasing*. Spatial aliasing shows in the form of jagged edges that look harsh and unpolished when static; and silhouettes, thin structures and shading that crawl in motion.

All these problems converged together a few years ago, when graphics engines found specially in videogames were becoming more and more sophisticated and realistic, but the hardware for consoles and commodity computers were not able to keep up. So, a renewed interest in real time *anti aliasing* emerged, where traditional multi sampling approaches were not effective or applicable because of hardware and software limitations. During the following years, a number of different techniques appeared [106], aiming to overcome the lack of samples, with additional processing based on perceptual cues to obtain more pleasing results. However, all of them presented different limitations that hampered their use or extension. Our work in this topic explores such limitations, presenting a unified solution that tackles every limitation with a modular approach, easy to integrate and be

fit in different scenarios.

On the other hand, in the real world, the number of samples available can be considered infinite. However, different hardware approaches are required to sample the different dimensions of the plenoptic function that defines light [6]. In traditional photography, all of them are integrated into a single wavelength-dependant radiance value when light reaches the sensor. Due to the sensitivity limitations of current materials, the range of radiance able to be captured by current digital sensors is typically smaller than the one actually found in real scenes (up to several orders of magnitude). So, automatic and manual metering techniques try to find then the best exposure settings to best capture the radiance values around an area of interest. However, they cannot avoid areas outside of the sensor range to be clamped in the shadows end, highlights, or both.

To overcome this issue, high dynamic range (HDR) imaging techniques have been available for a while [143, 54]. They make use of *computational photography* to control the capture process in order to acquire additional information, which can be used afterwards to recover information a priori not seen directly by the sensor. For the specific case of radiance, a bracketed sequence of images of the same scene with different exposure settings are usually captured. This sequence is then used to estimate the actual range of radiance values in the scene. Before smart mobile devices, such process implied the use of a camera and the posterior use of a computer to perform all the required processing. When mobile devices with decent computing power appeared, *mobile computational photography* emerged, with everything happening on-device. In this line of research, we explore the first commercially available platform specially targeted to mobile computational photography. We implement a streamlined pipeline for extended dynamic range imaging, avoiding expensive and sometimes unnecessary steps by using human perception cues to obtain again pleasing results.

1.3 3D RECONSTRUCTION AND STYLIZATION

As mentioned previously, traditional photographic systems capture only a 2D slice of the plenoptic function of light. However, there are lots of cases where having additional depth information about the scene capture can be very useful for advanced image editing or computer vision tasks [137, 153, 20]. Following this line of mobile computational photography, current smart phones and tablets can become valid computational imaging devices. Interesting challenges come from their design limitations, with fixed hardware configurations that cannot be usually altered. In this line of research, we explore depth from defocus approaches, general enough to be used in this restricted scenarios, presenting a new fast and flexible pipeline for obtaining detailed depth maps of scenes.

Other times, instead of a global reconstruction of a scene from a single point of view, we are interested in obtaining full 3D reconstructions of single objects. Thus 3D scanning has been a recurring topic in computer graphics and computer vision over the years, as it gives us tools to quickly transfer objects directly from the real world to the virtual one. Among all the objects of interest, humans have been always a hot topic. Having digital replicas of

people can be used in different fields like cinema, videogames, medicine... and more recently fabrication. With the increasing popularity of 3D printers, a direct application that has captured the imagination of people is having a miniature replica of ourselves. This has been attempted since the required scanning and printing technologies were available, increasing the accuracy of the results as the solutions improved. However, hair, one of the key components of our identity, has been traditionally challenging. While systems for reconstructing hairstyles at single strands level exist [90], they are not suitable for 3D fabrication for two main reasons: i) the scale of hair strands is too small and does not scale well; ii) the structure of single hair strands or wisps is too fragile. In this work we study the problem of capture and reconstruction of 3D hairstyles for 3D fabrication. Inspired by the abstraction process followed by sculptors over the centuries, we present a novel stylized hair capture system, able to reconstruct simplified 3D models that retain the main features that identify each unique hairstyle.

1.4 GOAL & OVERVIEW

The main goal of this Thesis is to provide practical solutions to different problems related with the capture and manipulation of data, and simulation of creative processes from the real world. While some of these problems may seem unrelated, the solutions share in common that they usually feature novel combinations of traditionally separated fields: fluids simulation and image processing, image stylization and 3D reconstruction, computational photography. Another recurring topic in this body of work is the relation of these solutions with the world of art and human perception. Sometimes they directly become creative tools for content generation, others simulate artistic processes; while others take into account how we perceive the world around us, to provide pleasing reconstructions where simple math or hardware limitations do not reach.

Maybe as a result of having practicality in mind, the solutions proposed in this Thesis can usually be seen as systems comprised of several well defined pieces. This has the advantage of being easy to tweak and adapt to existing scenarios, to be used in different contexts, and also leaves the door open to be replaced by new or improved future work.

OVERVIEW This Thesis is divided in three main parts:

- **PART II** explores the creation of novel image editing tools that leverage the natural user interaction tablet devices provide for more engaging user experiences. Chapter 2 describes a framework for the simulation of the chemistry and craftsmanship involved in some analog photographic processes [98, 101], and demonstrate its potential building different prototypes. In Chapter 3 we use part of the previous framework to replicate a different kind of creative manipulations, where the artist places liquid or refractive layers in front of the main lens of a camera to obtain very interesting distortions.
- **PART III** presents 2D image reconstruction algorithms that make use of perceptual cues to compensate for the missing data in two different domains: spatial and radiance. In Chapter 4, we present our *SMAA* anti aliasing filter, which performs a morphological analysis of the image to produce smooth but sharp results in real time. Then, Chapter 5 describes a practical pipeline for extended dynamic range imaging in

mobile devices, that perceptually fuses a bracketed sequence of images with different exposures.

- PART IV describes new algorithms and systems for 3D reconstruction of scenes and objects. In Chapter 6, a new depth from defocus algorithm for fast capture of depth maps is described. Then, Chapter 7 presents a system for stylized capture of hairstyles, where color and geometric information is abstracted to produce simplified models ready for 3D printing, showcasing a pleasing hand-crafted look and feel.

While I am the leading author in most of the works presented here, they have been done in collaboration with different colleagues. Thus, at the beginning of each chapter the work described is put in context, and my contribution is explicitly described when needed.

1.5 CONTRIBUTIONS AND MEASURABLE RESULTS

1.5.1 Publications

Most of the work presented in this thesis has been already published, in particular in four journals indexed in JCR, including one paper in ACM Transactions on Graphics and presented at SIGGRAPH; one peer-reviewed international conference, one book chapter and one granted US Patent:

- Computational Simulation of Alternative Photographic Processes (Chapter 2, Part II):
 - This work was accepted at Eurographics Symposium on Rendering (EGSR) 2013, and published in Computer Graphics Forum [61]. This journal has an impact factor of 1.902, and its position in the JCR index is 18th out of 104 (Q1) in the category Computer Science, Software Engineering (data from 2014).
- Simulation of the effects of liquids on a camera lens (Chapter 3, Part II):
 - This work was granted with a US Patent filed with Adobe Systems [206].
- SMAA: Enhanced Subpixel Morphological Antialiasing (Chapter 4, Part III):
 - This work was accepted at Eurographics 2012, and published in Computer Graphics Forum [105]. This journal has an impact factor of 1.902, and its position in the JCR index is 18th out of 104 (Q1) in the category Computer Science, Software Engineering (data from 2014).
 - The method was built on top of our previous GPU version of MLAA, published in the GPU Pro 2 book [107], a series collecting state-of-the-art techniques for game development.
- Mobile Computational Photography: Exposure Fusion on the N900 (Chapter 5, Part III):
 - This work was accepted at the Ibero-American Symposium in Computer Graphics (SIAGC) 2011 [59].
- Fast depth from defocus from focal stacks (Chapter 6, Part IV):

- This work was published in *The Visual Computer* [18]. This journal has an impact factor of 0.922, and its position in the JCR index is 50th out of 104 (Q2) in the category Computer Science, Software Engineering (data from 2014).
- Capturing and Stylizing Hair for 3D Fabrication (Chapter 7, Part IV):
 - This work was accepted at SIGGRAPH2014, and published in *ACM Transactions on Graphics* [58]. This journal has an impact factor of 4.096, and its position in the JCR index is 1st out of 104 (Q1) in the category Computer Science, Software Engineering (data from 2014). A US Patent application was also filed with Disney Enterprises [25].

In addition to the previous list of publications, during my PhD I have contributed significantly to other projects and publications:

- Convolution-based simulation of homogeneous subsurface scattering.
In this work, lead by Adolfo Muñoz, we present a novel rendering algorithm for homogeneous translucent materials. It was first accepted at CEIG 2010 [60], where it was selected Best Paper (1 in 2), to be later extended and published in *Computer Graphics Forum* [154].
- BSSRDF Estimation from Single Images.
In this work, lead also by Adolfo Munoz, we propose a single-image system to acquire plausible diffusion profiles for translucent materials from real world (Figure 1.1). It was accepted at Eurographics 2011 and published in *Computer Graphics Forum* [153]. It was also selected for *FMX 2012 Best of Eurographics* session, Europe's most influential conference for digital entertainment.
- Practical and Realistic Facial Wrinkles Animation.
In this work, lead by Jorge Jimenez, we propose a simple technique to achieve realistic facial wrinkles animation for characters in videogames (Figure 1.2). It was published in the *GPU Pro 2* book [104].
- Intrinsic Light Fields.
In this work, lead by Elena Garces, we present the first method to efficiently compute the intrinsic decomposition of 4D light fields, ensuring angular coherence and practical processing times [69].

1.5.2 Awards

We include here a list of awards and fellowships received throughout this thesis, that have allowed the realization of the work here presented:

- PIF Grant from the Universidad de Zaragoza (4-year PhD grant).
- Adobe Systems funding to extend the collaborative work after each of the research internships.

Additionally, some projects described in this thesis have received different awards or recognitions:



Figure 1.1: Results from our method to capture the appearance of translucent objects. Insets show the input images from which the materials were estimated. The renders show virtual objects rendered with our estimations.

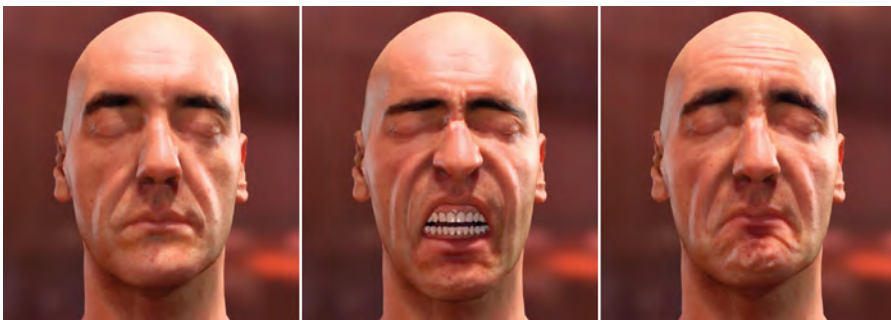


Figure 1.2: Results from our facial wrinkles simulations. Starting from a neutral pose and textures (left), our method is able to produce realistic wrinkles for complex poses and expressions (center and right), as can be seen in the forehead, chin and nose.

- Our work *Computational Simulation of Alternative Photographic Processes* was invited to the XXIV Spanish Conference in Computer Graphics (CEIG 2014).
- Our work *Capturing and Stylizing Hair for 3D Fabrication* was awarded as the Best Paper at the GOLEM Workshop.
- Our work *Capturing and Stylizing Hair for 3D Fabrication* was also selected for the SIGGRAPH 2014 *Technical Papers Preview* video.

1.5.3 Research Stays and Visits

Five research stays, totaling 12 months, were carried out during this PhD in three different institutions:

- April 2011 – May 2011 (one month): Early Stage Researcher within the *GOLEM* Marie Curie project at Porto Interactive Center (Portugal).
- June 2011 – August 2011 (three months): Research Intern at the *Advanced Technology Labs* at Adobe Systems (San Jose, California, USA). Supervisor: Dr. Gregg Wilensky. Publications [61, 206] were fruit of this collaboration.
- June 2013 – July 2013 (one month): Early Stage Researcher within the *GOLEM* Marie Curie project at Porto Interactive Center (Portugal).
- August 2013 – December 2013 (four months): Research Intern at the *Capture & Effects Group* at Disney Research Zurich (Switzerland). Supervisor: Dr. Thabo Beeler. We published [58] and applied for a US patent as the result of it [25].
- July 2015 - October 2015 (three months): Research Intern at the *Imagination Lab* at Adobe Systems (San Jose, California, USA). Supervisor: Dr. Byungmoon Kim. Worked on a novel solver for Poisson-based surface reconstruction.

1.5.4 Research Projects

During my PhD studies I have participated in the following research projects:

- *VERVE*: Vanquishing fear and apathy through e-inclusion: personalised and populated realistic virtual environments for clinical, home and mobile platforms. *European Commission (FP7-ICT-2011-7)*. Grant no.: 288914. PI (in Spain): Diego Gutierrez.
- *GOLEM*: Realistic Virtual Humans. *European Commission Marie Curie Industry-Academia Program, Seventh Framework*. Grant no.: 251415. PI: Diego Gutierrez.
- *MIMESIS*: Técnicas de bajo coste para la adquisición de modelos de apariencia de materiales. *Spanish Ministry of Science and Education (TIN2010-21543)*. PI: Diego Gutierrez
- *TANGIBLE*: Humanos realistas e interacción natural y tangible. *Spanish Ministry of Science and Education (TIN2007-63025)*. PI: Francisco J. Seron.
- Aumento del rendimiento gráfico, para sistemas de simulación y visualización en tiempo real, a través de técnicas de antialiasing morfológico. *Fundación ARAID (OTRI 2011/0180)*. PI: Diego Gutierrez.
- *Advanced Multimodal Audiovisual Technologies (TAMA)* *Gobierno de Aragón (regional government)*. PI: Carlos Orrite.
- Desarrollo de una herramienta digital para evaluar la función visual y los procesos cognitivos visuales en niños preverbales. *Carlos III Health Institute. Spanish Ministry of Economy and Competitvity*. PI: Victoria Pueyo.

1.5.5 *Professional Service*

I have been given the chance to serve the research community by reviewing and serving on the program committee of several journals and conferences. Over the years, I have reviewed papers for ACM SIGGRAPH and SIGGRAPH Asia, IEEE Transactions on Visualization and Computer Graphics (TVCG), Computer Graphics Forum, Computer & Graphics, ACM Transactions on Applied Perception, IET Computer Vision, Eurographics Symposium on Rendering, High Performance Graphics, Digital Media and Digital Content Management (DMDCM), CEIG and SIACG; and I have served on the committee of the IADIS Computer Graphics, Visualization, Computer Vision and Image Processing (CGVCVIP). Finally, I was on the local organizing committee for the Eurographics Symposium on Rendering (EGSR) 2013, held in Zaragoza, and hosted by our research group.

1.5.6 *Impact on the industry*

Apart from the previous academic publications, the impact of some of this work can also be found in the industry:

- Our stylized hair capture system presented in [58] is currently the state of the art in hair reconstruction for full 3D miniature fabrication, with no other method close to the look and feel of our results. It has been widely covered in both industry and mainstream media outlets such as NBC News, Gizmodo, Engadget, Xataka, El Mundo, 3D Printing Industry, 3Ders... A US Patent application was filed with Disney Enterprises [25].
- Our SMAA [105] and MLAA anti aliasing filters [107] have been extensively used in commercial videogames for consoles and PC, and also featured in gaming industry sites like Games Developer Magazine, GamesIndustry.biz and Eurogamer's Digital Foundry. SMAA is currently the state of the art in post processing anti aliasing filters, with more recent commercial solutions implementing similar ideas.
- The liquid on lens simulation has been recently granted with a US Patent [206], proving the relevance such ideas can have in the image editing apps to come in the next years.

Part II

CREATIVE MANIPULATION OF DIGITAL PHOTOGRAPHS

In this part we present novel interactive image editing tools inspired by analog photography and the physical manipulations that are required to create and develop such photographs. We first introduce a computational framework that allows the simulation of the chemical components and reactions that slowly create the final picture; and the user interactions to control them in their liquid state. We demonstrate it through different prototypes for different user experiences. Then, based on this framework, we also recreate interesting optical manipulations like placing liquid or refractive layers in front of the camera lens to produce very creative results.

In this Chapter we present a novel computational framework for physically and chemically-based simulations of analog alternative photographic processes. In the real world, these processes allow the creation of very personal and unique depictions due to the combination of the chemicals used, the physical interaction with liquid solutions, and the individual craftsmanship of the artist. Our work focuses not only on achieving similar compelling results, but on the manual process as well, introducing a novel exploratory approach for interactive digital image creation and manipulation. With such an emphasis on the user interaction, our simulations are devised to run on tablet devices; thus we propose the combination of a lightweight data-driven model to simulate the chemical reactions involved, with efficient fluids simulations that modulate them. This combination allows realistic gestures-based user interaction with constant visual feedback in real-time. Using the proposed framework, we built two prototypes with different tradeoffs between realism and flexibility, showing its potential to build novel image editing tools.

This work is published in *Computer Graphics Forum* and presented at *Eurographics Symposium on Rendering (EGSR) 2013*. It was developed in collaboration with Adobe Systems, which kindly funded the project until its completion.

J. I. Echevarria, G. Wilensky, A. Krishnaswamy, B. Kim & D. Gutierrez

COMPUTATIONAL SIMULATION OF ALTERNATIVE PHOTOGRAPHIC PROCESSES
Computer Graphics Forum, Vol.32 (4), EGSR 2013

2.1 INTRODUCTION

Digital photography brought along powerful, intuitive image editing techniques that allow the user to dramatically change the final look of the image. Some of these techniques are designed to mimic the results from analog photography, mainly by means of tone manipulation [70, 178, 17], or are encapsulated into single-button filters, such as the popular *Instagram*. It is interesting to see the way the general public has embraced such *vintage filters*. This presents an opportunity to revisit the real photographic processes that produce the looks they copy and to explore their potential for creating novel digital experiences and image editing tools; this is something currently neglected with the reduced selection of canned effects and textures available to the digital artist.

The resulting interaction metaphors seek to bring back the *craftsmanship* and exploratory approach to image creation of those analog *alternative photographic processes* [101], such as wet plate collodion or printmaking (as seen in Figure 2.2), noticeably lost in the transition from analog to digital photography. Currently kept alive by professionals and aficionados, they allow for a wide range of impressive tonal and emotional variations. Some are intended and carefully chosen, such as the exact composition of the chemicals involved; some arise serendipitously due to imperfections and the manual manipulation of liquid solutions; and some are due to the individual skills and particular methodology of each practitioner.



Figure 2.1: *Our image processing pipeline allows the simulation of the craftsmanship and final looks of analog alternative photographic techniques like wet plate collodion (left) or cyanotypes (right). The dynamic nature of our fluids-based chemical reactions and the required manual work interaction provide unique results, similar to their analog counterparts and their unique local heterogeneities. The small inset shows the original scene, courtesy of Tracie Tee.*

We present a novel computational framework for physically and chemically based simulations of some of these alternative photographic techniques, allowing real time physical interactions with the aqueous solutions involved. Over time, these fluids locally modulate the interactions between emulsions, substrates and the composition of the chemicals. Such complex chemical reactions are simulated using a lightweight data driven model. This creates heterogeneities and tonal ranges similar to the equivalent real analog processes (see Figure 2.1). While reliability, repeatability and total control over the editing pipeline is critical in many professional environments, our system presents a forward simulation of the photographic process for a more exploratory approach. Other uses for our framework may include educational photographic tools. To better convey the feeling of actual physical manipulation, we target tablet devices, which impose an additional challenge due to their limited computational power and our hard real-time constraint. Our prototypes showcase the flexibility of our framework to create pipelines with different tradeoffs between realism and simplicity of usage, depending on the target use cases.

Our main contributions are:

- A physically and chemically based framework that simulates alternative photographic processes, user interaction included, allowing the creation of novel image depictions from the ground up (Section 2.4).
- A novel and efficient 3D Navier-Stokes solver that takes into account vertical-to-horizontal flow transfer for natural liquid behavior and user interaction, coupled to a Lattice Boltzmann solver for accurate subsurface water percolation through paper fibers (Section 2.5).
- A lightweight data-driven model that simulates the dynamic chemical reactions of the aqueous solutions, based on their components and processing times (Section 2.6).
- Fully working prototypes, running on tablet devices (Section 2.7 and supplementary videos).

It is out of the scope of this work to describe and simulate each detail of these photographic processes. Instead, we are focusing on the essential steps needed to capture and develop an image, which are common to most photographic pipelines. Therefore, our simulations, while physically and



Figure 2.2: Examples of analog alternative processes. The top row shows wet plate collodion ambrotypes with varying degrees of heterogeneities caused by the aqueous solutions involved. The bottom row shows a rubytype (ambrotype over a colored glass plate), a cyanotype and a split toned print, respectively. Artworks by Deborah Parkin, Daniel Carrillo, Ian Ruhter, Indra Moonen, Mike Ware and Tim Rudman (used with permission).

chemically based, are still not accurate enough to serve as a predictive tool, but they serve to prove that plausible user experiences and results can already be built.

2.2 RELATED WORK

Multiple papers exist which aim to enhance the look of an image or alter its style; these usually follow data-driven or statistical approaches [88, 175, 198, 199]. As such, they do not focus on simulating any particular aspect of the analog photographic processes. There exist mobile apps like *Instagram* or plugins like *Exposure 4* that mimic analog looks. However, they are based on sets predefined filters and texture overlays for specific effects, while our framework emphasizes the user interaction and the creative process, providing more engaging experiences with procedural unique results.

Geigel and Musgrave [70] provide a tone reproduction operator to simulate the standard photographic pipeline for black and white images. Our work is related to this, although there are significant differences. First, we focus on simulating analog photographic techniques which present more local variations for which computational models do not exist. Second, our work allows the user to physically interact with a mobile device to simulate some of the manual work in those analog processes, such as pouring liquid emulsions or controlling local development time. Third, we allow the users to tweak the main parameters as they would do in the real world, including chemical components concentrations and processing times. Last, given the manual interaction required, two results will never be identical; each final image has a unique look, just like the actual processes.

Based also on analog photography, Bae et al. [17] focus on the management of tonal aspects for more expressive renditions of the original



Figure 2.3: *Different steps of the wet collodion process as demonstrated by artist Quinn Jacobson (used with permission). Left: Pouring of the emulsion. Center: Development of the latent image after exposure. Right: Fixing bath to obtain the final image.*

scenes, drawing inspiration from well-known photography artists; their system also allows for the transfer of pictorial styles between images. Reinhard et al. [178] introduce a tone mapping operator based on Ansel Adams' classic zones system. Last, German's work [72] presents a technique to reverse-engineer some aesthetic decisions made during the print making process, and applies them to a digital scan of the original negative. These techniques manipulate tonal ranges for a particular application, but do not focus on reproducing actual photographic processes nor on user control over them.

2.3 THE ANALOG PHOTOGRAPHIC PIPELINE

This section explains the basic concepts of the photographic pipelines and techniques we are simulating. We refer the reader to modern manuals for more in-depth explanations about them [181, 101, 98]. A general analog photographic pipeline consists of five core steps: the creation of a light-sensitive emulsion, its exposure to actinic light, the development of the latent image, the fixing of the final image and optional toning. Each photographic process deals with these differently; for instance, while in film-based photography the emulsion is created industrially, it is carefully handcrafted in the wet collodion processes (an example can be seen in Figure 2.3). In the following, we will describe the manual work required in each step, not focusing on any technique in particular.

The creation of the emulsion starts with the artist selecting the chemicals and formulas specific for the process, which give the liquid emulsion different density responses and spectral sensitivities. This is then poured onto a plate and distributed across its surface by manually tilting it, or it is alternatively coated over the supporting medium for a print. Both the liquid nature of the emulsion and the state of the chemicals involved favor local variations due to heterogeneities in the mix.

During exposure, light interacts with the emulsion creating small particles which form the so-called latent image. Then, the developer liquid makes those particles grow in size until they are clearly visible. Once the artist gets the desired density a liquid fixer is used for eliminating unexposed areas, obtaining the final image and avoiding further undesired chemical reactions. Additional steps like toning can be performed at this point. Toning is based on the (iterative) use of bleaching baths to re-enable a fixed image to keep reacting to developers and toners. The artist can control its timing to limit the reach of the toner gradually from highlights to shadows.

A key aspect of the described pipeline (and the unique results it produces) is the craftsmanship involved in most of the stages of the process. In

particular, the artists needs to physically interact with the different aqueous solutions to achieve the desired spatial distribution, development and setting times, etc. The simulation of this interactive pipeline is explained in the following Section, and is built around the same principle of unique physical interaction.

2.4 COMPUTATIONAL PIPELINE

Chemical reactions happening in photographic processes are complex low-level phenomena [77, 148, 151, 68], even more if one takes into account that they are achieved and controlled using liquid solutions prone to complex spatial distributions. A convenient way to represent the response of a photographic process are its characteristic (or Hurter-Driffeld) curves $g(e, \mathbf{b})$ [93]. These curves represent the final density of the processed emulsion as a function of exposure e , and the input parameters \mathbf{b} (chemical components, processing times), and they can be measured for a desired number of scenarios.

In our framework, we propose a dynamic data-driven characteristic curve model based on the main parameters artists control in real life. These parameters are modulated in different steps of the pipeline by using interactive fluids simulations to introduce heterogeneities and provide natural user interaction. These two key components are explained in detail in Sections 2.5 and 2.6. Figure 2.4 shows an overview of our pipeline, described below. To allow for more intuitive physical interaction, our techniques are devised to run on tablet devices. We refer the reader to the video in the supplementary material which shows actual interaction sessions.

Emulsion. First, the user selects the desired chemical mix, by adjusting values of three main components: halide concentration, iodide to bromide ratio and cadmium concentration (Section 2.6 explains this selection of chemicals). Different mixes yield different contrast, density and spectral sensitivity. Next, the user pours the virtual liquid emulsion over the surface of the virtual plate, by touching the screen to add liquid on the desired areas (see Figure 2.4, a), and using his or her fingers to stir and spread it over the surface. Prolonged contact adds more liquid. Additionally, tilting movements can be used for adding gravity and friction forces, estimated from the normal of the screen based on the gyroscope data and the current fluid distribution, respectively.

Our real-time, hybrid fluids system, described in Section 2.5, simulates the dynamics of the real liquid on and under the surface, giving immediate feedback to the user and updating a two-dimensional height map $\mathbf{H} = [h_{i,j}]$ describing its distribution over the surface¹. To model heterogeneities in the emulsion that may give rise to local variations in the results, we create an additional Perlin noise [169] map to modulate the iodide to bromide ratio in the chemical mix, which directly influences the contrast and spectral sensitivity of the final image. For natural results, we advect this map along with \mathbf{H} as will be explained in Section 2.5 (Figure 2.11 and Figure 2.12, show examples of these heterogeneities).

The spatially-varying setting time τ_s of the emulsion is dynamically calculated based on its coverage of the surface of the plate or paper:

$$\tau_s(t) = \tau_s(t - \Delta t) + \sigma_s(h) \cdot \Delta t, t > 0 \quad (2.1)$$

¹ We assume per-pixel values in all our equations and remove subindices from the variables.

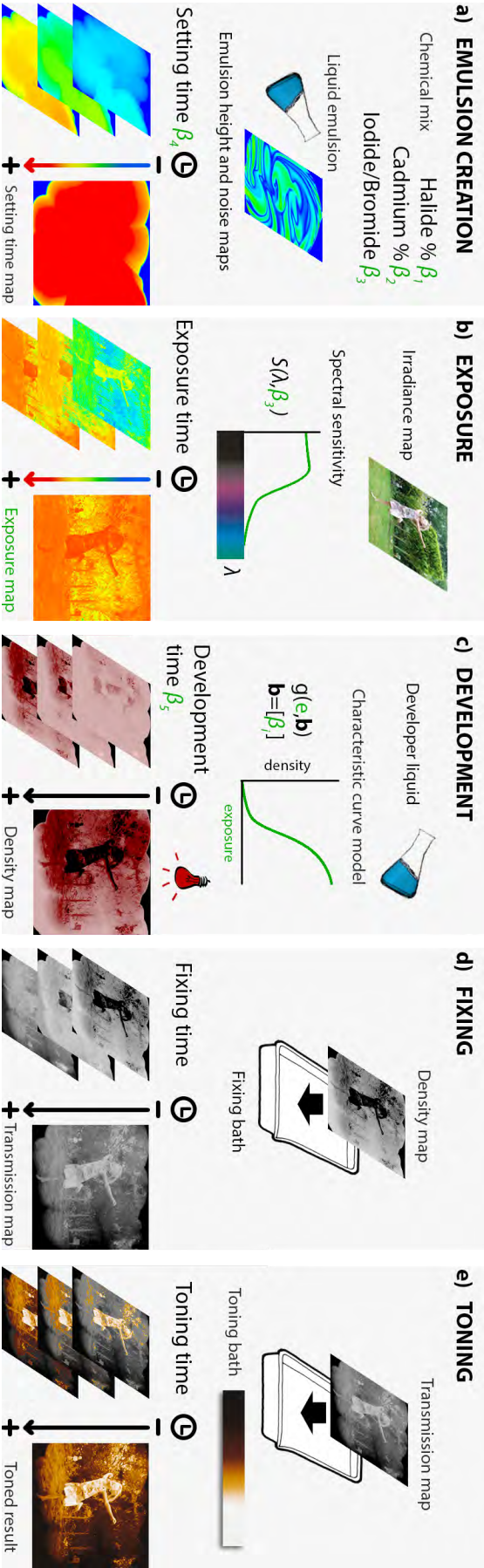


Figure 2.4: a) The emulsion is created by specifying the chemical mix, pouring the virtual liquid over a surface and allowing it to set, obtaining different maps that will be used in following steps. b) Next, an exposure map is obtained based on the irradiance, the spectral sensitivity of the emulsion and the exposure time. c) The resulting latent image is developed by pouring liquid developer which, as time passes, allows our dynamic curve model to calculate the density map based on previously defined β_i parameters. d) Inside the fixing bath, unexposed areas are cleared, obtaining the final transmission map. e) Optionally, a toning bath allows the artist to change the appearance of the image for specific looks.

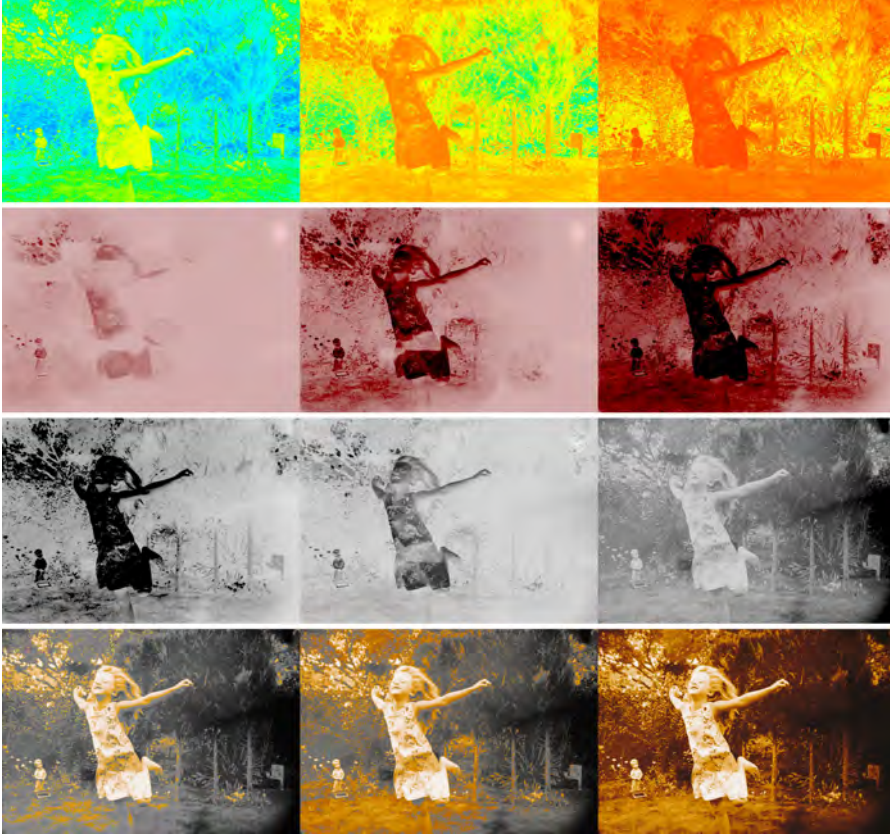


Figure 2.5: From top to bottom, temporal evolution (left to right) of the exposure, development, fixing and toning steps. We provide the user with continuous visual feedback to stop each process at the desired moment. The exposure map is shown for visualization in false color as a heat map, and the development step is shown as seen under safe light inside the darkroom. During fixing, inversion from negative to positive is achieved as in ambrotypes (underexposed negatives over dark backgrounds [98]).

where t is time, $\tau_s(0) = 0$, $\sigma_s(h)$ is a binary function ($\sigma_s(h) = 1$ for $h > 0$ and 0 otherwise), and Δt is the time step of the simulation. This yields a *setting time map* (Figure 2.4, a), which affects the final density of the image.

Exposure. After the emulsion has been allowed to set, it is exposed. Exposure e for a continuous spectral distribution of light is defined as:

$$e = \int_t \int_{\lambda} I(\lambda) \cdot S(\lambda) \cdot d\lambda \cdot dt \quad (2.2)$$

where $I(\lambda)$ is the irradiance from the exposing light for wavelength λ and $S(\lambda)$ is the spectral sensitivity. We obtain $I(\lambda)$ using HDR RGB values from the input images (and its corresponding CIE λ values). Alternatively, we can approximate it from linearized LDR inputs [76] or access the camera sensor of the device directly. $S(\lambda)$ in alternative processes is usually very different from regular digital RGB sensors, varying with the chemical mix (see Figure 2.6). To model this dependency, we use a data driven approach based on previous measures [186] (more on this selection in Section 2.6), interpolated using piecewise cubic Hermite polynomials and storing the result in a 2D lookup texture (Figure 2.7). The result of this step is an *exposure map*, as

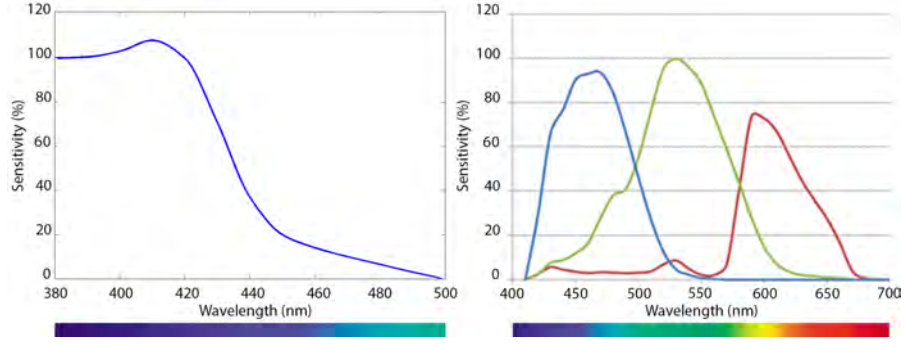


Figure 2.6: Left: Example of a spectral sensitivity curve for a collodion emulsion with a iodide/bromide ratio of 60/40 [186]. Right: Typical sensitivity curves for the RGB sensor of a digital camera (adapted from [122]). As can be seen, collodion emulsions are almost insensitive to yellow-red wavelengths, while being more sensitive to the UVA range ($< 400\text{nm}$) than digital sensors.

seen in Figure 2.5 (first row).

Development. Similar to the creation of the emulsion process, the user pours the developer liquid and interacts with it, while the development time τ_d is computed as:

$$\tau_d(t) = \tau_d(t - \Delta t) + \sigma_d(h) \cdot \Delta t, \quad t > 0 \quad (2.3)$$

where $\sigma_d(h)$ controls the development ratio. We use $\sigma_d(h) = \text{smoothstep}(0.0, 0.5, h)$, with a smooth Hermite interpolation between 0 and 1 when $0.0 < h < 0.5$, and set $\tau_d(0) = 0$ (Figure 2.5, second row).

As explained in Section 2.3, the goal of the development step is to achieve the desired particle density to make the latent image visible. We compute a *density map* $d(e, \mathbf{b})$ as a function of τ_d and the characteristic curve of the process $g(e, \mathbf{b})$ (see Section 2.6). Since unexposed areas of the image also become foggy over prolonged developing times, we add a small offset controlled by K_f . A value of $K_f = 0.003$ works well in our simulations:

$$d(e, \mathbf{b}) = g(e, \mathbf{b}) + K_f \cdot \tau_d(t) \quad (2.4)$$

Fixing. The role of the fixer is to dissolve the unexposed areas, preventing further exposure and development of the emulsion. Usually, this fixing step is performed by submerging the plate into the liquid fixer, so it is a uniform process over the whole surface. We therefore define the final *fixed density map* ψ based on a global fixing time τ_f :

$$\psi(\tau_f) = (d(e, \mathbf{b}) - K_d \cdot \tau_f) + (h - K_c \cdot \tau_f), \quad \psi(\tau_f) \geq 0 \quad (2.5)$$

where K_d and K_c are the dissolving and clearing power of the fixer. We use empirical values of 0.0025 and 0.1 respectively for generic realistic fixing times. Exposed areas are almost not affected by the fixer, but the image could actually disappear if left long enough inside the fixing bath.

This final density map ψ gives a measure of the opacity of the processed emulsion; so we can define a *light transmission map* $T = 1/10^\psi$ [70], with values in the $[0, 1]$ range, that can be used directly for display (Figure 2.5, third

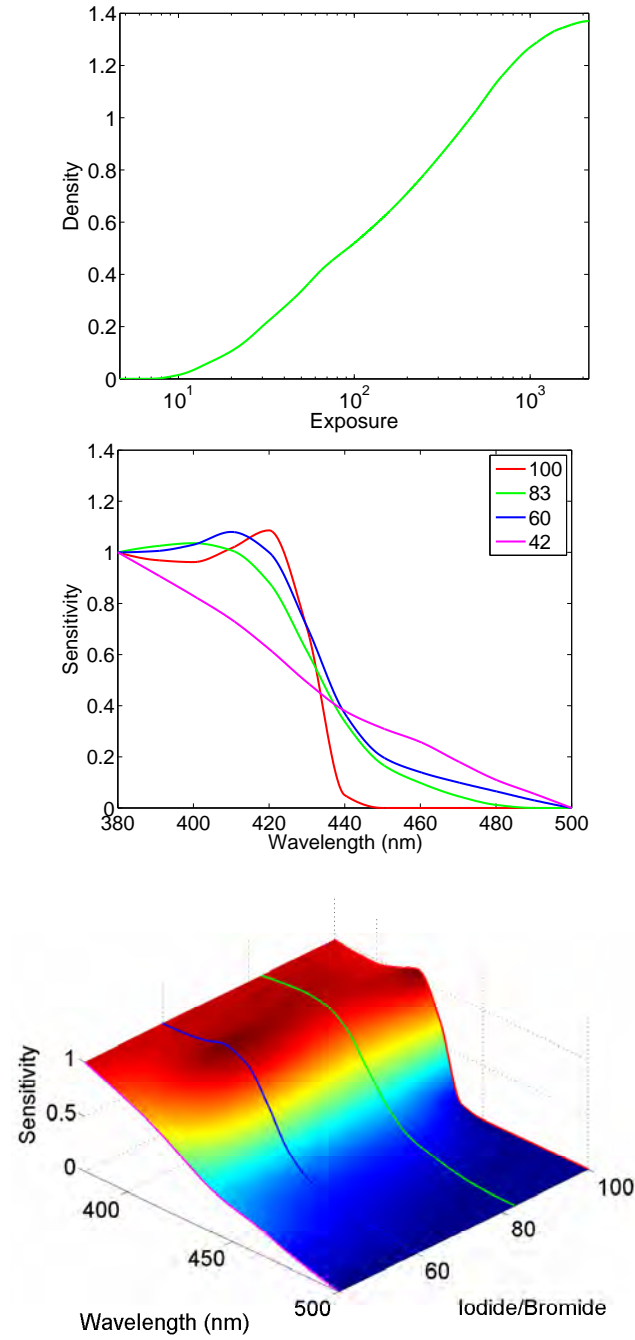


Figure 2.7: Top: Global optimal curve $f_0(e)$ obtained for parameters: halide concentration of 1.1, iodide/bromide ratio of 60/40, cadmium iodide ratio of 0.68, 60s of setting time and 30s of development. Middle: Sensitivity curves $S(\lambda, \text{iodide/bromide})$. Bottom: Lookup table obtained after interpolating $S(\lambda, \text{iodide/bromide})$ for intermediate iodide/bromide ratios, using piecewise cubic Hermite polynomials.

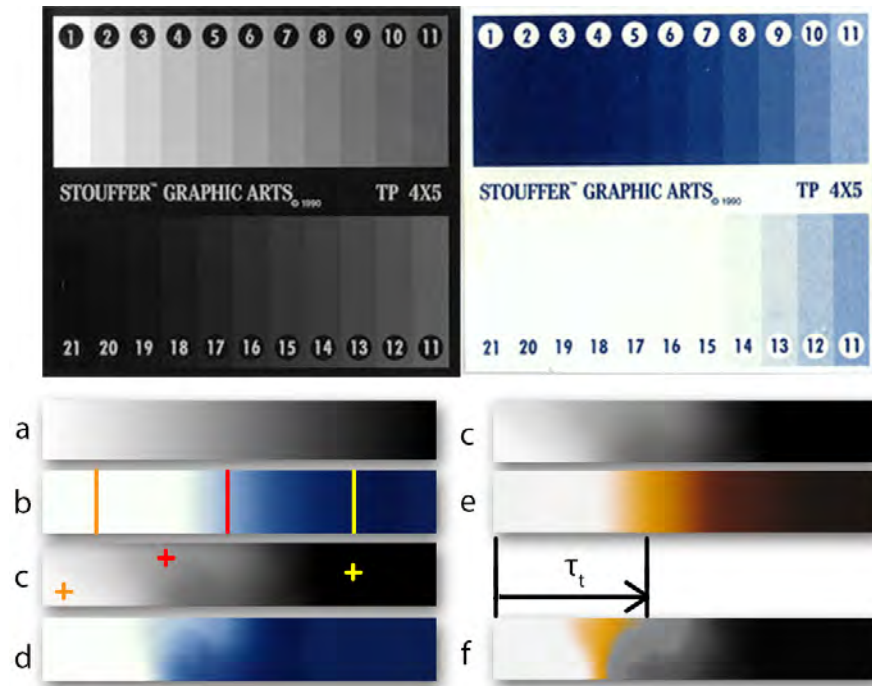


Figure 2.8: Top: A calibrated negative step wedge with differences of $1/2$ F-Stop in exposure between steps, and a result for the cyanotype process [116]. Bottom left: From a simple greyscale linear ramp (a), we obtain its corresponding cyanotype-looking ramp (b) by mapping its values to the calibrated step wedge on the left. We then process (a) with our framework, yielding the transmission map T (c). We map the $[0, 1]$ values in T to the $[0, 1]$ x -coordinates of (b) (examples of this mapping are shown in different colors), obtaining our final cyanotype look (d). Bottom right: Split-toning effect with the FSA toner [181]. From the transmission map T (c), and the toned ramp (e), we control τ_t to achieve the final result (f). τ_t effectively limits the range of transmission values that are affected by the toning step.

row) or further processing.

Toning. Real plates and prints can present colorful monochromatic looks as a result of the chemicals used or additional toning processes (Section 2.3). In the physical world, the final tone can be measured beforehand by exposing and processing a calibrated negative step wedge, producing a positive gradient with the resulting colors based on the transmission of such negative (Figure 2.8, left). In our framework, density is decoupled from final appearance, which allows us to map our neutral transmission map T to the specific looks of different techniques and toners. We do this by using the transmission values $[0, 1]$ as texture coordinates to sample their previously processed step wedges, which we store as lookup tables [33]. Toning progressively occurs from highlights to shadows, as a function of time (Figure 2.5, fourth row). So, by controlling toning time τ_t , and thus which areas get affected by the toner, split toning effects can be simulated. Figure 2.8, center and right, shows examples of global toning mimicking the looks of cyanotypes and split toning respectively.

Flexibility of the simulation. Our framework is flexible enough to simplify or omit some of the previous steps in the pipeline (more details in Sec-

tion 2.6), depending on its intended usage. The prototypes described in Section 2.7 and shown in the supplementary video leverage this to provide different tradeoffs between realism and simplicity of usage. Equally important, each step of the pipeline could be adjusted to fit different photographic processes if relevant data (characteristic and spectral sensitivity curves, color ramps...) is available.

2.5 HYBRID FLUIDS SIMULATION

To simulate similar heterogeneities, uniqueness in the results and engaging physical interactions as in the real processes, we rely on computational fluids to drive the dynamic chemical reactions which produce the final image. The Navier Stokes equations (NS) rewrite Newton’s law for fluids, treating the fluid as continuum. This is the natural choice to simulate liquids on the surface [67, 66]. When liquids seep into paper, however, they move by molecular interaction with paper fibers. In this case, the Lattice Boltzman equation (LB) becomes the natural choice [48]. We propose a novel hybrid method using both formulations that deals seamlessly with the interactions between the liquid and uniform surfaces like glass plates, or percolating surfaces like paper.

Hybrid methods have been studied before in the water color painting community [49, 124, 123]. All of these models used hybrid solutions combining the shallow water equation (a form of simplified NS) with custom models targeting particular effects. In contrast, we solve the 3D Navier-Stokes equations with an ultra-thin grid for emulsion and developer liquids on the surface, and the Lattice Boltzman method for in-paper behavior. Both methods yield numerical solutions that converge to the physically correct solution as the grid is refined [193].

On Surface: Navier-Stokes Equations. Our method is based on the incompressible Navier-Stokes (Eq. 2.6) and continuity (Eq. 2.7) equations:

$$\frac{\partial \mathbf{u}}{\partial t} = -\mathbf{u} \cdot \nabla \mathbf{u} + \nu \nabla \cdot \nabla \mathbf{u} - \frac{1}{\rho} \nabla P + \frac{\mathbf{f}}{\rho} \quad (2.6)$$

$$\nabla \cdot \mathbf{u} = 0 \quad (2.7)$$

where \mathbf{u} is the velocity vector of the fluid, ρ is the density of the fluid, P is the pressure, \mathbf{f} is external force such as $\rho \mathbf{g}$, where \mathbf{g} is the gravity vector, and ν is the kinematic viscosity. Equation 2.6 is the momentum equation that accounts for the forces acting on the fluid, and Equation 2.7 is called the incompressibility condition that keeps the volume of the fluid constant [35]. We follow the semi-Lagrangian advection and pressure projection proposed by Stam [190], given its stability when working with large simulation steps. The system starts from an initial state \mathbf{u}_0 and then Equation 2.6 is solved for each time step in four sequential steps: addition of external forces, advection of the fluid by itself, velocity diffusion due to viscous friction within the fluid, and projection onto the divergence-free velocity field (imposed by Equation 2.7). Additional scalar fields can be advected using the previously calculated velocity field. We refer the reader to [190] for additional details.

If the liquid layer is thin, the 3D NS equations may be first simplified to a 2D height field for efficiency. This can be achieved by removing the pressure term, yielding the shallow water equation [35]. This can be further simplified to shallow wave equations [113, 49, 200, 213]. Although computationally efficient, behaviors are limited to wave-like effects. All these approaches lack

the richness of 2D patterns that stem from vorticities produced by the pressure term. In addition, to simulate horizontal flow induced by liquid height variation, the shallow water equation needs careful tuning of time step, viscosity and horizontal force coefficients in order to be stable and suppress wave-like behaviors.

The emphasis for our framework upon user engagement is provided by modeling a believable behavior for the virtual fluid. We therefore develop an alternative method based on a 3D simulation grid (xyz) , with only two grid nodes in the vertical direction z , separated by a constant interval Δz : bottom and top layers covering $0 \leq z \leq \Delta z$ and $\Delta z \leq z \leq 2\Delta z$ respectively. Liquid may be higher than $2\Delta z$ on some locations. In this case, we treat the weight above the domain as an external force applied to the top. Let h be height. Combined with the weight of liquid in the cell itself, the force \mathbf{f} in (2.6) becomes

$$\mathbf{f} = \begin{cases} \rho \mathbf{g} \max(0, h - \Delta z)/\Delta z & \text{on top layer} \\ \rho \mathbf{g} \max(h, \Delta z)/\Delta z & \text{on bottom layer.} \end{cases} \quad (2.8)$$

Then, we use a full 3D projection step by solving the Poisson equation:

$$\nabla^2 P = \frac{\rho}{\Delta t} \nabla \cdot \tilde{\mathbf{u}}, \quad (2.9)$$

where $\tilde{\mathbf{u}}$ is the velocity obtained from the velocity diffusion step, which we apply before the pressure step. We set Neumann boundary conditions ($\partial P / \partial z = 0$) on the bottom and Dirichlet boundary condition ($P = 0$) on the top. Note that with boundary ghost cells, using only two grid nodes is enough to take the second pressure derivative. Since the grid has only two layers, the pressure projection is fast enough even for mobile devices.

Finally, the 3D continuity integrated over z yields the height rate by Navier-Stokes and continuity:

$$\left. \frac{\partial h}{\partial t} \right|_{\text{NS}} = \int_0^h \frac{\partial w}{\partial z} dz = - \int_0^h \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) dz. \quad (2.10)$$

Since the liquid is thin in z , by assuming that 2D flow difference along z is small, we approximate (2.10) as $\left. \frac{\partial h}{\partial t} \right|_{\text{NS}} = -h(\partial u / \partial x + \partial v / \partial y)|_{\text{bottom}}$. Thus, we approximate $\frac{\partial h}{\partial t}$ from divergence in the bottom layer. We did not use the divergence in the top layer since the flow should be similar to bottom layer, and also to reduce computations. This produced satisfactory expansion effects, although the height rate can be more precisely computed by using the top layer as well.

For efficiency, we use a multi-resolution scheme: we first calculate a 2D low resolution velocity field of $64 \times 64 \times 2$ which is then upsampled linearly to 256×256 for high resolution advections and user interaction. Since we want to modulate the chemical reactions in Section 2.6 according to different liquid distributions, we create the height map \mathbf{H} by advecting the liquid quantities injected by the user over the high resolution velocity field.

Subsurface: Lattice Boltzmann Equation. The simulation based on the Navier-Stokes equations is suitable for getting natural behaviors when the liquid moves over a uniform surface as happens with the plates used in wet plate photography. However, if we want to simulate realistic liquid-paper interactions for printing processes, this approach is difficult and impractical to extend for real-time simulations, given the high grid resolutions required

for fine scale details. A particle-based method was proposed by Lenaerts et al. [129], but the large number of particles needed makes it unsuitable for real time applications on mobile devices.

The Lattice Boltzmann equation (Eq. 2.11) [193] has been proved to be a more suitable approach when macroscopic physics fail to provide a practical description [212]. LB divides the simulation domain into a regular lattice, where each point \mathbf{x} is connected with its neighbors via lattice vectors \mathbf{e}_i . $\phi_i(\mathbf{x}, t)$ represents the expected number of particles moving along \mathbf{e}_i at a given time. During each time step Δt , propagation and collisions of these particles are simulated over the lattice, redistributing them towards their equilibrium distribution functions $\phi_i^{(eq)}$:

$$\phi_i(\mathbf{x} + \mathbf{e}_i \Delta t, t + \Delta t) = (1 - \omega)\phi_i(\mathbf{x}, t) + \omega\phi_i^{(eq)}(\mathbf{x}, t) \quad (2.11)$$

where ω is a relaxation parameter that can be used as global viscosity. To reduce compressibility, He and Luo's $\phi_i^{(eq)}$ formulation [83] can be adopted.

Chu and Tai [48] presented an interactive system for simulating eastern ink paintings, based on three layers of simulation: *surface*, where user interaction takes place; *flow*, where water percolation through fibers is simulated with LB; and *fixture*, where ink pigments and glue are deposited. For our real-time liquid-paper interaction, we remove the *fixture* layer since we do not deal with pigments nor glue. We also change the *surface* layer simulation, employing our Navier-Stokes model instead, given theirs is tailored to simulate brushwork and ink deposition over paper, with a difficult extension to handle stirring and surface tilting gestures.

Hybrid Simulator. Macroscopic movement of the liquid over the surface is described by the NS equations, which then feed LB for percolation through the paper. The fluid that is transferred from the surface to the paper is subtracted from the NS simulation at the start of its next simulation step, effectively connecting both approaches in a natural way. The rate of liquid transfer from surface to paper is:

$$\left. \frac{\partial h}{\partial t} \right|_{seep} = c_s (\rho_{max} - \rho_p), \quad (2.12)$$

where ρ_p is the current density of liquid inside the paper and ρ_{max} is the maximum liquid density that the paper can hold. c_s modulates the transfer rate depending on the global viscosity from the NS simulation, which also affects global viscosity inside the LB simulation.

This hybrid scheme allows us to provide the user with natural touch- and gesture-based interaction and realistic liquid behavior on the surface, along with intricate realistic liquid percolation. The supplementary video shows our simulation system in movement.

2.6 CHARACTERISTIC CURVES

As explained in Section 2.4, Equation 2.4 expresses the final density as a function of the characteristic curve of the process, which in turn is defined by exposure e , and the input parameter vector \mathbf{b} (specific for each process). Given the local modulations introduced by the usage of liquids and time dependent parameters like setting or development times, a continuous dynamic model for this curve is needed. Moreover, for an interactive use, it

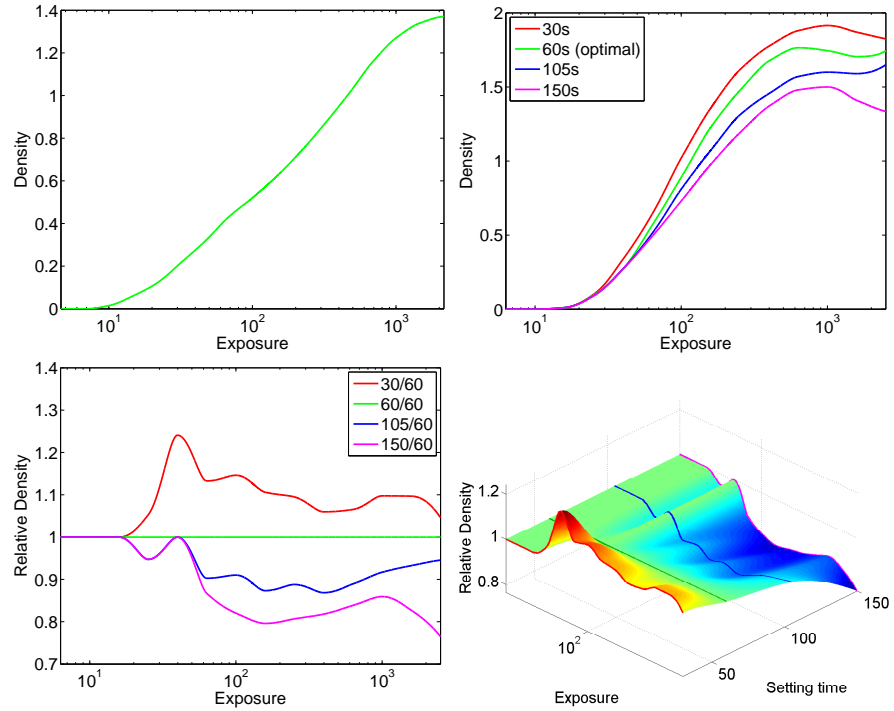


Figure 2.9: Top left: The global optimal curve $f_0(e)$. Top right: Characteristic curves $f_i(e, \beta)$ for the setting time parameter [186]. Bottom left: A new set of curves $f'_i(e, \beta)$ is obtained from Equation 2.13. Bottom right: 3D surface obtained by interpolating between the different $f'_i(e, \beta)$ curves, which we store as a regular 2D lookup table. The curves for all five parameters of our model can be found in the supplementary material.

needs to be both efficient and use intuitive parameters. Thus, low level computational models based on accurate simulations like Ramsden's one [177] are discarded. Instead, we adopt a data driven approach, and derive a model based on sparse sensitometric measurements.

We follow the work by Skladnikiewitz and colleagues [186], who focused on the wet collodion process given its historic importance as the first practical photographic technique. This leads to a five-dimensional vector \mathbf{b} , defining halide concentration, iodide to bromide ratio, cadmium concentration, setting time for the emulsion, and development time. Apart from matching the behaviors described in practical manuals [98], this data set has the additional advantage of using parameters similar to the ones an artist would use in the real world.

Although this model would formally be valid for accurate simulations of the collodion process, the approach described below can be adopted as an approximation to model other specific processes, if similar sensitometric data is available. We show how, using the global toning approach described in Section 2.4, our simulations can be extrapolated to other alternative processes like cyanotypes.

In [186], each parameter in \mathbf{b} is analyzed separately, obtaining a set of characteristic curves $f_i(e, \beta)$, where β is the value of the parameter i being analyzed. From these, optimal values of β are found², which define an optimal curve $f_i^{op}(e)$ for each parameter. As result of concatenated tests, a global

² The optimal value represents a good compromise between high final density and time allowed to physically handle the plates.

optimal characteristic curve $f_0(e)$ for the process is obtained (Figure 2.9, left), where all the parameters are fixed to their optimal values (refer to the Section 2.A for the complete set of data used).

However, we do not want to be constrained by optimal values. Instead, we want to give the user the freedom to modify any of the input parameters at will, which means that new characteristic curves should be used. So, we redefine the existing characteristic curves $f_i(e, \beta)$ (Figure 2.9, center left) with respect to its optimal ones (Figure 2.9, center right):

$$f'_i(e, \beta) = f_i(e, \beta) / f_i^{op}(e) \quad (2.13)$$

We then interpolate $f'_i(e, \beta)$ using piecewise cubic Hermite polynomials, obtaining a smooth 3D surface that can be stored and accessed efficiently as a 2D lookup table (Figure 2.9, right). Finally, we define a new global curve $g(e, \mathbf{b})$ as a function of the global optimal characteristic curve $f_0(e)$ and the redefined curves $f'_i(e, \beta)$:

$$g(e, \mathbf{b}) = f_0(e) \prod_{i=1}^5 f'_i(e, \beta) \quad (2.14)$$

Note that omitting any given parameter assumes its default optimal value, yielding $f'_i(e, \beta) = 1.0$. Although Equation 2.14 assumes each parameter i works independently from the others, our model yields coherent results, is based on ground-truth data, and is computationally very efficient. Coupled with our fluids simulation, it provides the real-time performance required for constant feedback and interactivity. Section 2.A includes an exploration of this model with different combinations of parameters.

2.7 IMPLEMENTATION

Using the described framework, we have built two prototypes, tailored to wet plate collodion and printmaking respectively. All the prototypes have been tested on an *Apple iPad*® (4th generation) with an A6X dual core CPU at 1.4GHz, 1GB of RAM and a PowerVR SGX 554 GPU. The Navier-Stokes fluids simulation runs on the CPU, and the Lattice-Boltzmann one on the GPU using OpenGL ES 2.0 shaders. Note that reading back from the GPU is required since the height map \mathbf{H} must be updated on the CPU with the result from the GPU LB simulation. Both simulations provide texture maps of 256x256 pixels, which are then up-sampled to the resolution of the results. The prototypes are CPU bound, with image processing times around 30 ms. For efficiency, the interactive image processing is performed using 1024x768 framebuffers; final results can be generated at the full resolution of the input image when saving them to memory. Our NS simulation runs at 27 frames per second, and our hybrid NS-LB at 22 fps, although suboptimal memory transfers from GPU to CPU (readback) limit the effective frame rate to 13 fps. This may be improved by offloading advection and upsampling to the GPU, and scheduling the readback before CPU workload, from separate threads.

Prototypes. Our collodion prototype follows the full pipeline described in Section 2.4, with the different maps and steps evolving in real time in front of the user, at actual speeds.

In order to show the versatility of our framework, we have implemented a second prototype with some simplifications and a more flexible pipeline. In

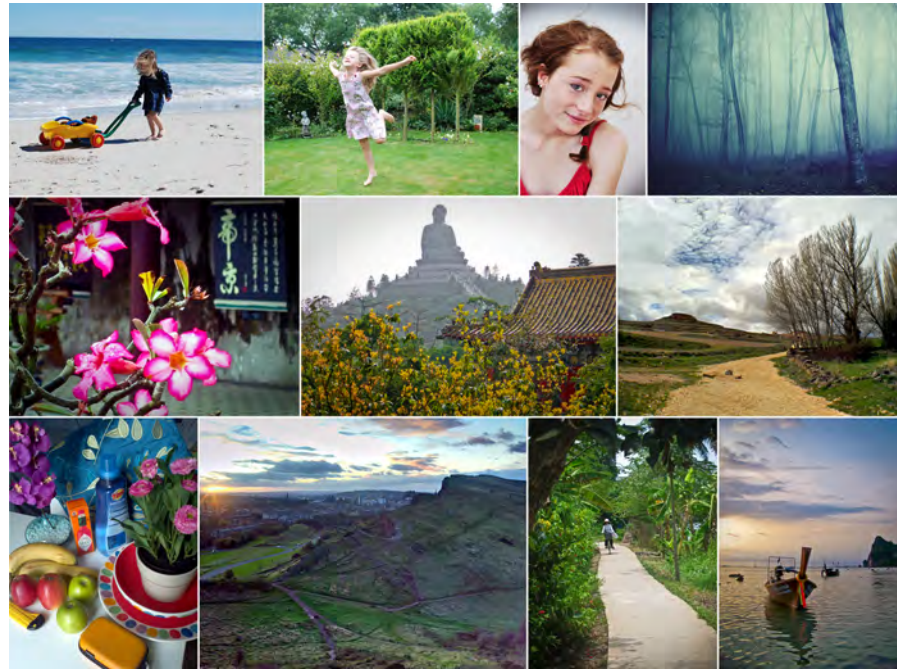


Figure 2.10: All the input images used for the results shown. Top row images courtesy of Michelle Meiring, Raven Cornelissen, Tracie Tee and Marius Adrian Rusu (respectively).

particular, the chemical mix and the spectral sensitivities can be changed at any time throughout the process, setting and exposure maps are based on global times set by the user, development time is obtained directly from the liquid (so pouring the emulsion and development occur simultaneously), and the fixing step is skipped, obtaining a transmission map directly from the density map.

2.8 RESULTS

Input images for the results shown can be found in Figure 2.10. We refer the reader to the accompanying video showing real-time demonstrations of our prototypes. Figures 2.1 and 2.11 show some results simulating wet plate photography, using the first prototype. Like in the real process, different combinations of the chemical components and processing times allow to achieve a rich range of tones in the final images, whereas varying degrees of heterogeneities are achieved by manipulating the liquid solutions (as shown also in Figure 2.12). In this case, no LB simulation is performed, since there is no percolation of the liquid within the glass plate.

We have created more results with our second prototype, which make use of our full hybrid NS-LB fluids simulation. Figure 2.13 shows additional results, including: i) wet plate rubytype; ii) cyanotype print; iii) a multi toning example with two different toners applied); iv) a basic color cross processing mode that modulates the luminance channel of the source image (in Lab color space) with the final transmission map, inspired by Bae et al. [17]; and v) a free-style printing result. The last two along with the cyanotype in Figure 2.1 show clearly visible results of liquid percolation. Different toners for analog prints are shown in Figure 2.14. Section 2.B includes Tables with the parameters used to create the results presented.



Figure 2.11: *Different results simulating the wet collodion process. Different combinations of the chemical mix and processing times, along with varying modulations by the aqueous solutions, lead to different, unique results, comparable to some results from Figure 2.2.*

2.9 CONCLUSIONS AND FUTURE WORK

We have presented a novel framework for digital image manipulations based on alternative analog photographic processes. We have implemented two fully working prototypes on a tablet device, for wet plate collodion and printmaking respectively, which provide an engaging way of physically manipulating digital images. Both realistic simulations and simplified physically based experiences can be built with our framework. In any case, our real-time data-driven model and hybrid fluids simulations lead to personal unique results, bringing a sense of craftsmanship and serendipity to the process. With this work, we have introduced a novel exploratory approach to image creation and manipulation when it comes to digital photography, something that was missing when compared with other artistic disciplines that have digital counterparts, like drawing and painting. We believe the ability to create engaging digital experiences by coupling physically-based simulations with natural user interfaces can appeal all kind of users, with potential uses for entertainment, educational or artistic purposes.

Our system is not free of limitations. Although based on real data, our simulations cannot be compared directly against real results because of many reasons. First, our data driven model does not take into account parameters that also influence the final outcome (temperature, aging of chemicals, additional chemicals that control the physical properties of emulsions and other liquids...). Second, our fluids simulations are simplified with respect to the behaviors on some rare supporting media used for prints, or the unique patterns that arise in the edges of plates due to dirt and organic matter. Third, digital cameras clamp IR and UVA wavelengths, whereas alternative photographic emulsions are very sensitive to UVA wavelengths specially, and so

differences in exposure and final look are expected. Fourth, it is outside of the scope of this paper to compare sensitometric data from real world and our simulations. All this, plus the inherent unique nature of the real results, makes side-by-side comparisons difficult. Nevertheless, one can see that our results share resemblances with the real examples shown in Figure 2.2, providing plausible results when performing similar manual work.

All of these limitations present interesting venues for future work, such as refinements in the final appearance based on grain formation during development, optical resolution of the emulsions during exposure, more general models for the chemical reactions or more detailed ones for specific processes, or extensions to the fluids simulations to deal with the very interesting artifacts that arise specially on the edges of the plates and other interesting liquid-paper interactions. We hope to also inspire extensions or adaptations of our framework to deal with more modern film stocks or other darkroom crafts based on the same concepts of physically based image manipulations, bridging the gap between current digital pipelines and these fascinating analog ones. Also it can be interesting to study and apply usability concepts over these new pipelines that favor not just the final result but also the processes that led to it.



Figure 2.12: Results with more visible modulations from the interaction with the liquids. Variations at different scales can be seen coming from either liquids themselves, heterogeneities in the mix, or both.



Figure 2.13: Top row: simulation of a rubytype and a cyanotype. Bottom row: a multitonned print (left) (as seen in Figure 2.2). Since the second prototype is not bounded by real physics, we can also create color (center) or more freestyle results (right).



Figure 2.14: Examples showing the use of different toners used in alternative printmaking, adapted from [181]: From left to right, top to bottom: cooper, selenium, polysulfide and blue toner. Note the reproduction of the solarization artifacts in the midtones for the cooper toner, as they occur in real life.

APPENDICES

2.A CHARACTERISTIC CURVES

Figures 2.15 to 2.19 show the data from [186] adapted to create our data-driven characteristic curve model presented in Section 2.6. Our model is explored in Figures 2.20 to 2.25. Starting from a reference result (Figure 2.20) we test different values for the input parameters **b**: halide concentration, iodide/bromide ratio, cadmium iodide ratio, setting and development times. The results show the model behaves in concordance with [186].

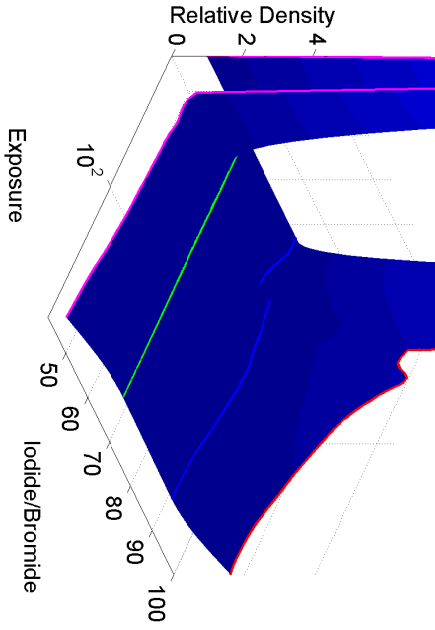
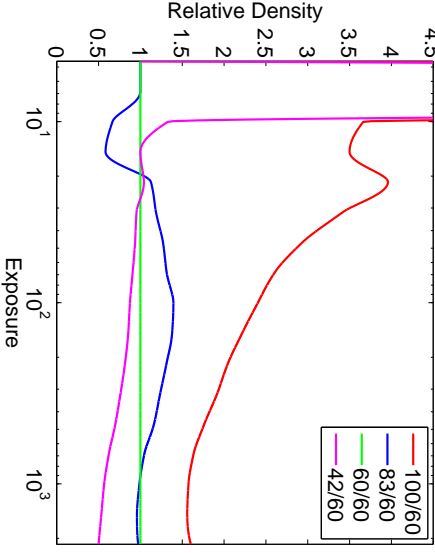
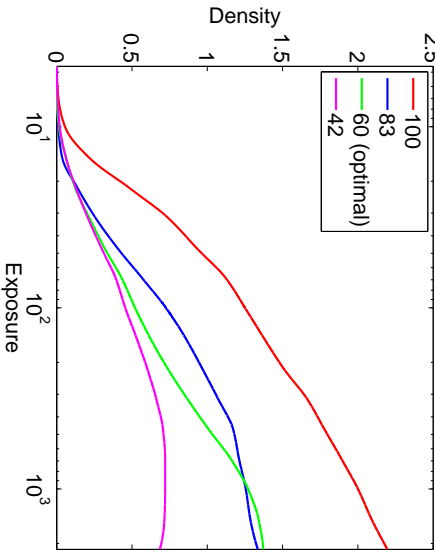
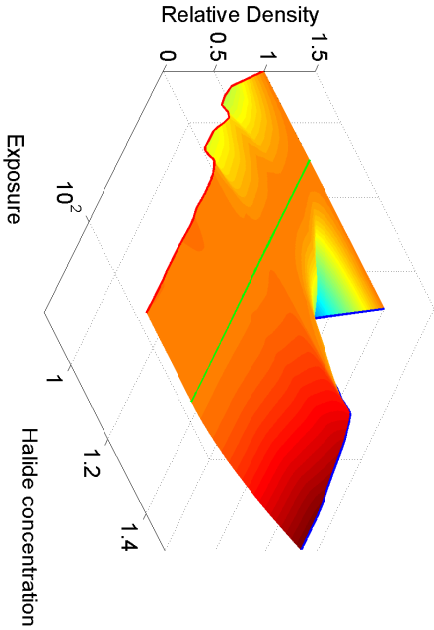
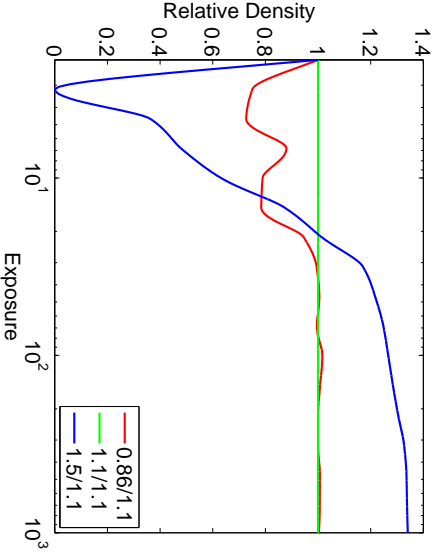
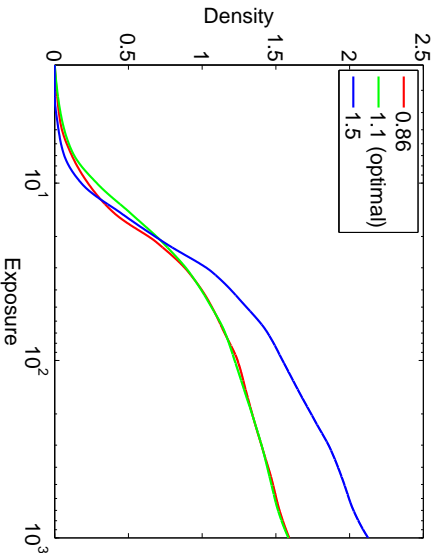


Figure 2.15: Halide concentration curves.

Figure 2.16: Iodide to bromide ratio curves. Curves for 100 and 42 clumped for a better visualization (peak values at 100 and 50 respectively).

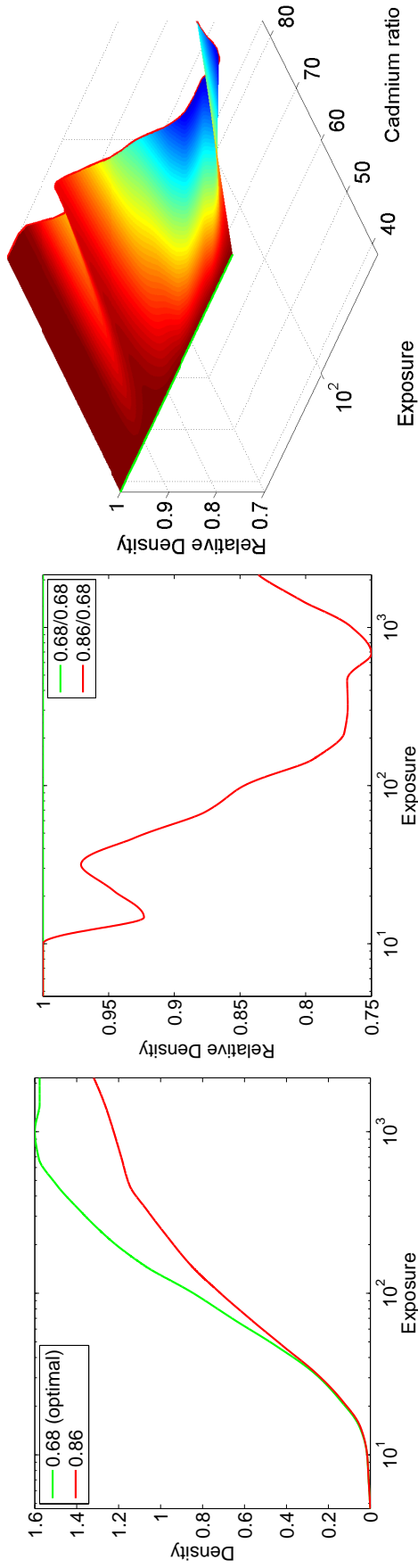


Figure 2.17: Cadmium iodide curves. Different ratios of cadmium and ammonium for the iodide part of the mix are tested.

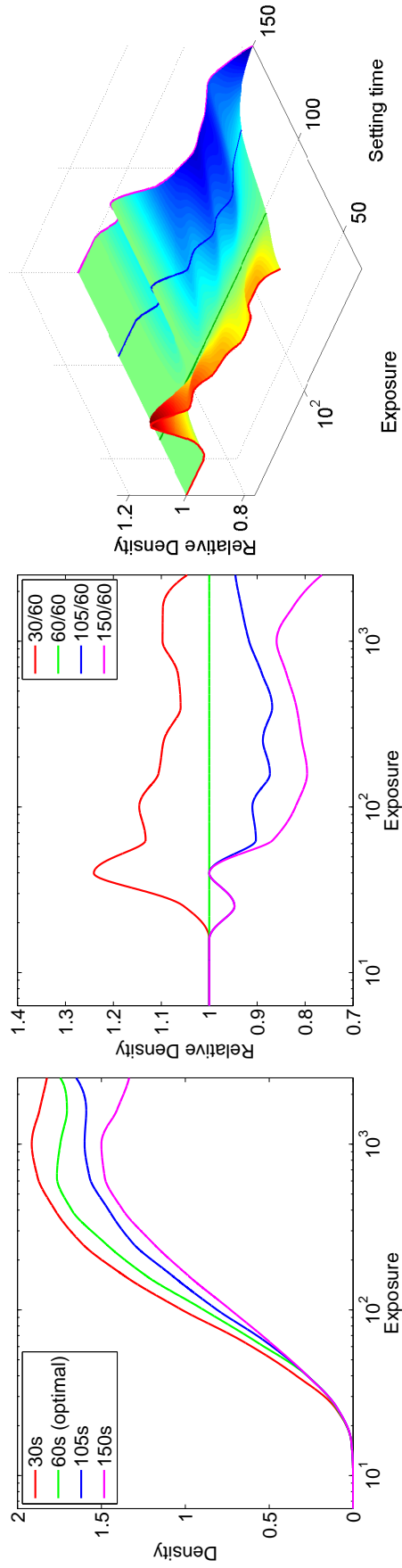


Figure 2.18: Setting time curves. We add $f_i(e, 0) = 0$ assuming there will be no image if emulsion is not allowed to set (not shown for clarity).

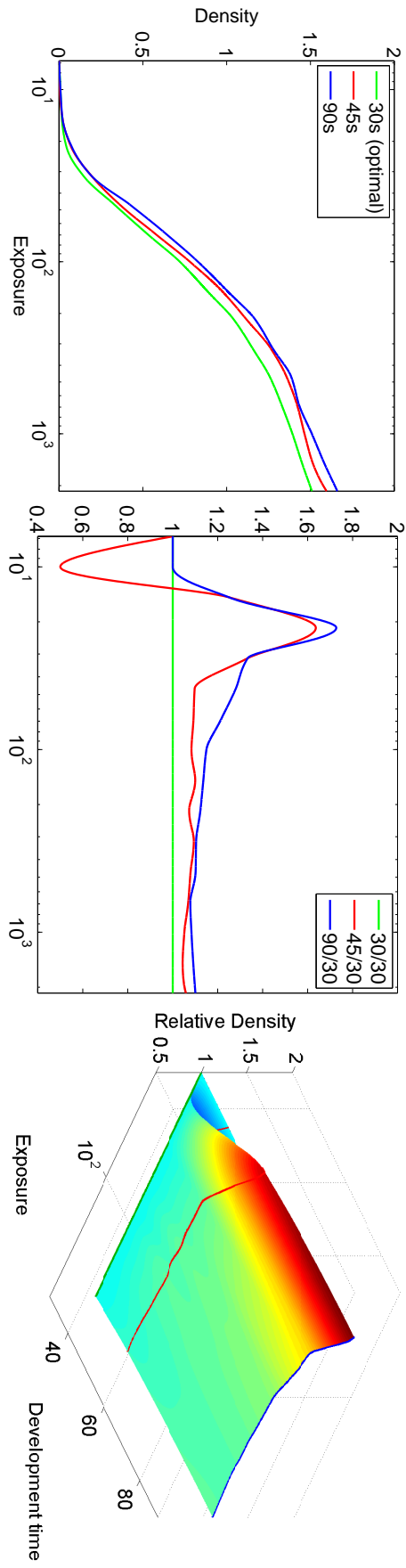


Figure 2.19: Development time curves. We add $f_i(e, 0) = 0$ assuming there will be no image if there is no development (not shown for clarity).



Figure 2.20: Reference image obtained with the following parameters: halide concentration of 1.0, iodide/bromide ratio of 60/40, cadmium ratio of 0.68, average setting time of 30s, 17 seconds of exposure and development time of 30s on average. Each of the following tests changes one of this parameters independently (except exposure time).



Figure 2.21: From left to right: Exploration of the iodide/bromide ratio parameter with values of 42/60, 70/30 and 100/0 respectively. A higher ratio produces an increase in contrast due to a steeper and shorter spectral sensitivity (Figure 2.7, center) and in final density (as seen in Figure 2.16).



Figure 2.22: From left to right: Exploration of the cadmium iodide ratio parameter with values of 0.68, 0.78 and 0.86 respectively. In concordance with Figure 2.17, the higher the amount of cadmium iodide, the lower the final density achieved.



Figure 2.23: From left to right: Exploration of the halide concentration parameter with values of 0.9, 1.2 and 1.5 respectively. Slight changes in density appear in the highlights with increasing values of this parameter, while density in shadows decreases more abruptly (Figure 2.15).



Figure 2.24: From left to right: Exploration of the setting time with average values of 30, 60 and 90 seconds respectively.



Figure 2.25: From left to right: Evolution of the results as development is calculated. Average values of 10, 20 and 30 seconds are shown. After 30 seconds, little more density is obtained, as seen in Figure 2.19.

2.B RESULTS







	β_1	β_2	β_3	β_4	β_5	τ_e	τ_f
	1.0	0.68	50/50	28s	24s	13s	32s
	1.0	0.68	45/55	31s	27s	10s	24s
	1.0	0.68	90/10	33s	23s	8s	22s
	1.0	0.68	42/58	31s	22s	21s	35s
	1.0	0.68	42/58	30s	21s	28s	30s
	1.0	0.68	70/30	31s	23s	13s	24s

Table 2.1: Parameters used for creating different results shown. β_1 is halide concentration, β_2 is cadmium ratio (the same for all results), β_3 is iodide/bromide ratio, β_4 is setting time τ_s , β_5 is development time τ_d , τ_e is exposure time and τ_f is fixing time.









	β_1	β_2	β_3	β_4	β_5	τ_e	τ_f
	1.0	0.68	55/45	30s	31s	13s	N/A
	1.0	0.68	60/40	25s	23s	12s	24s
	1.1	0.68	42/58	30s	34s	15s	N/A
	1.0	0.68	45/55	30s	27s	14s	N/A
	1.0	0.68	70/30	30s	23s	16s	N/A
	1.0	0.68	45/55	30s	24s	30s	N/A
	1.0	0.68	60/40	34s	23s	10s	31s
	1.0	0.68	42/58	30s	28s	27s	N/A

Table 2.2: Parameters used for creating different results shown. β_1 is halide concentration, β_2 is cadmium ratio, β_3 is iodide/bromide ratio, β_4 is setting time τ_s , β_5 is development time τ_d , τ_e is exposure time and τ_f is fixing time. N/A is used when the prototype used for creating the image did not implement that feature.



Figure 2.26: Results shown in the supplementary video. Top: result using the wet plate prototype. Mild fogging appears as result of a short exposure time and prolonged development. Bottom: Platinum print obtained with the printmaking prototype.

LIQUID-ON-LENS SIMULATION

In a similar spirit to the previous Chapter, here we focus on the manipulation of the optical path of a camera by placing liquids or refractive layers on the surface of the its lens. This is something artists have been using to create interesting and unique distortions. To provide an engaging interactive user experience, we ensure a natural interaction using the fluids solver from our previous framework, running on a tablet device with touch and gestures based input. To obtain realistic and predictable results, we use ray tracing to compute the deformation of the light rays passing through the manipulated optical paths. Given the relatively low computing power of mobile devices and the interactive rates required, we propose using efficient screen space ray tracing algorithms. Our prototype shows use cases impossible to achieve with existing systems and traditional image editing pipelines.

This work was developed during a three-months internship at the *Advanced Technology Labs* at Adobe Systems in San Jose (California, USA); and was recently granted with US Patent.

G. Wilensky, A. Krishnaswamy & J. I. Echevarria

SYSTEMS AND METHODS FOR SIMULATING THE EFFECTS OF LIQUIDS ON A
CAMERA LENS

US Patent US9176662 B2 (2015)

3.1 INTRODUCTION

One of the more straightforward manipulations a photographer usually performs during the capture of a scene is to alter the optical path of the rays from the scene to the sensor. This is usually done using conventional lenses and changing focal length, aperture size or focus. However, more specialized hardware allows additional degrees of freedom to achieve tilt-shift effects or selective focus. Taking this to the limit, the artist can place any refractive object in front of (or replacing) the main lens of the camera, or manipulate the surface of the lens by adding some liquid over it to obtain very creative distortions (Figure 3.1 shows some examples).

Drawing inspiration from these manipulations, we created a novel tool that allows the user to pour and stir liquid over a virtual lens, recapturing then a distorted image as would have been seen by the sensor of the camera.

3.2 OVERVIEW

Our system is composed of a virtual sensor, a virtual lens, the interactive liquid layer, and the scene to be captured (from the sensor feed in real-time) or manipulated (from a single image). Then, rays are traced from the sensor to the scene to obtain the final image. Figure 3.2 shows an overview of the system.



Figure 3.1: Left: Photo taken through an object made of crystal that alters the path of the light rays from the scene to the lens and sensor (by Paulina Aleshkina). Right: Photo taken after the selective application of a liquid over the lens (by Jill Auville).

3.3 IMPLEMENTATION DETAILS

Since we want real-time visual feedback, we need to render realistic refraction as fast as possible, thus discarding the use of conventional CPU-based raytracing given its computational cost. So, we make use of GPU screen-space ray tracing given its quality and efficiency [163]. Then, we model each component of the optical system (lens and liquid layer) efficiently using 2D height and normal maps (see Figure 3.2).

The virtual scene to capture is a 2D plane that will be sampled by the rays coming from the optical system. We simplify Oliveira and Brauwers’ approach [163] by assuming all rays through the lens and liquid are traced orthogonally from the virtual sensor, so we can use a simpler binary search for traversing the depth maps [174]. We extend these refraction calculations adding other optical effects like light dispersion, scattering and color tint for a wider range of results (see Figure 3.3 for some examples).

For light dispersion, we take the refracted outgoing direction from the liquid to the scene, and create two additional rays by using a user-defined dispersion coefficient. Then we use each ray for sampling the red, green and blue channels at close but different positions, so the effect of light dispersion is achieved. To be completely accurate, we should trace three different rays from the sensor, but this simpler approach produces physically-plausible results more efficiently.

Accurate scattering of light through the liquid would require accumulating several different incoming ray directions, making it an impractical approach for real-time. We simplify its simulation by creating an additional blurred version of the source image based on a user-defined scattering radius. This way, depending on whether the ray traverses the fluid layer or not, we sample the original image or the blurred one. For efficiency, we blur the image using a separable gaussian kernel applied as a GPU shader.

For the interactive liquid layer, we use our Navier-Stokes fluids solver presented in the previous Chapter (Section 2.5) with touch and gestures enabled interaction. As said before, we model the layer itself with height map extracted directly from the fluids simulation. A normal map is computed efficiently on the fly on the GPU.

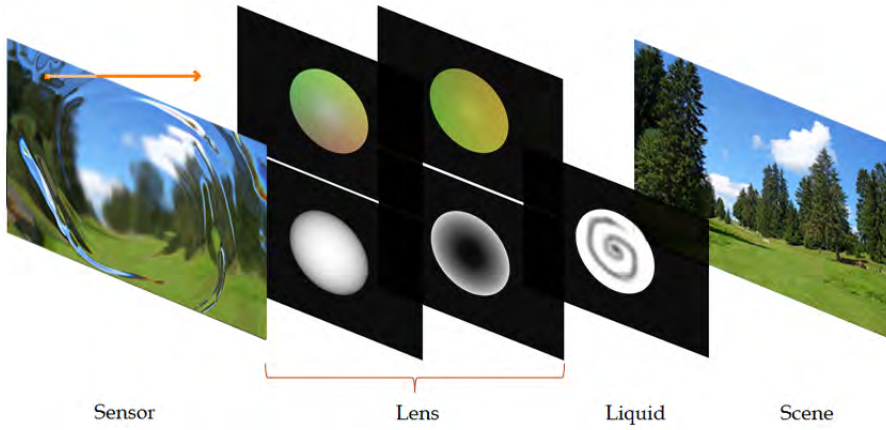


Figure 3.2: Schematic representation of our system. Assuming the distance between the lens and liquid elements is small, we trace orthogonal rays (in orange) from the virtual sensor to the scene. For efficiency, we trace just one ray per pixel on the final image. Given we are using a screen-space ray tracing approach, we model our virtual lens with normal and depth maps (for both the front and back facing sides of the objects). We do the same for the liquid distribution, getting its depth map from the fluids simulation and deriving its normal map on the fly inside the GPU shader.

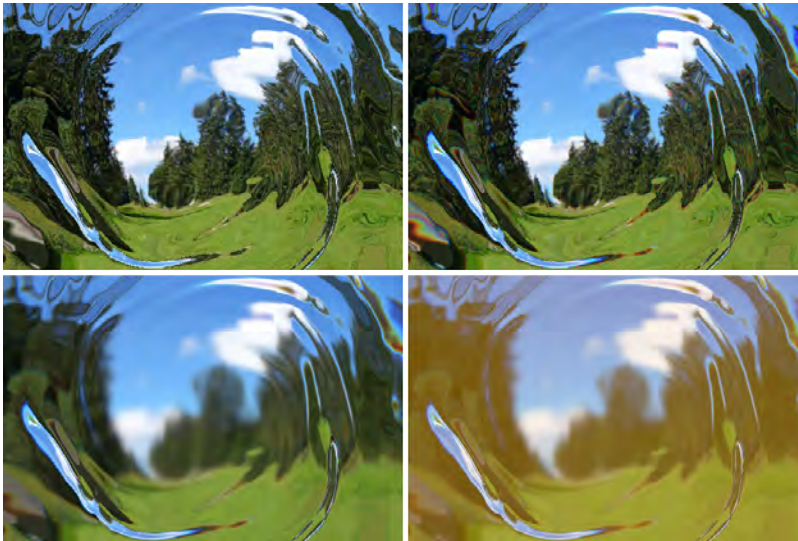


Figure 3.3: Top left: Refraction after [163]. Top right: Light dispersion added. Bottom left: Light dispersion plus scattering. Bottom right: Light dispersion plus scattering and a honey-like color tint.

3.4 RESULTS

We demonstrate a prototype running on an *Apple iPad*[®] (4th generation) with an A6X dual core CPU at 1.4GHz, 1GB of RAM and a PowerVR SGX 554 GPU. The fluids simulation run on the CPU while the ray tracing and additional processing runs entirely on the GPU.

Figure 3.4 shows a screenshot of the prototype, with the interaction with the fluid happening in the lens area located on the bottom left corner. The rest of the controls expose the different parameters that affect the result, both for the physics of the fluids simulation and for the optical properties of all

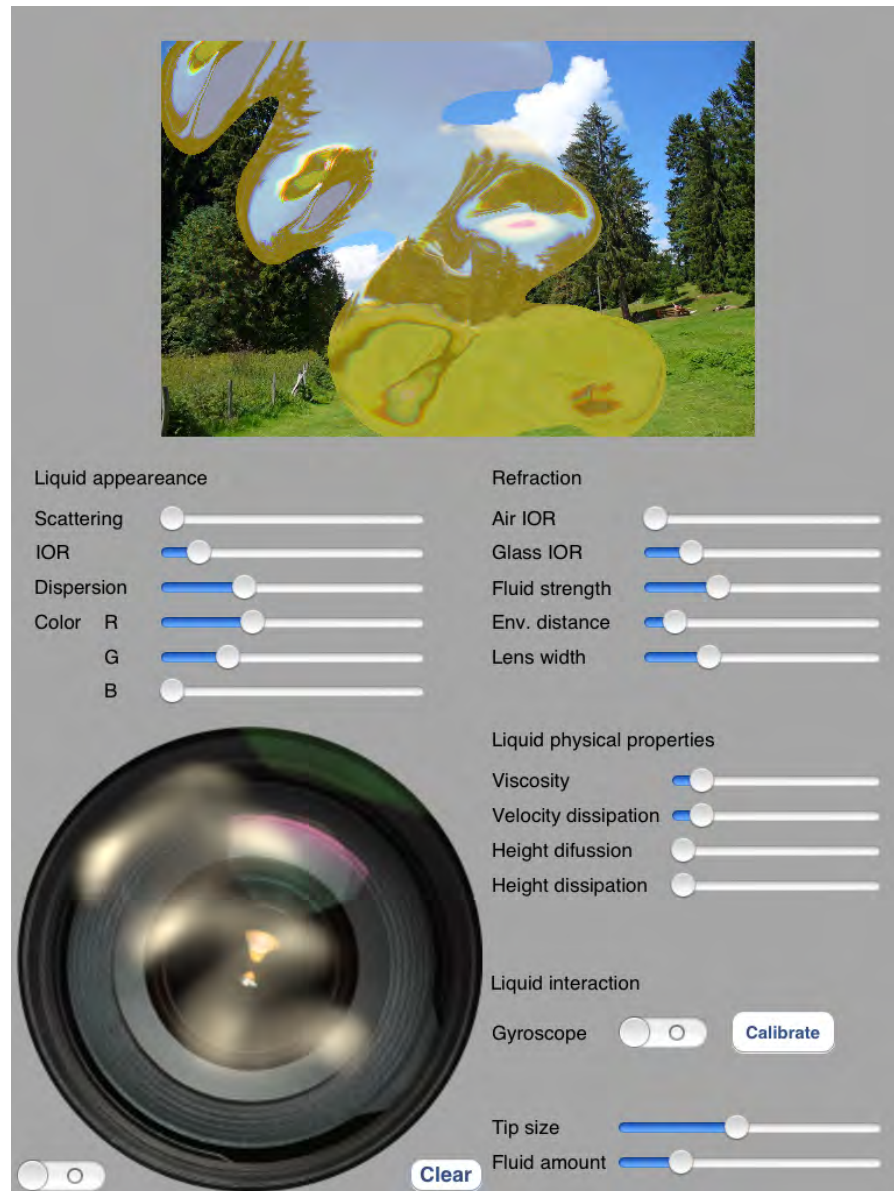


Figure 3.4: Screenshot showing our prototype. The development interface exposes all the parameters that control the physics simulation and the optical properties of the system.

the components. Figures 3.5 and 3.3 show different results obtained with our prototype. We refer the reader to the accompanying video for watching it in action.

Some of our results may present unnatural sharp transitions from the liquid to the clear areas. This happens mostly because we are currently assuming both the liquid and the background to be in focus at the same time. Those transitions would be smoother if proper defocus blur is calculated for each layer [86]. Related to this, we are currently modeling the scene as a 2D plane, so having per-pixel depth information would allow more accurate calculations for the refraction and also the defocus blur, being able to simulate more complex camera models [127, 92].



Figure 3.5: *Examples simulating different effects very difficult to achieve using regular image editing tools, but very easy to obtain using our prototype. It can be seen they present a similar look and feel to the real ones from Figure 3.1.*

3.5 CONCLUSION

We have presented a new image editing process based on physical manipulations. Although it could be run on any computer with some limitations, it is specially tailored for mobile devices, which by means of touch and gestures based user input can provide a more natural and engaging experience. In this work we used part of our previous framework to model one kind of creative manipulations an artist can easily perform over the optical path. Future work may extend the range of manipulations to general optical arrangements, from more traditional lens designs to arbitrary custom ones. With this work, we demonstrated mobile devices have the potential to provide novel digital experiences for digital image editing that go way beyond traditional approaches with simplified touch-based interfaces, something we would like keeping exploring in the future.

Part III

PERCEPTUAL RECONSTRUCTION OF 2D IMAGES

This part deals with reconstruction algorithms to obtain plausible and pleasing images from a perceptual point of view in two common but different scenarios. First we introduce our SMAA anti aliasing filter for videogames and other real time applications. We explain how to perform a conservative but comprehensive morphological analysis of the aliased image to obtain smooth but sharp reconstructions for different sample counts. Next, we demonstrate practical HDR imaging for mobile photography, describing strategies for capturing multiple exposures that are then perceptually fused directly on the device using computational photography techniques.

In this Chapter we present a new image-based, post-processing antialiasing technique, which offers practical solutions to the common, open problems of existing filter-based real-time antialiasing algorithms. Our method shows for the first time how to combine morphological antialiasing (MLAA) with additional multi/supersampling strategies (MSAA, SSAA) for accurate sub-pixel features, and how to couple it with temporal reprojection; always preserving the sharpness of the image. All these solutions combine synergies making for a very robust technique, yielding results of better overall quality than previous approaches while more closely converging to MSAA/SSAA references but maintaining extremely fast execution times. Additionally, we propose different presets to better fit the available resources or particular needs of each scenario.

This work is published in *Computer Graphics Forum* and presented at *Eurographics 2012*. It builds on top of our previous GPU version of MLAA, published in the *GPU Pro 2* book. Having participated in the original algorithm, for this project I worked with Jorge Jimenez analyzing the problems and limitations of the existing methods and discussing and implementing solutions to them.

J. Jimenez, J. I. Echevarria, T. Sousa & D. Gutierrez
SMAA: ENHANCED SUBPIXEL MORPHOLOGICAL ANTIALIASING
Computer Graphics Forum, Vol.31 (2), Eurographics 2012

J. Jimenez, B. Masia, J. I. Echevarria, F. Navarro & D. Gutierrez
PRACTICAL MORPHOLOGICAL ANTI-ALIASING (MLAA)
GPU Pro 2: Advanced Rendering Techniques (2011)

4.1 INTRODUCTION

Aliasing is one of the longest-standing problems in computer graphics, producing clear artifacts in still images (spatial domain) and introducing flickering animations (temporal domain). While using higher sampling rates can ameliorate its effects, this approach is too expensive and thus not suitable for real-time applications. During the last few years we have seen great improvements in real-time rendering algorithms, from complex shaders to enhanced geometric detail by means of tessellation. However, aliasing remains one of the major stumbling blocks for trying to close the gap between off-line and real-time rendering [10].

For more than a decade, *supersample antialiasing* (SSAA) and *multisample antialiasing* (MSAA) have been the gold standard antialiasing solutions in real-time applications and video games. However, MSAA does not scale well when increasing the number of samples and is not trivial to include in modern real-time rendering paradigms such as deferred lighting/shading [107, 11, 106]. To exemplify this problem with numbers, MSAA 8x takes an average of 5.4 ms in modern video games with state of the art rendering engines (increasing to 7.7 ms on memory bandwidth intensive games) on a NVIDIA GeForce GTX 470. Memory consumption in this mode can be as

high as 126 MB and 316 MB, for forward and deferred rendering engines respectively, taking 12% and 30% of the rendering time of a mainstream GPU equipped with 1GB of memory. This problem is aggravated when HDR rendering is used, as the memory consumption and bandwidth increases even further.

Recently, both industry and academia have begun to explore alternative approaches, where antialiasing is performed as a post-processing step [106]. The original *morphological antialiasing* (MLAA) method [180] gave birth to an explosion of real-time antialiasing techniques, rivaling in quality the results of MSAA and with a performance within the [0.1 – 5] ms range. However, analyzing the current generation of filter-based antialiasing techniques, they all share at least some of the following problems:

- Most edge detection methods only take into account numerical differences between pixels, ignoring the fact that the surroundings of an edge also affect how humans perceive them.
- The original shape of the objects is not always preserved; an overall rounding of the corners is most of the times clearly visible in text, sharp corners and subpixel features.
- Most approaches are designed to handle horizontal or vertical patterns only, ignoring diagonals.
- Real subpixel features and subpixel motion are not properly handled.
- Specular and shading aliasing is not completely removed, especially when it happens at subpixel level.

Addressing all these issues while maintaining practical real-time performance poses a real challenge. We propose a novel post-process antialiasing technique, *Enhanced Subpixel Morphological Antialiasing* (SMAA). Our approach follows the divide-and-conquer paradigm, and tackles these complex problems separately, offering simple, modular solutions. First, we extend the number and type of edge patterns in order to keep sharp geometric features while processing also diagonal lines. Second, by adding multi/supersampling and temporal reprojection to morphological antialiasing, we are able to reconstruct real subpixel features and handle subpixel motion. Last, we introduce a robust edge detection that exploits local contrast along with accelerated yet precise distance searches for a more accurate pattern classification.

Given the modular nature of our approach, specific features can be enabled or disabled, adjusting to the needs of each particular scenario and hardware configuration. We propose four different modes, from the simplest to the more sophisticated version, which includes a novel combination of antialiasing as a post-process filter, and both spatial and temporal supersampling. This flexibility allows for direct, practical use of our technique even in current mainstream hardware. Furthermore, we have made public all the source code at <http://iryoku.com/smaa/>, including very exhaustive comments for both implementation and integration, to ensure both reproducibility and an easy and fast adoption of the technique.

4.2 RELATED WORK

The simplest form of real-time antialiasing is *supersampling antialiasing* (SSAA), which involves rendering the scene at a higher resolution, then downsam-



Figure 4.1: Example of SMAA 4x integrated in the Crysis 2 game. The insets show the differences between MLAA [107], our novel SMAA T2x and 4x algorithms and MSAA 8x as reference. For 1080p frames, the average cost of SMAA T2x is 1.3 ms and 2.6 ms for SMAA 4x, measured on a NVIDIA GeForce GTX 470.

pling to the final resolution. It is also the basis of *multisampling antialiasing* (MSAA) [7], where the color of a pixel is only calculated once instead of running at subsample frequencies. To display the scene, all samples are aggregated using some filter (a resolve operation). Although recent related techniques like CSAA [211] and EQAA [9] reduce bandwidth and storage costs by decoupling coverage from color, depth and stencil, these methods still inherit MSAA drawbacks.

The addition of new real-time rendering paradigms such as deferred shading [55, 79, 71] and the lighting pre-pass [62], along with current limitations in graphics hardware, have recently motivated a great amount of exciting new research in this field [106]. Most of the recent antialiasing solutions handle the aliasing problem as a post-process, devising filters that are applied over the final, aliased image, usually rendered at final display resolution. The basic idea is to find discontinuities on the image and to blur them in clever ways, in order to smooth the jagged edges. While the approach is not entirely new [34, 196, 97], some advanced versions of it have been only recently applied in games [185, 119, 188]. All these techniques alleviate the aliasing problem, although the sharp definition of the edges is obviously lost to a degree. More refined solutions like *directionally localized antialiasing* (DLAA) [12], use smarter blurs that produce very natural results and good temporal coherence. Nevertheless, these approaches still yield blurrier results than MSAA.

Other solutions, such as *morphological antialiasing* (MLAA) [180], try to estimate the pixel coverage of the original geometry based on the color discontinuities found in the final image. Reshetov's original work provides great results, but the proposed CPU implementation is not fast enough to be used in real-time. This triggered a number of real-time implementations that run on different hardware platforms, such as the GPU [29, 8, 108], Playstation 3 SPUs and hybrid approaches that use both CPU and GPU [106, 53]. *Topological reconstruction antialiasing* (TMLAA) [28] uses topological information to recover subpixel features from the final image. However, this reconstruction can only fill one-pixel-sized holes, and it is not clear how well its assumptions work for animated sequences. *Fast approximate antialiasing* (FXAA) [138] approaches the subpixel problem by simply attenuating such features, which enhances the perceived temporal stability. However, its resulting images are still not at the quality level of standard methods like MSAA.

Deviating from pure image-based solutions, in the *distance-to-edge antialiasing* technique (DEAA) the forward rendering pass calculates and stores the distances of each pixel to near triangle edges with subpixel precision [106]. The post-process pass uses this information to derive blending coefficients. Similar in spirit, Persson's GPAA [170] and GBAA [106] use additional geometric information for coverage calculation. This produces almost perfect gradients with great temporal stability. However, working at final display resolution means they cannot handle subpixel features. Furthermore, they require either additional output buffers in the main pass or additional geometry passes. Providing better handling of subpixel features in deferred engines, *subpixel reconstruction antialiasing* (SRAA) [46] combines regular shading at final display resolution with supersampled geometry maps (normals and depth). Then, a super-resolution color image is built propagating the shaded samples over those maps; the resulting image is finally downsampled again to final screen resolution. Despite bringing subpixel features to the table, they are based on heuristic estimations and the resulting gradi-

ents are in general of lower quality when compared with other approaches. *Directionally adaptive edge antialiasing* [95] leverages MSAA subsample values for better gradient and color estimation. However, execution times are on the high side limiting the viability of the method to specific projects.

Finally, in very demanding realtime scenarios with complex shading and geometry, temporal antialiasing approaches have regained interest recently [160, 207] [106] (see section *Anti-Aliasing Methods in CryENGINE 3*). The main idea is to distribute the cost of supersampling over contiguous frames. Our work also takes this aspect into account, handling subsamples via temporal reprojection. In a different context, the work of Yang and colleagues [208] aims at restoring jagged edges that occur after nonlinear image processing filters, for which they require that the original, alias-free image be available.

Table 4.1 provides a detailed summary of the features supported for a representative selection of filter-based antialiasing techniques, including our work. This selection covers most of the recent major publications in the field, and includes all those for which implementations are available and are currently in use, in order to perform fair comparisons. It can be seen how each existing technique aims at solving a subset of all the problems involved, at the cost of leaving others out. In contrast, we provide a more holistic approach and systematically tackle all of them, while maintaining modularity by design.

LATER WORK After the publication of SMAA, new real time anti aliasing methods have appeared. Similar to our morphological component, CMAA [51] and AXAA [155] aim for a more conservative processing of edges, with a low performance hit. They are not able to handle several samples per pixel, though. Other methods also follow a similar approach to ours, with careful processing of borders and the gathering of additional samples per pixel for a better reconstruction of subpixel features [161, 162, 56]. However, their actual implementations are tied to specific hardware features from vendors like NVIDIA or AMD, so they are not universally available. A promising different route is to compute an analytical filtering to handle all the sources of aliasing at once [15], but its computational cost is still not practical for real time. In summary, it could be argued that SMAA is still the preferred anti aliasing solution for cutting edge videogames and other real time applications for PC, consoles and mobile devices; with its modular approach replicated by later AA solutions.

4.3 MORPHOLOGICAL ANTIALIASING

Morphological antialiasing (MLAA) [180], tries to estimate the pixel coverage of the original geometry. To accurately rasterize an antialiased triangle, the coverage area for each pixel inside the triangle must be calculated to blend it properly with the background (assuming a back-to-front rendering order). MLAA begins with an image without antialiasing (no coverage taken into account during rasterization), so it reverses the process by re-vectorizing the silhouettes, in order to estimate such coverage areas. Then, since the background cannot be known after rasterization, MLAA blends with a neighbor, assuming that its value is similar to the original background. Figure 4.2 describes this process; we refer the reader to the original publication for a more detailed explanation [180].

Supported features for a selection of filter-based antialiasing techniques. Memory footprint in terms of: backbuffer size/depth buffer/additional render targets. Performance is given for 1080p on a NVIDIA GeForce GTX 470, with exception of: a) Reshetov's [180] CPU-based implementation, which is measured on a Core i7 2620M @ 2.7GHz; b) AMD's exclusive MLAA [8], which is measured on a AMD Radeon HD 6870; and c) SRAA [46], whose times come from a GeForce GTX 480 (more powerful than the GeForce GTX 470). Please note that our SMAA T1S2x and 4x times include the resolves. MSAA performance numbers, calculated as the overhead over the same scenes without antialiasing, are 1.57 ms for MSAA 2x, 2.3 ms for MSAA 4x and 4.3 ms for MSAA 8x. Brute force SSAA overhead grows linearly with the number of samples, SSAA 16x takes an additional cost of 285 ms for the example in Figure 4-1.¹ For fairness, we measured the 4x mode of our algorithm on the same GPU, yielding a performance of 1.82 ms.² We measured the times of FXAA 3.11 in default (12) and extreme (39) presets.

	MLAA	MLAA	MLAA	FXAA _{3.11}	DLAA	SRAA	SMAA			
							1x	52x	T2x	4x
Sharp geometric features						yes	yes	yes	yes	yes
Diagonals							yes	yes	yes	yes
Subpixel features						yes		yes	yes	yes
Supersampled shading								yes	yes	yes
Local contrast adaptation				implicit	implicit	yes (n/a)	yes	yes	yes	yes
Accurate distance searches	yes	yes		depends	yes	yes (n/a)	yes	yes	yes	yes
Accurate gradients*	yes	yes	yes	yes	yes		yes	yes	yes	yes
Sharpness preservation*	medium	low	high	medium	medium	low	high	high	high	high
Ghosting-free	yes	yes	yes	yes	yes	yes	yes		yes	
Input (color/depth)	1x n/a	1x n/a	1x 1x	1x n/a	1x n/a	1x 4x	1x n/a	2x n/a	1x n/a	2x n/a
Memory footprint	1x/1x/n/a	1x/1x/n/a	1x/1x/1.5x	1x/1x/0x	1x/1x/1x	1x/4x/0x	1x/1x/2x	2x/2x/2x	1x/1x/4x	2x/2x/4x
Performance	350 ms	6.6 ms ¹	0.98 ms	0.62 ms / 0.83 ms ²	2.12 ms	2.5 ms	1.02 ms	2.04 ms	1.32 ms	2.34 ms

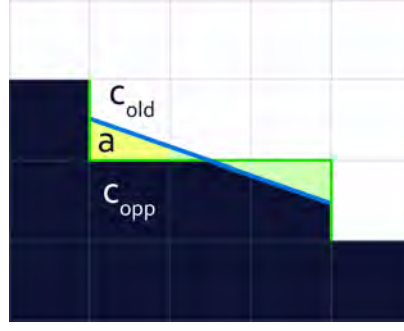


Figure 4.2: MLAA first finds edges by looking for color discontinuities (green lines), and classifies them according to a series of pre-defined pattern shapes, which are then virtually re-vectorized (blue line), allowing to calculate the coverage areas a for the involved pixels. These areas are then used to blend with a neighbor. For example, the pixel C_{opp} fills the area a of the pixel C_{old} : $c_{new} = (1 - a) \cdot c_{old} + a \cdot c_{opp}$.

Several morphological antialiasing implementations appeared after Reshetov's original paper [106]. Jimenez's MLAA [108] is one of the fastest and most documented. Its key feature is the use of novel *texture structures* for great performance improvements. These textures are used to encode the location of the edges and coverage areas, as well as the precomputed areas for blending. The algorithm works in three passes: edge detection (which is performed using depth or luma information), pattern detection plus calculation of coverage areas, and final blending. Pattern detection is performed by searching both ends of an edge (*distance searching*), halving the necessary iterations by using hardware bilinear filtering. Once the ends are reached, the algorithm looks at the *crossing edges*, which provide a mechanism for straightforward pattern classification; these crossing edges are the perpendicular edges with respect to the direction of a search (see for example the vertical green lines in Figure 4.2). With length and crossing edges information, the coverage area is retrieved with a single access to a precomputed texture, and used for the final blending. Figure 4.3 exemplifies the different steps and components of the pipeline. We choose this MLAA implementation as a starting point for our algorithm, and refer the reader to the original publication for a more comprehensive description [108].

4.4 SMAA: FEATURES AND ALGORITHM

In this section we present the core components of SMAA, their motivation and the main algorithmic ideas (see Figure 4.4). We build on Jimenez's MLAA pipeline, improving or completely redefining every step. In particular, we improve edge detection by using color information with local contrast adaptation for cleaner edges. We extend the number of patterns handled for sharp geometric features preservation and diagonals processing. In a similar fashion, we enhance pattern handling with accurate and fast distance searches for a more reliable edge classification. Last, we show how morphological antialiasing can be accurately combined with multi/supersampling and temporal reprojection. Although our new technique shares some of the core ideas of MLAA, it constitutes a major overhaul in terms of quality and robustness (see Figure 4.3).

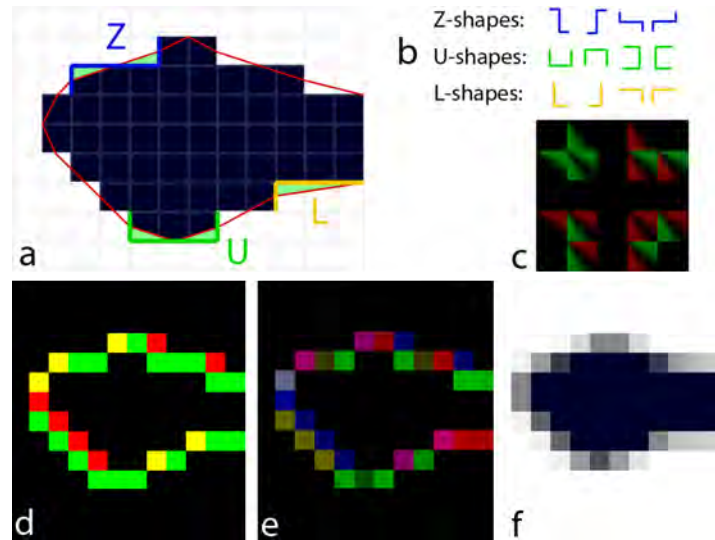


Figure 4.3: MLAA overview. (a) Input image, with the intended approximation outlined by red lines and the coverage areas shown in green. (b) Predefined patterns in the original algorithm [180]. (c) Precomputed areas texture in Jimenez’s GPU implementation [108]. (d) Detected edges. (e) Calculated coverage areas. (f) Final blending. Our SMAA algorithm overhauls the whole pipeline by extending (b) and (c) for sharp geometric features and diagonals handling. Local contrast adaptation removes spurious edges in (d). Extended patterns detection and accurate searches improve accuracy in (e). SMAA can handle additional samples in (f) for accurate subpixel features and temporal supersampling.

	Temporal Stability	Shape reconstruction	Subpixel handling	Supersampling
Local contrast adaptation	●	●		
Sharp geometric features detection		●		
Diagonal patterns detection	●	●		
Accurate distance searches	●	●		
Temporal supersampling	●	●	●	●
Spatial multisampling	●	●	●	

Figure 4.4: Overview of the key weaknesses of post-processing antialiasing filters (columns) and how the core elements of SMAA handle them (rows).

4.4.1 Edge detection

Edge detection is critical in all AA filters, since each undetected edge will remain aliased on the final image. On the other hand, too many blurred edges can reduce the quality of the antialiased image, while imposing unnecessary performance penalties. Different information can be employed for edge detection: RGB color, luma, depth, surface normal, object ID... or combinations of them. We choose to use luma based on four observations: first, MLAA expects edges to come specifically from color-based (either luma or RGB) discontinuities; otherwise artifacts may appear [106] (see section *MLAA on the PS3*). Second, as opposed to depth and normals, color information is *always* available. Third, it can handle shading aliasing. And fourth, it is faster than RGB color while usually yielding similar results. For efficiency, we only search for edges at the top and left boundaries of each pixel, since the bot-

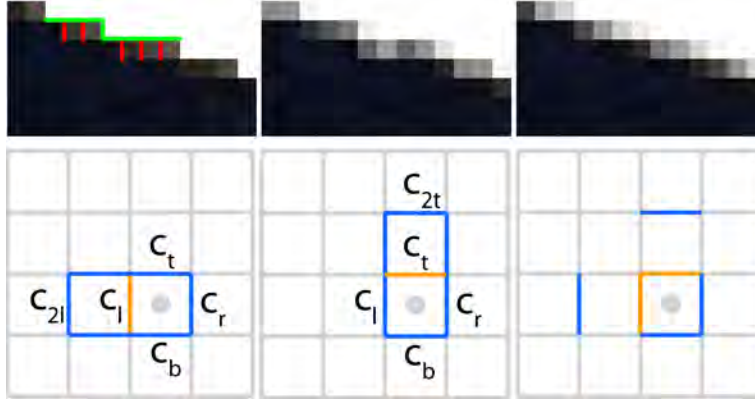


Figure 4.5: Top: Dominant contrast in green edges should mask the spurious red crossing edges (left). Not taking this local contrast into account leads to artifacts (center). Our SMAA algorithm corrects them (right). Bottom-left: left boundary (orange) of a given pixel (marked with a dot) and surrounding candidate edges (blue) that may dominate it, making it non-visible for human viewers. Bottom-middle: top boundary scenario. Bottom-right: candidate surrounding edges actually calculated.

tom and right ones can be retrieved from the neighbors.

Local contrast adaptation: The human visual system tends to mask low contrast edges in the presence of much higher contrasts in the surrounding area. Thus, a naive color edge detection based exclusively on local numerical differences will produce spurious edges (usually undetected by humans) that will affect pattern classification, downgrading image quality and temporal stability (see Figure 4.5, top). To avoid these spurious edges, we perform an adaptive double threshold which allows to: a) prevent line searches from stopping at non-perceptually-visible crossing edges; and b) choose the dominant (much higher contrast) edge when there are two parallel edges on a pixel (top-bottom, or left-right). This differs from previous approaches that take into account local contrast by simply checking the range of lumas found in the current pixel and its 4-neighborhood, and thus do not allow the notion of perceptual masking between edges [138].

Figure 4.5, bottom-left, shows the case for left edge (orange) of a given pixel (grey dot), plus the surrounding candidate edges (blue) that may *dominate* (mask) it. We calculate the maximum contrast c_{max} for all these edges and compare it with the contrast for the left edge. If the latter is above a threshold of $0.5 \cdot c_{max}$ the edge is preserved; otherwise, it is ignored. The threshold was chosen empirically and provides good results in all our tests. The bottom-middle image shows the similar case for the top edge. Since computing all these edges involves too many memory accesses, we select a subset that yields satisfactory results (bottom-right).

For the case of the left boundary, a straightforward algorithm would calculate $e_l = |L - L_l| > T$, where e_l is the boolean value that codes whether the edge is active, L and L_l represent luma values at the current and left pixels respectively, and T is a given threshold (usually between 0.05 and 0.2). We refine this naive approach with an additional test that can be expressed as:

$$\begin{aligned} c_{max} &= \max(c_t, c_r, c_b, c_l, c_{2l}) \\ e'_l &= e_l \wedge c_l > 0.5 \cdot c_{max} \end{aligned} \quad (4.1)$$

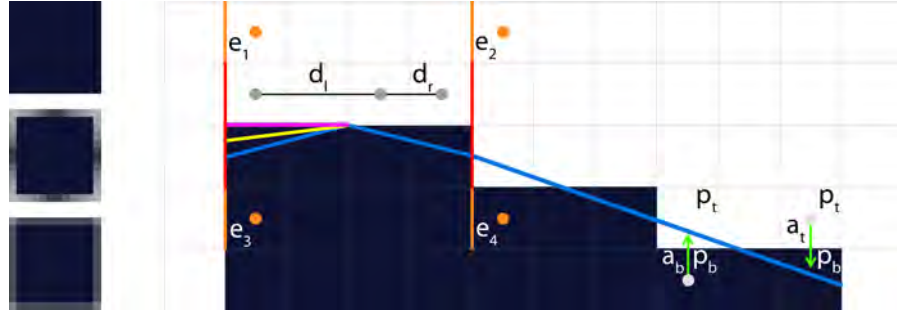


Figure 4.6: Left: Comparison between no antialiasing (top), a regular MLAA approach (middle), and the SMAA results (bottom). Notice how SMAA keeps the original shape of the object much better, while MLAA tends to round its shape. Right: Corners have crossing edges of length at least two (see the second pixel column), while aliased contour lines have crossing edges of just one pixel in length (stair-case towards the right). Fetching extended crossing edges (orange), in addition to regular edges (red), allows to discern between both cases, yielding a more accurate re-vectorization (pink), instead of rounding off corners (blue).

where $c_t, c_r, c_b, c_l, c_{2l}$ are the contrast deltas for the edges shown in Figure 4.5, and e'_l represents the final boolean value (active or not) for the left edge boundary. The edge at the top boundary, e'_t , is calculated in a similar fashion.

4.4.2 Pattern handling

Our new pattern detection allows to preserve sharp geometric features like corners, deals with diagonals and enables accurate distance searches.

Sharp geometric features: The re-vectorization of silhouettes of MLAA tends to round corners on the image (see Figure 4.6, left). Given that the crossing edges used for pattern detection are just one pixel long, it is not possible to distinguish a jagged edge from the actual corner of an object, which may be wrongly processed.

To avoid this, we make the key observation that crossing edges in contour lines have a maximum size of one pixel, whereas for sharp corners this length will most likely be longer. We thus fetch two-pixel-long crossing edges instead; this allows to detect actual corners and apply a less aggressive processing, thus retaining more closely the true shape of the object (see Figure 4.6, right). The degree of processing applied is defined by a rounding factor r , which scales the original coverage areas obtained by one-pixel-long crossing edges (blue lines in Figure 4.6, right). The recommended range for r is $[0.0 - 1.0]$. For example, values of $r = 1.0, 0.5$ and 0.0 yield the blue, yellow and pink lines respectively.

For the (academic) case of an horizontal line, we modify Jimenez's MLAA coverage areas calculation as follows:

1. We perform the original pattern detection, using the regular crossing edges (red edges on Figure 4.6, right). This yields two areas a_b and a_t per pixel belonging to the pattern. a_b is used to blend the bottom pixel p_b with its top neighbor p_t , whilst a_t is used to blend p_t with p_b (see [108] for details).

2. We refine the areas a_t and a_b according to the following:

$$a'_t = \begin{cases} r \cdot a_t & \text{if } d_l < d_r \quad \wedge \quad e_1 \\ r \cdot a_t & \text{if } d_l \geq d_r \quad \wedge \quad e_2 \\ a_t & \text{otherwise} \end{cases} \quad (4.2)$$

$$a'_b = \begin{cases} r \cdot a_b & \text{if } d_l < d_r \quad \wedge \quad e_3 \\ r \cdot a_b & \text{if } d_l \geq d_r \quad \wedge \quad e_4 \\ a_b & \text{otherwise} \end{cases} \quad (4.3)$$

where a'_t and a'_b are the modified area values, d_l and d_r are the distances to the left and to the right of the line for the current pixel, and e_i are booleans that indicate if an edge is active (see Figure 4.6).

Diagonal patterns: Most of the existing filter-based techniques search for patterns made exclusively of horizontal and vertical edges (orthogonal patterns). This translates into badly aliased results (in space *and* time) for diagonal lines (see Figure 4.7).

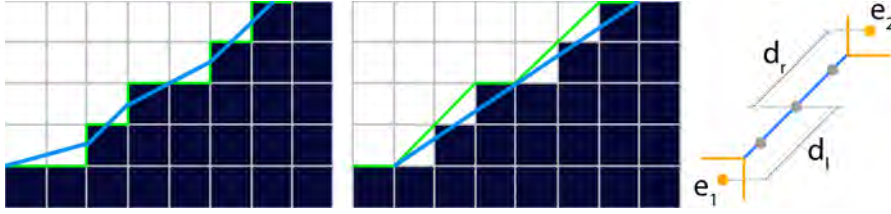


Figure 4.7: MLLA (left) and SMAA (center) re-vectorizations (blue lines) of near-45° diagonals. Thanks to our handling of diagonal patterns (green lines), SMAA reconstructs the edge accurately. Right: our approach just requires the same information as for the orthogonal case: distances d_l and d_r ; and crossing edges e_1 and e_2 (right).

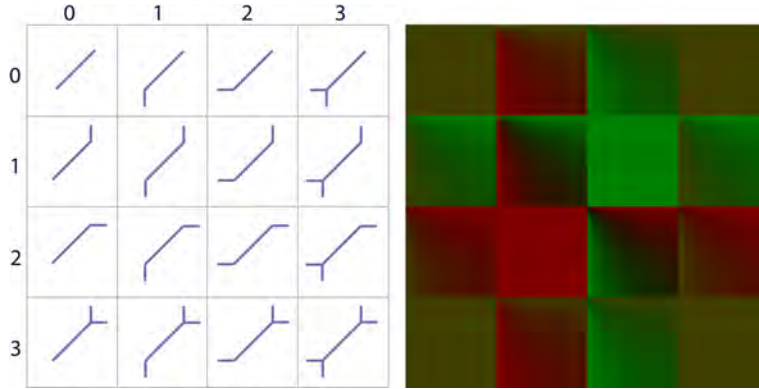


Figure 4.8: Diagonal patterns map (left) and their precomputed area texture (right).

We introduce a novel diagonal pattern detection that allows to detect these scenarios. In these cases, a diagonal re-vectorization (Figure 4.7, center) is used to yield coverage areas, instead of the original orthogonal re-vectorizations (Figure 4.7, left). The mechanism developed to handle diagonal patterns is inspired by the orthogonal patterns handling of Jimenez's MLLA. We introduce a precomputed texture that takes as input the diagonal pattern, defined by the distances to both ends of the diagonal line and

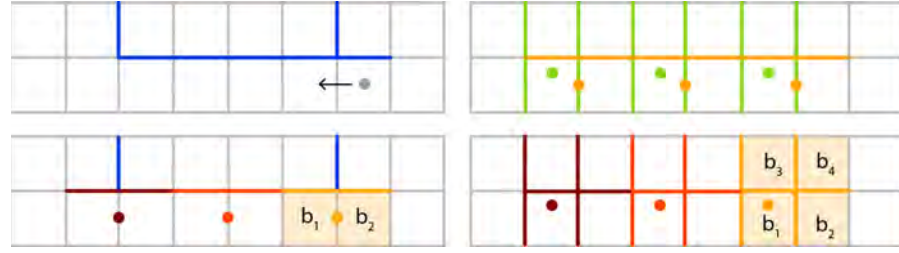


Figure 4.9: Top-left: Example of a search to find the left end of a horizontal edge (starting position marked with a dot). Top-right: Hardware filtered accesses performed by Jimenez's MLAA (orange dots) just check the orange edges. SMAA bilinear accesses (green dots) are able to additionally check all the crossing edges (green). In this example, this will make the search stop when the first crossing edge is found, instead of finishing at the left end of the horizontal edge. Bottom-left: The different iterations of the search (represented by different colors) and the values fetched by Jimenez's MLAA. Note how it misses all the crossing edges (in blue). Bottom-right: The same iterations and the values fetched by SMAA. It can be seen how SMAA is able to check four different pixels (shaded pixels) with just a single memory access.

the diagonal crossing edges information (Figure 4.7, right); and outputs the accurate coverage areas. Figure 4.8 shows the possible diagonal patterns and their corresponding pre-calculated areas.

Calculating diagonal coverage areas consists of the following steps, for both the top-left to bottom-right and the bottom-left to top-right diagonal cases:

1. We search for the diagonal distances d_l and d_r to the left and to right end of the diagonal lines.
2. We fetch the crossing edges e_1 and e_2 .
3. We use this input information (d_l , d_r , e_1 , e_2), defining the specific diagonal pattern, to access the precomputed area texture, yielding the areas a_t and a_b .

We perform this diagonal pattern detection before the orthogonal one in the coverage area calculation. If the diagonal pattern detection fails, we trigger the orthogonal detection. Otherwise, the areas produced by the diagonal pattern detection are used. This model allows to seamlessly perform the last blending step (step f in Figure 4.3) in a symmetric way for both orthogonal and diagonal patterns, given the fact that the semantics of the produced areas a_t and a_b are the same in both cases.

Accurate distances search: Key to pattern detection and classification is obtaining accurate edge distances (lengths to both ends of the line). Jimenez's MLAA makes extensive use of hardware interpolation (bilinear filtering) to accelerate this process. Hardware bilinear filtering can be used as a way of fetching and encoding up to four different values with a single memory access (otherwise it would be necessary to perform one memory access per value to fetch). This is exploited to fetch two edges at once, allowing to partially reduce bandwidth usage (see Figure 4.9, bottom-left). However, it does not check crossing edges during the search, which may lead to inaccuracies in pattern detection [108].

Unfortunately, fetching the crossing edges in the search loop following their scheme would imply two linearly filtered accesses per iteration, doubling the bandwidth usage. We generalize the approach for *two dimensional*

accesses, being able to fetch four different values with a single memory access (Figure 4.9, bottom-right).

Jimenez's MLAA uses a linear interpolation of two binary values producing a single floating point value:

$$f_x(b_1, b_2, x) = x \cdot b_1 + (1 - x) \cdot b_2, \quad (4.4)$$

where b_1 and b_2 are two binary values (either 0 or 1, given that the edges texture marks each edge as activated or not), and x is the interpolation value. If $x \neq 0.5$, this produces a set of four unique values: $\{0, 1 - x, x, 1\}$. So, it is possible to find a decoding function f^{-1} that recovers the original b_1 and b_2 binary values. Instead SMAA performs bilinear interpolation of four binary values as follows:

$$f_{xy}(b, x, y) = f_x(b_1, b_2, x) \cdot y + f_x(b_3, b_4, x) \cdot (1 - y), \quad (4.5)$$

where y is the interpolation value in the second dimension. By choosing a value of $y = 0.5x$, it is possible to create a binary base that allows to encode a bilinear interpolation between four binary values into a single one, and still be able to recover the sixteen possible original values. We exploit this fact to fetch the four b_1 , b_2 , b_3 and b_4 binary edge values (see Figure 4.9, bottom-right). We refer the reader to the source code for the specific details of this feature.

4.4.3 Subpixel rendering

MLAA algorithms work with a single sample per pixel. This translates into subsampling, which makes it impossible to recover real subpixel features (see Figure 4.10, no AA and MLAA). Having more samples per pixel allows for a better reconstruction of the antialiased image. A naive extension would involve using MLAA in conjunction with MSAA, applying it over each subsample group separately and then averaging them together. However, using such a simple approach leads to blurry results (see Figure 4.10, MSAA 4x with MLAA). This is due to MLAA and MSAA making different assumptions about the coverage of the samples, so they cannot converge even increasing the samples per pixel count:

- MLAA is designed to work on the silhouettes of objects and not with thin lines and features, as found in distant objects with high-frequency details (see Figure 4.10, MLAA). As it can be seen, not taking into account sharp geometric features leads to blurry results (with unnatural glows). Thus, sharp geometric features detection (Subsection 4.4.2) is critical when applying MLAA over subpixel features. Ultimately, this allows for corners to be conservatively reconstructed, in order to allow multi/supersampling to reconstruct their real shape (see Figure 4.10, SMAA 1x; notice how non-silhouette features are ignored).
- Given that MLAA assumes the positions of all the samples to be at the center of the pixel for the re-vectorization, this simply does not work due to the under-/overestimation of the corresponding coverage areas, producing gradients that do not match in the resolve step, which translate into a blurry appearance of distant objects.

In addition to sharp features detection, our solution is to take into account the offset position of each subsample inside the pixel, in order to calculate properly their coverage areas. This way, when the different subsample

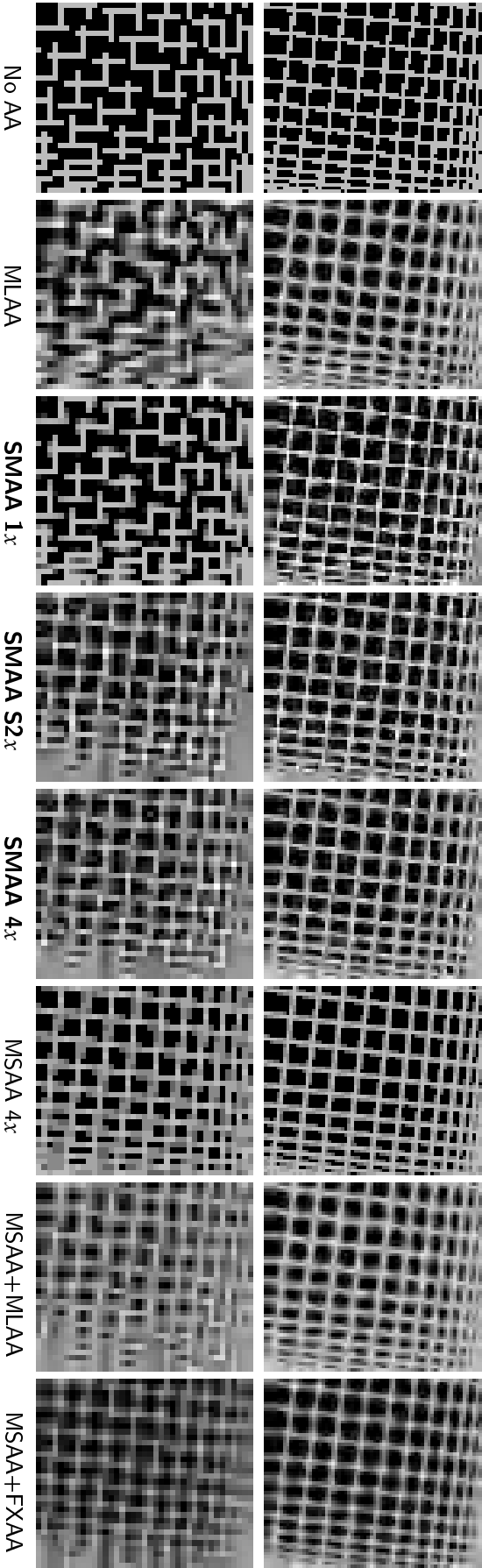


Figure 4.10: A difficult case for no AA, MLAA [108] and SMAA 1x: a white grid over a black background at mid-distance (top), prevents the reconstruction of accurate coverage; at a longer distance (bottom, zoomed in), the continuity of the grid is broken, preventing its recovery. Using extended patterns to deal with sharp geometric features and correct offsets allows for a more accurate area estimation, making SMAA S2.x and 4x converge to the MSAA 4x reference. Note how the naive application of MLAA over samples from MSAA 4x improves the connectivity of the grid, but blurring artifacts appear. We have also performed the same test using FXAA 3.11 (preset 39, max. quality) since it is one of the most used MLAA-like solutions. Figures 3, 4 and 5 in the supplementary material extend this with additional MSAA modes, and AA filters applied pre and post resolve.

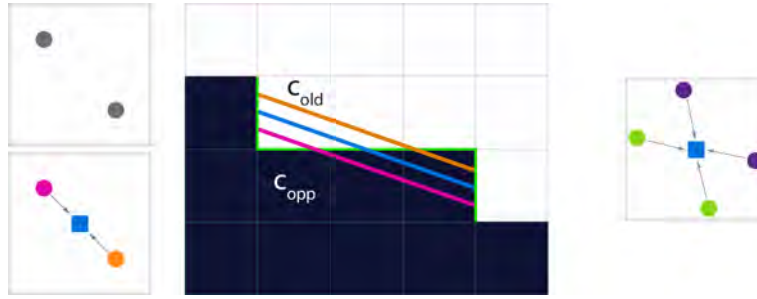


Figure 4.11: Left, top: usual MSAA 2x pattern, with offsets at $(-0.25, 0.25)$ and $(0.25, -0.25)$. Left, bottom: For combining multi/supersampling with MLAA (SMAA S2x and T2x), we have to offset the area calculations so that the average between the subsamples on top and bottom (pink and orange) corresponds to the color at the center of the pixel (blue). Middle: MLAA area calculations are devised to estimate the re-vectorization at the center of the pixel (blue). For SMAA S2x and T2x, these areas must be offset by -0.25 (pink) and $+0.25$ (orange). Right: Example of combining four subsamples (SMAA 4x) coming from both spatial multisampling and temporal supersampling, using two jittered results of SMAA S2x (purple and green).

groups are blended together, we obtain the average color at the center of the pixel (see Figure 4.11, left and middle). Then, the only required change to the pipeline is to use different precomputed areas textures for each subsample position. This approach is general enough to handle additional samples coming from standard approaches like temporal supersampling and spatial multisampling, so several configurations are possible. In particular, we have found the following modes to be the most interesting from a performance/quality perspective:

- SMAA 1x: includes accurate distance searches, local contrast adaptation, sharp geometric features and diagonal pattern detection.
- SMAA S2x: includes all SMAA 1x features plus spatial multisampling.
- SMAA T2x: includes all SMAA 1x features plus temporal supersampling.
- SMAA 4x: includes all SMAA 1x features plus spatial and temporal multi/supersampling.

The SMAA 4x mode requires to temporally jitter the SMAA S2x mode, as shown in Figure 4.11 (right). Figure 4.10 shows how SMAA 4x converges better to MSAA 4x than simply combining MLAA with MSAA.

4.4.4 Temporal reprojection

While temporal supersampling allows to efficiently render subpixel features, coupling it with a naive resolve approach like linear blending results in very noticeable residual artifacts, commonly referred to as *ghosting* (see Figure 4.12, left).

A better solution is to re-project instead the previous frames' subsamples into the current frame [160, 207][106] (Section *Anti-Aliasing Methods in CryENGINE 3*). However, disoccluded regions (occluded regions in the previous frame now visible in the current frame) still suffer from residual



Figure 4.12: Left: Using a naive resolve results in visible ghosting. Middle: Reprojection mitigates these artifacts but does not completely remove them. Right: The addition of velocity weighting allows to completely remove ghosting.

artifacts (see Figure 4.12, middle). To minimize them, we weight the previous subsample by w , which depends on the difference in velocity with respect to the current subsample:

$$w = 0.5 \cdot \max(0, 1 - K \cdot \sqrt{|\|v_c\| - \|v_p\||}), \quad (4.6)$$

where v_c and v_p are the velocity of current and previous frames, and K is a constant that determines how much we attenuate previous frame according to velocity differences (we use a value of 30 for all our examples). Then, the final resolve is performed as follows:

$$c = (1.0 - w) \cdot c_c + w \cdot c_p. \quad (4.7)$$

where c is the final resolved color, c_c the color in current frame, and c_p the color in the previous frame. Such a solution robustly handles disoccluded regions but at the expense of no antialiasing on such regions (see Figure 4.12, right). Nevertheless, the other components of our technique (either MLAA or spatial multisampling) will usually antialias these regions, effectively eliminating the problem.

A remaining problem of combining velocity weighting with a morphological strategy is that morphological antialiasing is actually blending pixels from both sides of the silhouette of an object at subpixel level. However, the velocity map remains aliased and so velocity is not propagated to the antialiased pixels, which leaves trails of blended pixels behind objects in motion. The solution is to apply SMAA also over the velocity buffer, in order to propagate velocities to the blended pixels. To efficiently perform this step, we coarsely store the velocity module in the alpha channel of the color buffer, so SMAA processes it for free.

4.5 RESULTS

Figure 4.13 shows a comparison of the subpixel modes of our technique against MLAA and SSAA 16x. Figures 1 and 2 from the supplementary material contain a more detailed comparison with a large number of selected antialiasing methods. We refer the reader to the supplementary material for additional examples both with still images and video. We recommend the

digital version of the article for proper examination. Performance metrics are measured on a NVIDIA GeForce GTX 470 using 1080p images. Typical execution times for our technique are of 1.02 ms for SMAA 1x, 1.32 ms for SMAA T2x, 2.04 ms for SMAA S2x and 2.34 ms for SMAA 4x. Subpixel modes allow higher thresholds for edge detection (see *better fallbacks* below), which lowers execution times without visible loss of image quality.

Local contrast: The first column of Figure 4.13 shows how a conventional edge detection approach (MLAA) usually fails to properly detect patterns in the presence of gradients. Note how our approach is able to detect and correctly antialias these difficult zones for smooth gradients.

Diagonal pattern detection: Our algorithm accurately reconstructs a perfectly straight diagonal line for the streetlamp silhouette. Traditional post-processing approaches generate aliasing artifacts, which can be clearly seen when looking at the image at native pixel resolution and in motion.

Sharp geometric features: Our technique manages to preserve the sharp corners in the base of the aerials (specially the one of the satellite dish), whereas most filter-based antialiasing techniques introduce some degree of roundness. This information is vital for multi/supersampling to reconstruct the accurate shape of an object. Also text present on textures or the user interface is better preserved.

Accurate searches: Blindly following edges without checking crossing edges at each step causes pattern detection to fail in some scenarios, as shown in the MLAA image. Our accurate search allows to enhance the antialiasing quality without increasing the number of memory accesses.

Subpixel features: Post-processing techniques using 1x (final display resolution) color inputs are unable to reconstruct *accurate* subpixel features (which accounts for all selected techniques with the exception of SSAA), producing artifacts like spurious pixels, gaps in surfaces and distracting effects due to subsampling. In contrast, our SMAA T2x mode is able to better preserve the connectivity of the lines, resembling more faithfully the results obtained with SSAA 16x. SMAA S2x and 4x modes also provide real subpixel features at the expenses of multisampling, an approach with varying viability depending on the complexity of the shaders.

Better fallbacks: Subpixel SMAA modes not only allow actual handling of subpixel features, but also provide better fallbacks for additional robustness. If the morphological component of SMAA leaves any edge unprocessed, the MSAA component of S2x and 4x modes will back that up. If the temporal reprojection present in T2x and 4x modes fails due to changes in the occlusion of objects between frames, the morphological and MSAA components will reduce aliasing. And the possible shading aliasing of S2x will be made up by temporal SSAA and MLAA in SMAA 4x, making it the most robust mode.

Discussion: Most of the features we have described solve limitations of not just MLAA in particular, but of *all* post-processing antialiasing filters in general. Performance-wise, in a forward rendering scenario SMAA 4x and SMAA T2x are about 1.46x and 4.09x faster than MSAA 8x respectively (the first taking into account the required multisampling 2x overhead). With respect to memory consumption, the most demanding configuration requires only 43% and 17% of the memory used by MSAA 8x, in a forward and deferred context respectively. Note that we are able to perform better than MSAA 8x, while delivering superior overall quality, both in gradients and

shading, resembling more accurately the results of SSAA 16x. In the case of a deferred engine, using MSAA 8x would incur an excessive drop of performance given the massive bandwidth required [11], along with the requirement of supersampling the edges at 8x.

In SMAA 1x and T2x modes, the execution times are within the same 1 ms ballpark as other solutions (see Table 1 in the supplementary material). The S2x and 4x modes are obviously more expensive due to multisampling (an average of 1.57 ms overhead for rendering at 2x, minus the resolve time), but they are still on-par with other techniques that handle subpixel features (SRAA and MSAA 8x), still yielding smoother results. Note that SRAA requires additional 4x multisampled depth and/or normal maps and possibly two geometry passes; while our approach multisamples color information at 2x. In the case of a deferred renderer, our approach would require supersampling the edges; however, stencil-masked implementations allow for efficient performance.

The overhead introduced by each of our solutions is either negligible or very affordable. In particular, local contrast adaptation is 0.08 ms, the sharp geometric features detection and accurate distance searches take less than 0.01 ms; diagonals processing and temporal supersampling introduce a small overhead of 0.12 ms and 0.3 ms respectively. Spatial multisampling adds 1.02 ms for filtering the second sample, and an additional average of 1.57 ms to render the scene at 2x. The delta that SMAA 4x adds on top of a 2x forward-rendered scene is as little as 2.09 ms, making it an attractive option for any scenario that can afford such a small multisample count.

4.6 CONCLUSIONS

We have presented a technique that tackles all the weak points remaining in filter-based antialiasing solutions. We have shown for the first time how to combine a filter-based antialiasing technique with standard multi/supersampling approaches and temporal reprojection. This novel combination of improved MLAA strategies with spatial and temporal multi/supersampling accounts for a very robust solution, combining the different synergies for better fallbacks. SMAA 1x delivers very accurate gradients, temporal stability and robustness, while introducing minimal overhead, making it an obvious choice for low-end configurations. SMAA T2x, for little additional cost, offers a very attractive tradeoff for any kind of rendering engine (deferred or forward), avoiding 2x multisampling while still reconstructing subpixel detail. SMAA S2x and SMAA 4x are the best options regarding image quality. We believe that SMAA will finally enable deferred engines to match the antialiasing quality of forward rendering engines.

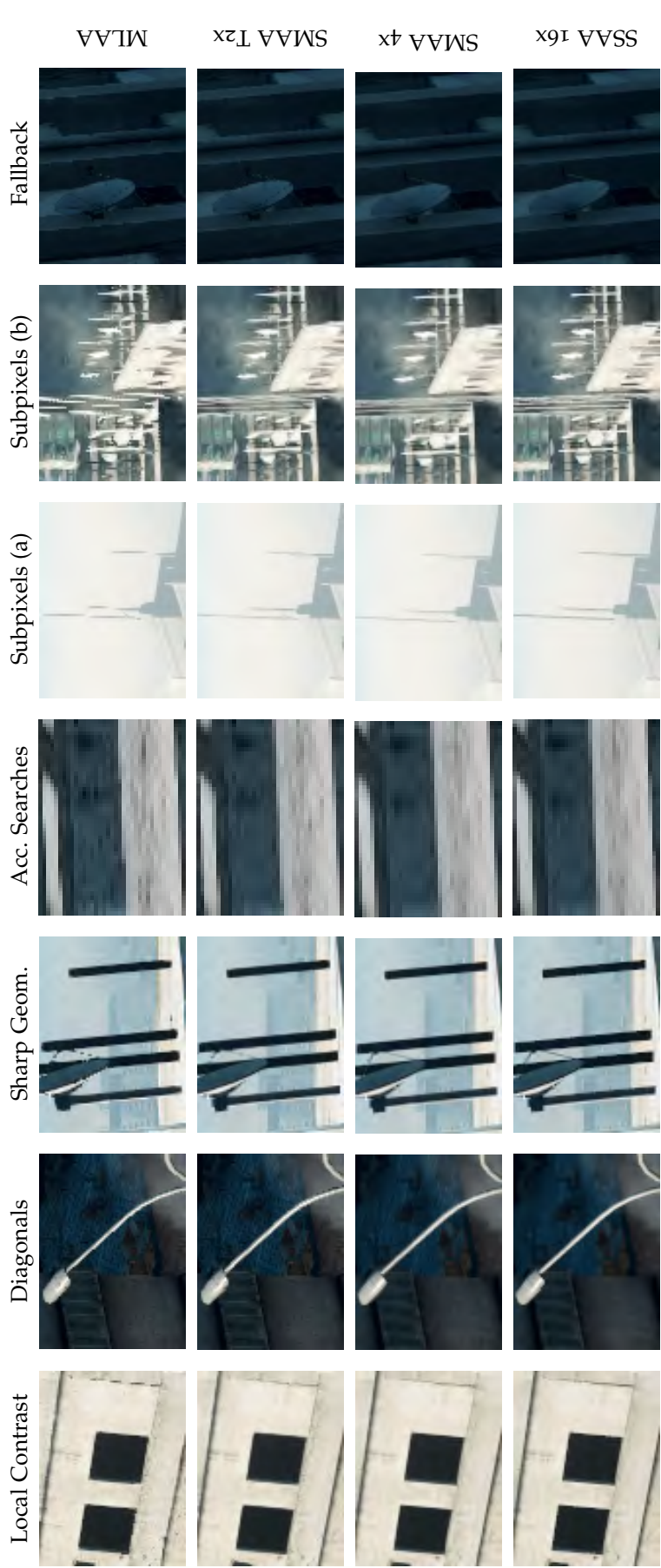


Figure 4.13: SMAA can produce results close to SSAA 16x, with SMAA T2x having a performance on par with the fastest MLAA implementation [108]. The improved edge/pattern detection allows to antialias difficult cases (first column). Diagonal pattern detection allows accurate reconstruction of such shapes (second column). The detection of sharp geometric features allows to better reconstruct corners and intersections (see second window in the first column, and bases of the aerials on the third column). Accurate searches allow to detect patterns in difficult scenarios (fourth column). Subpixel features handling allows to preserve connectivity and accurately represent distant objects (fifth and sixth columns). In zones with low-contrast edges, we fall back to MSAA 2x (seventh column), which provides good results and improves performance.

APPENDICES

4.A SUPPLEMENTARY MATERIAL

In the movie included, "*SMAA.mp4*", we show the behavior of the features of SMAA in motion. We compare against Jimenez's MLAA [108], FXAA 3.11 [138] (*preset 39* for the highest image quality), and CSAA 16xQ [211] (in order to have as reference the MSAA hardware implementation with the best image quality). We also show clips of SMAA integrated in the commercial game engine "*CryENGINE 3*" from *Crytek*.

File "*Battlefield3.psd*" contains the full resolution images used in Figure 4.13, 4.14, 4.15, 4.16 and 4.17. Each layer corresponds to a method for an easier and flexible comparison between them. We recommend to check them with software like Photoshop or GIMP, and swap between layers. Since MSAA was not available for this scene, we compare against different SSAA modes. Note that because Reshetov's criteria for edge detection and pattern handling can be enhanced [180], differences between his implementation and Jimenez's MLAA [108] can be appreciated (the same happens with AMD's implementation [8]). For example, while Reshetov utilizes RGB information for finding edges, Jimenez's MLAA uses luminance values. And while Reshetov takes into account the largest pattern a pixel belongs to, Jimenez's MLAA uses the shortest. However, Jimenez's threshold was tuned to be similar to Reshetov's. AMD's implementation cannot be customized, so it was applied with default settings. Additional screenshot comparisons can be found at the project website: <http://iryoku.com/smaa/>.

Figure 4.18 extends Figure 4.10 by comparing SMAA and MSAA modes with the same number of samples per pixel. It can be seen how SMAA gradually converges to the MSAA reference, in some cases surpassing the quality of MSAA modes with higher samples count.

Figure 4.19 demonstrates how offset positions must be taken into account for the area calculations when several samples per pixel are used. In this case, we test FXAA applied pre-resolve over each subsample coming from MSAA 4x in the same fashion we did with MLAA in Figure 4.10. As can be seen, not taking into account the exact position of each sample leads to blurry results that do not converge to the MSAA reference. Additionally, we test FXAA and MLAA applied post-resolve (Figure 4.20) over a resolved MSAA 2x input. The results show how suboptimal (even incorrect) it is to apply these antialiasing filters over an already antialiased input.

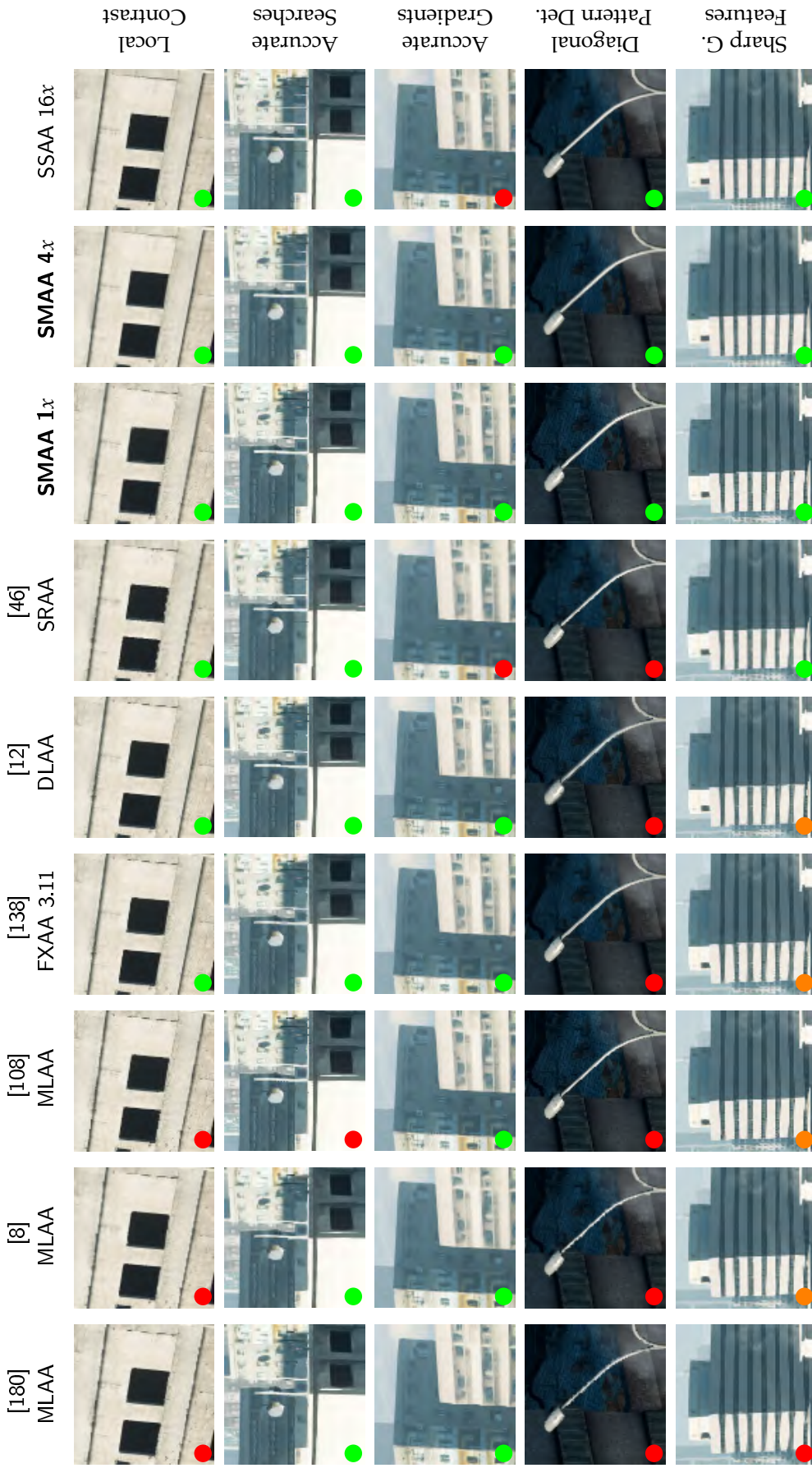
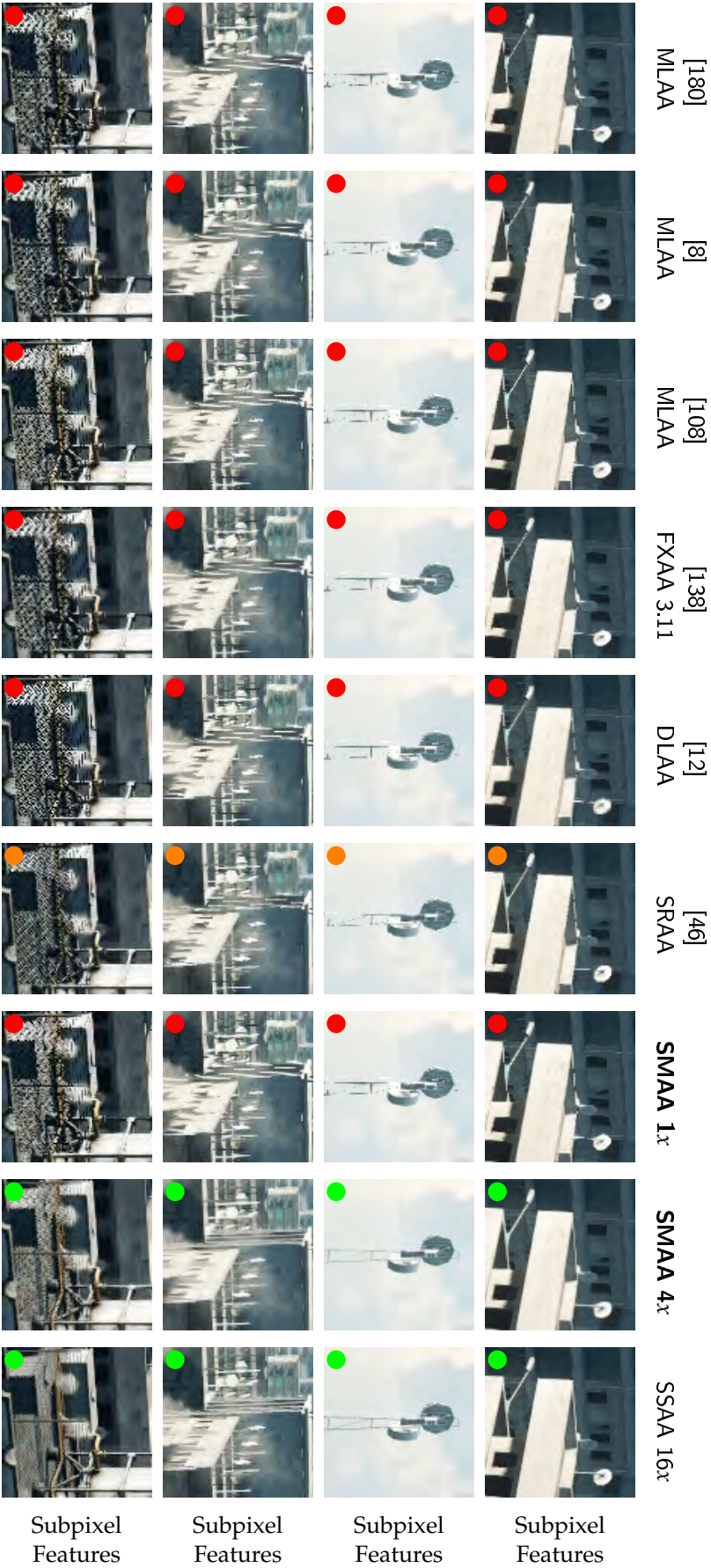


Figure 4.14: Comparison of the features (rows) of our approach with a selection of anti-aliasing techniques (columns). To help the reader navigate through this image matrix, we have color-coded the performance of each method for each particular feature tested, although this is ultimately a subjective criterion. Green, orange and red dots mark accurate, regular and inaccurate handling of a feature. Gradients from SSAA 16x are hampered in this case because of the use of an ordered grid SSAA. In the case of FXAA, we used preset 39 (maximum quality). The accompanying Photoshop file also includes FXAA's default preset 12, SMAA 1x High (as included in this Figure) and Ultra (with lower threshold for edge detection and larger patterns handling). Zoom into the digital version of this document or check the accompanying Photoshop files to see the details.



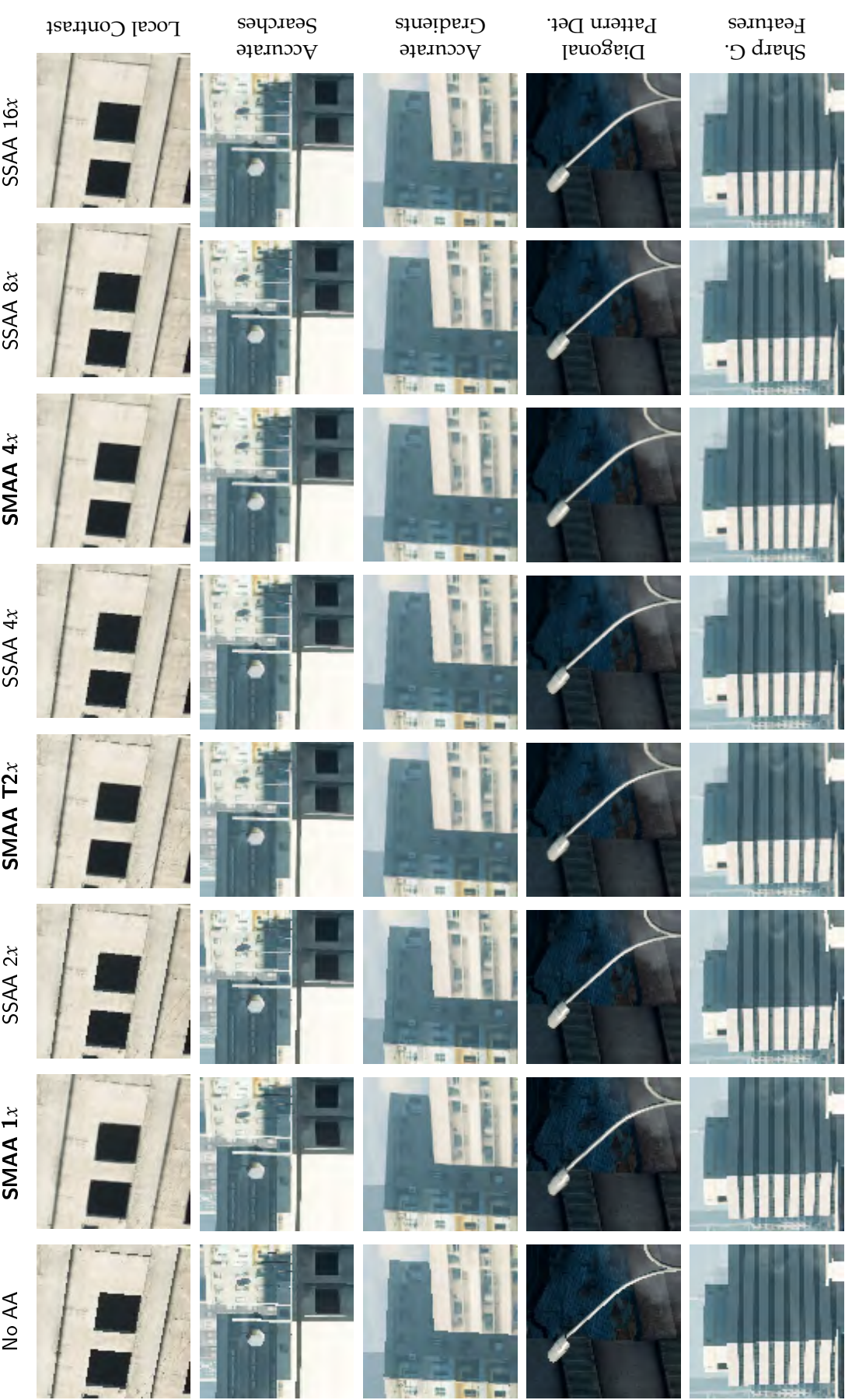


Figure 4.16: Comparison of SMAA modes against ordered-grid SSAA. Zoom into the digital version of this document or check the accompanying Photoshop files to see the details and comparison against all methods from Figure 4.14.

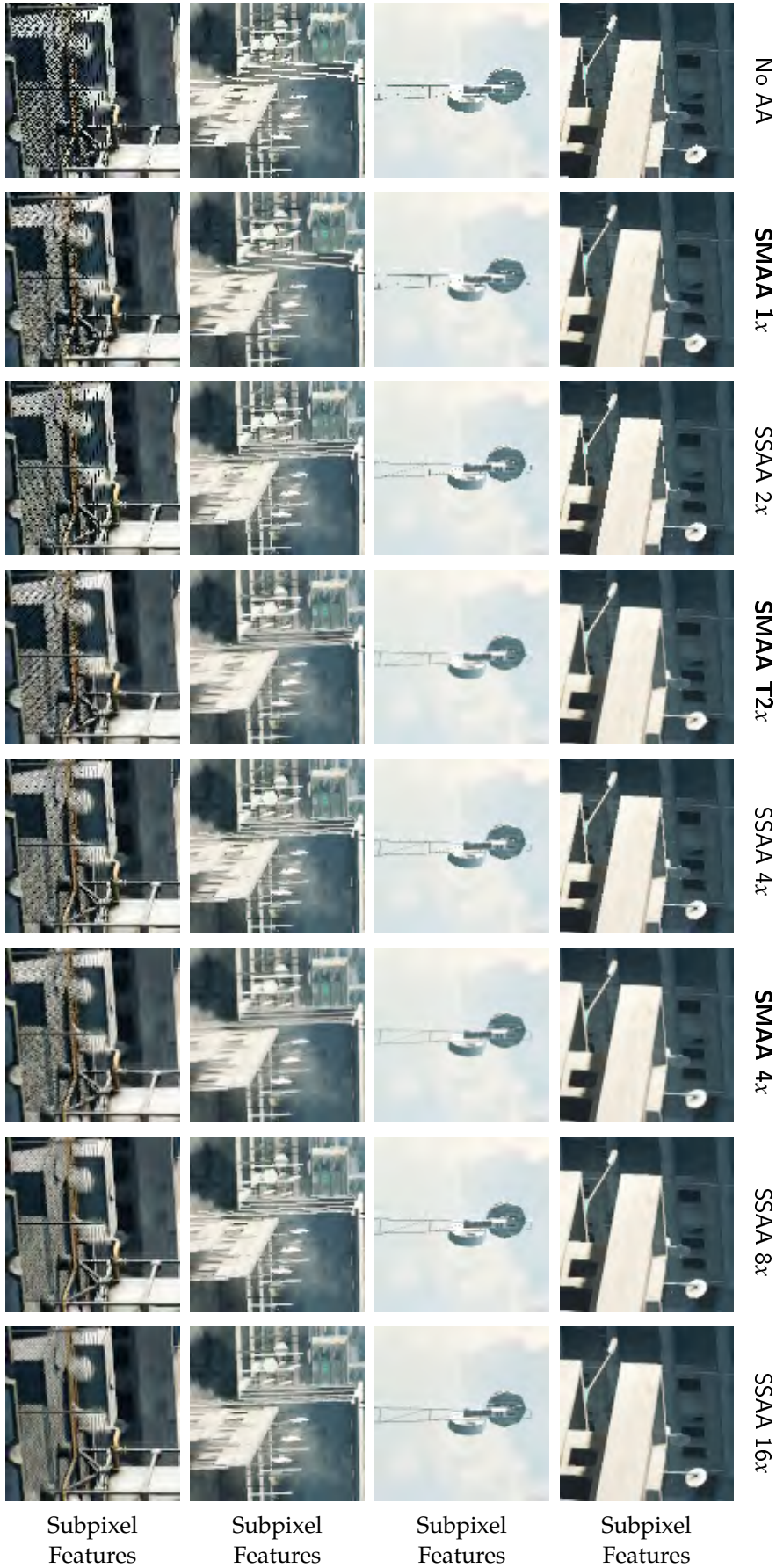


Figure 4.17: Comparison of SMAA modes against ordered-grid SSAA. Zoom into the digital version of this document or check the accompanying Photoshop files to see the details and comparison against all methods from Figure 4.14.

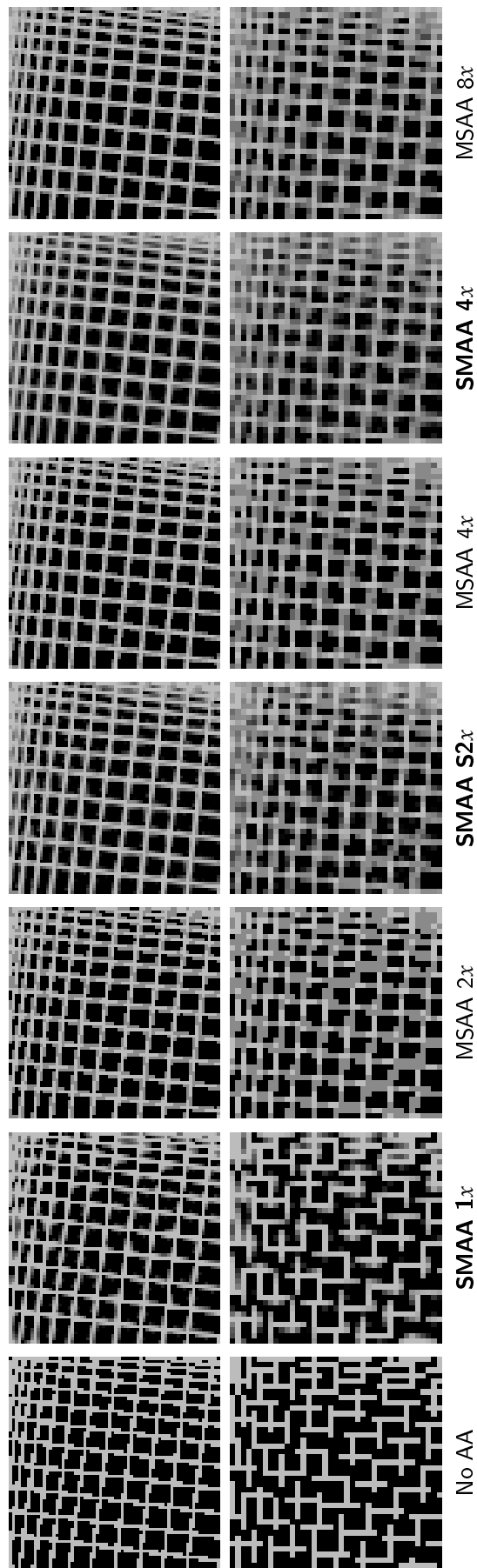


Figure 4.18: Comparison between SMAA and MSAA modes with the same number of samples per pixel in a very challenging scenario for subpixel features, with MSAA 8x included as the highest quality MSAA reference. As can be seen, using the same amount of information, SMAA provides smoother gradients than its corresponding MSAA counterparts, in some cases surpassing the quality of MSAA modes with a higher number of samples. Performance numbers can be found in Table 1.

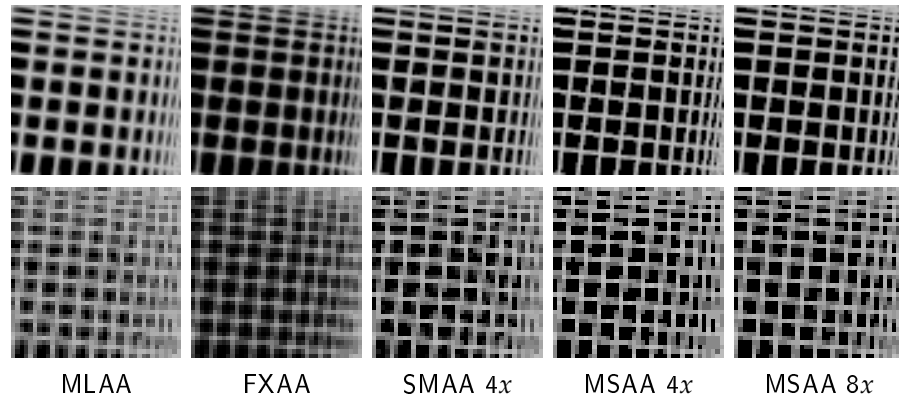


Figure 4.19: AA filters applied pre-resolve to each sample of a MSAA 4x input. Unlike SMAA, FXAA and MLAA do not take into account the offset position of the additional samples, thus leading to blurry results when compared against the MSAA 4x and 8x references. FXAA 3.11 preset 39 (max. quality) and Jimenez's MLAA were used in this test.

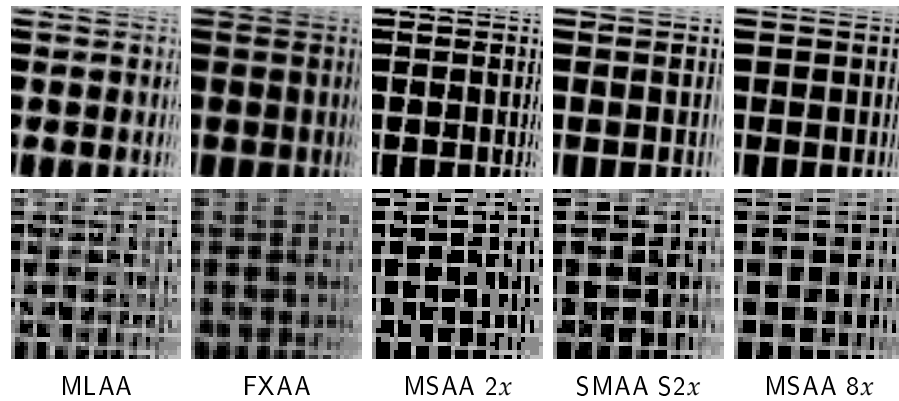


Figure 4.20: AA filters applied post-resolve over a resolved MSAA 2x input. It can be seen that when FXAA or MLAA are not fed with clean edges, they introduce artifacts that make the final image look not to converge to the MSAA reference. SMAA S2x and MSAA 8x included as a higher quality reference. FXAA 3.11 preset 39 (max. quality) and Jimenez's MLAA were used in this test.

This Chapter explores the Nokia N900 platform as a versatile playground for mobile computational photography. We present the implementation of a simple yet practical application to acquire high quality tone mapped HDR images, and a relatively straightforward white balance selection. This project was a way to test the flexibility and productivity of the platform, which was also the first commercially available realization of the *Fcam* architecture [4]. Back when we worked on this, HDR imaging in mobile devices was something uncommon, given the limited support from the operating systems and APIs. Nowadays, it is a standard feature in most of the camera apps, following the same core concepts and pipeline presented here.

This work was presented at the *Ibero-American Symposium in Computer Graphics (SIAGC) 2011*, and it was possible through a collaboration agreement with *Nokia Research Center*.

J. I. Echevarria & D. Gutierrez

MOBILE COMPUTATIONAL PHOTOGRAPHY: EXPOSURE FUSION ON THE N900
SIACG 2011

5.1 INTRODUCTION

Computational photography is an exciting and hot new field of research. In fact, forty percent of the 439 papers submitted to SIGGRAPH 2009 had to do with 2D imaging techniques applied to photography or video [131]. The overall goal of the field is to overcome the limitations of traditional image capture methods. By encoding the light reaching the sensor in smart ways, it is possible to later decode it and extract information that would have been lost otherwise. Thus, the information stored at the sensor is usually *not* the final image, but an encoded representation of the scene: a computational process needs to be applied afterwards, before the final image is ready for consumption. Examples of recent advances include extended dynamic range, deblurring out-of-focus images, digital refocusing or light field capture. These examples show how photography can be taken to the next level, and how consumer cameras could benefit from them, offering improved versions to the market.

A few years ago, the estimated number of digital cameras in the world broke the one *billion* mark. This is mainly due to the ubiquitous presence of mobile devices (mainly cell phones) with a built-in camera, which in turn has triggered an increasing number of photoblogs and online picture collections. Images can not only be taken anywhere, anytime and by anybody now; they can also be almost instantaneously shared, duplicated and manipulated. This poses the challenge of how to make computational techniques useful for this kind of mobile devices, which are hampered by several obvious limitations, both in terms of hardware (limited extensible lenses, reduced sensor size...) and software (less computational power).

One key limitation that was slowing progress down was the impossibility to access all the camera features, sometimes controlled by firmware and out of reach for the user (or programmer). It was only a few months ago that a team led by Stanford and Nokia Research [4] released an API to make

cameras fully programmable, allowing researchers to come up with new, customizable algorithms. The new open architecture is called the Frankencamera, and the authors offered implementations for two platforms: the F2 (built from off-the-shelf components), and the Nokia N900 smartphones.

In this Chapter, we focus on programming the complete high dynamic range imaging pipeline on the N900, from multi-bracketed capture of low dynamic range images, to final exposure fusion. Our implementation produces better results in general than the *HDR Capture* app provided by Nokia Beta Labs, reproducing more detail in over- and under-exposed areas. Finally, we leveraged the versatility of the platform to program a very simple white-balancing algorithm that cycles through different settings, in order to offer the user multiple depictions of the captured scene.

5.2 PREVIOUS WORK

Mobile computational photography is a quite recent research field, so little literature can be found yet. However, there are already good examples of some of its goals and applications.

The Frankencamera project [4] is the seminal work for the field. The goal was to provide the community a base hardware specification and an API for C++, to make cameras fully programmable. Its architecture permits control and synchronization of the sensor and the image processing pipeline. It also offers support for external hardware like lenses, flashes, accelerometers, etc. A more detailed overview of it can be found in Section 5.3. Along with the technical specifications, they also showcase the platform with applications like HDR viewfinding and capture, low-light viewfinding and capture, automated acquisition of extended-dynamic-range panoramas, foveal imaging, gyroscope-based hand-shake detection and rephotography.

During the gestation of Frankencamera, other interesting experiments were performed, mainly following the same approach as in this work: students were given the platform, and assigned a small task [131]. For instance, 4D Light fields capture and display was made possible by waving a cell phone around an object. The captured light field can then be displayed in the same or another phone, computing its spatial location and orientation based on a card with markers, and displaying the appropriate slice of the light field on the viewfinder. A virtual rearview mirror with no blind spots was also built by combining the video streams from five Nokia N95 phones mounted facing backwards on the roof of a car.

Real-time viewfinder alignment [3] is another example of algorithms that can be useful for a wide variety of mobile applications, from assistance in panorama capture, to low-light photography and camera-based controller input for games. Finally, focusing on external components apart from camera sensors, smart use of flash can provide favorable light conditions to take photos in and interesting illumination effects, as Adelsberger and colleagues demonstrated [5]. They presented a spatially adaptive photographic flash, in which the intensity of illumination is modified on a per-pixel basis depending on the depth and reflectivity of features in the scene. Their results show images that are better illuminated than the original scene, still looking natural and plausible.

LATER WORK After the publication of this work, several new methods for HDR imaging have been developed. Bracketed sequences of images are prone to artifacts due to alignment and ghosting [189], which can be ame-

liorated by using optical flow [223], patch-based reconstruction [182], or taking into account the noise distribution of color values [75]. Single-shot approaches have also appeared, all of them requiring specialized hardware with varying degrees of complexity [195, 47, 85, 217]. More recently, Serrano et al. [183] proposed the use of affordable masked sensors for convolutional sparse coding to capture HDR images and videos.

However, while providing higher quality reconstructions, from a practical point of view they are all still far from meeting the requirements of current mobile devices (commodity hardware, low computing power and processing times), which nowadays feature HDR capabilities based on the similar pipelines to the one presented here.

5.3 THE FRANKENCAMERA ARCHITECTURE

The Frankencamera [4] is both an open source architecture and an API for cameras. It includes a base hardware specification, a Linux-based operating system and a C++ API. It is intended to be flexible enough to implement most of the techniques proposed in the computational photography literature and also explore new ones. So, it presents a comprehensive list of features: the factor form and usability of consumer level cameras, touch-screen for designing new user interfaces, easy to program for with standard libraries, low-level access to principal components affecting the capture process on a per-frame basis, access to raw sensor pixels, acceptable processing power and compatibility and upgradability with standard and experimental camera accessories, to name a few.

The abstract architecture encompasses the image sensor, the fixed-function imaging pipeline that obtains and produces image data and optional photographic devices such as lens, flash, etc. Figure 5.1 shows a diagram explaining how these elements interact. The main characteristic of the image sensor is its stateless behavior, meaning that capture parameters must be set on a per-frame basis. The imaging processor has two main roles: to generate useful statistics like histograms and sharpness maps that are attached to each generated frame, and to perform basic operations as demosaicking, white-balancing, gamma correction, etc. Additional devices can be connected to the platform in order to expand the information obtained by the sensor [4].

Two platforms were constructed following the enumerated principles: the F2 camera, built from different off-the-shelf components; and the Nokia N900 smartphone. In this work we focus on our available version, which is the N900 smartphone.

5.3.1 The Nokia N900

The Nokia N900 is a smartphone provided with an ARM CPU running at 600 MHz, 256 MB of RAM plus 768 MB of virtual memory and a 5 MP camera with Carl Zeiss Optics, all of them governed by a *Maemo* Linux-based operating system, which makes the platform instantly compatible with tons of standard libraries for Unix development. Figure 5.2 shows some views of the unit.

To start developing for the N900, Nokia provides their Qt Creator SDK, an IDE to easily connect with the phone, which integrates the compiler and libraries needed to quickly start developing programs for the platform. Some additional setup needs to be performed on the device (firmware updates, drivers...), before it is finally ready to make the most of the unit through the

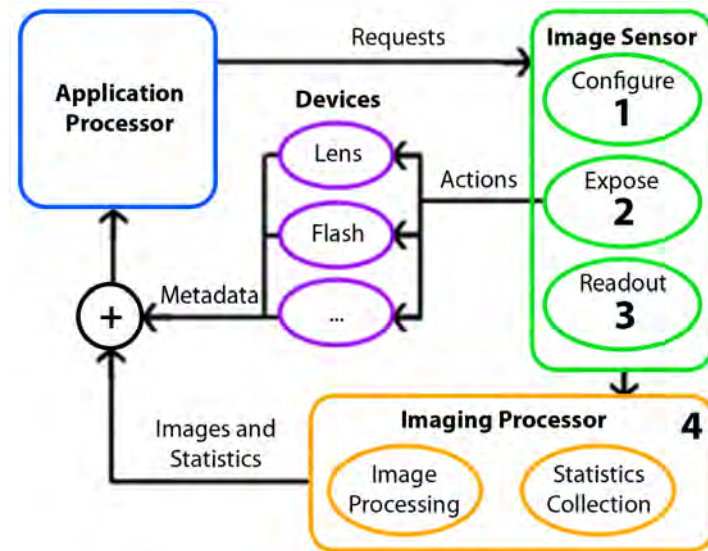


Figure 5.1: Overview of the Frankencamera architecture (adapted from [4]). The image sensor produces captures according to the requested parameters by the application. Optionally, external devices can be brought in during the exposure, altering the capture and providing useful additional info. Captures are then transformed into images by the imaging processor, which will be finally used by the application processor for further tasks.



Figure 5.2: Front and rear views of the Nokia N900 smartphone.

FCam API. This API provides low level access to most of the parameters of N900's sensor, lens and flash. With the FCam installed, the N900 turns into an affordable playground for computational photography.

5.4 HDR ACQUISITION

Our main goal is to study the Nokia N900 smartphone along with the FCam API as a platform for mobile computational photography. For this project, we have focused on the full pipeline of automatic acquisition of high-dynamic range (HDR) images, including multi-bracketing, exposure fusion and tone mapping. The objective is to have the N900 performing *all* the calculations without any user intervention, before presenting the final image. Note that a similar example application is presented in the original Frankencamera paper [4]. However, as we will see, we propose two different scene-independent mechanisms to choose the three input exposures used to

create the HDR image, which in general yield better results than Nokia’s own implementation.

Given the low dynamic range sensor of the N900, we start from a multi-exposure bracketed sequence of low dynamic range images. The usual next step would be to create an intermediate physically-based HDR image [54] which would be finally tone mapped [57, 179, 134]. However, keeping in mind our goals and the N900 specifications, we opt for using a more computational affordable but still high-quality solution: *Exposure Fusion* [150].

The original exposure fusion technique fuses a bracketed exposure sequence of low dynamic range images directly into a high quality low range tone mapped image. By skipping the intermediate HDR assembly it simplifies the acquisition process and camera calibration, still producing results comparable to widely used tone mapping operators. First, it measures the quality of each pixel from each image of the sequence according to their contrast, saturation and *well-exposedness*. Then it makes a weighted sum of all the images, using the corresponding weight maps, for obtaining the final result. In order to avoid artifacts due to this blending, each color image is transformed into a Laplacian Pyramid, and each weight map is transformed into a Gaussian Pyramid. The weighted sum is performed at each level of the pyramids, fusing them into a single one, which is finally collapsed to obtain the final full resolution image. In order to maximize the ratio between quality and required computational power, we choose to implement a simplified version of the algorithm, leveraging only well-exposedness information and simply ignoring contrast and saturation. Again, we have found that this not only streamlines the process, but it tends to yield better results than the implementation by Nokia Labs.

5.5 IMPLEMENTATION

First of all, we need to obtain the bracketed sequence of images with appropriate exposure values. In the original Frankencamera implementation, the authors propose to adjust the exposure times based on an analysis of the scene content, following the work by Kang et al [112].

In this work we aim to make this process scene-independent, while providing good results in most cases, so we follow a different route. We rely on the fact that on the N900, we can set the exposure time to any length we want (in microseconds) inside the supported range of the device. However, in digital photography it is more standard to work with EV (Exposure Value) stops [99], which depend on exposure time and aperture size. Given that the N900 aperture is fixed at $F/2.8$, we can obtain the range of N900 EVs by setting up exposure time values as in Table 5.1.

EV	4	6	8	10	12
Exposure time (s)	1/2	1/8	1/30	1/125	1/500

Table 5.1: Exposure times for the selected EV values that typically maximize the N900’s sensor range, given its $F/2.8$ aperture. N900’s minimum EV is 2 and maximum is 16.

For capturing input images with better quality, we have also added auto focus and white balance to the acquisition process. Source code for these tasks can be found in the documentation examples of the FCam API.

To create the final HDR image from the bracketed sequence, we use *Exposure Fusion* [150], which can blend together an arbitrary number of images. However, one important limitation of general HDR acquisition techniques is the fact that both the camera and the scene need to be static, to avoid ghosting artifacts. This suggests that finding the *minimum* number of necessary low-dynamic images is desirable, to reduce the total acquisition time. In our experiments, we found that nice results can be obtained with just three images (in accordance to the proposed method described in [4]). Instead of letting an analysis of scene content fix the three exposure times, we capture images with the five EVs that usually maximize the valid range of the sensor (EVs 4, 6, 8, 10 and 12). We then select the three resulting images containing the largest total number of useful pixel values, avoiding the two images with more over- or under-exposed pixel values (note that these two are not necessarily the lowest and highest EVs respectively; the discarded images will be a function of the scene original lighting level).

As an alternative, we have implemented another fully automated option, which makes use of the built-in metering functions inside the FCam API. Along with the auto focus and white balance, we additionally use the *auto exposure* to let the camera deduce what would the optimal exposure be, if just a single LDR image were to be taken. Then, we move two EV stops back and forth from that value (two steps proven to be the better choice in our tests). This way we have to capture just three images, lowering capture times and making HDR imaging obtention even more straightforward.

As outlined before, the original work proposes three different quality measures for each pixel: contrast, saturation and *well-exposedness*. As the authors discuss in their paper, the most influential measure seems to be the last one, while the other two add really subtle differences most of the times. So, due to our limited processing power, we use only the *well-exposedness* measure, which accounts for pixels that are not under nor overexposed, based on how close to 0.5 is the intensity of the pixel using a Gauss curve:

$$W_{ij,k} = \prod_c \exp \left(-\frac{(I_{ijc,k} - 0.5)^2}{2\sigma^2} \right) \quad (5.1)$$

where $I_{ijc,k}$ corresponds to the intensity of the pixel ij for each channel c , and σ equals to 0.2. This weights are calculated for each image k , and then normalized so $\sum_k \hat{W}_{ij,k} = 1$, with $\hat{W}_{ij,k}$ being the normalized weights.

At this point, if we make a naive weighted sum of all the images, artifacts will appear due to the different absolute intensities. So, before the blending, a Laplacian pyramid is built for the color images, $L\{I\}$, and a Gaussian one for the weight maps, $G\{\hat{W}\}$. Now, the blending can be performed at each level l , for each channel c :

$$L\{R\}_{ijc}^l = \sum_k G\{\hat{W}\}_{ijc,k}^l L\{I\}_{ijc,k}^l \quad (5.2)$$

The multi resolution approach provided by the Laplacian pyramid helps smoothing sharp discontinuities produced by Equation 5.2 during the collapse of $L\{R\}$ in order to obtain the final full resolution image. For building and collapsing the pyramids [37], we use the separable convolution kernel $f = [0.0625, 0.25, 0.375, 0.25, 0.0625]$ [150].



Figure 5.3: Two examples of photos captured and processed by our method. The insets show the bracketed sequence input. The complete pipeline takes less than 20s for a resolution of 640×480 . Left: auto exposure. Right: manual selection of three EVs out of five.

5.6 RESULTS AND DISCUSSION

All of our examples have been obtained by blending a bracketed sequence of three images with different EVs. In order to make computing times acceptable for taking photos on the go, we capture the images at 640×480 resolution. The three images are taken in 2s (5s for the sequence of five images), weights calculation (Equation 5.1) take 5s, and pyramids are built, mixed (Equation 5.2) and collapsed in 11s. Our pyramids have 8 levels, the maximum number of levels for the chosen resolution. As they have the biggest impact on the processing time of our application, we have tested less levels for sparing calculations. Unfortunately the results always show artifacts like halos and color shifts.

Figure 5.3 shows two challenging examples (as the low dynamic range insets show) where our application produces natural looking results. Figure 5.4 shows a comparison between two captures with our application (fully automatic version) and the results obtained via Nokia Beta Labs *HDR Capture* application [4], with automatic exposure enabled and forced to use three photos as input. We can see how, although built with the same tools, the flexibility this FCam API provides can make results vary significantly depending on its use.

Limitations of our application come in form of slightly loss of contrast and saturation due to the exclusive use of the *well-exposedness* quality measure, as explained in Section 5.5. However, we could directly introduce them, with a higher execution time for the complete process.

Just right now, there are already other commercial platforms on the market that are capable of simulate this kind of image processing [14, 73]. The difference is that those platforms just provide limited high-level access to some of the camera functions, thus making the exploration of other techniques difficult. In contrast, the N900 in conjunction with the FCam API offers direct low-level access to parameters that provide the flexibility and power needed for real mobile computational photography research.

There are virtually infinite applications that can be devised and implemented on the N900. Another simple example application is shown in Figure 5.5, where the camera cycles through seven white balance settings and

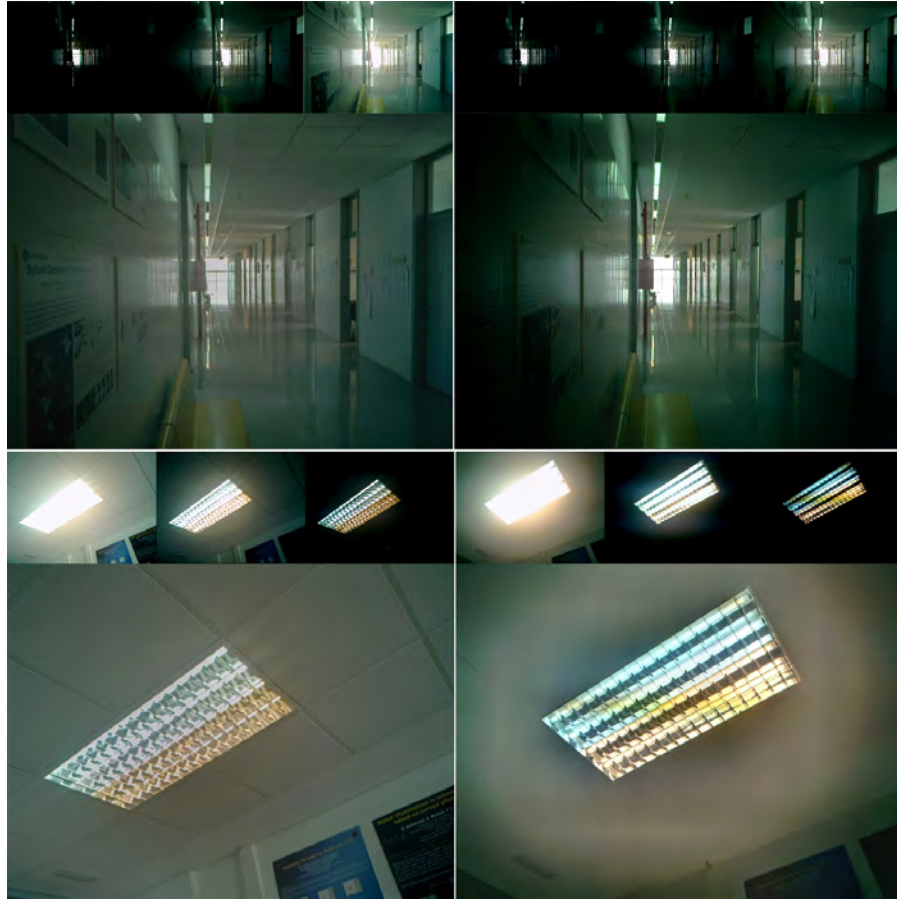


Figure 5.4: Results obtained with the N900 using our application (left) and HDR Capture app (right) by Nokia Beta Labs. Insets show the images used to build the final one. Examples taken with auto exposure in both applications. We can see how our application produces a more detailed and natural look.



Figure 5.5: From left to right, up to down: sequence of images obtained by capturing the same scene with white balance ranging from 2800K to 11000K. Bottom right shows the selection image presented to the user in order to choose the preferred color temperature.

offers the user a combined mosaic of the results. The user simply chooses the one that best depicts the captured scene, and the system keeps that one and (optionally) deletes the rest. In this case, the exposure time has been set at $50ms$, with a fixed gain of 1.0. The white balance range goes from 2800K to 11000K.

5.7 CONCLUSIONS AND FUTURE WORK

We have demonstrated how the Nokia N900, via the FCam API, can be used as a powerful and flexible platform for mobile computational photography. We have shown an efficient implementation of the full HDR pipeline, from the automatic acquisition of a bracketed sequence of images to the final fusion of the multiple exposures. We have obtained good results by just modifying exposure times of the sensor and adding an efficient algorithm for image processing. These results most of the times surpass Nokia's own implementation. Future work will include exploration of concepts and techniques that make use of more components of the mobile phone like lenses, flash, etc.

Part IV

3D RECONSTRUCTION AND STYLIZATION

In this last part we focus on obtaining 3D reconstructions in two different contexts. First, we introduce a new fast and flexible depth from defocus pipeline to obtain the depth map of a scene from the camera point of view. We explain how to use computational photography to control the capture process to obtain detailed and accurate depth maps efficiently. Next, we introduce a novel capture system to obtain stylized 3D reconstructions of hairstyles for 3D fabrication. We describe coherent color and geometry stylization in a multi-view environment, allowing varying levels of abstraction that still retain the main features that make each unique hairstyle recognizable.

In this Chapter we present a new depth from defocus method. It is related with the previous one in the sense that we use computational photography to have full control over the capture process of the scene. This allows us to make the assumption that a per pixel blur estimate (related with the circle of confusion), while ambiguous for a single image, can behave in a consistent way when applied over a focal stack of two or more images. Such assumption allows us to fit a simple analytical description of the circle of confusion to the different per pixel measures to obtain approximate depth values up to a scale. Our results are comparable to previous work while offering a faster and flexible pipeline.

This work was published in *The Visual Computer*, as the result of a 2-months research visit of Stephen Bailey from Vanderbilt University (Tennessee, USA), whom I supervised until its publication.

S. W. Bailey, J. I. Echevarria, B. Bodenheimer & D. Gutierrez
FAST DEPTH FROM DEFOCUS FROM FOCAL STACKS
The Visual Computer, Vol.31 (12) 2014

6.1 INTRODUCTION

Among single view depth cues, focus blur is one of the strongest, allowing a human observer to instantly understand the order in which objects are arranged along the z axis in a scene. Such cues have been extensively studied to estimate depth from single viewpoint monocular systems [64]. The acquisition system is simple: from a fixed point of view several images are taken, changing the focal distance consecutively for each shot. This set of images is usually called a focal stack, and depending on the number of images in it, different approaches to estimate depth can be taken. When the number of images is high, a *shape from focus* [192] approach aims to detect the focal distance with maximal sharpness for each pixel, obtaining a robust first estimate that can be further refined.

With a small number of images in the focal stack (as low as two), that approach is not feasible. *Shape from defocus* [202] techniques use the information contained in the blurred pixels based on the idea of the circle of confusion, which relates the focal position of the lens and the distance from a point to the camera with the resulting size of the out-of-focus blur circle in an image.

Estimating the degree of blur for a pixel in a single image is difficult and prone to ambiguities. However, we propose the hypothesis that those ambiguities are possible to disambiguate by applying and analyzing the evolution of the blur estimates for each single pixel through the whole focal stack. This process allows us to fit an analytical description of the circle of confusion to the different estimates, obtaining actual depth values up to a scale for each pixel. Our results demonstrate that this hypothesis holds, providing reconstructions comparable to those found in previous work, and making the following contributions:

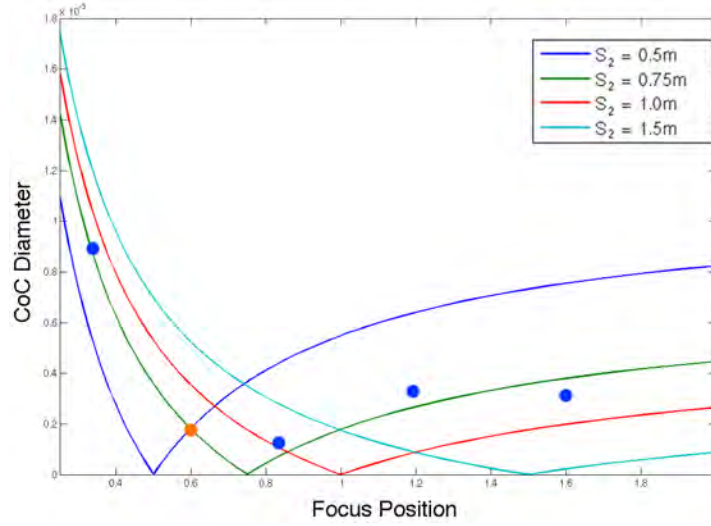


Figure 6.1: Circle of confusion (CoC) diameter vs. focus position of the lens for points located at different distances from the camera S_2 (axis units in meters). Plots show how points become focused (smaller CoC) as the focal distance gets closer to their actual positions. It can be seen how different combinations of focal and object distances produce intersecting CoC plots, so a CoC measure from a single shot (orange dot) is not enough to disambiguate the actual position of the object (potentially at $S_2 = 0.5$ or $S_2 = 0.75$ for the depicted case). Blue dots show estimations from additional focus positions that, even without being perfectly accurate, have the potential to be fitted to the CoC function that returns the actual object position $S_2 = 0.75$ (shown by the green line) when its output is zero.

- We show that single image blur estimates can behave in a robust way when applied over a focal stack, with the potential to estimate accurate depth values up to a scale.
- A fast and flexible method, with components that can be easily improved independently as respective state of the art advances.
- A novel normalized convolution scheme with an edge-preserving kernel to remove noise from the blur estimates.
- A novel global error metric that allows the comparison of depth maps with similar global shapes but local misalignments of features.

6.2 RELATED WORK

There is a vast amount of literature on the topic of estimating depth and shape based on monocular focus cues; we comment on the main approaches and how they relate to ours. First we discuss active methods that make use of additional hardware or setups to control the defocus blur. Next we discuss passive methods that depend on whether the information comes from focused or defocused areas.

ACTIVE METHODS Coded apertures that modify the blur patterns captured by the sensor for a better processing have been used [130, 145]. Moreno-Noguer et al. [152] project a dotted pattern over the scene during capture. In the depth from diffusion approach [220], an optical diffuser is placed near

the object being photographed. Lin et al. [135] combine a single-shot focal sweep and coded sensor readouts to recover full resolution depth and all-in-focus images. Our approach does not need any additional or specialized hardware, so it can be used with regular off-the-shelf cameras or mobile devices like smartphones and tablets.

PASSIVE METHODS: SHAPE FROM FOCUS These methods start computing a focus measure [171] for each pixel of each image in the focal stack. A rough depth map can then be easily built assigning to each of its pixels the position in the focal stack for which the focus measure of that pixel is maximal. As the resolution of the resulting depth map in the z-axis depends critically on the number of images in the focal stack, this approach usually employs a large number of them (several tens). Improved results have been obtained when focus measures are filtered [159, 141, 184] or smoother surfaces fitted to the previously estimated depth map [192]. Our method uses fewer images and the resolution in the z-axis is independent on the number of them.

PASSIVE METHODS: SHAPE FROM DEFOCUS In this approach, the goal is to estimate the blur radius for each pixel, which varies according to its distance from the camera and focus plane. Since the focus position during capture is usually known, a depth map can be recovered [168]. This approach significantly reduces the number of images needed for the focal stack, ranging from a single image to a few of them.

Approaches using only a single image [16, 156, 221, 40, 222, 39] make use of complex focus measures and filters to obtain good results in many scenarios. However, they are not able to disambiguate cases where the blur cannot be known to come from the object being in front of or behind the focus plane (see Figure 6.1). Cao et al. [41] solves this ambiguity through user input.

Using two or more images, Watanabe and Nayar [202] proposed an efficient set of broadband rational operators invariant to texture that produces accurate, dense depth maps. However, those sets of filters are not easy to customize. Favaro et al. [65] model defocus blur as a diffusion process based on the heat equation, then they reconstruct the depth map of the scene estimating the forward diffusion needed to go from a focused pixel to its blurred version. Our algorithm is not based on the heat diffusion model but on heuristics that are faster to compute. Favaro [63] imposes constraints for the reconstructed surfaces based on the similarity of their colors. The results presented there show great details, but as acknowledged by the author, color cannot be considered a robust feature to determine surface boundaries. Li et al. [133] use shading information to refine depth from defocus results in an iterative method.

Hasinoff and Kutulakos [81] proposed a method that uses variable aperture sizes along with focal distances for detailed results. However, such an approach needs the aperture size to be controllable and they use hundreds of images for each depth map.

Our work follows a shape from defocus approach with a reduced focal stack of at least two images. We use simple but robust per-pixel blur estimates, coupled with high quality image filtering to remove noise and increase robustness. We analyze the evolution of the blur at each pixel through the focal stack by fitting it to an analytical model for the blur size, which returns the distance of the object from the camera up to a scale.

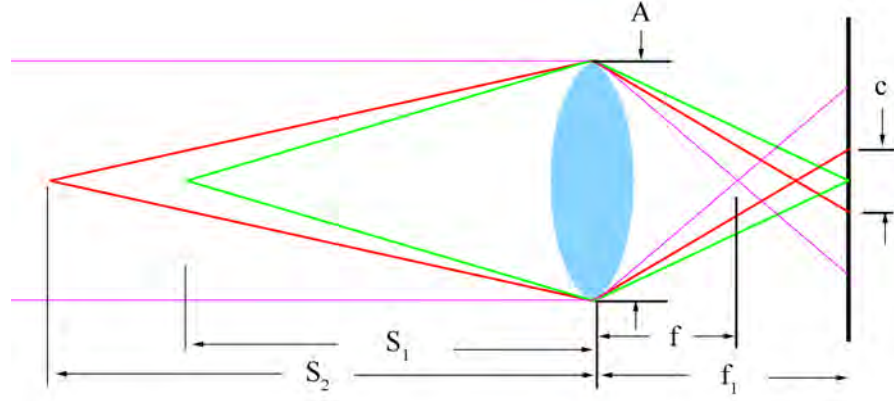


Figure 6.2: Diagram showing image formation on the sensor when points are located on the focal plane (green), or out of it (red and pink).

6.3 BACKGROUND

The circle of confusion is the resulting blur circle captured by the camera when light rays from a point source out of the focal plane pass through a lens with a finite aperture [84]. The diameter c of this circle depends on the aperture size A , focal length f , the focal distance S_1 , and the distance S_2 between the point source and the lens (see Figure 6.2). Keeping the aperture size, focal length, and distance between the lens and the point source constant, the diameter of the circle of confusion can be controlled by varying the focal position using the following relation when the focal position S_1 is finite:

$$c = c(S_1) = A \frac{|S_2 - S_1|}{S_2} \frac{f}{S_1 - f} \quad (6.1)$$

and when the focal position S_1 is infinite,

$$c = \frac{fA}{S_2}. \quad (6.2)$$

As shown in Figure 6.1, the relation between the focal position S_1 and c is non-linear. The behavior of Equation 6.1 is not symmetric around the distance of the focal plane (S_2), and approaches infinity for objects in front of the focal plane (making them disappear from the captured image) and asymptotically approaches the value given by Equation 6.2 for objects behind it.

Our goal is to obtain the distance of each object S_2 for each pixel in the image. But, as seen from Equation 6.1 and Figure 6.1, even knowing all parameters A , S_1 , c and f , there is ambiguity when recovering the position of S_2 with respect to the focus position S_1 . So, instead of just using one estimate for c , our approach is based on the assumption that additional $n \geq 2$ estimates of c , c_i , $1 \leq i \leq n$, for different known focal distances S_1 , S_1^i , will allow us to determine the single S_2 value that makes Equation 6.1 optimally approximate all the measures obtained.

6.4 ALGORITHM

Our shape from defocus algorithm starts with a series of images that capture the same stationary scene but vary the focal position of the lens, a *focal stack*. For each image in the focal stack, we compute an estimate of the amount of blur using a two-step process. First, a focus measure is applied to each pixel of each image in the stack. This procedure generates reliable blur estimates near edges. We next determine which blur estimates are unreliable or invalid, and extrapolate them based on the existing irregularly sampled estimates in each image. For this step, we propose a novel combination of normalized convolution [117] with an edge-preserving filter for its kernel.

With blur estimates for each pixel in each image, we proceed to estimate per-pixel depth values fitting our blur estimates to the analytical function for the circle of confusion. We construct a least squares error minimization problem to fit the estimates to that function. Minimizing this problem gives the optimal depth for a point in the scene.

6.4.1 Focal Stack

The input to our algorithm is a set of n images where $n \geq 2$. In our tests we use 2 or 3 images. Each image captures the same stationary scene from the same viewpoint. The only difference between each image is the focal distance of the lens when the image is captured. Thus, each point in the object space will have varying circles of confusion in each image of the focal stack. Additionally, the focal position S_1^i of the lens when the image is captured is saved, where i is the i^{th} image in the focal stack. While this information can be obtained easily from different sources (EXIF data, APIs to access digital cameras or physical dials on the lenses), in its absence a rough estimate of the focal distances based on the location of the objects in focus may suffice.

In this work we assume that the images are perfectly registered to avoid misalignments due to the magnification that occurs when the focal plane changes. This can be achieved using telecentric optics [202] or image processing algorithms [81, 63, 197].

6.4.2 Local Blur Estimation

Our first step is to apply a focus measure that will give a rough estimate of the defocus blur for each pixel and thus an estimation of its circle of confusion. Several different measures have been proposed previously [171]. In our case, Hu and De Haan's [89] provided enough robustness and consistency to track the evolution of blur over the focal stack.

Given user defined parameters σ_a and σ_b , representing the blur radii of two Gaussian functions with $\sigma_a < \sigma_b$, the local blur estimation algorithm is applied to the focal stack. The algorithm estimates a radius of the Gaussian blur kernel σ for each signal in each image in the focal stack. Note that σ_a and σ_b are chosen a priori and for the algorithm to work well $\sigma_a, \sigma_b \gg \sigma$. We empirically chose $\sigma_a = 4$ and $\sigma_b = 7$ for images of size 720x480. For the one-dimensional case, the radius of the Gaussian blur kernel, σ , is estimated as follows:

$$\sigma(x) \approx \frac{\sigma_a \cdot \sigma_b}{(\sigma_b - \sigma_a) \cdot r_{max}(x) + \sigma_b} \quad (6.3)$$

with

$$r_{max}(x) = \frac{I(x) - I_a(x)}{I_a(x) - I_b(x)} \quad (6.4)$$

where x is the offset into the image, and $I(x)$ is the input image; $I_a(x)$ and $I_b(x)$ are blurred versions of $I(x)$ using the blur kernels σ_a and σ_b , respectively. For 2-D images, isotropic 2D Gaussian kernels are used. We work with luminance values from the captured RGB images.

Because this algorithm depends on the presence of edges (discontinuities in the luminance), regions of the image far from edges or significant changes in signal intensities need to be estimated by other means. Consider a region of the image that is sufficiently far from an edge; for example, around $3\sigma_a$ from an edge, the intensities of the original image $I(x)$ and the blurred images $I_a(x)$ and $I_b(x)$ will be close to each other because the intensities in a neighborhood around x in the original image I are similar. This similarity causes the difference ratio maximum $r_{max}(x)$ from Equation 6.4 to go to zero if the numerator approaches zero or to infinity if the denominator approaches to zero. If $r_{max}(x)$ approaches zero, then from Equation 6.3 the estimated blur radius approaches σ_a , and if $r_{max}(x)$ approaches infinity, then the estimate approaches zero. Figure 6.3 shows an example of the blur maps obtained with this method.

It is important to note that similar to other single image blur measures, the method in [89] is not able to disambiguate an out-of-focus edge from a blurred texture. However, since we are using several images taken with different focus settings, our algorithm will seamlessly deal with their relative changes in blur during the optimization step (Section 6.4.4).

6.4.3 Noise Filtering and Data Interpolation

Because of the assumption that $\sigma_a, \sigma_b \gg \sigma$, the above algorithm does not perform well in regions of the image far from edges where $\sigma \rightarrow \sigma_a$. Moreover, for constructing our depth map, we assume that discontinuities in depth correspond to discontinuities in the edge signals of an image, but the converse does not hold since they can come from discontinuities due to changes in texture, lighting, etc. The local blur estimation algorithm performs better over such discontinuities, but leaves uniform regions with less accurate estimations. Thus, we need a way of reducing noise by interpolating data to those areas. A straightforward approach to filter noise is to process pixels along with their neighbors over a small window. However, choosing the right window size is a problem on its own [142, 126] as large windows can remove detail in the final results. So, we propose a novel combination of normalized convolution [117] with an edge-preserving filter for its kernel.

We use normalized convolution since this method is well-suited for interpolating irregularly sampled data. Normalized convolution works by separating the data and the operator into a signal part $H(x)$ and a certainty part $C(x)$. Missing data is given a certainty value of 0, and trusted data a value of 1. Using $H(x)$ and $C(x)$ along with filter kernel $g(x)$ to interpolate, normalized convolution is applied as follows:

$$\bar{H}(x) = \frac{H(x) * g(x)}{C(x) * g(x)} \quad (6.5)$$

where $\bar{H}(x)$ is the resulting data with interpolated values for the missing data.

As the first step, we categorize good blur radius estimates and poor ones, which we then mark as missing data. Poor estimates will correspond to estimates for discrete signals in the input image that are sufficiently far from detectable edges, and can be identified by their values being close to σ_a . Thus, we define good estimates as any blur estimate σ contained in the interval $[0, \sigma_a - \delta)$ and invalid estimates are contained in the interval $[\sigma_a - \delta, \sigma_a]$ where $\delta > 0$. In our experiments, we found that a value of $0.15\sigma_a$ worked well for δ . The confidence values for normalized convolution are then generated as follows:

$$C(x) = \begin{cases} 1 & \text{if } \sigma(x) < \sigma_a - \delta \\ 0 & \text{otherwise} \end{cases} \quad (6.6)$$

where $\sigma(x)$ is from Equation 6.3. Figure 6.3 shows the confidence maps for the sparse blur map generated from the prior stage of pipeline. Similarly, the discrete input signal for normalized convolution is generated as follows:

$$H(x) = \begin{cases} \sigma(x) & \text{if } \sigma(x) < \sigma_a - \delta \\ 0 & \text{otherwise.} \end{cases} \quad (6.7)$$

With the resulting confidence values and input data, we only need to select a filter kernel $g(x)$ to use with normalized convolution.

Since we have estimates for discrete signals near edges in the image and need to interpolate signals far from edges, we want to use an edge preserving filter. A filter with this property ensures that discontinuities between estimates that are caused by discontinuities in intensity in the original input signal are preserved, while spatially close regions with similar intensities will be interpolated based on valid nearby estimates that share similar intensities in the original image from the focal stack. There are several filters that have this property including the joint bilateral filter [172] and the guided image filter [82]. We use the guided image filter because of its efficiency and proven effectiveness [22]. In the absence of better guides, we use the original color images from the focal stack as the guides for the corresponding blur maps. With this filter as the kernel, we apply normalized convolution as described in Equation 6.5. We use this technique to generate refined blur estimates for each image in the focal stack. The size of the spatial kernel for the guided image filter needs to be large enough to create an estimation of the Gaussian blur radius for every discrete signal in the image. Therefore, sparser maps require larger spatial kernels. The guided image filter has two parameters, the radius of the window and a value ϵ related to edge and detail preservation. Experimentally, we found that a window radius of between 15 and 30, and ϵ of $7.5e-3$ works well for our focal stacks. The end result is a set of n maps, $\hat{H}_i(x)$, that estimate the radius of the Gaussian blur kernel in image i of the focal stack. Since the circle of confusion can be modeled as a Gaussian blur, these maps can be used to estimate the diameter of the circle of confusion for each pixel in each image of the focal stack. Figure 6.3 shows the output of the normalized convolution for each image in the focal stack.

6.4.4 Fit to the Analytical Circle of Confusion Function

Through the previous steps, each image I_i in the focal stack of size n is accompanied by the focal distance of the shot S_1^i . We can then estimate

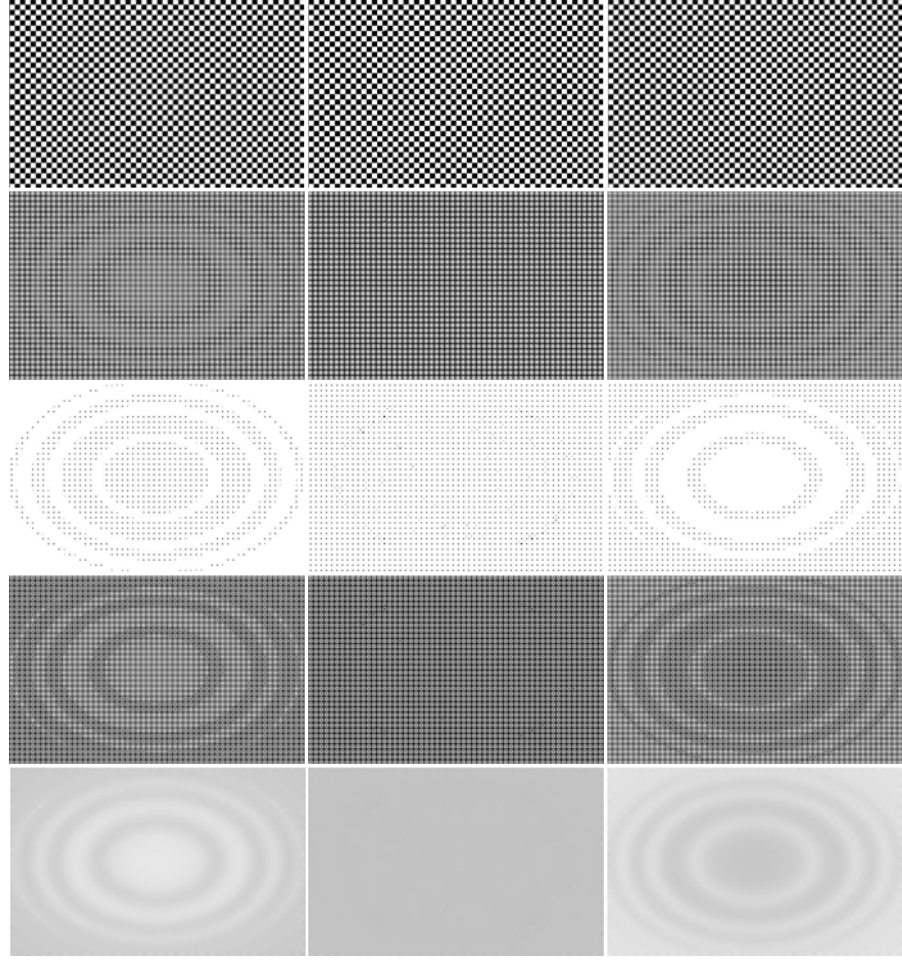


Figure 6.3: From top to bottom, the different steps of our algorithm: Input focal stack consisting of three images (left to right) from a synthetic dataset (more details in Subsection 6.5.1). Initial blur estimations. Confidence maps from Equation 6.6. Masked blur maps after Equation 6.7. Refined blur maps after the application of normalized convolution. It can be seen how we are able to produce smooth and consistent blur maps to be used as the input for our fitting step. Final reconstruction for this example is shown in Section 6.5

actual depth information. We first show how to do this for one pixel and its n circle of confusion estimations.

Given Equation 6.1 for the circle of confusion, every variable is currently known or estimated except for S_2 , the unknown depth. Solving for S_2 using only one estimate for the circle of confusion is not possible because of the ambiguity shown in Figure 6.1; otherwise, there will be two possible values for S_2 , as shown in the following equation:

$$S_2 = \frac{S_1}{\pm \frac{c(S_1 - f)}{Af} - 1}. \quad (6.8)$$

To find a unique S_2 , a system of non-linear equations is constructed where we attempt to solve for S_2 that satisfies all of the equations. Each equation

solves for depth given the circle of confusion estimates c_i for one image of the focal stack:

$$S_2 = \frac{S_1^i}{\pm \frac{c_i(S_1^i - f)}{Af} - 1} \text{ for all } i = 1, \dots, n \quad (6.9)$$

Since these equations are not, in almost all cases, satisfied simultaneously, we use a least squares method to minimize the error where we want to reduce the error in measured value for the circle of confusion. Thus, we obtain the following function to minimize:

$$\sum_{i=1}^n \left(c_i - A \frac{|S_2 - S_1^i|}{S_2} \frac{f}{S_1 - f} \right)^2 \quad (6.10)$$

This equation leads to a single-variable non-linear function whose minimizer is the best depth estimation for the given blur estimates. The resulting optimization problem is tractable using a variety of methods [176]. In our implementation we use quadratic interpolation with the number of iterations fixed at four. This single variable optimization problem can then be extended to estimate depth for each discrete pixel in the image. The result is a depth map that can be expressed as:

$$D(x) = \min \left[\sum_{i=1}^n \left(c_i(x) - A \frac{|S_2 - S_1^i|}{S_2} \frac{f}{S_1 - f} \right)^2 \right] \text{ for } S_2$$

To make our optimization run quickly, we assume bounds on the range of values that S_2 can have for each pixel. In particular, we assume that the depth of at every point in the scene lies between the nearest focal length and the farthest focal length of all the images in the focal stack [202]. Note that this assumption is only necessary for fast optimization; methods that have an unbounded range exist [176].

However, because of this assumption every blur estimate needs to be scaled to ensure there are local minimizers of Equation 6.10 that lie somewhere within the assumed range of depth. As shown in Appendix 6.A, to ensure that there is a minimizer on the interval between the closest and farthest focal distances, an upper bound on the blur estimates c_i must be imposed. This bound is given by

$$\frac{Af}{S_1^j - f} = r \geq 2c. \quad (6.11)$$

Furthermore, we know that all blur estimates generated from normalized convolution are between 0 and σ_n . Thus some positive scalar s can be defined as follows:

$$s \leq \frac{Af}{2\sigma_n(S_1^n - f)} \quad (6.12)$$

where S_1^n is the largest focal distance in the stack. Multiplying each blur estimate by s ensures that Equation 6.11 is satisfied for all blur estimates, which implies that under normal conditions, there will be at least one local minimizer for Equation 6.10 between the nearest and farthest focal distances. Figure 6.5 shows the final depth map for the focal stack from Figure 6.3.

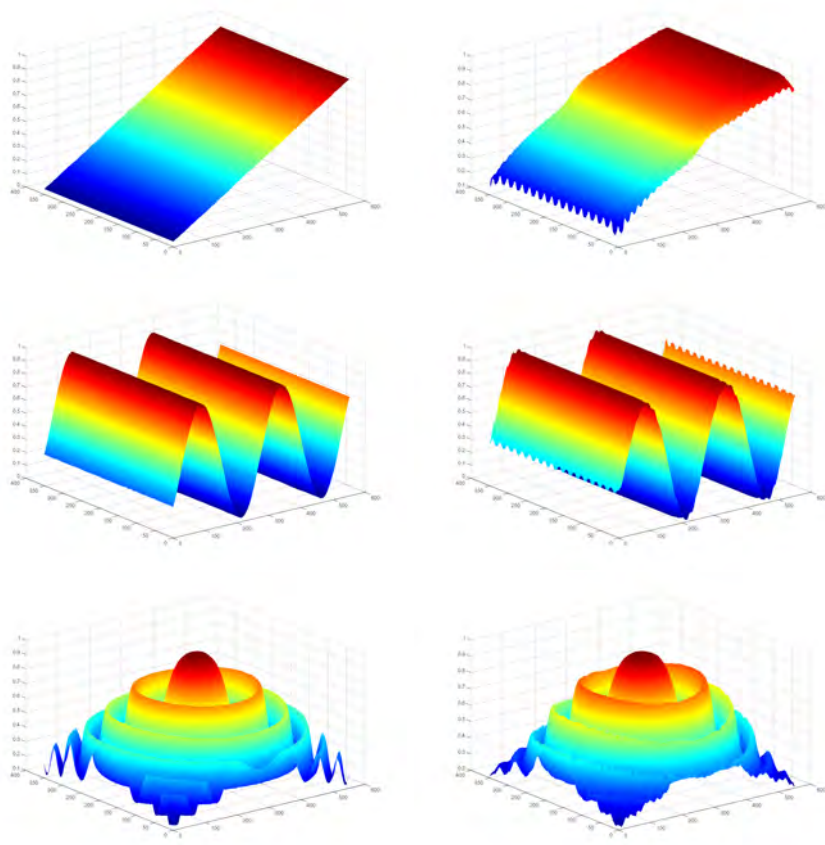


Figure 6.4: 3D Visualizations of the original depth maps (left) and our estimated depth maps (right). As can be seen, the global shape of the object is reconstructed in a recognizable way in all cases.

6.5 RESULTS

In the following, we test our algorithm with synthetic scenes. Next, we run it over real scenes from previous work to allow visual comparisons between methods. Our algorithm can run in linear time. The C++ implementation of our algorithm takes less than 10s to generate the final depth map for 640×480 inputs on an Intel Core i7 2620M @ 2.7GHz.

6.5.1 Synthetic Scenes

To validate the accuracy of our algorithm, we generated synthetic focal stacks similar to those in prior work [65, 141]. In particular, we used the slope, sinusoidal and wave objects shown in Figure 6.4.

To create the synthetic focal stacks, we start from an in-focus image and its depth map. Using Equation 6.1 we are able to estimate the amount of blur c to be applied to each pixel of the image. We assume the depth map ranges between 0.45m and 0.7m, and the lens parameters are $f = 30\text{mm}$ and f-number $N = 2.5$. We then obtain three different images for each focal stack, with focal distances set to $S_1^1 = 0.4\text{m}$, $S_1^2 = 0.6\text{m}$ and $S_1^3 = 1.0\text{m}$ (the resulting focal stack for the wave example can be found in Figure 6.3).

Figure 6.4 shows the results of running our algorithms over these focal stacks, compared against the ground truth data. As can be seen, the global

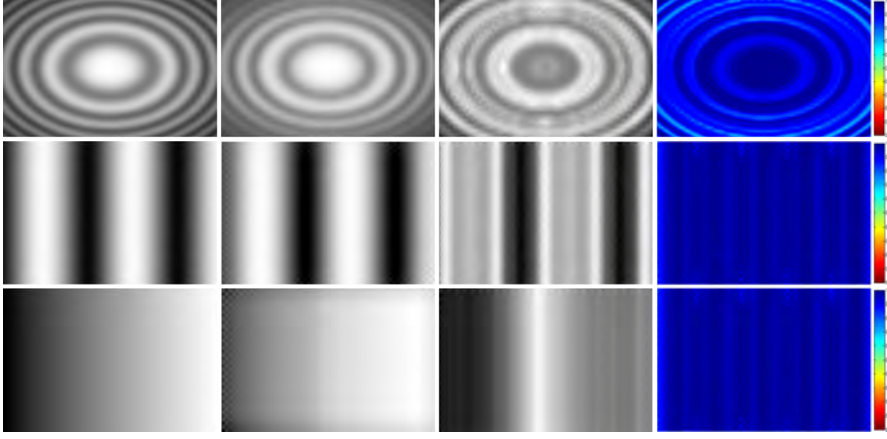


Figure 6.5: Comparison of original depth maps (on the left) with our estimations (middle left). Local error from the curve fitting step (middle right) where the errors ranged between a magnitude of 10^{-9} and 10^{-8} (black and white respectively for better visualization), and our global accuracy metric (right). In this last case a value of one means a perfect match. Our local and global accuracy metrics clearly show that while local errors may occur, the reconstructed global shape of the object has a good resemblance with the ground truth one, as appreciated also in Figure 6.4.

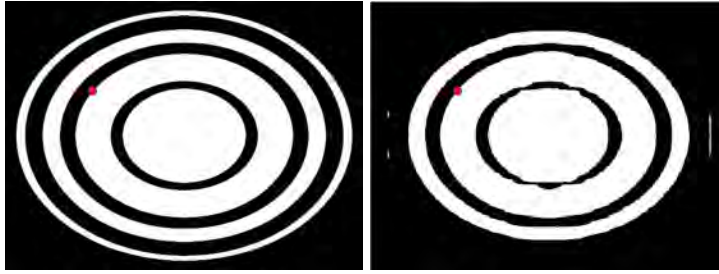


Figure 6.6: Example of estimating the global accuracy of a pixel (marked in red) for the wave object from Figure 6.4. Pixels with depth values greater or equal to it are marked in white, while the rest keep unmarked (black). This is done for both the ground truth depth map (left) and the estimated depth map (right). A similarity measure for that pixel is then computed by marking with one all the pixels with matching values and dividing that number by the total size of the map.

shape of the object is properly captured, but there are also noticeable local errors at different scales. Standard error metrics are thus difficult to apply because of their aggregation of these local error measures. Thus we propose a novel error metric that favors the global shape comparing relativity between original and estimated depth values.

6.5.2 Global and Local Error Metrics

We start choosing a reference pixel in the original depth map and mark (with 1) all pixels in the map that are greater than or equal to the depth value at that pixel. All other pixels remain unmarked (with 0). We repeat this process for the estimated depth map using the same reference pixel, as seen in Figure 6.6. We then compute a similarity map by comparing per-pixel values in both previous maps, obtaining final values of 1 only for matching pixel values. An accuracy value for the reference pixel is computed by taking

the sum of all values in the similarity map and dividing it by the total number of pixels of the map. So values closer to 1 are more accurate than the ones closer to 0. This process is repeated for each pixel in the depth maps to obtain accuracy maps as seen on the right in Figure 6.5.

In addition to our global accuracy metric, we can also obtain per-pixel error maps from the optimization step. Such maps show the squared error obtained when fitting Equation 6.1 to the estimated blur values for one pixel through the focal stack, to obtain its final depth value. Examples of these maps can be found in Figure 6.5 (middle right).

Looking at the blur estimates used for the optimization reveals that small blurs were over-estimated while large blurs were under-estimated. These inaccuracies caused the algorithm to compress the depth estimates such that the range of estimated depths is smaller than the actual range. However, since blur estimate errors are consistent across the entire image, the depth estimates are still accurate relative to each other, and so the global shape captures the main features of the ground truth.

6.5.3 *Real Scenes*

We also tested our algorithm with real scenes. We again used examples from prior work [202, 65, 63] to allow direct visual comparisons with our results. In these examples, the number of images for each focal stack is two. As can be seen in Figure 6.7, we obtain plausible reconstructions comparing favorably with both Watanabe and Nayar [202] and Favaro [65], even though our depth maps look blurrier due to the filtering explained in Subsection 6.4.3. Our work presents an interesting tradeoff between accuracy and speed, as it is significantly faster than the 10 minutes reported in [63]

Additional examples from real scenes can be found in Figure 6.8. The first two rows show plausible reconstructions for different stuffed toys. The bottom row shows a difficult case for our algorithm. Given the asymptotic behavior of the circle of confusion function (Figure 6.1), objects from a certain distance show small differences in blur. Since our blur estimations are not in real scale, this translates into either unrelated distant points recovered into the same background plane, or inaccurate and different depth values for neighboring pixels. This happens usually in outdoor scenes, so our algorithm is better suited for close-range scenes.

6.6 CONCLUSIONS

We have presented an algorithm that estimates depth from a focal stack of images. This algorithm uses a least-squares optimization to obtain depth values from a set of pixel measurements up to a scale. We have shown that it compares well with prior work but runs significantly faster.

As mentioned previously, our algorithm possesses some limitations. The focus measure we employed [89] has difficulties in estimating large blur radii, producing an undesired flattening of the estimated depth map. It would be interesting to test other measures included in Pertuz et al. [171] to see their effect. In Figure 6.9, we show that our algorithm can robustly handle small inaccuracies in focal distances, and it would be interesting to analyze the effect of these inaccuracies in future work. Also, the guided filter [82] used as the kernel for the normalized convolution shows texture-copy artifacts sometimes given the suboptimal use of the color images as the guides for the filter. However, it is not clear what could be a good guide for



Figure 6.7: Close focus (left), Far focus (middle left), our estimated depth map (middle right), and its corresponding 3D visualization (right). Colors and shading added for a better visualization.



Figure 6.8: Close focus (left), Far focus (middle left), our estimated depth map (middle right), and its corresponding 3D visualization (right). Colors and shading added for a better visualization. The estimated depth map for the top scene used parameters $f=24\text{mm}$, $f/8$, close focal distance of 0.35m , and far focal distance of 0.75m . The estimated depth map for the middle scene used parameters $f=26\text{mm}$, $f/8$, close focal distance of 0.4m , and far focal distance of 0.8m . The estimated depth map for the bottom scene used parameters $f=18\text{mm}$, $f/8$, close focal distance of 0.5m , and far focal distance of 15m .

this tasks, with possible choices like intrinsic images [218] being ill-posed problems that may introduce their own artifacts. Finally, while our current optimization step is already using interpolated blur data that took into account the confidence of each sample, it could be interesting to combine those confidence values in order to place additional constraints during this step.

We believe our method presents an interesting tradeoff between accuracy and speed when compared with previous works. The modularity of our

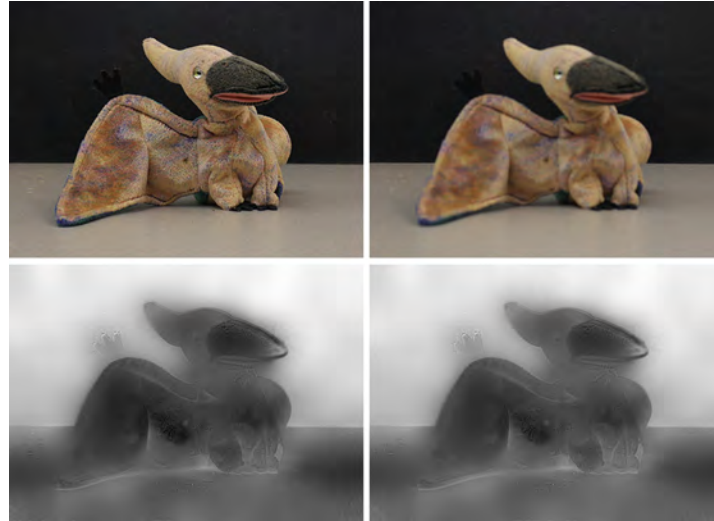


Figure 6.9: Comparison between accurate and estimated focus positions. Top: Input images captured with focal distances of $0.4m$ (left), and $0.8m$ (right). Bottom left: estimated depth map using those focal distances. Bottom right: results using estimates of $0.3m$ and $1.0m$ respectively. As can be seen, our algorithm can handle small inaccuracies robustly.

approach makes it straightforward to study alternatives to the chosen algorithms at each step, so it can greatly benefit from separate advances that occur in the future.

APPENDICES

6.A LEAST SQUARES FUNCTION ANALYSIS

In this appendix we show how to cast the depth estimation problem as an optimization problem. Consider the optimization problem for a single signal with n blur estimates, and each c_i is captured with a focal position S_1^i . Let

$$g_i(x) = \left(c_i - A \frac{|x - S_1^i|}{x} \frac{f}{S_1 - f} \right)^2 \quad (6.13)$$

The function $g_i(x)$ has a critical point at S_1^i because the derivative at S_1^i of $g_i(x)$ does not exist due to the term $|x - S_1^i|$ in the function. Furthermore, if the blur estimate c_i is less than the circle of confusion size

$$c = \frac{f^2}{N(S_1^i - f)} \quad (6.14)$$

for a depth x at infinity, then the function will have two local minimizers, as shown in Figure 6.10, at the points $g(x) = 0$ where

$$x = \frac{S_1^i A f}{A f - c_i (S_1^i - f)} \quad (6.15)$$

and

$$x = \frac{S_1^i A f}{A f + c_i (S_1^i - f)}. \quad (6.16)$$

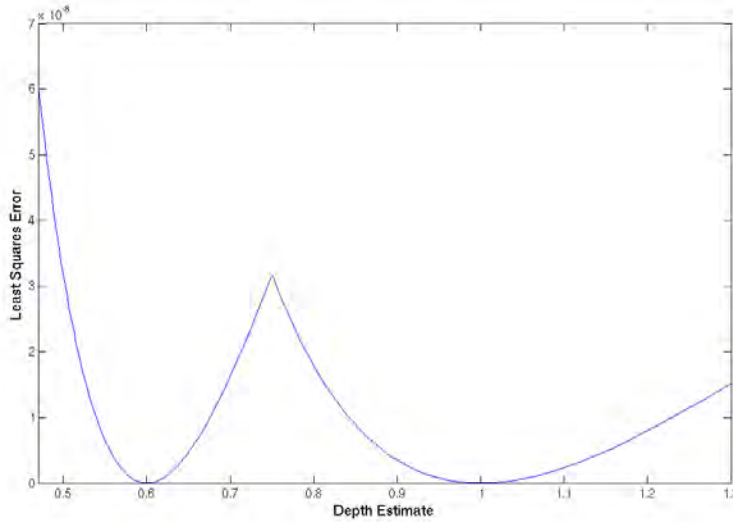


Figure 6.10: Plot of $g_i(x)$ showing a local maximizer at the point $S_1^i = 0.75m$ and two local minimizers on either side of the maximizer.

However, if $c_i = 0$ then the function will have one minimizer at $x = S_1^i$, and similarly if x is larger than the circle of confusion size for a depth at infinity, then $g_i(x)$ will have only one minimizer somewhere within the interval $(0, S_1^i)$.

For the purposes of optimization, we assume that

$$0 < c_i < \frac{f^2}{N(S_1^i - f)}. \quad (6.17)$$

This assumption introduces the restriction that the depth of a signal in the focal stack cannot be too close to the lens.

A further restriction for the depth x is that $S_1^1 < x < S_1^n$ where $0 < S_1^1 < S_1^2 < \dots < S_1^n$. This restriction limits the depth of any point in the focal stack to be between the closest focal position of the lens and the farthest focal position.

With these assumptions, we can now look at the least squares optimization equation

$$z(x) = \sum_{i=1}^n g_i(x). \quad (6.18)$$

Because each $g_i'(x)$ is undefined at $x = S_1^i$ for all $i = 1, \dots, n$, the function $z(x)$ has critical points at S_1^1, \dots, S_1^n . Furthermore, $z(x)$ is continuous everywhere else for $x > 0$ because the functions $g_i(x)$ are continuous where $x > 0$ and $x \neq S_1^i$. Because $g_i(x)$ has a local maximizer at S_1^i , this point may be a local maximizer for $z(x)$. This gives us $n - 1$ intervals on which $z(x)$ is continuous for $S_1^1 < x < S_1^n$, and these intervals are $(S_1^1, S_1^2), (S_1^2, S_1^3), \dots, (S_1^{n-1}, S_1^n)$. These open intervals may or may not contain a local minimizer, and if an interval does contain a local minimizer, it might be the global minimizer of $z(x)$ on the interval (S_1^1, S_1^n) .

Under certain conditions, $z(x)$ is convex within the interval (S_1^1, S_1^{i+1}) for all $i = 1, \dots, n - 1$. Note that $g_j(x)$ is convex within the open interval for all $j = 1, \dots, n$. To see this, assume that Equation 6.11 holds and that the focus position of the lens is always greater than the focal length f of the lens so that $r > 0$. We also assume that

$$S_1^n \leq \frac{3rS_1^j}{2c_j}. \quad (6.19)$$

If $x < S_1^j$ then the absolute value term $|x - S_1^j|$ in $g_i(x)$ becomes $-x + S_1^j$. From this, we know that

$$rS_1^j \geq 2c_j x \quad (6.20)$$

from Relation 6.11 and because x and S_1^j are positive. Rearranging the relation, we get

$$-2c_j S_1^j + rS_1^j \geq 0. \quad (6.21)$$

Since $x < S_1^j$, $2rx < 2rS_1^j$ and $2rS_1^j - 2rx > 0$. Therefore,

$$\begin{aligned} -2c_j x + 3rS_1^j - 2rx &= \\ &= -2c_j x + rS_1^j + (2rS_1^j - 2rx) \\ &\geq 2rS_1^j - 2rx \\ &> 0 \end{aligned} \quad (6.22)$$

Furthermore, since $x > 0$, $r > 0$, and $S_1^j > 0$, we know that

$$\frac{2rS_1^j}{x^4} > 0. \quad (6.23)$$

Therefore, we know that

$$g_j''(x) = \frac{2rS_1^j(-2c_jx + 3rS_1^j - 2rx)}{x^4} > 0 \quad (6.24)$$

for $0 < x < S_1^j$.

If $x > S_1^j$ then

$$x < S_1^n \leq \frac{3rS_1^j}{2c_j} \quad (6.25)$$

from Equation 6.19 and that $x < S_1^n$. Since $c_j > 0$, we can multiply the relation by $2c_j$ to get

$$3rS_1^j > 2c_jx. \quad (6.26)$$

From relation (6.11), we can say that

$$2r - 2c_j \geq 4c_j - 2c_j = 2c_j. \quad (6.27)$$

Therefore,

$$3rS_1^j > x(2r - 2c_j) \geq x(2c_j). \quad (6.28)$$

Distributing x in the above relation, we get

$$3rS_1^j > 2rx - 2c_jx \quad (6.29)$$

Rearranging the terms, we get

$$2c_jx + 3rS_1^j - 2rx > 0. \quad (6.30)$$

Multiplying by the left hand side of (6.23), we get

$$g_j''(x) = \frac{2rS_1^j(2c_jx + 3rS_1^j - 2rx)}{x^4} > 0 \quad (6.31)$$

for $S_1^j < x < S_1^n$.

As shown above, the second derivative of $g_j(x)$ is always positive on the interval (S_1^1, S_1^n) except at the point S_1^j for all $j = 1, \dots, n$. Since $z(x)$ is the summation of all $g_j(x)$, $z(x)$ is also convex on the interval except at the points $S_1^1, S_1^2, \dots, S_1^n$. Therefore, $z(x)$ is convex in the intervals (S_1^i, S_1^{i+1}) for all $i = 1, 2, \dots, n-1$. As a consequence, if S_1^i and S_1^{i+1} are local maximizers, then there is some local minimizer within the open interval (S_1^i, S_1^{i+1}) . From this, a global minimizer can be identified which gives the best depth estimate for the given signal on the interval (S_1^1, S_1^n) . Figure 6.11 shows an example of $z(x)$ with the local maximizers and minimizers.

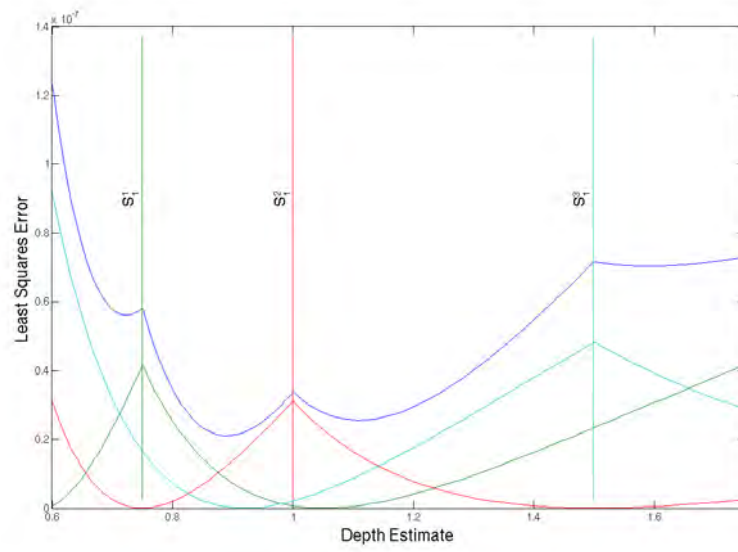


Figure 6.11: Plot of $z(x)$ shown in dark blue with $g_1(x)$, $g_2(x)$, and $g_3(x)$ shown in red, light blue, and green respectively. This shows $z(x)$ with local maximizers at $S_1^1 = 0.75$, $S_1^2 = 1$, and $S_1^3 = 1.5$ and local minimizers in the intervals (S_1^1, S_1^2) and (S_1^2, S_1^3) .

CAPTURING AND STYLIZING HAIR FOR 3D FABRICATION

Recently, we have seen a growing trend in the design and fabrication of personalized figurines, created by scanning real people and then physically reproducing miniature statues with 3D printers. This is currently a hot topic both in academia and industry, and the printed figurines are gaining more and more realism, especially with state-of-the-art facial scanning technology improving. However, current systems all contain the same limitation – no previous method is able to suitably capture personalized hair-styles for physical reproduction. Typically, the subject’s hair is approximated very coarsely or replaced completely with a template model.

In this Chapter we present the first method for stylized hair capture, a technique to reconstruct an individual’s actual hair-style in a manner suitable for physical reproduction. Inspired by centuries-old artistic sculptures, our method generates hair as a closed-manifold surface, yet contains the structural and color elements stylized in a way that captures the defining characteristics of the hair-style. The key to our approach is a novel multi-view stylization algorithm, which extends feature-preserving color filtering from 2D images to irregular manifolds in 3D, and introduces abstract geometric details that are coherent with the color stylization. The proposed technique fits naturally in traditional pipelines for figurine reproduction, and we demonstrate the robustness and versatility of our approach by capturing several subjects with widely varying hair-styles.

This work is published in *ACM Transactions on Graphics* and presented at *SIGGRAPH 2014*. It was developed during a four-months internship at the *Capture & Effects Group* at Disney Research Zurich (Switzerland), and a US patent application was filed with The Walt Disney Company.

J. I. Echevarria, D. Bradley, Diego Gutierrez & T. Beeler
CAPTURING AND STYLIZING HAIR FOR 3D FABRICATION
ACM Transactions on Graphics, Vol.33 (4), SIGGRAPH 2014

T. Beeler, D. Bradley & J. I. Echevarria
CAPTURING AND STYLIZING HAIR FOR 3D FABRICATION
US Patent Application US20160071316 A1

7.1 INTRODUCTION

A mainstream goal in computer graphics is to create data-driven methods for building geometric models of humans. In recent years we have seen advances in 3D face and body scanning, motion capture and real-time performance capture. While human scanning has many applications in fields like video games, films and medical analysis, a fast-growing field is the physical reproduction of miniature statues or figurines. Physical reproduction in general, and particularly of humans, has become a hot topic both in academia [133, 191, 194] and industry [2, 1, 50, 164, 173]. Recently, *3D-Systems* even announced the release of a 3D-photobooth, which will facilitate 3D-portraits for the masses. The underlying pipeline of all these systems is essentially the same: A person is scanned, the resulting mesh is processed



Figure 7.1: Our work is inspired by artistic sculptures of hair in the real world and those created by digital artists in professional modeling software.

often with artist interaction, and the figurine is printed using a 3D printer. Consequently, all systems have similar drawbacks, and in particular, no previous approach can capture personalized *hair-styles* with adequate details, while being suitable for physical reproduction.

Almost as much as the face, a person’s hair-style is a defining characteristic of an individual. The reproduction of figurines without properly capturing the hair is a severe limitation of current systems, since the hair-style contributes so substantially to the person’s identity. Existing research in hair capture methods either focus on reconstructing highly-detailed individual wisps or hair strands [139, 24, 167], which do not meet the physical manufacturing constraint of a closed manifold surface, or they produce coarser reconstructions [140] that lack the level of stylization, detail or colors required to produce appealing 3D-printed models. In this work we present the first method for *stylized* hair capture, which addresses current limitations of physical reproduction systems, enabling the faithful miniaturization and physical reproduction of figurines with drastically varying hair-styles. Our method automatically reduces the complexity of hair to an abstract, printable 3D surface, while still capturing the essential structural and color elements that define its style. The proposed method fits naturally into existing physical reproduction pipelines, and so our work has the potential to significantly impact the growing industry of figurine reproduction.

Our work is inspired by existing artistic sculptures of hair. For centuries, artists have shown that the essence of a hair-style can be represented on a continuous manifold surface, such as marble or clay, through stylized sculpting of geometric details (Figure 7.1, left). Today in the digital world, CG modelers follow the same principles by virtually sculpting hair structure on a 3D mesh (Figure 7.1, right). Our goal is to computationally achieve a similar level of abstraction in the captured hair-style of an individual.

To this end, we start by obtaining a smooth surface representation of the hair-style from multi-view stereo reconstruction. As our primary contribution, we introduce a novel color stylization operator that works directly over the geometric mesh domain, and can be applied over non-uniform manifold surfaces. This way, color information can be sampled, stored and processed in a consistent way with respect to the input views. As it is important to retain the appearance of directional wisps and the overall flow of hair, the color is stylized over the mesh using a combination of directional smoothing and orthogonal shock filters, inspired by analogous 2D image stylization [111, 121]. The per-vertex stylized color is then used to generate coher-



Figure 7.2: *We present the first method to capture an individual’s hair style (left) in a manner suitable for miniaturization and physical reproduction while still faithfully preserving the essential visual look of the style (right).*

ent geometric displacements over the surface, effectively stylizing the shape as well. The final result is a printable surface that can be miniaturized so that both geometry and color convey the hair-style of the captured person (see Figure 7.2). Our method allows the user to adjust the level of abstraction to match the scale of the final printout and to achieve different visual styles, which behave in a consistent way no matter the complexity of the original hair-style. Ultimately, the stylized hair is combined with state-of-the-art face scanning and traditional 3D printing methods for fabricating full-head figurines. We show the flexibility of our approach by reconstructing a large number of varied, complex hair-styles, and even non-human furry objects.

7.2 RELATED WORK

Our work is related to methods for creating personalized figurines, reconstructing hair, geometry abstraction and image stylization.

PERSONALIZED FIGURINES. A large variety of methods are being employed to capture a person in 3D, ranging from depth sensors such as the Kinect to photogrammetric systems. Li et al. [133] present a system to capture 3D self portraits from a 3D sensor, based on non-rigid registration of several partial scans and Poisson texture blending for smooth colors from the input views. Sturm et al. [191] propose a signed distance function that is updated at interactive rates directly from the sensor feed, obtaining dense 3D models. Tena et al. [194] develop a semi-automated, commercial system to seamlessly integrate customer faces into figurines. However, all methods have problems when it comes to reconstructing hair, either approximating the hair-style in low-resolution or replacing it completely with a pre-modeled template.

In addition to research efforts, the consumer market is exploding with products and services that offer physical reproduction of personal figurines [2, 1, 50, 164, 173]. However, these systems suffer from the same drawback that the hair-style is not sufficiently captured. Our work is complementary to these efforts, proposing the first approach to address this limitation and provide personalized hair-styles suitable for 3D reproduction.

HAIR RECONSTRUCTION. Early work on reconstructing hair from images targeted simple hair-styles [118] or reconstructed only partial hair [74]. More recently we have seen several advances in generating photo-realistic hair reconstructions [166, 204, 167, 100, 24, 87, 44, 139, 90] and synthesis [201]. These methods aim to reconstruct individual strands of hair, which do not meet the physical reproduction constraint of a manifold surface. In contrast, we aim to create 3D-printable hair with enough geometric detail to convey the same *style* as the original. Luo et al. [140] obtain a hair surface from multiple images, based on the observation that orientation fields are reasonably coherent across views. However, hair-style results tend to lack the level of stylization, detail and color required to produce appealing 3D-printed models. Finally, coarse hair reconstructions have been obtained from a single image for the application of advanced image editing [45], which differs significantly from our goal of stylized 3D hair capture.

GEOMETRY ABSTRACTION. Several works have recently appeared on the topic of abstraction of geometrical features for simplification of complex models [214, 157, 149]. However, those focus on man-made shapes, which are very different from the geometry of hair-styles.

IMAGE AND VIDEO STYLIZATION. The field of non-photorealistic rendering is traditionally very active in studying the problem of stylizing and abstracting 2D images and videos (refer to Kyprianidis et al. [120] for a detailed survey). Specifically related to hair stylization are methods that stylize images while preserving the directionality of the most prominent features [111, 121]. These techniques are designed for single images represented as 2D regular grids, and are thus unsuited for 3D hair stylization. Even in the context of multi-view styling, consistency of existing 2D stylization algorithms across views is not guaranteed. We draw inspiration from these 2D operators, and propose a novel extension to irregular 3D geometric domains and multi-view settings. This allows us to stylize hair while handling its view-dependent appearance and geometric complexity in a coherent way.

LATER WORK After the publication of the proposed method, new work has appeared about 3D hair reconstruction and fabrication. Some of it focus on the obtention of complex detailed 3D models, still not suitable for successful 3D printing [91, 19]. While not being their main goal, other works are able to produce realistic 3D reliefs from single images [42, 43], which can be easily printed with current technologies. However, it is not clear how they should be extended to produce full 3D models. Finally, focusing on the printing technology itself, the method of Laput et al. [125] is able to print single strands of hair. They demonstrate results for different objects, including simple hairstyles and facial hair, although it is yet to be seen how more complex models would look, and if they would be affected by the lack of stylization [215].

In summary, our work represents the first and only approach for reconstructing full 3D hair that is suitable for manufacturing personal figurines, yet stylized with subject-specific details that capture the identity of the individual.

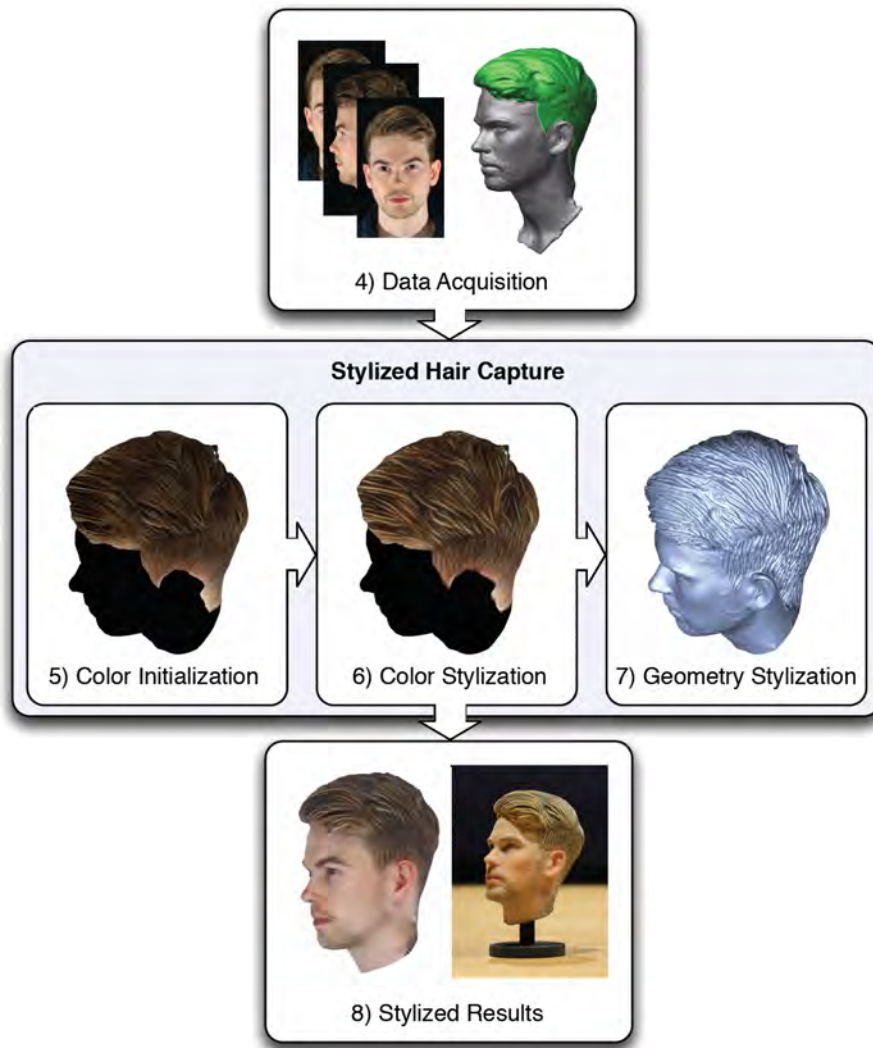


Figure 7.3: Our approach starts with a data acquisition process to construct a coarse proxy surface of the hair from multiple images. Stylized hair capture then begins by initializing, and then stylizing color information over the proxy surface using a novel stylization filter. We then extract detailed structure from the stylized colors in order to consistently stylize also the geometry. The result is a 3D-printable surface that captures the defining features of the hair-style.

7.3 METHOD OVERVIEW

Figure 7.3 shows a diagram of our pipeline. Our algorithm takes as input several color images of a person with a given hair-style from a multi-view capture system (Section 7.4). Using these images, an initial smooth and coarse proxy geometry is obtained using multi-view stereo [23]. This proxy is then initialized with per-vertex color information from the input images, split into multiple frequencies (Section 7.5). Next, stylization operators are applied over the color information to achieve the desired level of abstraction (Section 7.6). From the stylized color information, new geometric details for the proxy are synthesized giving the artist control over the geometric appearance (Section 7.7). The final output of our method is a stylized mesh which abstracts the complexity of a real hair-style while still preserving its

defining features (Section 7.8). This mesh is suited for miniaturization and 3D printing (Figure 7.2, right).

7.4 DATA ACQUISITION

As a domain for computing stylized effects, our algorithm requires a coarse, low-resolution geometric proxy surface of the hair. This proxy may be generated with any 3D capture system that provides geometry and images, such as Li et al. [133]. In this work, we use a multi-view reconstruction setup as described in the following.

Ideally, the geometry proxy would be captured in a single shot from sufficient viewpoints to cover the full head. Since this approach requires a significant amount of camera and lighting equipment, we describe a technique that makes use of only limited hardware. We place ten digital SLR cameras in a quarter-spherical setup and photograph the subject under four consecutive orientations defined by 90-degree rotations (Figure 7.4).



Figure 7.4: Our capture setup consists of 10 cameras placed in a quarter-sphere and we capture 4 different orientations of the subject.

Although we take care not to change the hair-style or alter the facial expression drastically between the four takes, minor differences are tolerated since we require only a low-resolution proxy shape. We use a multi-view stereo reconstruction algorithm [23] to compute partial reconstructions from each of the four orientations (Figure 7.5, left). These reconstructions are aligned rigidly through the Iterative Closest Points (ICP) algorithm [27], and a single surface is obtained through Poisson reconstruction [114] of the combined point cloud. This surface represents our geometric proxy, including both hair and face (Figure 7.5, right). We manually identify the hair region through simple masking. The proxy will serve as a base for synthesizing stylized details in both shape and color. The four rigid transformations computed from ICP are also applied to the calibrated camera views to produce virtual cameras surrounding the proxy. In total we could obtain 40 virtual cameras; however, while such a dense view sampling is advantageous for multi-view stereo, we found that a subset of eight views is sufficient for hair

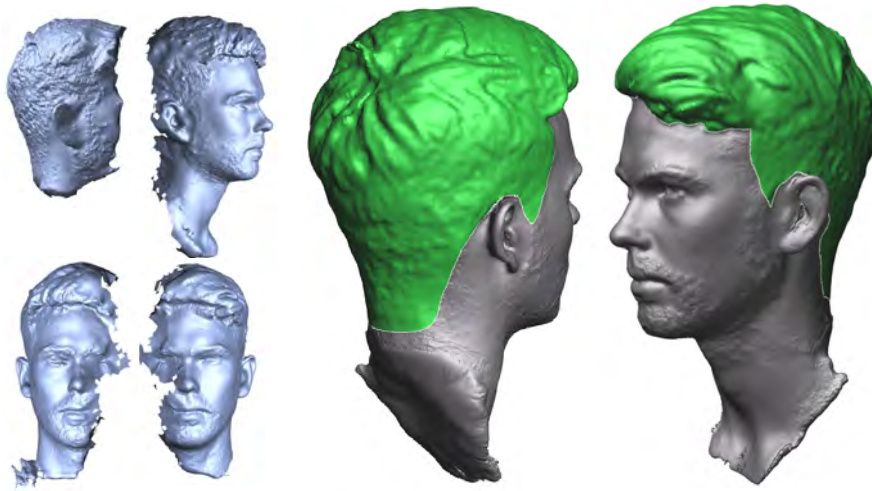


Figure 7.5: *Partial reconstructions from four sequential orientations (left) are combined to form a proxy geometry (right) shown with the hair masked in green.*

stylization once the geometry is reconstructed. To this end, we select the eight views that cover the hair volume from front, back, and both sides at two different elevations. We refer only to this subset of inputs in the rest of this Chapter.

7.5 COLOR INITIALIZATION

Once the proxy geometry has been obtained, the next step is to assign colors to its vertices. Contrary to face colors, which come from [23], coloring the hair requires special treatment because the proxy surface only poorly approximates the volumetric nature of hair and is not geometrically accurate, so there will inevitably be inconsistencies between different views. Furthermore, hair has a very complex appearance, with strong view dependent effects, such as specular reflection, translucency and occlusions. Obtaining a sharp and seamless colorization from multiple views is therefore extremely challenging. On the one hand, assigning colors per vertex based on a single-best viewpoint will lead to strong color seams (Figure 7.6 (a)). On the other hand, computing color by averaging multiple viewpoints will lead to blurry results (Figure 7.6 (b)). As shown in Figure 7.6 (c), color seams due to single-best viewpoint selection are most apparent in the lower frequencies, while they are masked in the high frequency bands. In contrast, blurriness due to color averaging has the biggest impact on the high-frequency components (Figure 7.6 (d)). Following from this observation, we separate color information into two frequency bands, low and high, using a Difference of Gaussians (DoG) filter. These color bands are processed separately according to their complementary nature and combined back together on the mesh (Figure 7.6 (e-f)).

Subsequent steps in the pipeline will make use of the intensity values to enhance contrast (Section 7.5.3) and guide geometric stylization (Section 7.7). To be able to directly operate on intensity values we convert the images from RGB to HSV color space as a preprocessing step.

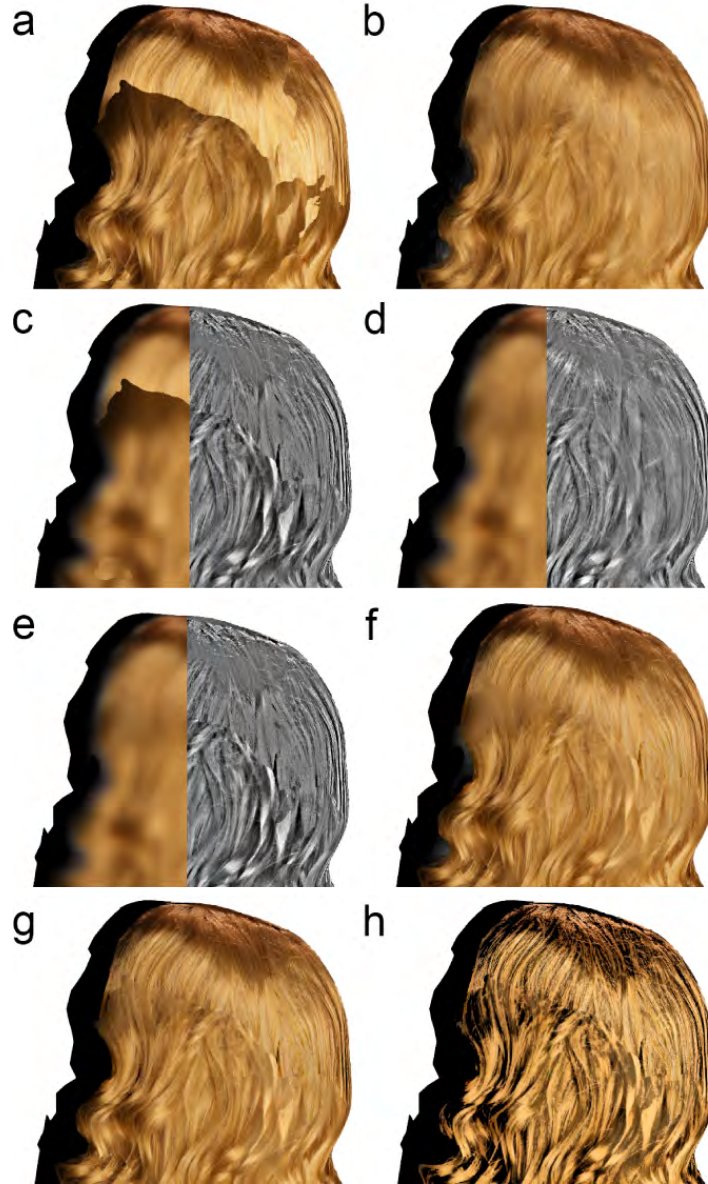


Figure 7.6: a) All-frequency colorization from the single-best view introduces strong seams but locally preserves sharpness. b) All-frequency colorization by averaging multiple views leads to blurry results, but reduces seams. c) Seams are most apparent in low-frequency bands, while they are masked in the high-frequency bands (shown in false color given they are offsets from the DoG). d) Loss of sharpness on the other hand is most apparent for high frequencies. e,f) We colorize the proxy by averaging low-frequency information from multiple views but sample high-frequency information from the single-best view, leveraging to the complementary nature of the two approaches. g) This approach might cause a change in contrast, which we equalize as described in Subsection 7.5.3. h) Contrast may be further boosted by scaling the intensity of the high-frequency band to help stylization. (Dataset courtesy of Luo et al. [140]).

7.5.1 Low-Frequency Color

For low-frequency color information, we assign a color \mathbf{c} to each vertex of the proxy mesh by averaging color samples \mathbf{c}^j from the set of views \mathcal{V} , weighted by their foreshortening angle ω^j

$$\mathbf{c} = \frac{1}{\sum_{j \in \mathcal{V}} \omega^j} \sum_{j \in \mathcal{V}} \omega^j \mathbf{c}^j, \quad (7.1)$$

with

$$\omega^j = \max(\langle \mathbf{n}, \mathbf{v}^j \rangle, 0),$$

$$\mathbf{c}^j = I^j(P^j \mathbf{p}),$$

where \mathbf{n} is the normal and \mathbf{p} is the position in world space of the vertex. I^j , \mathbf{v}^j and P^j are the low-frequency HSV image, view vector and projection matrix of view j respectively. Equation 7.1 is applied to each HSV channel separately. This approach effectively removes visible color seams and attenuates view-dependent color changes (Figure 7.6 (d)).

7.5.2 High-Frequency Color

As mentioned previously, the proxy geometry is only a coarse approximation of the hair volume, and averaging high-frequency color from multiple views as described for the low-frequency components would lead to blurry results and exhibit ghosting (Figure 7.6 (d)). We therefore sample the high-frequency details only from the single-best view j^* , which we consider to be the one with the highest foreshortening angle ω^j (Figure 7.6 (e)).

7.5.3 Contrast Equalization

Local contrast or dynamic range is directly related to incident light intensity. When averaging the low-frequency color from multiple views, the resulting intensity l might be very different from the low frequency intensity l^{j^*} in the single-best view used to extract the high-frequency component h^{j^*} due to view dependent effects; this results in a perceived loss of contrast when frequencies are re-combined (Figure 7.6 (f)). To alleviate this problem, we perceptually adjust the intensity value of the high frequency components h as

$$h = \xi h^{j^*} = \frac{l}{l^{j^*}} h^{j^*}. \quad (7.2)$$

In the case that the combined l is darker than l^{j^*} we use $1/\xi$ instead, since the goal here is to increase the contrast.

Figure 7.6 (g) shows the effect of equalizing the contrast when compared to Figure 7.6 (f). The impact is most apparent in areas around the seams, where the averaged low-frequency intensity l differs substantially from the intensity found in the single-best view l^{j^*} .

Finally, contrast may be boosted globally by uniformly scaling h to help color stylization (Figure 7.6 (h)).

7.6 COLOR STYLIZATION

As motivated in the introduction, our intent is to create a representation of the hair that can be miniaturized and printed. However, the level of detail of real hair is overwhelmingly high, and we thus need to find a means to reduce the complexity while preserving its defining features. We approach this problem by employing a specialized stylization filter.

Recently, Kyprianidis and Kang [121] have proposed an anisotropic, feature-preserving stylization filter for images, which exhibits interesting color stylization effects in hair regions. Their 2D filter can successfully reduce hair complexity, while still maintaining the overall appearance of a hair-style. Unfortunately, their method is not immediately suited for stylizing hair in 3D, since (i) it requires a regular 2D-domain, while our hair proxy constitutes an irregular manifold embedded in 3-space; and (ii) it is designed for single-image processing, while we operate within a multi-view scenario, where consistency between different views is essential. Nevertheless, the 2D filter of Kyprianidis and Kang contains several of the properties we desire in our application of stylized hair capture, and thus we propose to employ a similar mathematical foundation in a novel multi-view stylization algorithm that operates on irregular manifolds in 3D. Our approach will be to couple 2D and 3D stylization. Therefore, we will first provide an overview of the foundations for feature-preserving stylization in 2D (as presented by Kyprianidis and Kang), and then proceed to describe the key extensions that enable color stylization of hair in 3D.

7.6.1 Feature-Preserving 2D Filter

Feature-preserving or enhancing directional filters are based on three main components [111, 110]: The estimation of the local structure, an integration/smoothing operation to reduce complexity, and a sharpening operator to enhance the desired features. In the following, we describe how these components are achieved, focusing first on a single image.

LOCAL STRUCTURE ESTIMATION. Estimating the directionality of features in a 2D image has been studied before, with some specific applications to hair [166, 110, 140]. Given its fast computation and proven effectiveness in stylization contexts, we first compute the structure tensor $\mathcal{S}(\mathbf{x})$ [36] at each pixel \mathbf{x} . To attenuate noise, we filter the structure tensor using an isotropic 2D Gaussian [121] with standard deviation $\sigma_d = 4$.

Based on the structure tensor we introduce the *orientation tensor* $\mathcal{O}(\mathbf{x}) \in \mathbb{R}^{2 \times 2}$, which consists of the two eigenvectors of \mathcal{S} scaled by their respective eigenvalues. For our purposes $\mathcal{O}(\mathbf{x})$ is equivalent to $\mathcal{S}(\mathbf{x})$ but has some advantages when extending to multiple views, as we will see in Section 7.6.3. For ease of notation, we will omit the pixel (\mathbf{x}) in the orientation tensor and refer simply to \mathcal{O} in the rest of this discussion. The first row of the orientation tensor \mathcal{O}_g , corresponds to the image gradient, while the second row \mathcal{O}_t coincides with the tangent. The gradients and tangents will be used for directional smoothing and directional sharpening as described in the following. We will use $\vec{\mathcal{O}}_{g,t}$ to indicate the normalized gradient and tangent directions, respectively.

DIRECTIONAL SMOOTHING/INTEGRATION. Once we have a smooth orientation field, stylization starts by performing directional smoothing fol-

lowing the tangents $\vec{\mathcal{O}}_t$. This operation consists of a line integral convolution [38]. To obtain the resulting color $I'(\mathbf{x}_0)$ of a given pixel \mathbf{x}_0 , additional color samples $I(\mathbf{x}_k^\pm)$ are interpolated from both directions of the flow, and then averaged using a 1D Gaussian function $G(d_k)$ of standard deviation $\tilde{\sigma}_t$, where d_k represents the geodesic distance from the sampling point \mathbf{x}_k to \mathbf{x}_0 . We aim to filter with a standard deviation $\sigma_t = 10$, and account for local anisotropy by computing $\tilde{\sigma}_t$ as

$$\tilde{\sigma}_t = \frac{1}{4}\sigma_t \left(1 + \frac{\|\mathcal{O}_g\| - \|\mathcal{O}_t\|}{\|\mathcal{O}_g\| + \|\mathcal{O}_t\|} \right)^2. \quad (7.3)$$

We then have:

$$I'(x_0) = \frac{1}{w} \left[G(0)I(\mathbf{x}_0) + \sum_{k=1}^l G(d_k) [I(\mathbf{x}_k^+) + I(\mathbf{x}_k^-)] \right], \quad (7.4)$$

where $I(\mathbf{x})$ is the original color of a pixel \mathbf{x} , $w = \sum_{k=-l}^l G(d_k)$ is a normalization factor, and $l = \lceil 2\tilde{\sigma}_t \rceil$ represents the cut-off of $G(d_k)$. Equation 7.4 is applied to each HSV channel independently. Several ways to compute $I(\mathbf{x}_k^\pm)$ have been proposed [111, 121]; we have found that a second-order Runge-Kutta integration scheme provides the best results in our context. Figure 7.7 (center) shows the result after this step for an image patch of hair (left).



Figure 7.7: A schematic visualization of the smoothing and shocking operations. Left: Input image. Center: Line integral convolution following the tangents (in blue). Orange dots show integrated samples $I(\mathbf{x}_k^\pm)$ for pixel \mathbf{x}_0 (green). Right: Application of the shock operator in the direction of the gradients (shown in red).

DIRECTIONAL SHARPENING. To further stylize certain features, a directional *shock filter* can be applied in the direction of the gradient $\vec{\mathcal{O}}_g$ [165, 110]. This filter consists of morphological operations of dilation and erosion, which create ruptures between local maxima and minima, while enhancing flow-like patterns in the image [205]. For efficiency, it can be approximated by a min/max filter applied over a neighborhood of radius r , depending on the sign of a luminance-based Laplacian of Gaussian (LoG) with standard deviation σ_g [121]. In our case, we apply the LoG over the value channel in HSV space. We use $r = \sigma_g = 3$ for the results shown. Figure 7.7 (right) shows the final result, after both directional smoothing and sharpening.

7.6.2 Extension to Irregular Manifolds in 3-Space

This section describes how the stylization filter defined on a regular 2D image domain can be extended to operate on an irregular manifold embedded in 3-space. Color processing over manifolds has been studied before for applications like de-noising [187]. Here we extend a different set of operators that additionally deal with per-vertex information obtained from several input images, which need to be processed in a coherent way.

A possible alternative would be to compute a UV-parameterization that maps the manifold into 2-space, and then use the 2D stylization filter. However, we discarded such an approach for two reasons. First, the mesh proxy is far from being a developable surface, which would cause distortions in the mapping. These distortions would vary spatially and so the stylization operators would have to take them into account. Second, the proxy may have arbitrary genus, which can occur due to curls or ponytails (for example, see Figure 7.13), which would cause a discontinuous parameterization and require cutting the mesh - a challenging problem on its own. Furthermore, the stylization operators would have to be adapted to handle the discontinuity caused by such cuts. While these approaches can be found in previous work on texture synthesis [128, 203], we avoid these challenges by operating directly in 3-space.

As described in Section 7.5, we compute and store color information directly on the mesh, and will do the same for orientation tensors. Operations such as directional smoothing and shock filtering are then performed on a per-vertex basis, using the geodesic distance on the mesh surface. To do so, analogous to 2D directional smoothing, we require to look up the color value along a direction \mathbf{t} at a geodesic distance δ from the current vertex \mathbf{x} on the mesh. Since the mesh is not planar, the point $\mathbf{x}'_k = \mathbf{x} + \delta\mathbf{t}$ might be off surface and needs to be projected back on to get \mathbf{x}_k (Figure 7.8 (a)). If δ is larger than the local tessellation of the mesh, then \mathbf{x}_k will not reside in the one-ring neighborhood of \mathbf{x} and the process is repeated recursively using the closest point \mathbf{x}_e on the edge of the one-ring as new starting point and subtracting $\|\mathbf{x}_e - \mathbf{x}\|$ from $\delta\mathbf{t}$, effectively rolling down the original vector $\delta\mathbf{t}$ onto the mesh. Figure 7.8, b shows this roll-down schematically. Once \mathbf{x}_k is computed, the value is interpolated using barycentric coordinates.

The important parameter in this process is the step size δ , which should be similar to the local vertex density to avoid sampling issues. Fortunately, our meshes have very uniform vertex density allowing us to use a global δ that corresponds to the average edge length of the mesh.

7.6.3 Extension to Multiple Views

As motivated in Section 7.5, information from different views will be naturally misaligned since the proxy geometry is an approximation of the hair volume and the appearance of hair may differ substantially in different views. Consequently we merge low-frequency color information by averaging the contributions of the individual views to avoid color seams (Section 7.5.1), but we must combine high-frequency components using the single-best view to avoid blurring (Section 7.5.2). As a result, the high-frequency color information on the mesh will contain seams, and this can adversely impact orientation tensor computation if performed directly on the mesh. This is demonstrated in Figure 7.9 on a synthetic mesh patch. The high-frequency components are sampled from three different views

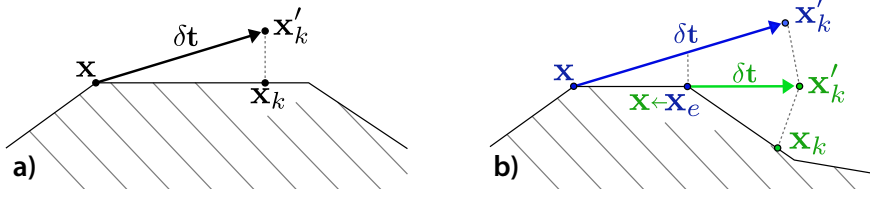


Figure 7.8: a) A point x_k' in the direction of the tangent t at distance δ from x may be off surface, and thus needs to be projected back on (x_k) in order to sample interpolated vertex information. b) If the sampling distance δ (blue) is larger than the local tessellation, the sampling point x_k will reside outside of the direct neighbourhood of x . In this case the process is repeated recursively using the closest point x_e on the edge of the one-ring as new starting point x (green) and adjusting δ accordingly, effectively rolling down the original vector δt onto the mesh.

(color-coded for better visualization) using the single-best view per vertex (Figure 7.9 (a)). The seams are clearly visible due to the change in orientation of the features. If we compute the orientation tensor on the mesh, it will inevitably follow such seams and so will do the directional smoothing (Figure 7.9 (b) and (f)). As a consequence, we will compute the orientation tensors from the images and transfer them to the mesh vertices.

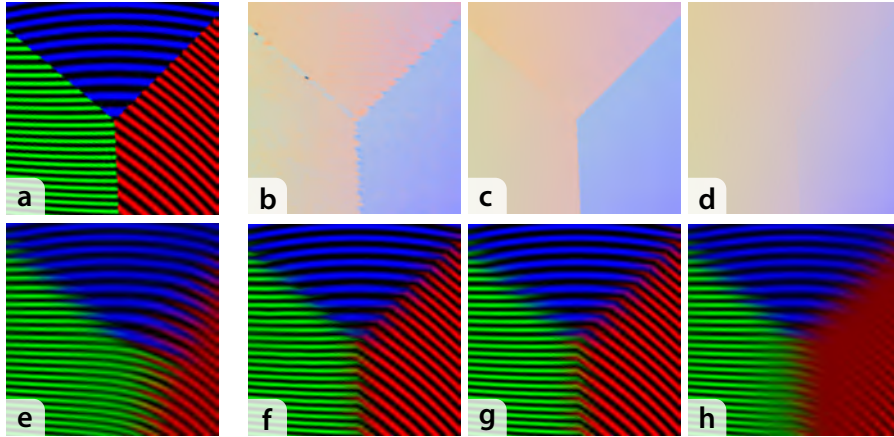


Figure 7.9: Synthetic example with high-frequency values from different views (color-coded for better visualization). a) Using only the single-best view will cause sharp transitions (blue-red, green-red). b) These transitions would adversely impact orientation tensor computation (colors encode the dominant orientation). c) Computing and transferring the tensor from the single-best view avoids these artifacts, but exhibits the same discontinuities. d) Using the proposed tensor combination yields a smooth tensor field. f,g,h) Tensor fields (b,c,d) applied to (a). The orientation is only continuous in (h) but the spatial frequencies are attenuated around seams as the colors don't match the tensor anymore. e) The final gathering step re-establishes this consistency and smoothly combines the structural elements from the different views without degradation.

Since the orientation tensors capture the continuous style of the hair, their spatial variation is essentially low-frequency, even though they are computed from the high-frequency components of the images. Analogously to vertex colorization discussed in Section 7.5, computing per-vertex orientation tensors from the single-best view would lead to strong seams (Figure 7.9 (c) and (g)), and thus we need to combine the tensors from multiple views

using an averaging scheme instead (Figure 7.9 (d)). In the following section we will explain how to combine them to produce a smooth tensor field on the mesh.

ORIENTATION TENSOR BACKPROJECTION. Similar to Paris et al. [167], transferring an orientation tensor \mathcal{O}^j from a view j to a point \mathbf{x} in space happens via backprojection, although our proxy geometry makes the process more straightforward. We project the normalized gradient direction $\vec{\mathcal{O}}_g$ onto the tangent plane at \mathbf{x} . The tangent direction $\vec{\mathcal{O}}_t$ is computed as the orthogonal vector in the tangent plane. Both vectors are re-scaled to their original magnitudes $\|\mathcal{O}_{g,t}^j\|$. Note that the sign of these vectors is arbitrary. We thus ensure consistency of the direction vectors of all views by reversing vectors that do not agree with the orientation of the single-best view. In the following, \mathcal{O} will denote this backprojected orientation tensor.

ORIENTATION TENSOR COMBINATION. Unlike structure tensors, which are oriented absolutely with respect to each view, orientation tensors from different views can be directly combined since they are oriented with the gradients. The combined gradient direction $\vec{\mathcal{O}}_g$ is thus computed as

$$\vec{\mathcal{O}}_g = \frac{1}{w} \sum_{j \in \mathcal{V}} \omega^j \theta^j \vec{\mathcal{O}}_g^j, \quad (7.5)$$

where \mathcal{V} is the set of views and $w = \sum_{j \in \mathcal{V}} \omega^j \theta^j$ is the normalization factor. The contributions are weighted using foreshortening ω as defined in 7.5.1 as well as the misalignment θ , which is computed as the discrepancy between the orientation tensors of the view j and the single-best view j^*

$$\theta^j = \langle \vec{\mathcal{O}}_g^j, \vec{\mathcal{O}}_g^{j^*} \rangle. \quad (7.6)$$

Given the gradient direction $\vec{\mathcal{O}}_g$ on the mesh, the tangent direction $\vec{\mathcal{O}}_t$ can again be derived since they are orthogonal.

Finally, the magnitudes of the vectors are computed as

$$\|\mathcal{O}_{g,t}\| = \frac{1}{w} \sum_{j \in \mathcal{V}} \omega^j \theta^j \|\mathcal{O}_{g,t}^j\|, \quad (7.7)$$

using the same weights and normalization as in Equation 7.5.

DIRECTIONAL GATHERING AND SMOOTHING. In the previous section we described how to combine the orientation tensors from the different views to produce a continuous orientation tensor field on the mesh (Figure 7.9 (d)). Unfortunately, since the high-frequency color information has been computed from the single-best view and the orientation tensors are produced by averaging multiple views, the two will be inconsistent around seams. This discrepancy will cause the directional smoothing to filter across color boundaries in these areas and produces blurry results (Figure 7.9 (h)). To overcome this problem, we propose an approach to update the color information, using a measure of the discrepancy between the gradient computed from the mesh colors $\vec{\mathcal{O}}'_g$ and the gradient from the combined multi-view orientation tensor $\vec{\mathcal{O}}_g$ as

$$\beta = 1 - \langle \vec{\mathcal{O}}_g, \vec{\mathcal{O}}'_g \rangle. \quad (7.8)$$

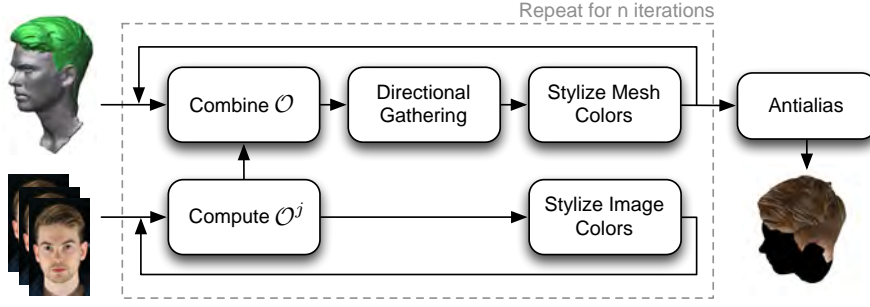


Figure 7.10: The proposed color stylization pipeline couples 2D image (bottom row) with 3D mesh stylization (top row). The final output is the mesh-proxy with stylized per vertex colors.

Given this measure, we introduce a *gathering* step, which ensures that color is consistent with the combined orientation tensor field. For a vertex with discrepancy β larger than a given threshold τ , we search along the tangent field in both directions for color samples with low discrepancies, using the same second-order Runge-Kutta scheme as for directional smoothing. The vertex color \mathbf{c} is then updated by linearly interpolating between those color samples based on their geodesic distance from the current vertex. The discrepancy β is then updated in the same way. Overall, discrepancy is usually lower than 0.1 in our examples, and we found a value of $\tau = 0.02$ produces sufficient consistency. The updated discrepancy β is further used as weighting factor to attenuate these samples during directional smoothing.

The final result is a smooth orientation tensor field with consistent coloring, the required basis for our stylizations (Figure 7.9 (e)) as it preserves the important structural elements from the different viewpoints while combining them together in a continuous way.

7.6.4 Coupled Mesh-View Stylization

Now that we have explained all the components required for multi-view hair stylization on a manifold in 3D, we present the complete stylization pipeline, as shown conceptually in Figure 7.10. It is important to see that the final result is based on coupled stylization of both the mesh and the input views. This is required to keep the orientation tensor, which is computed from the views, consistent with the colors on the mesh, which are stylized using this tensor.

The degree of stylization is emphasized by iteratively re-applying the method, which allows for direct artistic control. We use two to three iterations for the results presented here. After stylization, an antialiasing step is employed to refine the color transitions. The final output is the mesh-proxy with stylized per vertex colors.

7.7 GEOMETRY STYLIZATION

Up to this point, we have stylized high-frequency information in HSV color space over the mesh. We wish to also stylize the geometric details of the hair, such that they are consistent with the color style. To this end, we will compute spatially-varying surface offsets $d(\mathbf{x})$, and displace the vertices of the proxy geometry along the normal direction $\mathbf{n}(\mathbf{x})$ by $d(\mathbf{x})\mathbf{n}(\mathbf{x})$.

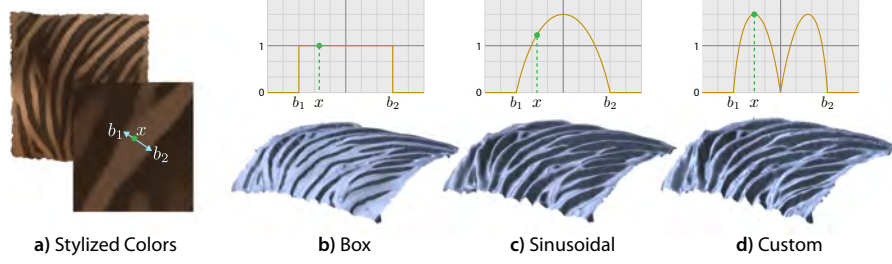


Figure 7.11: a) As a consequence of the shock filter, vertices along the gradient will have the same color within the boundaries $b_{1,2}$ of a wisp. This enables to compute the relative position x of a vertex \mathbf{x} within the cross-section and to apply different wisp profile functions (b,c,d) to artistically alter the geometric appearance.

Most of the perceived high-frequency contrast is encoded in the value channel while hue and saturation vary less substantially in general. This contrast can be largely attributed to shading changes caused by hair geometry, which is why we look to the stylized color intensity to determine the structural offsets. This is the underlying motivation for operating in HSV color space. Converting the high-frequency stylized intensity offsets $v(\mathbf{x})$ defined in the range of $[-1, 1]$ to displacements $d(\mathbf{x})$ is essentially a tone mapping application

$$d(\mathbf{x}) = \varphi \Phi(v(\mathbf{x})), \quad (7.9)$$

where φ is an artistic parameter that controls the strength of the shape stylization. For the results shown, φ varies between 3 and 6, and as tone mapping operator Φ , we apply a simple gamma correction with γ ranging between 0.5 and 0.7.

WISP PROFILES. The shock operation in color stylization has the property of creating uniform intensities (and thus uniform displacement values) for all the vertices within a shock, in the direction of the gradient (Figure 7.11, a), producing wisps with a locally flat appearance (Figure 7.11, c). While this is often sufficient to create successful hair stylizations, additional geometric styles can be obtained by modulating the displacements artistically.

Our idea is to find the relative position of each vertex within the cross-section of a wisp, and use this position to modulate displacement according to a user-defined *wisp profile*. To this end, for each vertex we search for discontinuities in intensity following the direction of the gradient \vec{O}_g (as done previously for the shock filter) by checking the relative change in intensity with respect to the starting vertex. Discontinuities in each direction mark the cross-sectional boundaries $b_{1,2}$ of the wisp (Figure 7.11, a). We then apply a 1D-window function to the geometry offsets to control its profile. In our results, we employ three different wisp profile functions - box, sinusoidal and custom - which produce different artistic looks (Figure 7.11, b-d). Defining additional profile functions is straightforward, providing a simple way to artistically control the style of the hair. To reduce aliasing, the modulated displacements can be smoothed in the direction of the tangents using the same directional smoothing operator explained in Section 7.6.

ALL-FREQUENCY COLOR AND GEOMETRIC INFORMATION. After applying the stylized high-frequency displacements, low- and high-frequency



Figure 7.12: *Our approach is versatile, and can be used for stylized capture of more than just human hair. Here we show additional examples of a stuffed toy dog (top) and fur (bottom).*

color information is combined and converted from HSV to RGB, obtaining the final colored and sculpted result, which is suitable for miniaturization and physical reproduction using traditional 3D printing processes.

7.8 RESULTS

We demonstrate the robustness and versatility of our method by capturing several subjects with widely varying hair-styles, and show that the proposed technique fits naturally in traditional pipelines for personalized figurine creation by physically reproducing several figurines with stylized hair.

A primary goal of our algorithm is to reconstruct an individual’s hair in a way that captures the identifying characteristics of the hair-style. Not only should the result be recognizable as a replica of a particular person, but it should be clearly recognized as a reconstruction of that person with their hair in a particular style. Our technique successfully achieves this goal, which we demonstrate by capturing the same two subjects each with four different hair-styles, as shown in Figures 7.13 and 7.14. Each hair-style is clearly recognizable in both shape and color, even though it is captured in a stylized and 3D-printable way.

We show the robustness of our approach in Figure 7.15, by capturing several different subjects with a wide range of hair-styles and hair colors. Our method can even stylize the reconstruction of facial hair, as seen for the subjects in the bottom two rows.

In Figure 7.12 we further demonstrate the versatility of the proposed technique, by capturing stylized reconstructions of non-human hair. In these examples, we capture a fur collar and a stuffed toy dog. Despite their complexity, both reconstructions are stylized manifold surfaces, suitable for 3D printing.

In our pipeline we describe a data acquisition step (Section 7.4) to build a low-resolution proxy surface. However, our hair stylization method can be applied to any proxy geometry with available camera views. This is demon-

strated in Figure 7.16, where we apply stylization to a dataset provided by Luo et al. [140]. As can be seen in the zoom regions, the benefit of our approach over Luo et al. [140] is that we capture the important structural and color elements of the hair-style, even starting from a proxy that only coarsely approximates the hair.

Different levels of stylization can be achieved by changing the shock filter width, as shown in Figure 7.19 (a-c). These styles can even be combined in any artistic way, which we show in Figure 7.19 (d). As mentioned, our technique can be applied naturally in physical reproduction pipelines for manufacturing personalized figurines. We miniaturized and 3D-printed several of our results, as shown in Figure 7.2 (right), Figure 7.3 (rightmost), Figure 7.20 and Figure 7.21. Physical printouts were created using a *ZCorp* printer. Our method can produce smaller figurines or handle lower-resolution printers by creating coarser hair wisps, as seen in Figure 7.19 (c). Figure 7.17 shows a comparison against a different high quality approach for the creation of personalized figurines.



Figure 7.13: Here we show four different captured hair-styles for one person. For each style we show the final result with both color and geometry stylization, as well as the result without color in order to best visualize the geometry stylization. Our method is able to faithfully capture the essence and identifying characteristics of each hair-style, even when the subject is the same.



Figure 7.14: Here we show four different captured hair-styles for one person. For each style we show the final result with both color and geometry stylization, as well as the result without color in order to best visualize the geometry stylization. Our method is able to faithfully capture the essence and identifying characteristics of each hair-style, even when the subject is the same.



Figure 7.15: Our stylized hair capture technique can reconstruct a wide range of hair-styles, including the facial hair on the subjects in the bottom two rows. Here we show one of the input images (left), the stylized geometry without color (center), and the final result with stylized shape and color (right).

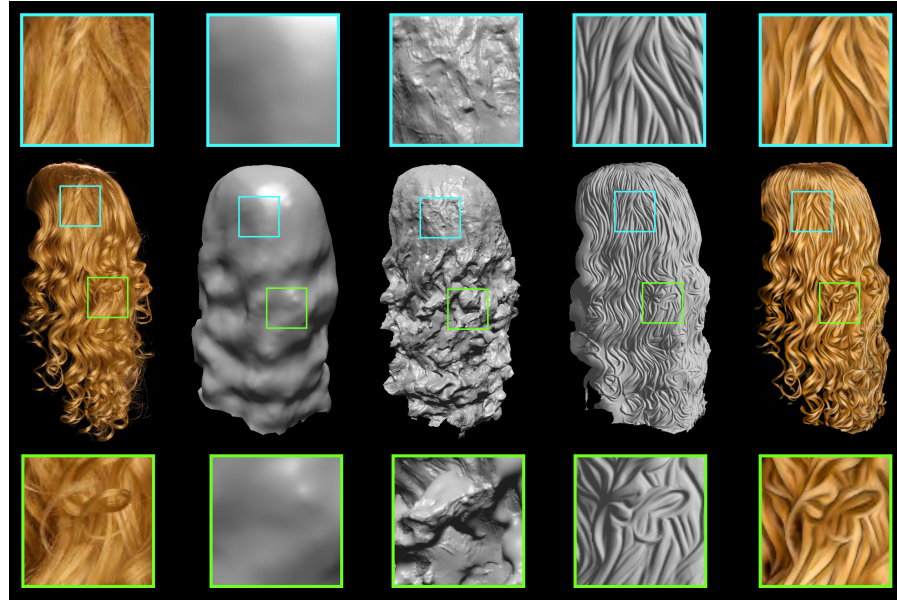


Figure 7.16: Although we do not share the same final goal, we show our method applied over a dataset from Luo et al. [140]. From left to right: one of the input views. Initial coarse reconstruction. Reconstructed results by Luo et al. Our stylized results starting from the same low-resolution proxy. In addition to sculpted geometry, our method generates also stylized color information. Our detailed stylization improvements over Luo et al. are clearly visible in the zoomed regions.



Figure 7.17: Since our method is reconstructing the actual hairstyle instead of approximating it with a limited set of templates, it does a better job capturing the identity of the subject, no matter how intricate his hairstyle is. Left: Reference input image. Center: Our reconstruction. Right: Result using Tena et al. approach [194].

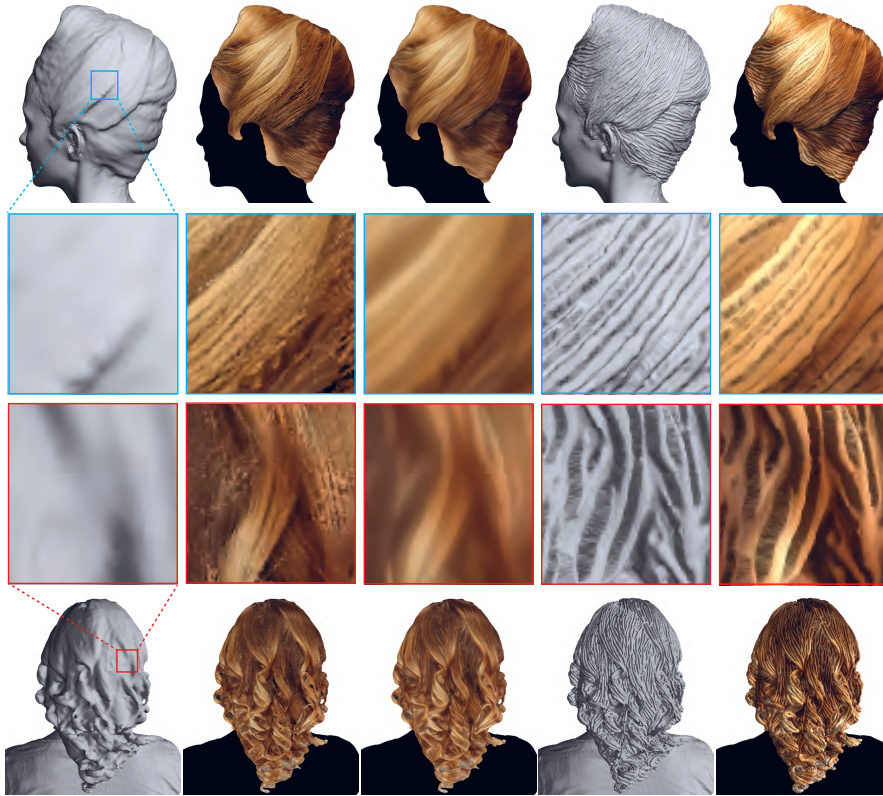


Figure 7.18: We illustrate the intermediate steps of our pipeline on two different datasets. From left to right: the input proxy geometry, color initialization on the proxy, color stylization alone, geometry stylization alone, and the final result with color and geometry stylization.

For a better assessment of our contribution to capturing and stylizing hair, we illustrate the intermediate steps of our pipeline in Figure 7.18, for a subset of our datasets. This figure shows the proxy mesh generated by Beeler et al. [23], the proxy with initialized colors, both our color and geometry stylization results on their own, and the final combined result of our algorithm. Our method increases the color and structural elements of the hair-style substantially compared to the initial proxy reconstruction.

We computed our results on a desktop PC with a six-core Intel 3930K CPU. It took our prototype implementation 4-7 hours to process meshes between 300K and 600K hair vertices; however, we believe this could be substantially improved using optimized data structures and by exploiting the extremely parallelizable nature of the problem on the GPU (following Kyprianidis et al. [121]). The parameters given in Subsection 7.6.1 were determined for input images of 2592×1728 pixels and transferred to the mesh domain for the 3D filter via re-projection onto the proxy geometry.

DISCUSSION. In our implementation, the total processing time and the overall quality of the results depend on the mesh discretization. It is preferable to operate on uniform triangulations with reasonable resolution. In particular, we found that between 300K and 600K hair vertices are sufficient to obtain sharp color and geometry details. Most multi-view capture techniques require many viewpoints when reconstructing complex objects with strong self-occlusions, which is why we used 40 viewpoints for the initial proxy reconstruction. For stylization, however, we found that a subset of

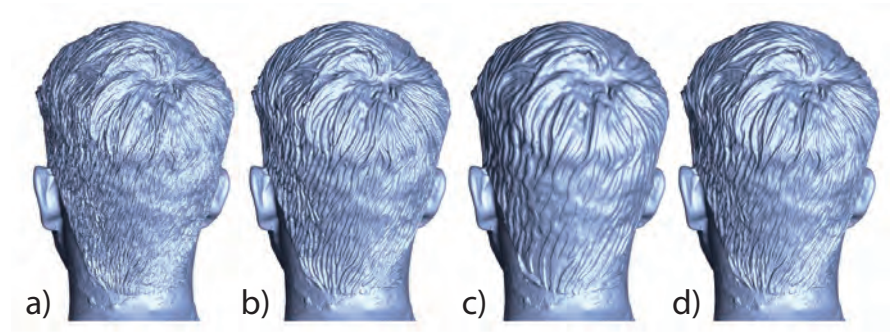


Figure 7.19: Here we show different levels of stylization (a-c) and a combination of styles (d) computed as $0.3(a) + 0.4(b) + 0.4(c)$.



Figure 7.20: Our method fits naturally in the physical reproduction pipeline for personalized figurines. A stylized capture result was printed with and without color.

about 8 views is sufficient. We expect the global shape of the hair to be provided by the proxy geometry, which might fail to reproduce some of the finer structures such as individual curls. Identifying and addressing such structures would be an interesting area for future research.

In conclusion, we present the first method for *stylized* hair capture, a technique to reconstruct an individual's actual hair-style in a manner suitable for physical reproduction. Our method generates hair as a closed-manifold surface, yet contains the structural and color elements stylized in a way that captures the defining characteristics of the hair-style.



Figure 7.21: Here we show the actual printouts of some examples from Figures 7.13 and 7.14.

Part V

CONCLUSION

CONCLUSIONS AND FUTURE WORK

In this Thesis we have presented a variety of practical solutions to problems related with the capture and manipulation of data, and the simulation of processes from the real world. Most of them feature novel combinations of traditionally separated fields: fluids simulation and image processing, image stylization and 3D reconstruction, or computational photography. They also share their relation with visual arts, creativity and human perception. Some of them are inspired by processes followed by artists in the real world, others became direct tools for content creation, while others take into account how people perceive the world in 2D and 3D.

In the following we summarize the conclusions for each of the three main parts of this thesis.

CREATIVE MANIPULATION OF DIGITAL PHOTOGRAPHS In Chapter 2 we presented a novel framework for digital image manipulations based on alternative analog photographic processes from the real world. We showcased different prototypes, which provide new engaging ways of manipulating digital images with a physical meaning, bringing a newfound sense of craftsmanship and serendipity to the process. Using the fluids solver from our framework, we can simulate other artistic processes as well, as shown in Chapter 3. There, a liquid layer is deposited on top of a virtual lens, which refracts the light coming from the (virtual) scene, obtaining very appealing distortions. We demonstrated the potential of this process as an interactive tool, with results that resemble the ones artists create in real world.

With this work, we introduced a novel exploratory approach to digital photography, something we believe was missing when compared with other artistic disciplines that have digital counterparts, like drawing or painting. We believe this ability to create engaging digital experiences by coupling physically-based simulations with natural user interfaces can appeal all kind of users, with potential uses for entertainment, educational or artistic purposes.

PERCEPTUAL RECONSTRUCTION OF 2D IMAGES Chapter 4 introduced SMAA, an anti aliasing post process filter for real time, that produces natural results taking into account how humans perceive aliasing and sharpness. It performs a comprehensive morphological analysis of the image to analytically compute pixel coverage, to produce smooth or sharp results where required. Our method is extended to deal with additional samples coming from spatial and temporal multi/supersampling, converging to the supersampled references, and achieving better temporal stability in subpixel features and shading. Our solution is modular, allowing the creation of different presets that can target different scenarios and power budgets. This work also showcases how taking into account the human visual system can help producing feasible and pleasing reconstructions in the absence of additional data. SMAA is currently a well known and well regarded anti aliasing solution, featured in lots of video games for consoles and PC. It still can be considered the state of the art in anti aliasing filters for real time, with more recent solutions based on similar concepts and approaches [161, 162].

In Chapter 5, we made use of the Nokia N900 smartphone, the first commercially available platform for mobile computational photography, to demonstrate practical pipelines for mobile HDR imaging. Our approach captures a standard bracketed sequence of low dynamic range exposures, to compose an extended dynamic range image based on perceptual cues, with everything being happening on-device. By the time the paper was published, cameras found in mobile devices were not as evolved as today, mostly because of the software support. Nowadays, HDR imaging is a standard feature in smart phones and tablets following the same principles exposed here, camera APIs are more capable and computing power is increasing at an impressive pace. Coupled with additional movement and positioning sensors, this makes them great platforms for practical mobile computational photography beyond HDR.

3D RECONSTRUCTION AND STYLIZATION In another example of (mobile) computational photography, in Chapter 6 we presented a new and efficient algorithm to estimate depth maps from a small focal stack of images. We base our approach on the assumption that pixel blur measures, while spatially inconsistent for a single focal distance, can behave in a consistent way for each single pixel through a focal stack. This allows the fitting of an analytical model of the defocus blur to recover an estimation of the distance of each point in the scene to the camera. We believe our method presents an interesting tradeoff between accuracy and speed when compared with previous works. The modularity of our approach makes it straightforward to study alternatives to the chosen algorithms at each step, so it can greatly benefit from separate advances that occur in the future. In our opinion, mobile computational photography is posed to have an increasingly critical role over the years. Having the ability to control the capture process in a way that complimentary information about the scene can be easily acquired, can have a big impact on existing image processing algorithms that currently have to estimate such information from a single image (often the illest-posed scenario) [137, 153], or use specialized hardware [20].

Finally, in Chapter 7 we presented the first method for *stylized* hair capture, a technique to reconstruct an individual's actual hair-style in a manner suitable for physical reproduction. Our method generates hair as a closed-manifold surface, yet contains the structural and color elements stylized in a way that captures the defining characteristics of the hair-style. Our results show a pleasing hand-crafted look and feel, in some cases very similar to what artists sculpt for commercial figurines and fine arts. Since its publication, more recent methods also reconstruct printable hairstyles [42] producing impressive results from a single input image. However, they aim for different applications and it is not clear how it would extend to full 3D models, or scale for different print sizes. Apart from creative purposes, we believe abstraction and stylization algorithms can also become powerful tools to deal with noisy data or to hallucinate details in pleasing ways to the human visual system.

FUTURE WORK Apart from the future work described for each of the previous projects in their corresponding Chapters, in the following we describe briefly ongoing related projects.

Related with the manipulation of digital images, we are currently exploring novel ways of using 4D light fields [132] for creative purposes. In contrast to traditional 2D photos, 4D light fields store additional angular information by not integrating (or disintegrating) the light rays hitting the sensor. While commercial light field cameras have been available already for a few years, there is still a small number of editing tools for them trying to efficiently extend concepts from 2D to 4D [103, 30, 31, 13, 216, 32, 69], or finding the preferred user interfaces for point-based editions in 4D [102, 144]. However, we believe that this additional angular information should allow editing capabilities and tools beyond what is possible with 2D images. Early examples like refocusing of perspective shift have been demonstrated in the past [96], but (un)fortunately, they can also be simulated to some extent by using combinations of color images and/or depth information [21, 16, 209].

We think about 4D light fields as a collection of 3D light rays capturing the radiance of the original scene, which naturally resonates with ray tracing. In our work, we are focusing on the simulation of what we call *light field rephotography*, that is, obtaining a new 2D photo of a scene through a different optical arrangement. While this concept has been timidly demonstrated in the past [92], the focus has been on the simulation of the optical systems themselves, and not dealing with the singularities of working with structured 4D light fields and the limited amount of ray samples they store. In contrast, our work focuses on a more general interaction of light fields with arbitrary optical arrangements, from traditional lens designs to more creative ones involving a variety of refractive objects and atypical configurations that expose the sensor directly. The core of our approach is the extension of ray differentials [94] to 4D light fields, namely *light field differentials*, which provide two main benefits: i) the cost of tracing bundles of nearby rays is greatly reduced; ii) the analytical description of the light transport provided by the ray differentials enables more sophisticated access and interpolation of the original samples.

We believe this framework has the potential to enable novel tools and uses like the illusion of interactively re-photographing a previously captured scene, or the simulation of optical setups different from the fixed one used during the capture to either correct optical distortions or to obtain them.

Following our work on 3D reconstruction, we are currently developing a new solution to the more general problem of 3D surface reconstruction from point clouds [26]. One of the most popular approaches for that matter is posing the problem as a Poisson Equation, where an oriented point cloud is converted into a vector field, so an implicit indicator function can be reconstructed elegantly [114]. Then, such implicit function is converted into a mesh [52]. Although current solutions are quite robust to noise while producing very detailed results, they also aim for watertight solutions that have lots of advantages, but also critical limitations. One of the most important ones is related with the ability to model non-manifold arrangements with very thin structures, like the ones usually found in clothes and other thin objects like sheets of paper. In such cases, a watertight solution implies a binary partitioning of the space around such features (inside/outside), impossible to model them properly. To overcome this limitation, we are solving the same Poisson Equation, but using a regional level set [219, 115]. Regional level sets can represent explicitly an arbitrary number of regions, providing

a more comprehensive partitioning of the space around the point cloud. The challenge here is to use this very limited mathematical element to solve such equation. Once done, we can apply a number of PDE-based manipulations to post process the initial solution, obtaining a non-manifold implicit function, which requires also extensions to current polygonization algorithms.

We expect our solution to improve the robustness of 3D scanning pipelines that fail to produce proper reconstructions in those cases, specially when the point cloud is already capturing such thin features and arrangements.

PERSONAL CONCLUSIONS During the development of this thesis I have been able to work on a variety of projects, on different topics and fields. This has allowed me to expand my knowledge, and to explore novel solutions that arose from unusual intersections between fields like fluids simulation, image processing, image stylization or 3D reconstruction. I think that is one of the most interesting aspects of the work I have done: instead of becoming an expert in one single field, I have ended up being knowledgeable in several different ones. This is something that was not planned from the beginning, and I am very thankful for it. Be it because of our multidisciplinary lab, or the different internships that I have done, working on so different problems teaches you to become better at evaluating the contributions of previous works, and putting them into perspective with respect to their own field and others. Over time, I have found this open-mindedness very useful when trying to find the "right" approach for each specific problem, specially with some recurring ones where traditional approaches leave small room for improvement, or when assessing the work of others.

Through these years, I have had the chance to work with people from different parts of the world, and from different backgrounds. Working in such diverse teams has been very enlightening: although all of us are researchers in the end, each field has its own subtleties, and I feel very lucky to have been able to learn from all of them to broaden my vision about research as a whole. Something similar has happened with the internships I have done. Apart from the different company cultures that I was able to experience, doing research inside the industry is something very valuable that I believe complements greatly the academic side. It is amazing being surrounded by experts around you from very different backgrounds, which paves the way for very powerful synergies. I was also lucky to work in projects whose results can be appreciated by all kinds of people, which makes everything more rewarding for me.

Related with this last point, during this thesis I have realized the importance of dissemination of results to make knowledge advance. From academic publications and oral presentations, to high level educational talks for everyone, it is very important to hone your communication skills to actually connect with your audience, so they can receive your message to build on top of it, expand their knowledge about the technologies that surround them or the work we researchers actually do, or inspire the following generations of researchers.

All in all, during these years I have acquired a very valuable set of skills not only useful for research, but also for my personal life, helping to shape the person I am today. It was not easy, but it was fun!

BIBLIOGRAPHY

- [1] 3D-U. ThreeDee-You. <http://www.3d-u.es/>, 2010. Accessed: January 18, 2014.
- [2] 3DSYSTEMS. 3DMe Photobooth. <http://www.3dsystems.com/>, 2013. Accessed: January 10, 2014.
- [3] ADAMS, A., GELFAND, N., AND PULLI, K. Viewfinder alignment. *Computer Graphics Forum* 27, 2 (2008), 597–606.
- [4] ADAMS, A., JACOBS, D. E., DOLSON, J., TICO, M., PULLI, K., TALVALA, E.-V., AJDIN, B., VAQUERO, D., LENSCH, H. P. A., HOROWITZ, M., PARK, S. H., GELFAND, N., BAEK, J., MATUSIK, W., AND LEVOY, M. The franken-camera: an experimental platform for computational photography. *ACM Trans. Graph.* 29 (July 2010), 29:1–29:12.
- [5] ADELSBERGER, R., ZIEGLER, R., AND GROSS, M. Spatially adaptive photographic flash, 2008.
- [6] ADELSON, E. H., AND BERGEN, J. R. *The Plenoptic Function and the Elements of Early Vision*. MIT Press, 1991, pp. 3–20.
- [7] AKELEY, K. Reality engine graphics. In *SIGGRAPH '93: Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (1993), pp. 109–116.
- [8] AMD. Morphological anti-aliasing, 2010.
- [9] AMD. EQAA modes for amd 6900 series graphics cards. Tech. rep., AMD, 2011.
- [10] ANDERSSON, J. 5 major challenges in interactive rendering. In *ACM SIGGRAPH Courses* (2010).
- [11] ANDERSSON, J. DirectX 11 Rendering in Battlefield 3. In *Game Developers Conference 2011* (2011).
- [12] ANDREEV, D. Directionally localized anti-aliasing (DLAA). In *Game Developers Conference 2011* (2011).
- [13] AO, H., ZHANG, Y., JARABO, A., MASIA, B., LIU, Y., GUTIERREZ, D., AND DAI, Q. *Light Field Editing Based on Reparameterization*. Springer International Publishing, Cham, 2015, pp. 601–610.
- [14] APPLE. ios 4.1 update. <http://arstechnica.com/apple/news/2010/09/hdr-photography-with-iphone-4-and-ios-41.ars>, September 2010.
- [15] AUZINGER, T., MUSIALSKI, P., PREINER, R., AND WIMMER, M. Non-Sampled Anti-Aliasing. In *Vision, Modeling & Visualization* (2013), M. Bronstein, J. Favre, and K. Hormann, Eds., The Eurographics Association.
- [16] BAE, S., AND DURAND, F. Defocus magnification. *Comput. Graph. Forum* 26, 3 (2007), 571–579.

- [17] BAE, S., PARIS, S., AND DURAND, F. Two-scale tone management for photographic look. *ACM Trans. Graph.* 25 (2006).
- [18] BAILEY, S. W., ECHEVARRIA, J. I., BODENHEIMER, B., AND GUTIERREZ, D. Fast depth from defocus from focal stacks. *The Visual Computer* (2014), 1–12.
- [19] BAO, Y., AND QI, Y. Realistic hair modeling from a hybrid orientation field. *The Visual Computer* 32, 6 (2016), 729–738.
- [20] BARRON, J. T., AND MALIK, J. Intrinsic scene properties from a single rgb-d image. In *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition* (Washington, DC, USA, 2013), CVPR '13, IEEE Computer Society, pp. 17–24.
- [21] BARSKY, B. A., HORN, D. R., KLEIN, S. A., PANG, J. A., AND YU, M. *Computational Science and Its Applications — ICCSA 2003: International Conference Montreal, Canada, May 18–21, 2003 Proceedings, Part III*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003, ch. Camera Models and Optical Systems Used in Computer Graphics: Part II, Image-Based Techniques, pp. 256–265.
- [22] BAUSZAT, P., EISEMANN, M., AND MAGNOR, M. Guided image filtering for interactive high-quality global illumination. *Computer Graphics Forum* 30, 4 (2011), 1361–1368.
- [23] BEELER, T., BICKEL, B., BEARDSLEY, P., SUMNER, B., AND GROSS, M. High-quality single-shot capture of facial geometry. *ACM Trans. Graph.* 29, 3 (2010), 40:1–40:9.
- [24] BEELER, T., BICKEL, B., NORIS, G., BEARDSLEY, P., MARSCHNER, S., SUMNER, R. W., AND GROSS, M. Coupled 3D reconstruction of sparse facial hair and skin. *ACM Trans. Graph.* 31, 4 (2012), 117:1–117:10.
- [25] BEELER, T., BRADLEY, D., AND ECHEVARRIA, J. I. Capturing and stylizing hair for 3d fabrication, Mar. 10 2016. US Patent 20160071316 A1.
- [26] BERGER, M., TAGLIASACCHI, A., SEVERSKY, L. M., ALLIEZ, P., LEVINE, J. A., SHARF, A., AND SILVA, C. T. State of the Art in Surface Reconstruction from Point Clouds. In *Eurographics 2014 - State of the Art Reports* (2014), S. Lefebvre and M. Spagnuolo, Eds., The Eurographics Association.
- [27] BESL, P. J., AND MCKAY, N. D. A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. Mach. Intell.* 14, 2 (1992), 239–256.
- [28] BIRI, V. Morphological antialiasing and topological reconstruction. In *GRAPP 2011* (2011).
- [29] BIRI, V., HERUBEL, A., AND DEVERLY, S. Practical morphological antialiasing on the GPU. In *ACM SIGGRAPH 2010 Talks* (2010), SIGGRAPH '10, pp. 45:1–45:1.
- [30] BIRKLBAUER, C., AND BIMBER, O. Light-field retargeting. *Comput. Graph. Forum* 31, 2pt1 (May 2012), 295–303.
- [31] BIRKLBAUER, C., AND BIMBER, O. Panorama light-field imaging. *Computer Graphics Forum* 33, 2 (2014), 43–52.

- [32] BIRKLBAUER, C., SCHEDL, D. C., AND BIMBER, O. Nonuniform spatial deformation of light fields by locally linear transformations. *ACM Trans. Graph.* 35, 5 (June 2016), 156:1–156:12.
- [33] BJORKE, K. Color controls. *GPU Gems* (2004).
- [34] BLOOMENTHAL, J. Edge inference with applications to antialiasing. *SIGGRAPH Comput. Graph.* 17 (1983), 157–162.
- [35] BRIDSON, R. *Fluid Simulation for Computer Graphics*. A K Peters/CRC Press, 2008.
- [36] BROX, T., BOOMGAARD, R., LAUZE, F., WEIJER, J., WEICKERT, J., MRÁZEK, P., AND KORNPORST, P. Adaptive structure tensors and their applications. In *Visualization and Processing of Tensor Fields*, Mathematics and Visualization. 2006, pp. 17–47.
- [37] BURT, P. J., EDWARD, AND ADELSON, E. H. The laplacian pyramid as a compact image code. *IEEE Transactions on Communications* 31 (1983), 532–540.
- [38] CABRAL, B., AND LEEDOM, L. C. Imaging vector fields using line integral convolution. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (1993), SIGGRAPH '93, pp. 263–270.
- [39] CALDERERO, F., AND CASELLES, V. Recovering relative depth from low-level features without explicit t-junction detection and interpretation. *International Journal of Computer Vision* (2013), 1–31.
- [40] CAO, Y., FANG, S., AND WANG, F. Single image multi-focusing based on local blur estimation. In *Image and Graphics (ICIG), 2011 Sixth International Conference on* (2011), pp. 168–175.
- [41] CAO, Y., FANG, S., AND WANG, Z. Digital multi-focusing from a single photograph taken with an uncalibrated conventional camera. *Image Processing, IEEE Transactions on* 22, 9 (Sept 2013), 3703–3714.
- [42] CHAI, M., LUO, L., SUNKAVALLI, K., CARR, N., HADAP, S., AND ZHOU, K. High-quality hair modeling from a single portrait photo. *ACM Trans. Graph.* 34, 6 (Oct. 2015), 204:1–204:10.
- [43] CHAI, M., SHAO, T., WU, H., WENG, Y., AND ZHOU, K. Autohair: Fully automatic hair modeling from a single image. *ACM Trans. Graph.* 35, 4 (July 2016), 116:1–116:12.
- [44] CHAI, M., WANG, L., WENG, Y., JIN, X., AND ZHOU, K. Dynamic hair manipulation in images and videos. *ACM Trans. Graph.* 32, 4 (2013), 75:1–75:8.
- [45] CHAI, M., WANG, L., WENG, Y., YU, Y., GUO, B., AND ZHOU, K. Single-view hair modeling for portrait manipulation. *ACM Trans. Graph.* 31, 4 (2012), 116:1–116:8.
- [46] CHAJDAS, M. G., MCGUIRE, M., AND LUEBKE, D. Subpixel reconstruction antialiasing for deferred shading. In *Symposium on Interactive 3D Graphics and Games* (2011), pp. 15–22.

- [47] CHO, H., KIM, S. J., AND LEE, S. Single-shot high dynamic range imaging using coded electronic shutter. *Computer Graphics Forum* 33, 7 (2014), 329–338.
- [48] CHU, N. S.-H., AND TAI, C.-L. Moxi: real-time ink dispersion in absorbent paper. *ACM Trans. Graph.* 24, 3 (2005).
- [49] CURTIS, C. J., ANDERSON, S. E., SEIMS, J. E., FLEISCHERY, K. W., AND SALESIN, D. H. Computer-generated watercolor. In *Proceedings of SIGGRAPH '97* (1997).
- [50] D-TECH ME. D-tech me. <http://www.insidethemagic.net/tag/d-tech-me/>, 2012. Accessed: January 18, 2014.
- [51] DAVIES, L., AND STRUGAR, F. Conservative morphological anti-aliasing, 2014.
- [52] DE ARAÚJO, B. R., LOPES, D. S., JEPP, P., JORGE, J. A., AND WYVILL, B. A survey on implicit surface polygonization. *ACM Comput. Surv.* 47, 4 (May 2015), 60:1–60:39.
- [53] DE PEREYRA, A. MLAA: Efficiently moving antialiasing from the gpu to the cpu. Tech. rep., Intel, 2011.
- [54] DEBEVEC, P. E., AND MALIK, J. Recovering high dynamic range radiance maps from photographs. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1997), SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., pp. 369–378.
- [55] DEERING, M., WINNER, S., SCHEDIWY, B., DUFFY, C., AND HUNT, N. The triangle processor and normal vector shader: a vlsi system for high performance graphics. *SIGGRAPH Comput. Graph.* 22 (June 1988), 21–30.
- [56] DROBOT, M. Hybrid Recosntruction Antialiasing. In *GPU Pro 6*. AK Peters Ltd., 2015.
- [57] DURAND, F., AND DORSEY, J. Fast bilateral filtering for the display of high-dynamic-range images. *ACM Trans. Graph.* 21 (July 2002), 257–266.
- [58] ECHEVARRIA, J. I., BRADLEY, D., GUTIERREZ, D., AND BEELER, T. Capturing and stylizing hair for 3d fabrication. *ACM Transactions on Graphics (SIGGRAPH 2014)* 33, 4 (2014).
- [59] ECHEVARRIA, J. I., AND GUTIERREZ, D. Mobile computational photography: Exposure fusion on the nokia n900. Grupo Português de Computação Gráfica.
- [60] ECHEVARRIA, J. I., MUNOZ, A., SERON, F., AND GUTIERREZ, D. Screen-space rendering of translucent objects. In *Congreso Español de Informática Gráfica* (2010).
- [61] ECHEVARRIA, J. I., WILENSKY, G., KRISHNASWAMY, A., KIM, B., AND GUTIERREZ, D. Computational simulation of alternative photographic processes. *Computer Graphics Forum (Proc. EGSR 2013)* 32, 4 (2013).
- [62] ENGEL, W. Light pre-pass renderer, 2008.

- [63] FAVARO, P. Recovering thin structures via nonlocal-means regularization with application to depth from defocus. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on* (2010), pp. 1133–1140.
- [64] FAVARO, P., AND SOATTO, S. *3-D Shape Estimation and Image Restoration: Exploiting Defocus and Motion-Blur*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
- [65] FAVARO, P., SOATTO, S., BURGER, M., AND OSHER, S. J. Shape from defocus via diffusion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 30, 3 (2008), 518–531.
- [66] FOSTER, N., AND FEDKIW, R. Practical animation of liquids. In *Proceedings of ACM SIGGRAPH 2001* (2001).
- [67] FOSTER, N., AND METAXAS, D. Realistic animation of liquids. *Graph. Models and Image Processing* 58 (1996).
- [68] FUJITA, S. *Organic Chemistry of Photography*. Springer, 2004.
- [69] GARCES, E., ECHEVARRIA, J. I., ZHANG, W., WU, H., ZHOU, K., AND GUTIERREZ, D. Intrinsic light fields, 2016.
- [70] GEIGEL, J., AND MUSGRAVE, F. K. A model for simulating the photographic development process on digital images. In *Proceedings of SIGGRAPH '97* (1997), ACM Press/Addison-Wesley Publishing Co.
- [71] GELDREICH, R., PRITCHARD, M., AND BROOKS, J. Deferred lighting and shading. In *Game Developers Conference 2004* (2004).
- [72] GERMAN, D. M., AND RIGAU, J. Improving scans of black and white photographs by recovering the print maker’s artistic intent. *Computers and Graphics* 33, 4 (2009).
- [73] GOOGLE. Android 2.2 update. <http://developer.android.com/sdk/android-2.2-highlights.html>, September 2010.
- [74] GRABLI, S., SILLION, F., MARSCHNER, S. R., AND LENGYEL, J. E. Image-based hair capture by inverse lighting. In *Proc. Graphics Interface* (2002), pp. 51–58.
- [75] GRANADOS, M., KIM, K. I., TOMPKIN, J., AND THEOBALT, C. Automatic noise modeling for ghost-free hdr reconstruction. *ACM Trans. Graph.* 32, 6 (Nov. 2013), 201:1–201:10.
- [76] GRITZ, L., AND D’EON, E. The importance of being linear. *GPU Gems* 3 (2008).
- [77] GURNEY, R. W., AND MOTT, N. F. The theory of the photolysis of silver bromide and the photographic latent image. *Proceedings of the Royal Society of London. Series A - Mathematical and Physical Sciences* 164, 917 (1938).
- [78] GUTIERREZ, D., SERON, F. J., LOPEZ-MORENO, J., SANCHEZ, M. P., FANDOS, J., AND REINHARD, E. Depicting procedural caustics in single images. *ACM Trans. Graph.* 27, 5 (Dec. 2008), 120:1–120:9.
- [79] HARGREAVES, S. Deferred shading. In *Game Developers Conference 2004* (2004).

- [80] HASINOFF, S. W., AND KUTULAKOS, K. N. Coded focal stack photography. *International Journal of Computer Vision* 81, 1 (2009), 82–104.
- [81] HASINOFF, S. W., AND KUTULAKOS, K. N. Confocal stereo. *International Journal of Computer Vision* 81, 1 (2009), 82–104.
- [82] HE, K., SUN, J., AND TANG, X. Guided image filtering. In *Proceedings of the 11th European conference on Computer vision: Part I* (Berlin, Heidelberg, 2010), ECCV'10, Springer-Verlag, pp. 1–14.
- [83] HE, X., AND LUO, L.-S. Lattice Boltzmann Model for the Incompressible Navier-Stokes Equation. *Journal of Statistical Physics* 88, 3 (1997).
- [84] HECHT, E. *Optics*, 3rd ed. Addison-Wesley, Aug. 1997.
- [85] HEIDE, F., STEINBERGER, M., TSAI, Y.-T., ROUF, M., PAJAK, D., REDDY, D., GALLO, O., LIU, J., HEIDRICH, W., EGAZARIAN, K., KAUTZ, J., AND PULLI, K. Flexisp: A flexible camera image processing framework. *ACM Trans. Graph.* 33, 6 (Nov. 2014), 231:1–231:13.
- [86] HELD, R. T., COOPER, E. A., O'BRIEN, J. F., AND BANKS, M. S. Using blur to affect perceived distance and size. *ACM Trans. Graph.* 29, 2 (2010).
- [87] HERRERA, T. L., ZINKE, A., AND WEBER, A. Lighting hair from the inside: A thermal approach to hair reconstruction. *ACM Trans. Graph.* 31, 6 (2012), 146:1–146:9.
- [88] HERTZMANN, A., JACOBS, C. E., OLIVER, N., CURLESS, B., AND SALESIN, D. H. Image analogies. In *Proceedings of SIGGRAPH '01* (2001), ACM.
- [89] HU, H., AND DE HAAN, G. Adaptive image restoration based on local robust blur estimation. In *Proceedings of the 9th international conference on Advanced concepts for intelligent vision systems* (Berlin, Heidelberg, 2007), ACIVS'07, Springer-Verlag, pp. 461–472.
- [90] HU, L., MA, C., LUO, L., AND LI, H. Robust hair capture using simulated examples. *ACM Trans. Graph.* 33, 4 (2014).
- [91] HU, L., MA, C., LUO, L., AND LI, H. Single-view hair modeling using a hairstyle database. *ACM Trans. Graph.* 34, 4 (July 2015), 125:1–125:9.
- [92] HULLIN, M. B., HANIKA, J., AND HEIDRICH, W. Polynomial Optics: A construction kit for efficient ray-tracing of lens systems. *Computer Graphics Forum (Proceedings of EGSR 2012)* 31, 4 (2012).
- [93] HURTER, F., AND DRIFFIELD, V. Photo-chemical investigations and a new method of the determination of the sensitiveness of photographic plates. *Journal of the Society of Chemical Industry* 9 (1890).
- [94] IGEHY, H. Tracing ray differentials. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1999), SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., pp. 179–186.
- [95] IOURCHA, K., YANG, J. C., AND POMIANOWSKI, A. A directionally adaptive edge anti-aliasing filter. In *Proceedings of the Conference on High Performance Graphics 2009* (2009), pp. 127–133.

- [96] ISAKSEN, A., MCMILLAN, L., AND GORTLER, S. J. Dynamically reparameterized light fields. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., pp. 297–306.
- [97] ISSHIKI, T., AND KUNIEDA, H. Efficient anti-aliasing algorithm for computer generated images. In *ISCAS (4)* (1999), pp. 532–535.
- [98] JACOBSON, Q. B. *Chemical Pictures: The Wet Plate Collodion Photography Book*. Quinn Jacobson Photography and Studio Q, 2010.
- [99] JACOBSON, R. E., RAY, S. F., ATTERIDGE, G. G., AND AXFORD, N. R. *The Manual of Photography: Photographic and Digital Imaging*, 9th ed. Focal Press, 2000.
- [100] JAKOB, W., MOON, J. T., AND MARSCHNER, S. Capturing hair assemblies fiber by fiber. *ACM Trans. Graph.* 28, 5 (2009), 164:1–164:9.
- [101] JAMES, C. *The Book of Alternative Photographic Processes*. Delmar Cengage Learning, 2009.
- [102] JARABO, A., MASIA, B., BOUSSEAU, A., PELLACINI, F., AND GUTIERREZ, D. How do people edit light fields? *ACM Trans. Graph.* 33, 4 (July 2014), 146:1–146:10.
- [103] JARABO, A., MASIA, B., AND GUTIERREZ, D. Efficient propagation of light field edits. In *Proc. of the V Ibero-American Symposium in Computer Graphics* (2011), SIACG 2011, pp. 75–80.
- [104] JIMENEZ, J., ECHEVARRIA, J. I., OAT, C., AND GUTIERREZ, D. *GPU Pro 2*. AK Peters Ltd., 2011, ch. Practical and Realistic Facial Wrinkles Animation.
- [105] JIMENEZ, J., ECHEVARRIA, J. I., SOUSA, T., AND GUTIERREZ, D. Smaa: Enhanced morphological antialiasing. *Computer Graphics Forum (Proc. EUROGRAPHICS 2012)* 31, 2 (2012).
- [106] JIMENEZ, J., GUTIERREZ, D., YANG, J., RESHETOV, A., DEMOREUILLE, P., BERGHOFF, T., PERTHUIS, C., YU, H., MCGUIRE, M., LOTTES, T., MALAN, H., PERSSON, E., ANDREEV, D., AND SOUSA, T. Filtering approaches for real-time anti-aliasing. In *ACM SIGGRAPH Courses* (2011).
- [107] JIMENEZ, J., MASIA, B., ECHEVARRIA, J. I., NAVARRO, F., AND GUTIERREZ, D. *GPU Pro 2*. AK Peters Ltd., 2011, ch. Practical Morphological Anti-Aliasing.
- [108] JIMENEZ, J., MASIA, B., ECHEVARRIA, J. I., NAVARRO, F., AND GUTIERREZ, D. Practical Morphological Anti-Aliasing. In *GPU Pro 2*. AK Peters Ltd., 2011, pp. 95–113.
- [109] JIMENEZ, J., SUNDSTEDT, V., AND GUTIERREZ, D. Screen-space perceptual rendering of human skin. *ACM Trans. Appl. Percept.* 6, 4 (Oct. 2009), 23:1–23:15.
- [110] KANG, H., AND LEE, S. Shape-simplifying image abstraction. *Computer Graphics Forum* 27, 7 (2008), 1773–1780.

- [111] KANG, H., LEE, S., AND CHUI, C. K. Flow-based image abstraction. *IEEE Transactions on Visualization and Computer Graphics* 15, 1 (2009), 62–76.
- [112] KANG, S. B., UYTENDAELE, M., WINDER, S., AND SZELISKI, R. High dynamic range video. *ACM Trans. Graph.* 22 (July 2003), 319–325.
- [113] KASS, M., AND MILLER, G. Rapid, stable fluid dynamics for computer graphics. In *Comput. Graph. (Proc. of SIGGRAPH 90)* (1990), vol. 24.
- [114] KAZHDAN, M., BOLITHO, M., AND HOPPE, H. Poisson surface reconstruction. In *Symposium on Geometry Processing* (2006), pp. 61–70.
- [115] KIM, B. Multi-phase fluid simulations using regional level sets. In *ACM SIGGRAPH Asia 2010 Papers* (New York, NY, USA, 2010), SIGGRAPH ASIA '10, ACM, pp. 175:1–175:8.
- [116] KING, S. Ultraviolet light sources for printing with the alternative processes. <http://unblinkingeye.com/Articles/Light/light.html>, 2001. [Online; accessed January-2013].
- [117] KNUTSSON, H., AND WESTIN, C.-F. Normalized and differential convolution: Methods for interpolation and filtering of incomplete and uncertain data. In *Proceedings of Computer Vision and Pattern Recognition ('93)* (New York City, USA, June 16–19 1993), pp. 515–523.
- [118] KONG, W., TAKAHASHI, H., AND NAKAJIMA, M. Generation of 3D hair model from multiple pictures. In *Proc. Multimedia Modeling* (1997), pp. 183–196.
- [119] KOONCE, R. Deferred Shading in Tabula Rasa. In *GPU Gems 3*. Addison Wesley, 2007, pp. 429–457.
- [120] KYPRIANIDIS, J. E., COLLOMOSSE, J., WANG, T., AND ISENBERG, T. State of the Art: A taxonomy of artistic stylization techniques for images and video. *IEEE Transactions on Visualization and Computer Graphics* 19, 5 (2013), 866–885.
- [121] KYPRIANIDIS, J. E., AND KANG, H. Image and video abstraction by coherence-enhancing filtering. *Computer Graphics Forum* 30, 2 (2011), 593–602.
- [122] LABS, D. How do sensors see colors? <http://www.dxomark.com/index.php/About/In-depth-measurements/Measurements/Color-sensitivity>, November 2012.
- [123] LAERHOVEN, T. V., , AND REETH, F. V. Real-time simulation of watery paint. *Computer Animation and Virtual Worlds* 16 (2005).
- [124] LAERHOVEN, T. V., LIESENBORG, J., AND REETH, F. V. Real-time watercolor painting on a distributed paper model. In *Proceedings of CGI* (2004).
- [125] LAPUT, G., CHEN, X. A., AND HARRISON, C. 3d printed hair: Fused deposition modeling of soft strands, fibers, and bristles. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (New York, NY, USA, 2015), UIST '15, ACM, pp. 593–597.

- [126] LEE, I.-H., SHIM, S.-O., AND CHOI, T.-S. Improving focus measurement via variable window shape on surface radiance distribution for 3d shape reconstruction. *Optics and Lasers in Engineering* 51, 5 (2013), 520–526.
- [127] LEE, S., EISEMANN, E., AND SEIDEL, H.-P. Real-time lens blur effects and focus control. *ACM Trans. Graph.* 29 (2010).
- [128] LEFEBVRE, S., AND HOPPE, H. Appearance-space texture synthesis. *ACM Trans. Graph.* 25, 3 (2006), 541–548.
- [129] LENAERTS, T., ADAMS, B., AND DUTRÉ, P. Porous flow in particle-based fluid simulations. *ACM Trans. Graph.* 27, 3 (2008).
- [130] LEVIN, A., FERGUS, R., DURAND, F., AND FREEMAN, W. Image and depth from a conventional camera with a coded aperture. *ACM Transactions on Graphics, SIGGRAPH 2007 Conference Proceedings, San Diego, CA* (2007).
- [131] LEVOY, M. Experimental platforms for computational photography. *IEEE Computer Graphics and Applications* 30 (2010), 81–87.
- [132] LEVOY, M., AND HANRAHAN, P. Light field rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1996), SIGGRAPH '96, ACM, pp. 31–42.
- [133] LI, C., SU, S., MATSUSHITA, Y., ZHOU, K., AND LIN, S. Bayesian depth-from-defocus with shading constraints. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on* (June 2013), pp. 217–224.
- [134] LI, Y., SHARAN, L., AND ADELSON, E. H. Compressing and companding high dynamic range images with subband architectures. *ACM Trans. Graph.* 24 (July 2005), 836–844.
- [135] LIN, X., SUO, J., WETZSTEIN, G., DAI, Q., AND RASKAR, R. Coded focal stack photography. In *IEEE International Conference on Computational Photography* (2013).
- [136] LOPEZ-MORENO, J., JIMENEZ, J., HADAP, S., REINHARD, E., ANJYO, K., AND GUTIERREZ, D. Stylized depiction of images based on depth perception. In *Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2010), NPAR '10, ACM, pp. 109–118.
- [137] LOPEZ-MORENO, J., JIMENEZ, J., HADAP, S., REINHARD, E., ANJYO, K., AND GUTIERREZ, D. Stylized depiction of images based on depth perception. In *Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2010), NPAR '10, ACM, pp. 109–118.
- [138] LOTTES, T. FXAA. Tech. rep., NVIDIA, 2011.
- [139] LUO, L., LI, H., AND RUSINKIEWICZ, S. Structure-aware hair capture. *ACM Trans. Graph.* 32, 4 (2013), 76:1–76:12.
- [140] LUO, L., LI, H., SYLVAIN, WEISE, T., PAULY, M., AND RUSINKIEWICZ, S. Multi-view hair capture using orientation fields. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2012), pp. 1490–1497.

- [141] MAHMOOD, M. T., AND CHOI, T.-S. Nonlinear approach for enhancement of image focus volume in shape from focus. *Image Processing, IEEE Transactions on* 21, 5 (2012), 2866–2873.
- [142] MALIK, A. Selection of window size for focus measure processing. In *Imaging Systems and Techniques (IST), 2010 IEEE International Conference on* (2010), pp. 431–435.
- [143] MANN, PICARD, MANN, S., AND PICARD, R. W. On being ‘undigital’ with digital cameras: Extending dynamic range by combining differently exposed pictures. In *Proceedings of IS&T* (1995), pp. 442–448.
- [144] MASIA, B., JARABO, A., AND GUTIERREZ, D. Favored workflows in light field editing. In *Proceedings of CGVCVIP ’14* (2014).
- [145] MASIA, B., PRESA, L., CORRALES, A., AND GUTIERREZ, D. Perceptually optimized coded apertures for defocus deblurring. *Computer Graphics Forum* 31, 6 (2012), 1867–1879.
- [146] MASIA, B., WETZSTEIN, G., ALIAGA, C., RASKAR, R., AND GUTIERREZ, D. Display Adaptive 3D Content Remapping. *Computers & Graphics, Special Issue on Advanced Displays* 37, 8 (2013), 983–996.
- [147] McNAMARA, A., MANIA, K., AND GUTIERREZ, D. Perception in graphics, visualization, virtual environments and animation. In *SIGGRAPH Asia 2011 Courses* (New York, NY, USA, 2011), SA ’11, ACM, pp. 17:1–17:137.
- [148] MEES, K. *The theory of the photographic process*. The MacMillan Company, 1942.
- [149] MEHRA, R., ZHOU, Q., LONG, J., SHEFFER, A., GOOCH, A., AND MITRA, N. J. Abstraction of man-made shapes. *ACM Trans. Graph.* 28, 5 (2009), 137:1–137:10.
- [150] MERTENS, T., KAUTZ, J., AND VAN REETH, F. Exposure fusion: A simple and practical alternative to high dynamic range photography. *Computer Graphics Forum* 28, 1 (2009), 161–171.
- [151] MITCHELL, J. W. Photographic sensitivity. *Reports on Progress in Physics* 20, 1 (1957).
- [152] MORENO-NOGUER, F., BELHUMEUR, P. N., AND NAYAR, S. K. Active refocusing of images and videos. In *ACM SIGGRAPH 2007 papers* (New York, NY, USA, 2007), SIGGRAPH ’07, ACM.
- [153] MUNOZ, A., ECHEVARRIA, J. I., SERON, F., LOPEZ-MORENO, J., GLENCROSS, M., AND GUTIERREZ, D. BSSRDF estimation from single images. *Computer Graphics Forum* 30 (2011), 455–464.
- [154] MUNOZ, A., ECHEVARRIA, J. I., SERON, F. J., AND GUTIERREZ, D. Convolution-based simulation of homogeneous subsurface scattering. *Computer Graphics Forum* 30, 8 (2011), 2279–2287.
- [155] NAH, J.-H., KI, S., LIM, Y., PARK, J., AND SHIN, C. Axa: Adaptive approximate anti-aliasing. In *ACM SIGGRAPH 2016 Posters* (New York, NY, USA, 2016), SIGGRAPH ’16, ACM, pp. 51:1–51:2.

- [156] NAMBOODIRI, V., AND CHAUDHURI, S. Recovery of relative depth from a single observation using an uncalibrated (real-aperture) camera. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* (2008), pp. 1–6.
- [157] NAN, L., SHARE, A., XIE, K., WONG, T.-T., DEUSSEN, O., COHEN-OR, D., AND CHEN, B. Conjoining gestalt rules for abstraction of architectural drawings. *ACM Trans. Graph.* 30, 6 (2011). 185:1–185:10.
- [158] NAVARRO, F., CASTILLO, S., SERÓN, F. J., AND GUTIERREZ, D. Perceptual considerations for motion blur rendering. *ACM Trans. Appl. Percept.* 8, 3 (Aug. 2011), 20:1–20:15.
- [159] NAYAR, S., AND NAKAGAWA, Y. Shape from focus. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 16, 8 (1994), 824–831.
- [160] NEHAB, D., SANDER, P. V., LAWRENCE, J., TATARCHUK, N., AND ISIDORO, J. R. Accelerating real-time shading with reverse reprojection caching. In *Proceedings of the 22nd ACM SIGGRAPH/EUROGRAPHICS Symposium on Graphics Hardware* (2007), pp. 25–35.
- [161] NVIDIA. Txaa. <http://www.geforce.com/hardware/technology/txaa>, September 2012.
- [162] NVIDIA. Mfaa. <http://www.geforce.com/hardware/technology/mfaa>, September 2014.
- [163] OLIVEIRA, M. M., AND BRAUWERS, M. Real-time refraction through deformable objects. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games* (2007), ACM.
- [164] OMOTE 3D. Omote 3D. <http://www.omote3d.com/>, 2012. Accessed: January 18, 2014.
- [165] OSHER, S., AND RUDIN, L. I. Feature-oriented image enhancement using shock filters. *SIAM Journal on Numerical Analysis* 27, 4 (1990), 919–940.
- [166] PARIS, S., BRICEÑO, H. M., AND SILLION, F. X. Capture of hair geometry from multiple images. *ACM Trans. Graph.* 23, 3 (2004), 712–719.
- [167] PARIS, S., CHANG, W., KOZHUSHNYAN, O. I., JAROSZ, W., MATUSIK, W., ZWICKER, M., AND DURAND, F. Hair photobooth: Geometric and photometric acquisition of real hairstyles. *ACM Trans. Graph.* 27, 3 (2008), 30:1–30:9.
- [168] PENTLAND, A. P. A new sense for depth of field. *Pattern Analysis and Machine Intelligence, IEEE Transactions on PAMI-9*, 4 (1987), 523–531.
- [169] PERLIN, K. An image synthesizer. In *Proceedings of SIGGRAPH '85* (1985), ACM.
- [170] PERSSON, E. Geometric post-process anti-aliasing, 2011.
- [171] PERTUZ, S., PUIG, D., AND GARCIA, M. A. Analysis of focus measure operators for shape-from-focus. *Pattern Recognition* 46, 5 (2013), 1415–1432.

- [172] PETSCHNIGG, G., SZELISKI, R., AGRAWALA, M., COHEN, M., HOPPE, H., AND TOYAMA, K. Digital photography with flash and no-flash image pairs. In *ACM SIGGRAPH 2004 Papers* (New York, NY, USA, 2004), SIGGRAPH '04, ACM, pp. 664–672.
- [173] POCKETSIZE ME. PocketSize Me. <http://www.pocketsizeme.ch/>, 2013. Accessed: January 18, 2014.
- [174] POLICARPO, F., OLIVEIRA, M. M., AND COMBA, J. A. L. D. Real-time relief mapping on arbitrary polygonal surfaces. *ACM Trans. Graph.* 24, 3 (2005).
- [175] POULI, T., AND REINHARD, E. Progressive histogram reshaping for creative color transfer and tone reproduction. In *Proceedings of NPAR '10* (2010), ACM.
- [176] PRESS, W. H., TEUKOLSKY, S. A., VETTERLING, W. T., AND FLANNERY, B. P. *Numerical recipes: The art of scientific computing*, 3rd ed. Cambridge university press, 2007.
- [177] RAMSDEN, J. J. Computing photographic response curves. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 406, 1830 (1986).
- [178] REINHARD, E., STARK, M., SHIRLEY, P., AND FERWERDA, J. Photographic tone reproduction for digital images. *ACM Trans. Graph.* 21, 3 (2002).
- [179] REINHARD, E., STARK, M., SHIRLEY, P., AND FERWERDA, J. Photographic tone reproduction for digital images. *ACM Trans. Graph.* 21 (July 2002), 267–276.
- [180] RESHETOV, A. Morphological antialiasing. In *Proceedings of the Conference on High Performance Graphics 2009* (2009), pp. 109–116.
- [181] RUDMAN, T. *The Photographer's Toning Book: The Definitive Guide*. Am-photo Books, 2003.
- [182] SEN, P., KALANTARI, N. K., YAESOUBI, M., DARABI, S., GOLDMAN, D. B., AND SHECHTMAN, E. Robust patch-based hdr reconstruction of dynamic scenes. *ACM Trans. Graph.* 31, 6 (Nov. 2012), 203:1–203:11.
- [183] SERRANO, A., HEIDE, F., GUTIERREZ, D., WETZSTEIN, G., AND MASIA, B. Convolutional sparse coding for high dynamic range imaging. *Computer Graphics Forum* 35, 2 (2016).
- [184] SHIM, S.-O., AND CHOI, T.-S. A fast and robust depth estimation method for 3d cameras. In *Consumer Electronics (ICCE), 2012 IEEE International Conference on* (2012), pp. 321–322.
- [185] SHISHKOVTSOV, O. Deferred Shading in S.T.A.L.K.E.R. In *GPU Gems 2*. Addison Wesley, 2005, pp. 143–166.
- [186] SKLADNIKIEWITZ, P., HERTEL, D., AND SCHMIDT, I. The wet collodion process – a scientific approach. *The Journal of Imaging Science and Technology* 42, 5 (1998).
- [187] SOCHEN, N., DERICHE, R., AND PEREZ, L. The beltrami flow over implicit manifolds. In *Ninth IEEE International Conference on Computer Vision (ICCV)* (2003), vol. 3, pp. 832–839.

- [188] SOUSA, T. Vegetation Procedural Animation and Shading in Crysis. In *GPU Gems 3*. Addison Wesley, 2007, pp. 373–385.
- [189] SRIKANTHA, A., AND SIDIBÉ, D. Ghost detection and removal for high dynamic range images: Recent advances. *Image Commun.* 27, 6 (July 2012), 650–662.
- [190] STAM, J. Stable fluids. In *Proceedings of SIGGRAPH '99* (1999), ACM Press/Addison-Wesley Publishing Co.
- [191] STURM, J., BYLOW, E., KAHL, F., AND CREMERS, D. CopyMe3D: Scanning and printing persons in 3D. In *Pattern Recognition*, vol. 8142. 2013, pp. 405–414.
- [192] SUBBARAO, M., AND CHOI, T. Accurate recovery of three-dimensional shape from image focus. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 17, 3 (1995), 266–274.
- [193] SUCCI, S. *The Lattice Boltzmann Equation: For Fluid Dynamics and Beyond*. Numerical Mathematics and Scientific Computation Series. Clarendon Press, 2001.
- [194] TENA, J. R., MAHLER, M., BEELER, T., GROSSE, M., YEH, H., AND MATTHEWS, I. Fabricating 3D figurines with personalized faces. *IEEE Computer Graphics and Applications* 33, 6 (2013), 36–46.
- [195] TOCCI, M. D., KISER, C., TOCCI, N., AND SEN, P. A versatile hdr video production system. *ACM Trans. Graph.* 30, 4 (July 2011), 41:1–41:10.
- [196] VAN OVERVELD, C. W. A. M. Application of morphological filters to tackle discretization artifacts. *Vis. Comput.* 8 (1992), 217–232.
- [197] VAQUERO, D., GELFAND, N., TICO, M., PULLI, K., AND TURK, M. Generalized autofocus. In *IEEE Workshop on Applications of Computer Vision (WACV'11)* (Kona, Hawaii, January 2011).
- [198] WANG, B., YU, Y., WONG, T.-T., CHEN, C., AND XU, Y.-Q. Data-driven image color theme enhancement. *ACM Trans. Graph.* 29, 6 (2010).
- [199] WANG, B., YU, Y., AND XU, Y.-Q. Example-based image color and tone style enhancement. *ACM Trans. Graph.* 30, 4 (2011).
- [200] WANG, H., MILLER, G., AND TURK, G. Solving general shallow wave equations on surfaces. In *Proceedings of Symposium on Computer Animation (SCA)* (2007).
- [201] WANG, L., YU, Y., ZHOU, K., AND GUO, B. Example-based hair geometry synthesis. *ACM Trans. Graph.* 28, 3 (2009), 56:1–56:9.
- [202] WATANABE, M., AND NAYAR, S. Rational Filters for Passive Depth from Defocus. *International Journal on Computer Vision* 27, 3 (May 1998), 203–225.
- [203] WEI, L.-Y., AND LEVOY, M. Texture synthesis over arbitrary manifold surfaces. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (2001), SIGGRAPH '01, pp. 355–360.
- [204] WEI, Y., OFEK, E., QUAN, L., AND SHUM, H.-Y. Modeling hair from multiple views. *ACM Trans. Graph.* 24, 3 (2005), 816–820.

- [205] WEICKERT, J. Coherence-enhancing shock filters. In *Pattern Recognition*, vol. 2781 of *Lecture Notes in Computer Science*. 2003, pp. 1–8.
- [206] WILENSKY, G., KRISHNASWAMY, A., AND ECHEVARRIA, J. I. Systems and methods for simulating the effects of liquids on a camera lens, Nov. 3 2015. US Patent 9,176,662.
- [207] YANG, L., NEHAB, D., SANDER, P. V., SITTHI-AMORN, P., LAWRENCE, J., AND HOPPE, H. Amortized supersampling. *ACM Trans. Graph.* 28 (2009), 135:1–135:12.
- [208] YANG, L., SANDER, P. V., LAWRENCE, J., AND HOPPE, H. Antialiasing recovery. *ACM Trans. Graph.* 30 (2011).
- [209] YANG, Y., LIN, H., YU, Z., PARIS, S., AND YU, J. Virtual dslr: High quality dynamic depth-of-field synthesis on mobile platforms. *Electronic Imaging 2016*, 18 (2016, publication date =).
- [210] YE, G., GARCES, E., LIU, Y., DAI, Q., AND GUTIERREZ, D. Intrinsic video and applications. *ACM Transactions on Graphics (SIGGRAPH 2014)* 33, 4 (2014).
- [211] YOUNG, P. Coverage sampled antialiasing. Tech. rep., NVIDIA, 2006.
- [212] YU, D., MEI, R., LUO, L.-S., AND SHYY, W. Viscous flow computations with the method of lattice boltzmann equation. *Progress in Aerospace Sciences* 39, 5 (2003).
- [213] YUKSEL, C., HOUSE, D. H., AND KEYSER, J. Wave particles. *ACM Trans. Graph.* 26, 3 (2007).
- [214] YUMER, M. E., AND KARA, L. B. Co-abstraction of shape collections. *ACM Trans. Graph.* 31, 6 (2012), 166:1–166:11.
- [215] ZELL, E., ALIAGA, C., JARABO, A., ZIBREK, K., GUTIERREZ, D., McDONNELL, R., AND BOTSCH, M. To stylize or not to stylize?: The effect of shape and material stylization on the perception of computer-generated faces. *ACM Trans. Graph.* 34, 6 (Oct. 2015), 184:1–184:12.
- [216] ZHANG, F. L., WANG, J., SHECHTMAN, E., ZHOU, Z. Y., SHI, J. X., AND HU, S. M. Plenopatch: Patch-based plenoptic image manipulation. *IEEE Transactions on Visualization and Computer Graphics PP*, 99 (2016), 1–1.
- [217] ZHAO, H., SHI, B., FERNANDEZ-CULL, C., YEUNG, S. K., AND RASKAR, R. Unbounded high dynamic range photography using a modulo camera. In *Computational Photography (ICCP), 2015 IEEE International Conference on* (April 2015), pp. 1–10.
- [218] ZHAO, Q., TAN, P., DAI, Q., SHEN, L., WU, E., AND LIN, S. A closed-form solution to retinex with nonlocal texture constraints. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 34, 7 (July 2012), 1437–1444.
- [219] ZHENG, W., YONG, J.-H., AND PAUL, J.-C. Simulation of bubbles. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2006), SCA '06, Eurographics Association, pp. 325–333.

- [220] ZHOU, C., COSSAIRT, O., AND NAYAR, S. Depth from Diffusion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Jun 2010).
- [221] ZHUO, S., AND SIM, T. On the recovery of depth from a single defocused image. In *Computer Analysis of Images and Patterns*, X. Jiang and N. Petkov, Eds., vol. 5702 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2009, pp. 889–897.
- [222] ZHUO, S., AND SIM, T. Defocus map estimation from a single image. *Pattern Recogn.* 44, 9 (Sept. 2011), 1852–1858.
- [223] ZIMMER, H., BRUHN, A., AND WEICKERT, J. Freehand hdr imaging of moving scenes with simultaneous resolution enhancement. *Computer Graphics Forum* 30, 2 (2011), 405–414.

This document was typeset using the typographical look-and-feel classicthesis developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography *"The Elements of Typographic Style"*.

Final Version as of September 15, 2016 (classicthesis version 1.0).

In this Thesis we focus on providing practical solutions to different problems related with the capture and manipulation of data, and simulation of processes from the real world.

We dedicate one part of this work to explore novel use cases for tablet devices, leveraging the natural user interaction they enable to prototype more engaging image capture and editing tools. We present a novel framework for the simulation of the craftsmanship involved in analog photography techniques and other interesting optical manipulations.

A second part is devoted to image reconstruction algorithms that share the use of perceptual cues to compensate for the missing data. First, we introduce our *SMAA* anti aliasing filter for real time applications, where a comprehensive morphological analysis is performed to provide smooth but sharp results. Next, a simple procedure is described to capture extended dynamic range images with mobile devices, using computational photography techniques.

The last part deals with the capture of 3D data from the real world. We present a new *depth-from-defocus* algorithm for obtaining detailed depth maps of scenes. Finally, we describe the first system for stylized capture of hair, producing results suitable for 3D fabrication inspired by the abstraction process performed by sculptors.

