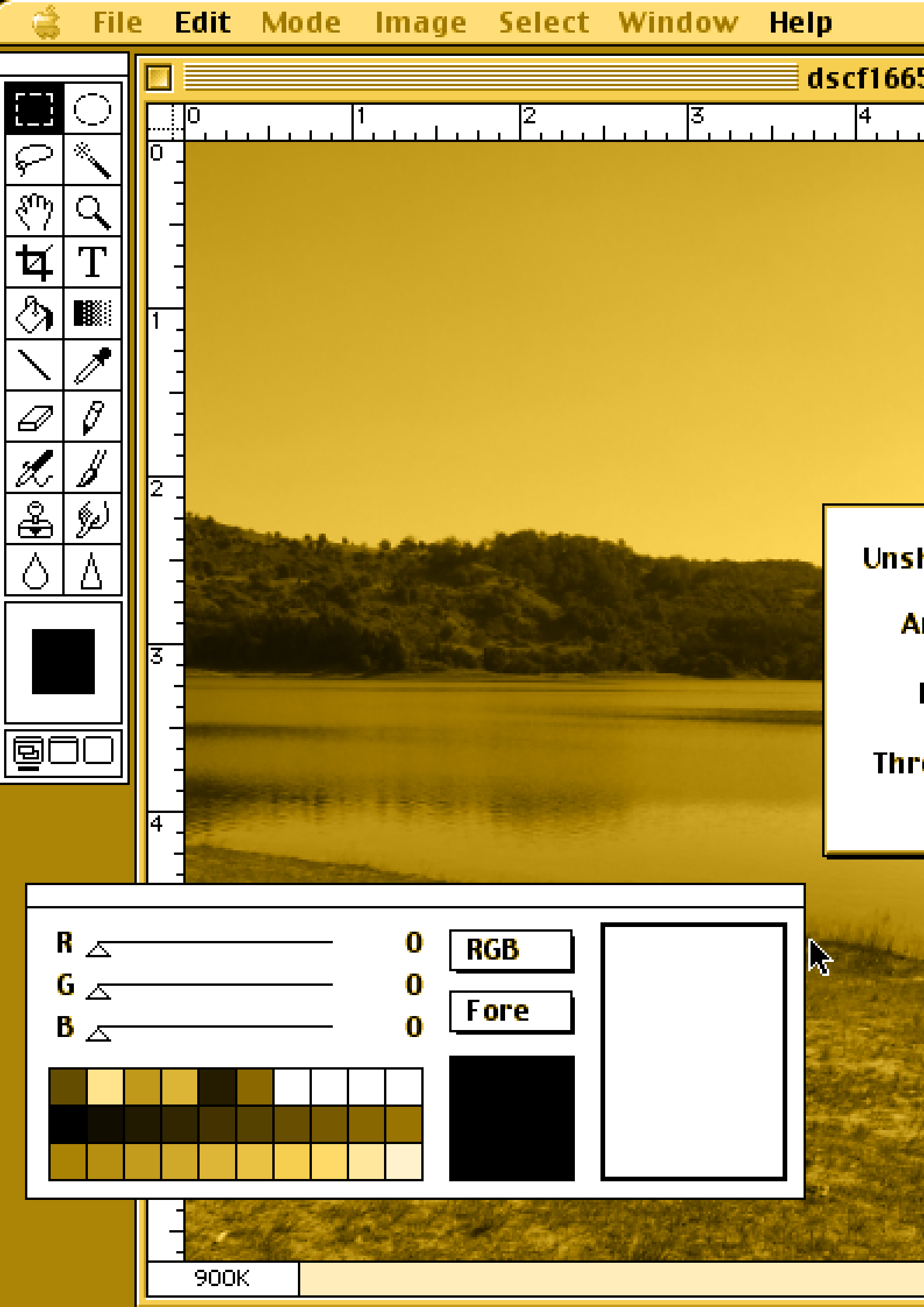
File Edit Mode Image Select Window Help
dscf1665
R 0
G 0
B 0
RGB
Fore
900K

PROCESSED PICTURES, PHOTOSHOP, AND UNSHARP MASK

TILL A. HEILMANN

ABSTRACT | The project presented here examines techniques and practices of digital image processing using the consumer application Adobe Photoshop as an example. My study aims at a theory of contemporary visual culture that addresses digital images in their distinct quality of being processed pictures instead of in general terms or abstract notions of digitality. To achieve this goal, I will conduct an in-depth investigation of Photoshop at the levels of both the cultural layer of interfaces and uses, and the computational layer of code and data structures. Using the methodologies of software studies and media archaeology, the project will give a first comprehensive account of Photoshop as one of the most influential tools for the production of digital culture. I illustrate the application of the software studies approach to image processing by giving a short but exemplary analysis of Photoshop's popular Unsharp Mask feature.

KEYWORDS | contemporary visual culture, digital/digitized, image processing, media theory, photography

Premise and Goal of the Project

The project "The Processed Picture" aims at improving our understanding of contemporary visual culture by way of studying digital image processing. To this end, I am conducting an in-depth cultural analysis of the software Adobe Photoshop, one of the most popular tools for digital image processing and editing.[1]

The project has three main goals: first, to give a comprehensive account of Photoshop's cultural logic of image manipulation; second, to establish a framework for reflecting digital image processing in its significance for visual culture and a general theory of digital media; and third, to conceptualize the phenomenon of digital images in their distinct quality of being processed pictures instead of in general terms or abstract notions of digitality.

The premise of the project, echoing a famous slogan in German media theory from the early 2000s, is that the digital image does not exist. However, unlike authors like Hagen and Pias[2] I want to argue that while there is no such thing as 'the' digital image, scarcely conceivable as a singular abstraction, there are many different kinds of digital images. These images can therefore only be properly analyzed and interpreted if their manifold modes of

existence as video game graphics, computer tomography, visual effects in movies, scientific data visualizations, retouched photographs, and so on are all taken into account. Digital processing is but one moment in the complex operational chain of producing, storing, transmitting, displaying, and interacting with digital images. Nevertheless, it is an important one that has mainly been discussed in relation to "the danger posed to 'truth' by computer-manipulated photographic imagery."[3] In my project, on the other hand, I will focus on the productive effects of digital image processing for epistemic and aesthetic purposes. The sciences and new media art provide ample material for illustration.

Today, Photoshop is slowly losing ground to new competitors in the PC software market. At the same time, it is facing the challenge of new photographic devices like smartphones and new visual regimes like computational photography. The dominance of Photoshop as the premier tool for image editing and processing is coming to an end. But this does not mean that Photoshop has lost its importance. On the contrary, the program's normalizing effect on digital visual culture over the last three decades can be seen not only in the images themselves but also, and maybe more importantly, in the increasing automation of image processing by mobile photography apps and computational photography. Instagram's popular filters, for example, were modeled on Photoshop's algorithms.[4] Even if Photoshop were to disappear altogether as a discrete software application, its processing logic has already permeated our contemporary visual culture and is deeply ingrained in everyday imaging technologies.

Methodology

Methodologically, I will take the software studies approach, using analytic and interpretive techniques like interface critique and critical code reading. To elaborate on the historical and practical dimensions of digital image processing, I will rely on the methods of media archaeology, oral history, and discourse and media analysis. On the theoretical side, I will build on and continue work on the nexus between imaging, knowledge, and mediation.[5] In particular, I will draw upon recent studies on aesthetic and epistemic aspects of digital imaging.[6]

Software studies is a relatively new branch of academic research that emerged from media studies in the early 2000s in response to the digital transformation of media culture and also as a reaction to the narrow focus of media theory on hardware issues.[7] The digitization of telecommunication and media industries since the 1980s has led to traditional media like print, television, and photography being increasingly remodeled or remediated in software.[8] Since the core component of computing hardware, the microchip, is usually of a general-purpose architecture, the actual specification or shaping of a digital medium is determined by the program(s) running on the hardware. Therefore, the task of describing and interpreting digital media has effectively become the task of analyzing software. Because of the relative novelty of the approach and the fact that there have been very few actual case studies of major consumer applications like Photoshop, my project will also serve as a methodological test run for software studies.

In software studies, we examine the object in question in its peculiar dual existence, i.e. on the cultural layer of user interfaces, features provided, typical uses and effects, etc., and on the computational layer of algorithms, data structures, protocols, file formats, and so on.[9] When it comes to programs like Adobe Photoshop, this means that we systematically relate the aesthetic 'surface' of digital images to their technical 'subface'[10] in order to better understand the link between cultures and codes of digital images.

A frequent obstacle to the study of software is the fact that the source codes of programs, i.e. the human-readable forms of software written in high-level programming languages, are not disclosed to the public. This is particularly true in the case of commercial applications like Adobe Photoshop where the source codes typically remain well guarded and legally protected corporate secrets. Analyses and interpretations of software often have to admit defeat at this point. Luckily, Adobe released the source code for the first version of Photoshop in 2013.[11] This makes it possible to see how Photoshop's image processing features and data structures were originally implemented in version 1.0 (written in Object Pascal and 68000 assembly language) for Macintosh computers. Since this first version of the program is also available for emulation on modern machines, we can study Photoshop 1.0 in detail on both the cultural and the computational layer.

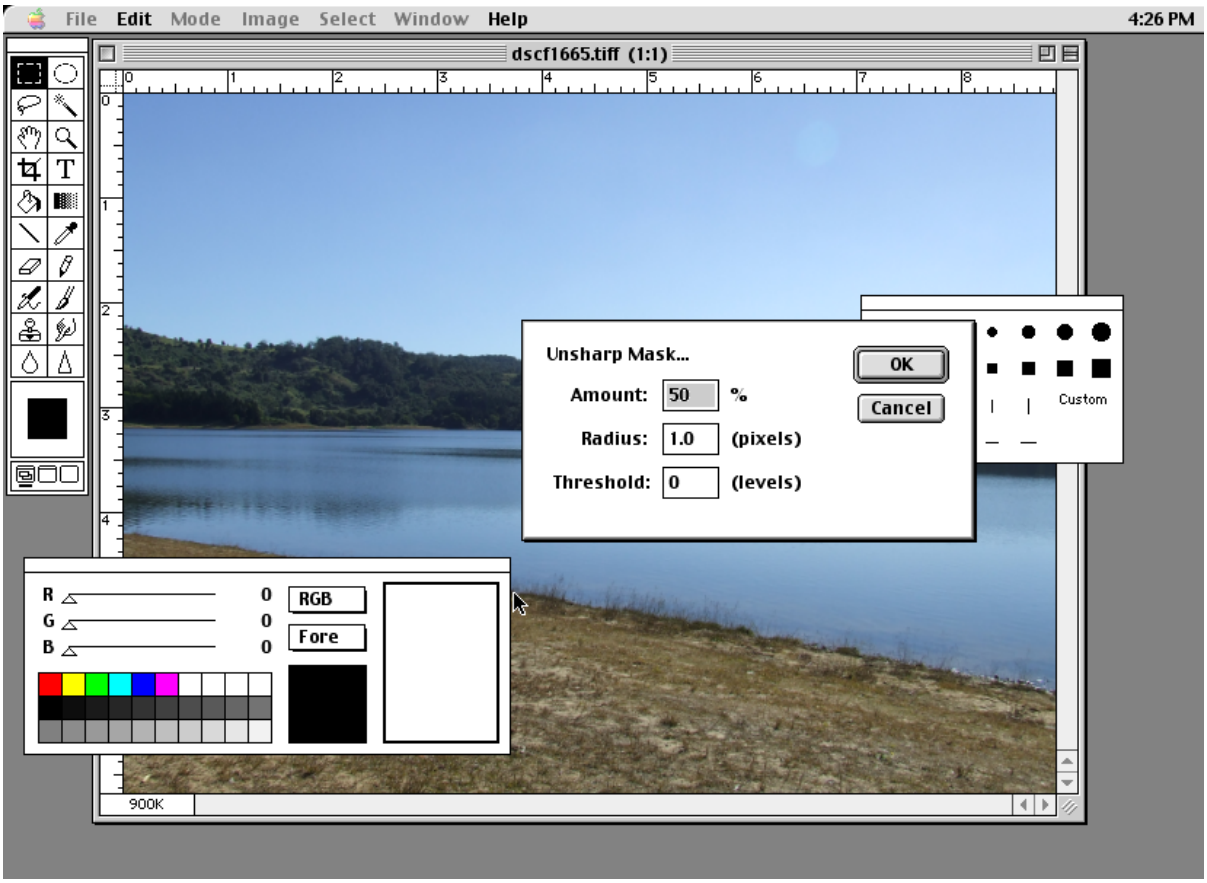

Figure 1. Unsharp Mask dialog box in Adobe Photoshop 1.0.7; 2021; screenshot.

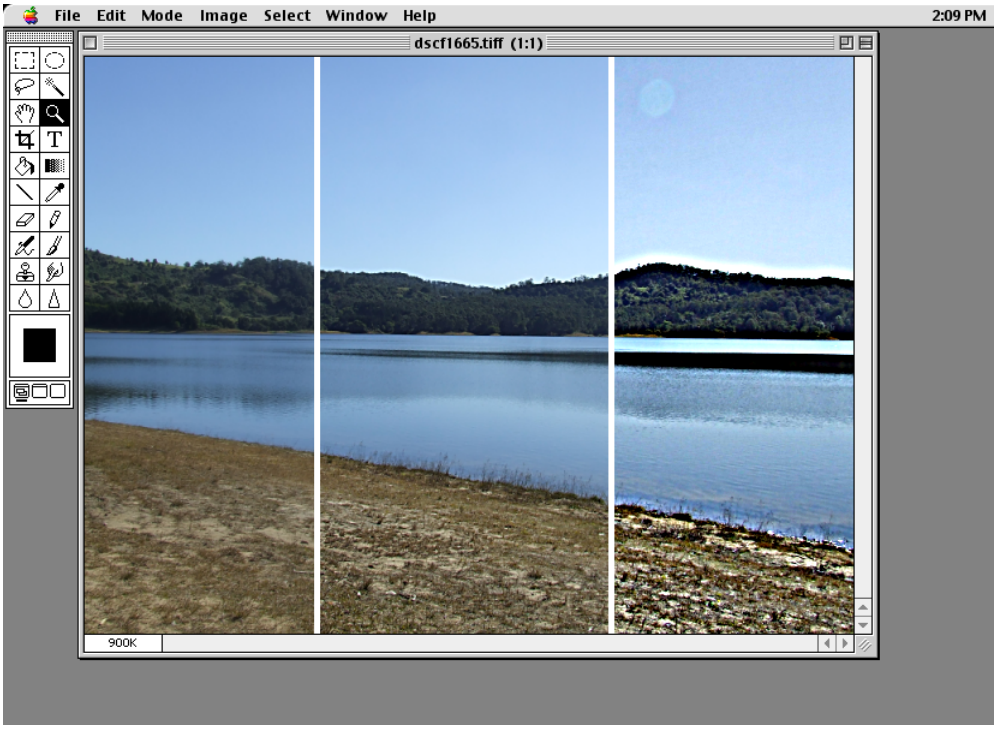

Figure 2. Comparison of Unsharp Mask filter effects in Adobe Photoshop 1.0.7. Left pane: filter not applied; middle pane: Amount: 80, Radius: 2.0, Threshold: 1; right pane: Amount: 300, Radius: 5.0, Threshold: 0; 2021; screenshot.

Unsharp Mask: Cultural Layer

In the remainder of this text, I will demonstrate a short analysis of one particular image processing feature of Photoshop to illustrate the program's visual logic.[12] The feature in question is the Unsharp Mask filter. Unsharp Mask is considered by many professionals to be one of the most important and historically significant features of Photoshop. The image filter—its name somewhat misleading for laymen—has been an integral part of the software since the first version of the program and serves as the primary tool for sharpening images. As the Official Adobe Photoshop Handbook from 1991 states succinctly, "Unsharp Mask is the most accurate way of creating a controlled sharpening effect."[13] In the following pages, I can only give a cursory impression of what analyzing and interpreting Photoshop from the perspectives of software studies and media archaeology might look like. I will limit myself to a few remarks on three aspects of the Unsharp Mask filter as it was realized in the first version of the program in 1990: its interface, its implementation in code, and its genealogy and evolution.

Our analytical and archaeological descent[14] into Adobe Photoshop's workings and history starts at the surface or on the cultural layer of the program, with a description of the Unsharp Mask filter as it appears to the user. How does Unsharp Mask present itself and its operations graphically on the screen?

Unsurprisingly, perhaps, for the first version of a program from the beginning of the 1990s, the feature is quite plain in appearance. Unsharp Mask is listed under its name as the last item of twenty-two filters in the according menu. The dialog box that pops up when you select the filter from the menu is also kept very simple and shows only three input fields for entering numerical parameters labeled Strength, Radius, and, since version 1.0.7, Threshold (fig. 1). After you have chosen the parameters and pressed the OK button, Photoshop's algorithms will increase the contrast at edges automatically determined in the image by emphasizing changes in brightness or color, i.e. by lightening light edges and darkening dark edges. If applied correctly, Unsharp Mask can make pictures clearer and crisper. If the parameters are set too high, the noise in the image will become more noticeable, colors can shift and halos may appear (fig. 2).

The rather simple controls and functionality of the Unsharp Mask filter have not changed substantially since the first version of Photoshop. Part of the feature's simplicity lies, of course, in the fact that we commonly do not have insight into the computational processes that take place 'below' the cultural layer of the graphical user interface (GUI). On the surface of the application, the actual processing is revealed merely through an animated progress bar (in early versions of the program), through a small preview area (only in later versions of the program), and, of course, in the resulting image at the end. Exactly how Photoshop determines edges in the picture and how the chosen parameters affect the operation of the filter remains hidden from the user's view.

As a 'surface' phenomenon of the GUI, the Unsharp Mask filter—like the rest of Photoshop's filters—stands in stark contrast to the picture it processes in three different ways: firstly, in spatial opposition insofar as the filter feature occupies its own place in the GUI as a list item in the menu, as a dialog box, and as a progress bar placed next to or 'over' the image, but in any case outside the picture itself; secondly, in logical opposition insofar as the feature represents the measure of its impact on the picture in a definitely nonpictorial manner, i.e. through text and, above all, through numbers; and thirdly, in temporal opposition insofar as it occurs in between two states of the picture—the unprocessed version before and the processed version after—without making the transition from one state to the other, the process of processing, visible in the picture itself.

Unsharp Mask: Computational Layer

If one wants to know more about how the Unsharp Mask feature works on the computational layer, the user manual for Photoshop (again: the one for the first release of the program) provides some hints:

> *The Unsharp Mask filter sharpens pixels using a variable radius. You specify a radius (in pixels) around the current pixel, which is being evaluated. The Unsharp Mask filter then blurs a selection according to the specified radius. A fraction of this blurred result is then subtracted from the original data, resulting in a sharpening effect. The larger the radius, the more information is included in the filter's calculations. If you specify a high value for the radius, the lower frequencies will be amplified; if you specify a low value,*

only high-frequency areas will be amplified. You can also specify the percentage of the filter's effect. The higher the percentage, the stronger the effect of the Unsharp filter mask on a selection. If you specify a low value, only a fraction of the effect is applied; if you specify a high value, most of the effect is applied.[15]

The Unsharp Mask feature works, we learn, by using the brightness information from a blurred copy of the image to alter the original image. The blurred or 'unsharp' copy—from which the filter gets its name—acts as a digital mask through which the unprocessed image data is algorithmically filtered to emerge from the process with increased edge contrast.

While analyses of computer programs usually come to a halt at the boundaries of the software's cultural layer, in the case of Adobe Photoshop we may continue our descent down to the 'subface' computational layer of codes, algorithms, and data structures. Of course, having access to the source for Photoshop 1.0 alone is not enough. To examine the program according to the principles of Critical Code Studies,[16] further conditions must be met. In particular, at least some basic knowledge of the programming languages used is necessary. Within the scope of this article, I can only give a sketch of how such an analysis might be conducted.

The source code for version 1.0.1 of Photoshop comprises 178 individual files (excluding the text file ChangeHistory.txt with version notes).[17] It is from these files that the ready-to-run application for consumers was compiled (with the help of many additional files from the MacApp framework).

Even before looking inside the files and examining the code in detail, we can learn important things by performing a basic 'distant reading' of the source code. The majority of files—144 to be exact—were written in the high-level programming language Pascal (extensions .p and .inc). Twenty-one files were written in assembly language (extension .a), the rest are files containing data encoded in different formats (extensions .r, .t, .h) and files controlling the compilation process. A simple count with the basic Unix tool wc ($ wc -l *.p *.inc *.a) gives us a total of 116,587 physical lines of code, including comments and blank lines. Using the more specialized tool cloc ($ cloc --force-lang="Assembly",a --force-lang="Pascal",inc --exclude-ext=r *), we find that there are 63,004 lines of Pascal and 8,228 lines of assembly in Photoshop's source, not counting comments and blank lines—a ratio of approximately 8:1.

In the form of a special object-oriented version developed by Apple, Pascal was the main programming language for writing software for Macintosh computers until the early 1990s, when it was gradually replaced in that role by C++. Object Pascal was closely linked to Apple's MacApp application framework which provided the generic functions, structures, and objects (such as menus, dialog boxes, input fields, etc.) needed by programs to fit seamlessly into the environment of the Macintosh operating system and GUI.[18] Taken together, Object Pascal and MacApp made it possible for software developers to build applications more quickly and more easily from ready-made parts than would have been the case if they had been designing them from the ground up. The choice of (Object) Pascal as the main implementation language for Photoshop, therefore, signals a desire for maximum conformity with Apple's digital ecosystem as well as the economic constraints of commercial software development. Lastly, the language design of Pascal stands for a certain style of writing programs in a clear, efficient, and logical manner—a programming paradigm which had been known as structured programming since the 1970s and which promised better control and management of the ever-increasing complexity of software.[19]

Assembly code, on the other hand, is the counterpart to high-level languages like Pascal: a terse symbolic transliteration of the computer's monotonous binary machine language, strictly tied in its mnemonic vocabulary of opcodes to the specific architecture of the microprocessor (in this case, a Motorola 6800 series CPU). Since the 1980s, assembly has only been used where and when maximum efficiency of code is required, i.e. for the highest possible speed of calculation and the smallest possible memory footprint. It is therefore hardly surprising that only a few files of Photoshop's source code are written in assembly. And it is all the more telling when we look to see what these files are about. The second largest one of them ($ wc -l *.alsort -r) is called UFilters.a.[20] The name suggests that the code in this file concerns elementary algorithmic procedures for Photoshop's filter features and that these procedures are among the most computationally expensive parts of the program. Implementing the basic filter procedures on the computational layer in the much more convenient Pascal language would not have resulted in the calculating speed expected on the cultural layer of the program. Letting the user stare at the screen while she waited for the computations to complete and the processed picture to appear was simply not an option.

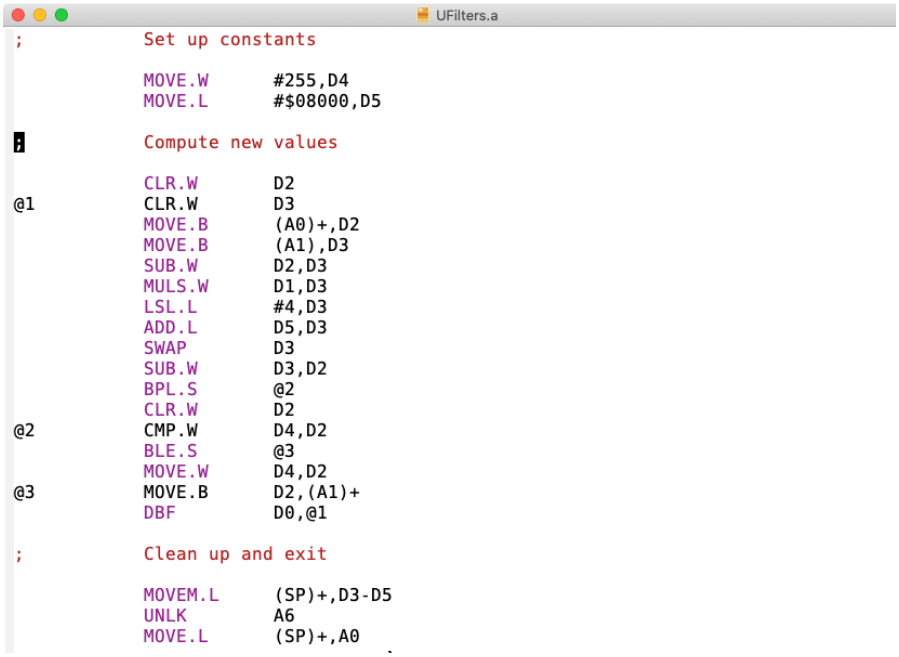

Figure 3. File UFilters.a from source code for Adobe Photoshop 1.0.1, lines 1299–1326; 2021; screenshot.

Examining the file UFilters.a more closely, we can find within "the endless litany of 'read,' 'write,' 'move,' and 'load' [...] called *assembly language*"[21] some reader-friendly labels like DoBoxFilter, DoWeightedFilter, DoMedianFilter, and also DoUnsharpMaskLine (in lines 95, 182, 1336, and 1258). Presumably, these are the names by which the respective procedures are called from elsewhere in the source code of Photoshop. And without any further knowledge of assembly language, we can see that the latter procedure referencing the Unsharp Mask feature is only thirty-three lines of code long (not counting comments, directives for constants, and blank lines). And the core of this procedure, marked "Compute new values" in the comment on line 1304, counts a mere seventeen assembly commands (lines 1306-1322). Compared to the total amount of Pascal and assembly code (71,232 lines), the presumed algorithmic nucleus of the Unsharp Mask feature seems downright tiny (fig. 3).

The spatial, logical, and temporal break separating the Unsharp Mask filter from the image to be filtered on the cultural layer is reflected on the computational layer. There, it appears as the divide between the concise and direct assembly instructions for speed-optimized computation of the pixel values on the one hand and the highly structured and abstract Pascal objects for integrating the feature into the application's graphical user interface on the other. The tension between machine-oriented procedures for elementary image processing operations and the overall framework of the program, abstracted away from the underlying hardware in a higher programming language and structured as a complex hierarchy of objects, runs through the entire source code of Photoshop.

Here, I want to end my brief description of Photoshop's computational layer. I would only add that a further search for traces of the Unsharp Mask feature in the program's source with the help of the Unix tool grep ($ grep -ni unsharp *) brings to light sixteen more places, among them a procedure of the Pascal class TUnsharpMaskFilter called .DoFilter (UFilter.p, lines 1268-1328). This procedure does not call only the procedure DoUnsharpMaskLine (line 1313) but, before it, a .DoFilter procedure (line 1293) 'inherited' by its parent class TGaussianFilter. This inherited .DoFilter procedure, in turn, finally calls an assembly procedure called DoWeightedFilter (UFilters.inc1.p, lines 168 and 220).

Before it proceeds to compute the new brightness of pixels with DoUnsharpMaskLine, the Unsharp Mask feature seems to apply a Gaussian filter—i.e. one weighted according to the normal distribution—to the original image (DoWeightedFilter). Apparently, this step of processing creates the blurred copy of the image mentioned in the Adobe Photoshop User Guide ("The Unsharp Mask filter then blurs a selection according to the specified radius"), which is subtracted from the original image to achieve the desired sharpening effect ("A fraction of this blurred result is then subtracted from the original data, resulting in a sharpening effect").[22]

Unsharp Mask: Genealogy

With the subject of the Gaussian filter, we have arrived at the question of the Unsharp Mask's genealogy. For the blurred image produced by Photoshop's DoWeightedFilter assembly procedure has two distinct precursors in media history. On the one hand, the use of filters for signal processing predates digital computing, or at least digital image processing, by quite a bit. Image processing features like the Gaussian filter or the high-pass filter in Photoshop are really digital simulations of earlier electronic filters developed for analog signal processing. Methods for brightness interpolation and noise removal had been realized with analog circuitry since the first half of the 20th century, primarily within the context of wireless and telephone technology.[23] On the other hand, and this brings me to my final point, the technique of the Unsharp Mask was known long before it was turned into digital code. Again, the first Photoshop User Guide tells us more: "The [Unsharp Mask] filter is commonly used in pre-press production to enhance details in the separations by producing exaggerated density at the borders of a color change."[24]

Historically, the technique of unsharp masking came to digital image processing from the graphics industry. There, it was used as a pre-press procedure to prepare the printing plates. A common problem in the reproduction of color images is that dissimilar but adjacent colors can 'bleed' or mix and their tones get distorted. To reduce color distortions and improve the print quality, the images to be printed are first filtered through unsharp masks to enhance the edge contrast of colors.[25] Before digital Gaussian filters, unsharp masks were created photographically by copying the original negative with soft light or a diffusor plate. The result was a slightly blurred positive. When this low-contrast positive—the unsharp mask—was combined in-register with the original negative for producing a color separation, it acted as a filter that partially canceled information in the negative. In bright areas of the original negative, it let less light pass through, and vice versa. Because it had been blurred (intentionally), the unsharp mask only canceled low-frequency information from the original. The result was a color separation with increased acutance and less color bleeding in the prints.

The parallels between the pre-press technique of unsharp masking and the digital Unsharp Mask filter are obvious. The Gaussian filter used for blurring images in the digital domain corresponds to the soft light or diffusor plate used to produce the low-contrast positive in analog printing; the mathematical operation of subtraction from the original data in the computer corresponds to the combined projection of low-contrast positive and original negative in photomechanical reproduction. In both cases, an original image is first copied and then enhanced by combining it with its low-quality duplicate. In this perspective, the Unsharp Mask feature of Photoshop constitutes a digital simulation of an analog technique from photography and the printing industry. Or put the other way around: In retrospect, the older technique turns out to be an analog form of image processing.

However, neither the historical nor the technological relationship of analog and digital unsharp masking are quite as straightforward as my brief account suggests. For the digital Unsharp Mask feature of Photoshop has gradually reduced the method to a single purpose: the increase of acutance in photographic images. As the repeatedly cited user guide for the program shows, this was not always the case. In the pre-digital stage of image processing, unsharp masking was used primarily to solve technical problems in the reproduction of images (like color bleeding in the printing industry).

Unsharp masking was popularized, above all, in the 1970s by David Malin, a British-Australian astronomer. Malin used the technique to detect and preserve faint structures and details in astronomical photographs. Though these were present on the photographic plates that Malin had exposed, they could not be seen and reproduced when copied onto film or paper for subsequent reproduction without the application of unsharp masking techniques.[26] And indeed, this was the main purpose of the photographic method since its earliest days: to enable a controlled reduction of the dynamic range of original negatives. The first description of the technique was given around 1930 by the German medical physicist Gottfried Spiegler and his radiographer Kalman Juris at the Röntgentechnische Versuchsanstalt (X-ray lab) of the Vienna General Hospital. Spiegler and Juris did not use the term 'unsharp mask' but called their invention "a new copying process for producing ideally harmonious copies from high-contrast negatives."[27] The challenge they faced was how to copy images with large high-density regions—as is typical for X-ray photographs—without losing any diagnostically important details in very dark or very light parts of the picture. The problem to be solved was not the poor quality of radiographic pictures but the technical difficulties with their reproducibility or 'duplicability.'[28]

Without going into further technical details, we note the following: Before the triumph of digital image processing, sharpening of edges achieved by photographic unsharp masking techniques was a means to an end (e.g., in the preparation of color separations for printing) or an unintended, if useful, side effect (e.g., in Spiegler's and Juris's radiography or Malin's astrophotography), not an end in itself. Improving picture quality was not the intention; preserving picture quality in reproductions was. The problem, in other words, was not a lack of information in the original picture; on the contrary, it was the high amount of information in images, their broad range of tonal values, that could not be reproduced without loss when transferred to other media. In a nutshell, unsharp masking was not for correcting defects in images but for dealing with deficiencies of imaging technology.

Remarkably, this was still true in the early days of image processing with personal computers. Before digital cameras took the consumer market by storm in the 2000s, photographs had to be digitized with a scanner before you could edit and process them using a program like Photoshop.[29] Due to the low resolution of scanners at the time and constraints inherent to the sensor technology, scanned negatives and prints often suffered from slight blurring at the edges. And the sharpening features of Photoshop were originally intended as a remedy against the blur introduced by the scanning process. Photoshop's digital image filter, to put it briefly, was designed to dissimulate the digitizing of images.[30]

Early Photoshop user manuals and handbooks tell you about this. Adobe's first official tutorial for the program from 1993, for instance, remarks: "[T]he scanning process can cause an image to appear slightly out of focus or 'soft.' You can sharpen an image using the Adobe Photoshop sharpening filters."[31] One year later, the second edition of the tutorial has this to say: "The Unsharp Mask filter adjusts the contrast of edge detail, creating the illusion of more image sharpness. This filter can be useful for refocusing an image that has become blurry from interpolation or scanning."[32] And even in the year 2002, most of the reasons for using the Unsharp Mask feature given by Photoshop's online help concerned the reproduction of images: "The Unsharp Mask filter corrects blurring introduced during photographing, scanning, resampling, or printing."[33]

Of course, things changed some time ago. The pictures that are processed and edited today are usually 'born digital', captured with digital cameras, while the sensor technology has improved considerably. Compensating for losses from the scanning process with digital sharpening filters hardly seems necessary anymore. But sharpening of photographs with Unsharp Mask and the like still happens, possibly more than ever. By the end of the 1990s, the Unsharp Mask feature had completed its cultural transformation from a specialized filter for correcting scanned pictures to a universal image enhancement tool. In 1998, for example, renowned Photoshop expert Dan Margulis described it as an all-around optimization instrument in his popular column Makeready: "Unsharp masking is an artificial method of making images appear more in focus. It is useful in virtually all graphic scenarios[.]"[34] And a few years later, Margulis followed up with the statement that digital sharpening not only benefits even high-quality pictures but should, in fact, be done routinely: "Almost every picture needs it, and not because photographers don't know how to focus their cameras."[35]

The changed purpose of the filter is also reflected in the literature on Photoshop. Tellingly, Adobe's current reference for the software omits the photographic tradition of unsharp masking and mentions the filter's original intent only in passing. The program's various sharpening filters are now summarized as follows: "Whether your images come from a digital camera or a scanner, most images can benefit from sharpening."[36] If you want further proof of the feature's transformation, YouTube offers plenty of evidence. In popular video tutorials hosted by the site, Unsharp Mask is discussed almost exclusively as a general method for image improvement.

The most momentous shift, however, concerns the filter's role within the Photoshop program itself. Since the turn of the millennium, digital unsharp masking has been elevated to an integral part of Adobe's image processing pipeline. Introduced in 2003, the plug-in software Camera Raw is a digital darkroom, so to speak, that allows users to work on the 'raw', unprocessed information from the camera before it is converted into an image in the TIFF, JPEG, Photoshop's proprietary PSD, or some other file format for more editing and compositing.[37] In addition to various features for image adjustment such as cropping, setting the white balance, adjusting contrast and color, and so on, Camera Raw presents users with a range of default settings that control the conversion of raw data into pictures—including one for sharpening the image with the Unsharp Mask algorithm (fig. 4).[38] Unsharp masking has become an 'always-on' filter that all Photoshop images must pass through.

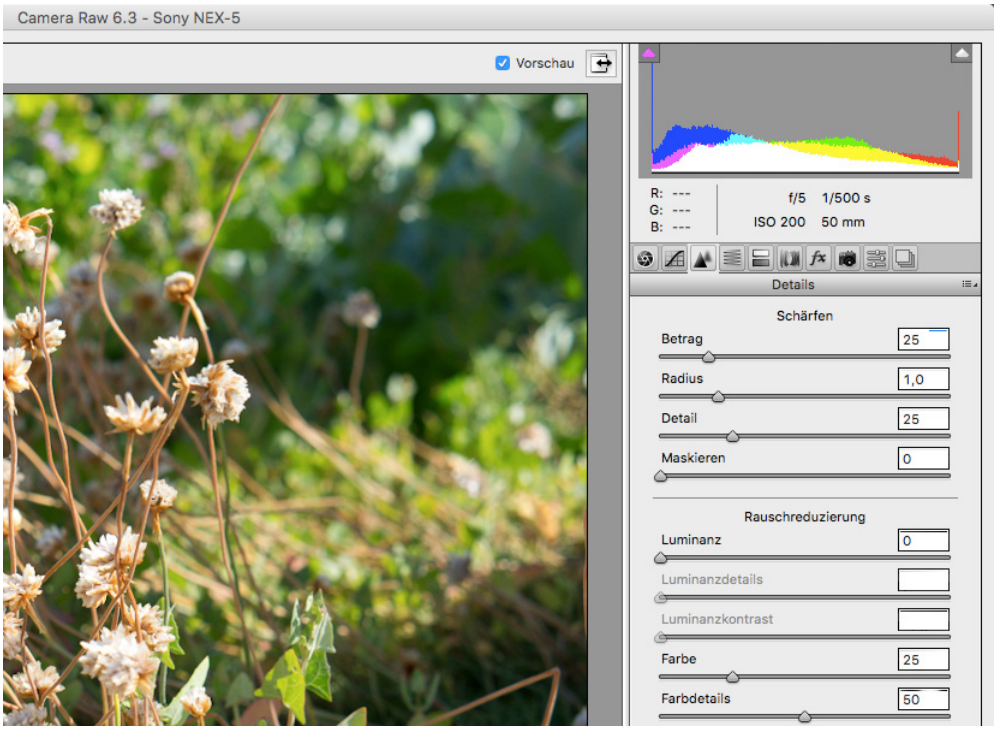

Figure 4. Unsharp Mask defaults in Adobe Camera Raw 6.3. German localization "Betrag", "Radius", "Detail", and "Maskieren" for Amount, Radius, Detail, and Masking; 2021; screenshot.

It could be argued that digital images should always be sharpened with Unsharp Mask or similar image processing features. Because the color values of individual pixels captured by regular sensor technology with Bayer filters have to be interpolated from multiple adjacent sensor pixels, edges in digital photographs are always somewhat 'blurred'. However, this does not change the fact that the method of unsharp masking has turned from a photographic technique to control the density range of pictures in the first half of the twentieth century into an algorithmic image enhancement procedure that is now routinely applied to all digital photographs.

Conclusion

Over the course of its evolution from the Röntgentechnische Versuchsanstalt Wien to Photoshop, the digital transformation of unsharp masking has resulted in a reduction of the method to one single purpose—edge sharpening—on the one hand, and in a generalization of this purpose for all photographs on the other hand. Image processing and editing applications for PCs and digital cameras themselves have already automated unsharp masking to a large extent. Current and future imaging soft- and hardware, like Instagram and computational photography with next-generation smartphones[39] will amplify this effect. We may assume that all of this contributes to a change in our perception of images, our expectations of what good, 'sharp' pictures should look like as well. This topic, however, needs to be addressed in another study.

NOTES

[1] A note on terminology: We commonly distinguish between image processing and image editing. While the former term means global modifications made to images through largely automated processes (e.g. the redistribution of tonal values), the latter term refers to manually applied corrections to select parts of an image (e.g. the gradual retouching of skin blemishes in a portrait with a digital repair brush). In practical use, however, the two forms of image manipulation often overlap and rely on similar or the same computational methods and processes. Therefore, I will use the two terms synonymously.

[2] Wolfgang Hagen, "Es gibt kein 'digitales Bild'," *Archiv für Mediengeschichte*, no. 2 (2002): 103–111; Claus Pias, "Das digitale Bild gibt es nicht: über das (Nicht-)Wissen der Bilder und die informatische Illusion," *zeitenblicke* 2, no. 1 (2003), accessed February 17, 2021, url: http://www.zeitenblicke.de/2003/01/pias/.

[3] Martha Rosler, "Image Simulations, Computer Manipulations," *Afterimage*, November 1989: 7.

[4] Kevin Systrom, "Answer to What Do the Different Image Filters on Path, Instagram, Oink, Etc. Actually Do?" January 4, 2012, accessed February 17, 2021, https://www.quora.com/What-do-the-different-image-filters-on-Path-Instagram-Oink-etc-actually-do/answer/Kevin-Systrom.

[5] William J.T Mitchell, *Image Science: Iconology, Visual Culture, and Media Aesthetics* (Chicago: University of Chicago Press, 2018); Hartmut Winkler, *Prozessieren: Die dritte, vernachlässigte Medienfunktion* (Paderborn: Wilhelm Fink, 2015); Vilém Flusser, *Into the Universe of Technical Images* (Minneapolis, MN: University of Minnesota Press, 2011); Karin Knorr-Cetina, *Epistemic Cultures: How the Sciences Make Knowledge* (Cambridge, MA: Harvard University Press, 1999); Jay David Bolter and Richard Grusin, *Remediation: Understanding New Media* (Cambridge, MA: MIT Press, 1999).

[6] Steve F. Anderson, *Technologies of Vision: The War Between Data and Images* (Cambridge, MA: MIT Press, 2017); Ingrid Hoelzl and Rémi Marie, *Softimage: Towards a New Theory of the Digital Image* (Chicago, Ill.: University of Chicago Press, 2015); Hubertus Kohle, *Digitale Bildwissenschaft* (Glückstadt: Werner Hülsbusch, 2013); Inge Hinterwaldner, *Das systemische Bild: Ikonizität im Rahmen computerbasierter Echtzeitsimulationen* (Munich: Wilhelm Fink, 2010); Mark B. N. Hansen, *New Philosophy for New Media* (Cambridge, MA: MIT Press, 2006).

[7] Lev Manovich, *The Language of New Media* (Cambridge, MA: MIT Press, 2001), 45–48; Lev Manovich, *Software Takes Command* (New York: Bloomsbury Academic, 2013), 10–20, accessed February 17, 2021, url: http://issuu.com/bloomsburypublishing/docs/9781623566722_web?e=3257035/4651685; Friedrich Kittler, "There Is No Software." In *The Truth of the Technological World*, ed. Hans Ulrich Gumbrecht (Stanford, CA: Stanford University Press, 2013), 219–229.

[8] Bolter and Grusin, *Remediation*, 19.

[9] Manovich, *The Language of New Media*, 46–47.

[10] Frieder Nake, "The Disappearing Masterpiece: Digital Image & Algorithmic Revolution," In *4th Conference on Computation, Communication, Aesthetics and X*, Bergamo, July 7–8, 2016, 12–27.

[11] Len Shustek, "Adobe Photoshop Source Code" (Computer History Museum, February 13, 2012), https://computerhistory.org/blog/adobe-photoshop-source-code/.

[12] The following is adapted from my German article Till A. Heilmann, "Blackbox Bildfilter: Unscharfe Maske von Photoshop zur Röntgentechnischen Versuchsanstalt Wien," *Navigationen*, no. 2 (2020): 75–93.

[13] David Biedny and Bert Monroy, *The Official Adobe Photoshop Handbook* (Toronto et al.: Bantam Books, 1991), 204.

[14] Jussi Parikka, *What Is Media Archaeology?* (Cambridge–Malden, MA: Polity Press, 2012), 80.

[15] Adobe Systems, *Adobe Photoshop User Guide* (Mountain View, CA: Adobe Systems, 1990), 306. https://archive.computerhistory.org/resources/access/text/2013/01/102640940-05-01-acc.pdf.

[16] Mark C. Marino, *Critical Code Studies* (Cambridge, MA: MIT Press, 2020).

[17] You can download the source code from the website https://computerhistory.org/blogs/ photoshop-software-license-agreement/.

[18] Kurt J. Schmucker, *Object-Oriented Programming for the Macintosh* (Hasbrouck Heights, N. J.: Hayden Book Company, 1986).

[19] Ole-Johan Dahl, Edsger W. Dijkstra, and Charles Antony Richard Hoare, *Structured Programming* (London: Academic Press Ltd., 1972).

[20] The largest assembly file of the Photoshop source code is USeparation.a, the part of the program performing the complex calculations involved when converting images from RGB to CMYK and vice versa.

[21] Friedrich Kittler, "Protected Mode," in *The Truth of the Technological World*, ed. Hans Ulrich Gumbrecht (Stanford, CA: Stanford University Press, 2013), 217.

[22] Adobe Systems, *Adobe Photoshop User Guide*, 306.

[23] Vitold Belevitch, "Summary of the History of Circuit Theory," *Proceedings of the IRE* 50, no. 5 (1962): 848–855.

[24] Adobe Systems, *Adobe Photoshop User Guide*, 306.

[25] Helmut Kipphan, ed, *Handbuch der Printmedien: Technologien und Produktionsverfahren* (Berlin: Springer, 2000), 519.

[26] Adobe Systems, "Stargazer" (Adobe Systems, 2006), accessed February 17, 2021, https://www.adobe.com/digitalimag/pdfs/davidmalin.pdf; David F. Malin, "Unsharp Masking," *AAS Photo-Bulletin*, no. 16 (1977): 10–13.

[27] Gottfried Spiegler and Kalman Juris, "Ein neues Kopierverfahren zur Herstellung ideal harmonischer Kopien nach kontrastreichen Negativen," *Fortschritte auf dem Gebiet der Röntgenstrahlen* 42 (1930): 509.

[28] Spiegler and Juris, "Ein neues Kopierverfahren zur Herstellung ideal harmonischer Kopien nach kontrastreichen Negativen," 513.

[29] Biedny and Monroy, *The Official Adobe Photoshop Handbook*, 394.

[30] In a remarkable recurrence of events twenty years later, the mobile photo app Instagram was designed to make up for the low quality of early smartphone photography. Company founder and app developer Kevin Systrom introduced Instagram's filters

feature because his girlfriend at the time did not like the pictures she had taken with her iPhone 4 and wanted a way to make them look better; Jemima Kiss, "Instagram Ceo Kevin Systrom: 'We're Working on Time Travel'," *The Guardian*, October 2, 2015, accessed February 17, 2021, https://www.theguardian.com/technology/2015/oct/02/instagram-kevin-systrom-interview-working-on-time-travel.

[31] Adobe Systems, *Adobe Photoshop: Classroom in a Book* (Carmel, IN: Hayden Books, 1993), 134.

[32] Adobe Systems, *Advanced Adobe Photoshop: Classroom in a Book* (Indianapolis, IN: Hayden Books, 1994), 179.

[33] Adobe Systems, *Adobe Photoshop 7.0: Classroom in a Book* (Berkeley, CA: Peachpit Press, 2002), 96.

[34] Dan Margulis, "Sharpening with a Stiletto," *Electronic Publishing*, no. 2 (1998), accessed February 17, 2021, https://www.ledet.com/margulis/Makeready/MA27-Sharpening_With_Stiletto.pdf.

[35] Dan Margulis, "Life on the Edge," *Electronic Publishing*, no. 1 (2005), accessed February 17, 2021, https://www.ledet.com/margulis/Makeready/MA69-Life_on_the_Edge.pdf.

[36] Adobe Systems, "Adobe Photoshop CC Help," February 2, 2018, 436, accessed February 17, 2021, https://helpx.adobe.com/pdf/photoshop_reference.pdf.

[37] Most consumer cameras and smartphones, however, will save pictures as JPEGs and sharpen them automatically.

[38] Adobe Systems, *Adobe Photoshop CS User Guide* (Mountain View, CA: Adobe Systems, 2003), 68.

[39] Vasily Zubarev, "Computational Photography: From Selfies to Black Holes," *Vas3k Blog*, July 1, 2019, https://vas3k.com/blog/computational_photography/.

BIBLIOGRAPHY

Adobe Systems. *Adobe Photoshop User Guide*. Mountain View, California: Adobe Systems, 1990. https://archive.computerhistory.org/resources/access/text/2013/01/102640940-05-01-acc.pdf.

Adobe Systems. *Adobe Photoshop: Classroom in a Book*. Carmel, Indiana: Hayden Books, 1993.

Adobe Systems. *Advanced Adobe Photoshop: Classroom in a Book*. Indianapolis, Indiana: Hayden Books, 1994.

Adobe Systems. *Adobe Photoshop 7.0: Classroom in a Book*. Berkeley, California: Peachpit Press, 2002.

Adobe Systems. *Adobe Photoshop CS User Guide*. Mountain View, California.: Adobe Systems, 2003.

Adobe Systems. "Stargazer." Adobe Systems, 2006. https://www.adobe.com/digitalimag/pdfs/davidmalin.pdf.

Adobe Systems. "Adobe Photoshop CC Help," February 2, 2018. https://helpx.adobe.com/pdf/photoshop_reference.pdf.

Anderson, Steve F. *Technologies of Vision: The War Between Data and Images*. Cambridge, MA: MIT Press, 2017.

Belevitch, Vitold. "Summary of the History of Circuit Theory." *Proceedings of the IRE 50*, no. 5 (1962): 848–855.

Biedny, David, and Bert Monroy. *The Official Adobe Photoshop Handbook*. Toronto et al.: Bantam Books, 1991.

Bolter, Jay David, and Richard Grusin. *Remediation: Understanding New Media*. Cambridge, MA: MIT Press, 1999.

Dahl, Ole-Johan, Edsger W. Dijkstra, and Charles Antony Richard Hoare. *Structured Programming*. London: Academic Press Ltd., 1972.

Flusser, Vilém. *Into the Universe of Technical Images*. Minneapolis, MN: University of Minnesota Press, 2011.

Hagen, Wolfgang. "Es gibt kein ‚digitales Bild'." *Archiv für Mediengeschichte*, no. 2 (2002): 103–111.

Hansen, Mark B. N. *New Philosophy for New Media*. Cambridge, MA: MIT Press, 2006.

Heilmann, Till A. "Blackbox Bildfilter: Unscharfe Maske von Photoshop zur Röntgentechnischen Versuchsanstalt Wien." *Navigationen*, no. 2 (2020): 75–93.

Hinterwaldner, Inge. *Das systemische Bild: Ikonizität im Rahmen computerbasierter Echtzeitsimulationen*. Munich: Wilhelm Fink, 2010.

Hoelzl, Ingrid, and Rémi Marie. *Softimage: Towards a New Theory of the Digital Image*. Chicago, Illinois: University of Chicago Press, 2015.

Kipphan, Helmut, ed. *Handbuch der Printmedien: Technologien und Produktionsverfahren*. Berlin: Springer, 2000.

Kiss, Jemima. "Instagram Ceo Kevin Systrom: 'We're Working on Time Travel'." *The Guardian*, October 2, 2015. https://www.theguardian.com/technology/2015/oct/02/instagram-kevin-systrom-interview-working-on-time-travel.

Kittler, Friedrich. "Protected Mode." In *The Truth of the Technological World*, by Friedrich Kittler, edited by Hans Ulrich Gumbrecht, 209–218. Stanford, California: Stanford University Press, 2013.

Kittler, Friedrich. "There Is No Software." In *The Truth of the Technological World*, by Friedrich Kittler, edited by Hans Ulrich Gumbrecht, 219–229. Stanford, California: Stanford University Press, 2013.

Knorr-Cetina, Karin. *Epistemic Cultures: How the Sciences Make Knowledge*. Cambridge, MA: Harvard University Press, 1999.

Kohle, Hubertus. *Digitale Bildwissenschaft*. Glückstadt: Werner Hülsbusch, 2013.

Malin, David F. "Unsharp Masking." *AAS Photo-Bulletin*, no. 16 (1977): 10–13.

Manovich, Lev. *The Language of New Media*. Cambridge, MA: MIT Press, 2001.

Manovich, Lev. *Software Takes Command*. New York: Bloomsbury Academic, 2013. https://issuu.com/bloomsburypublishing/docs/9781623566722_web?e=3257035/4651685.

Margulis, Dan. "Sharpening with a Stiletto." *Electronic Publishing*, no. 2 (1998). https://www.ledet.com/margulis/Makeready/MA27-Sharpening_With_Stiletto.pdf.

Margulis, Dan. "Life on the Edge." *Electronic Publishing*, no. 1 (2005). https://www.ledet.com/margulis/Makeready/MA69-Life_on_the_Edge.pdf.

Marino, Mark C. *Critical Code Studies*. Cambridge, MA: MIT Press, 2020.

Mitchell, William J. T. *Image Science: Iconology, Visual Culture, and Media Aesthetics*. Chicago: University of Chicago Press, 2018.

Nake, Frieder. "The Disappearing Masterpiece: Digital Image & Algorithmic Revolution." In *4th Conference on Computation, Communication, Aesthetics and X*, Bergamo, July 7–8, 2016, 12–27.

Parikka, Jussi. *What Is Media Archaeology?* Cambridge–Malden, MA: Polity Press, 2012.

Pias, Claus. "Das digitale Bild gibt es nicht: über das (Nicht-)Wissen der Bilder und die informatische Illusion." *zeitenblicke* 2, no. 1 (2003). http://www.zeitenblicke.de/2003/01/pias/.

Rosler, Martha. "Image Simulations, Computer Manipulations." *Afterimage*, November 1989, 7–11.

Schmucker, Kurt J. *Object-Oriented Programming for the Macintosh*. Hasbrouck Heights, N. J.: Hayden Book Company, 1986.

Shustek, Len. "Adobe Photoshop Source Code." *Computer History Museum*, February 13, 2012. https://computerhistory.org/blog/adobe-photoshop-source-code/.

Spiegler, Gottfried, and Kalman Juris. "Ein neues Kopierverfahren zur Herstellung ideal harmonischer Kopien nach kontrastreichen Negativen." *Fortschritte auf dem Gebiet der Röntgenstrahlen* 42 (1930): 509–518.

Systrom, Kevin. "Answer to What Do the Different Image Filters on Path, Instagram, Oink, Etc. Actually Do?" January 4, 2012, accessed February 17, 2021, https://www.quora.com/What-do-the-different-image-filters-on-Path-Instagram-Oink-etc-actually-do/answer/Kevin-Systrom.

Winkler, Hartmut. *Prozessieren: Die dritte, vernachlässigte Medienfunktion*. Paderborn: Wilhelm Fink, 2015.

Zubarev, Vasily. "Computational Photography: From Selfies to Black Holes," July 1, 2019, https://vas3k.com/blog/computational_photography/.

TILL A. HEILMANN is Principal Investigator of the project The Processed Picture at the University of Bonn. He works on the theory, history, and aesthetics of digital media. His recent publications include: "Friedrich Kittler's Alphabetic Realism." Classics and Media Theory. Ed. Pantelis Michelakis. Oxford: Oxford University Press 2020, pp. 29–51; "Reciprocal Materiality and the Body of Code." Digital Culture & Society 1/1 (2015): 39–52; "'Tap, tap, flap, flap.' Ludic Seriality, Digitality, and the Finger." Eludamos. Journal for Computer Game Culture 8/1 (2014): 33–46.

Correspondence e-mail: mail@tillheilmann.info