Manovich, Lev. "Conclusion." *Software Takes Command*. New York: Bloomsbury, 2013. 329–342. *Bloomsbury Collections*. Web. 26 Apr. 2023. <<http://dx.doi.org/10.5040/9781472544988.0006>>.

Downloaded from Bloomsbury Collections, www.bloomsburycollections.com, 26 April 2023, 13:01 UTC.

Copyright © Lev Manovich 2013. You may share this work for non-commercial purposes only, provided you give attribution to the copyright holder and the publisher, and provide a link to the Creative Commons licence.

CONCLUSION

Software, hardware, and social media

In 1977 Alan Kay and Adele Goldberg imagined that a computer would become a “metamedium” that would contain “a wide range of already-existing and not-yet-invented media.” Exactly as they predicted, computers have been used to invent a number of *new types of media that are not simulations of prior physical media*. The examples include navigable three-dimensional spaces constructed with computer graphics, media databases, or “sim” video games such as SimCity, The Sims, and Civilization. And, of course, the Internet in particular has been a very productive host for inventing new types of communication and collaboration: email, forums, blogs, microblogging (e.g. Twitter), wiki, RSS, social networks, etc.

At the same time, the computer metamedium was also evolving in a second direction. While it indeed came to contain “a wide range of already-existing” media, I have argued that once each of these media was simulated in a computer their identities changed. Sutherland, Engelbart, Nelson, Kay, and other inventors of computer metamediums understood that *the simulations of previously existing physical or electronic media could add many new properties to these media*. As Kay and Goldberg wrote in their article, a simulated medium can become a “new medium with new properties.”

A computer “breathes new life” into the physical and electronic media it simulates. Media can become “dynamic” (to use another of Kay’s terms which he preferred to “interactive”). They can also be “intelligent”: think of Sketchpad, which could automatically “clean up” the sketches made by designers by satisfying constraints such as parallelism. Further, they can become collectively sharable

and collectively “editable”—think of large-scale social software projects such as Wikipedia and OpenStreetMap. Media objects such as pictures, sound, video, text, and code can leap from machine to machine in a truly magical fashion: from a mobile phone to a media player, then to a memory card, a laptop, a netbook, the Web, and so on and so on. Imagine that you live in the sixteenth century and you are told that you can order an image in a painting to travel by itself and appear in another painting in another country, or that a text in one book can lift itself and replace a text in another book? And yet this is exactly what many of us are doing every day without even thinking how magical this is.

I think that the scale, diversity and radicalism of these “additions”—those already invented and many more yet to be invented in the future—is of such magnitude that exploring what can be created with them will occupy us for a long time. And this is one of the reasons why the “digital revolution” is different from all previous techno-cultural revolutions. The ability to simulate not simply one or two, but most media in a computer—combined with computer abilities to control processes in real-time, calculate, transform inputs, test what/if scenarios, and send information over networks—opens a practically unbounded space of creative possibilities.

As we followed the evolution of the computer medium from the first stage of invention and experimentation (1960s–1970s) to the second stage of commercialization and wide adoption (1980s–1990s), we discovered the third direction of the computer metamedium evolution—*hybridization*. Translated into software, different types of media started acting like species within a common ecology—in this case, a shared software environment. Once they were “released” into this environment they begin interacting, mutating, and making hybrids. Both the simulated and new media types—text, hypertext, still photographs, vector graphics, digital video, 2D animation, 3D models and animation, navigable 3D spaces, maps, location information, messages and scripts—became building blocks for many new media combinations. As my examples illustrate, such hybrids can be found at different scales—from large-scale software systems such as Google Earth to the single images and short motion graphics created by individual designers.

The rise of social media and social networking on the web in the middle of the 2000s, their expansion to mobile platforms in

the next few years, and the development of apps markets for these platforms lead to new types of hybrids. The functional elements of social and mobile media—search, rating, wall posting, subscription, text messaging, instant messaging, email, voice calling, video calling, etc.—form their own media ecosystem. Like the ecosystem of techniques for media creating, editing and navigation realized in professional media software during the 1990s, this new ecosystem enables further interactions of its elements. These elements are combined in a variety of ways on different platforms and in different social media apps. New “features” of social and mobile software enter this ecosystem, and expand the pool of its elements.

The technologies behind these elements, such as multi-tier web applications and server-side scripting, make creating new combinations and new elements particularly easy in comparison to desktop applications.¹ While new versions of desktop applications such as Photoshop, InDesign or Maya that may contain new techniques are released infrequently, online applications such as Facebook, YouTube or Google Search can be updated by their companies very frequently (both Google and Facebook update their code daily) and new features can be added at any time.

The competition between leading social network services has been one engine of new elements as well as variations on existing elements. To give two examples from 2011, Google+ introduced the “Circles” feature to enable users to organize other people using the service into groups and to control what they post and share with each group;² Facebook introduced a Subscribe button to allow its users to follow public updates from particular users they like.³ In the same year, the new social photo sharing service Pinterest rose meteorically, due to its new features such as a flexible layout of photos in contrast to the fixed grid layout of Facebook and Google+.

¹ For a good explanation of how web applications work technically, see http://en.wikipedia.org/wiki/Web_application

² Vic Gundotra, “Introducing the Google+ project: Real-life sharing, rethought for the web,” googleblog.blogspot.com, June 28, 2011, <http://googleblog.blogspot.com/2011/06/introducing-google-project-real-life.html>

³ Meghan Peters, “Facebook Subscribe Button: What It Means for Each Type of User,” mashable.com, September 15, 2011, <http://mashable.com/2011/09/15/facebook-subscribe-users/>

While this book focused on the evolution of software for media creation and editing, this evolution is intrinsically connected with the parallel development of *hardware*, including desktops, and later laptops and mobile platforms, networks, servers, render farms, video cards, displays, and so on. More high resolution displays, larger and cheaper storage, faster networks, easier connectivity between devices for media capture, storage, editing, distribution, and playback—all these developments automatically extend the capacities of computational media, changing what can be imagined and designed. For example, a computer display with 35,840 x 8,000 pixel resolution (e.g. the HIPerSpace super-visualization computer that my lab has been fortunate to use since 2008) is not only quantitatively but also qualitatively different from a display which only has 1024 x 768 pixels (the desktop standard in the 1990s).⁴ Similarly, the experience of using the Internet via broadband connection is qualitatively different from a modem to dial-in via analog telephone line (also the standard in the 1990s).

When Kay's colleagues implemented various media editors on their "interim Dynabook" in the first part of the 1970s, most of this software could not compete with their physical and electronic equivalent tools. I remember my experience working on a first generation Macintosh in 1984; it could only show sixteen levels of grey on a 512 x 384 pixel screen measuring 9 inches. Obviously this was not sufficient for me to immediately drop my oil brushes and paints and switch to computers. So, in some sense, the first period of media computerization—between the completion of Sketchpad in 1963 and the release of PageMaker in 1985—was theoretical. During this period, the conceptual principles and the key algorithms necessary for detailed simulation of physical media were developed, before the sufficiently cheap hardware was available. For instance, during the 1960s many computer scientists learned about Sketchpad by reading Sutherland's PhD thesis since the machine on which it run—the TX-2 computer—existed only at MIT. (This is another interesting characteristic of computer media revolution—it was theorized in detail before it occurred in practice.)

⁴ Photographs of my lab's work with the HIPerSpace super-visualization computer are available at <http://lab.softwarestudies.com/2008/12/cultural-analytics-hiperspace-and.html>. The HIPerSpace computer is described at http://vis.ucsd.edu/mediawiki/index.php/Research_Projects:_HIPerSpace

But during the second part of the 1990s PC hardware became advanced enough to run simulations of most media at a sufficient fidelity that was comparable with the professional standards already in place. This included software for graphic design, CAD, 3D modeling, 2D and 3D animation, print layout, digital photo manipulation, and audio and video non-linear editing. In many cases, these applications were now responding to users' actions sufficiently fast to approach the level of interactivity conceptualized by Kay and Goldberg in 1977 (although 3D animation and video compositing even today may require long rendering times). As a result, simulated media became truly useful, and also accessible, to large numbers of people outside of computer laboratories and large media companies. Within half a decade, most culture professionals abandoned physical media for their simulated equivalents.

When I visited well-known electronic musician, writer and artist DJ Spooky in his Tribeca apartment in New York in January 2005, I did not find any musical instruments, traditional or electronic. The only "instrument" Paul Miller (aka DJ Spooky That Subliminal Kid) owned was his 15-inch PowerBook laptop, made by Apple. This was his "Dynabook": a "self-contained knowledge manipulator in a portable package the size and shape of an ordinary notebook."⁵ Although this "Dynabook" did not have Smalltalk, it ran another programming environment which was powerful, fast and allowed for visual programming—MAX, the language of choice worldwide for tens of thousands of electronic musicians, VJs, dancers, theatre performers and others working with different forms of real-time performance.

While the evolution of hardware enabled dissemination of media software to professional communities in the 1990s, in the 2000s this evolution also enabled the new stage in software design and use—social media (2004–). The few companies which dominated the field of professional media applications (Adobe, Apple, Autodesk) were joined by a multitude of new companies and countless start-ups focused on developing tools and services for the Web and mobile platforms.

The new software categories include social networking (e.g. Facebook), micro-content services (e.g. Twitter), media sharing

⁵ Kay and Goldberg, "Personal Dynamic Media," p. 394.

websites (YouTube, Vimeo, Picasa, Flickr, etc.); consumer-level software for media organization and light editing (e.g., iPhoto); blog editors (Blogger, WordPress); and many others. Keep in mind that software—especially web and mobile applications and services designed for consumers—continuously evolves, so some of the categories above, their popularity, and the features of particular applications and social network services may change by the time you are reading this. One graphic example is the shift in the identity of Facebook. During 2007, it moved from being yet another social media application that was competing with MySpace to a “social OS” that aimed to combine the functionality of previously different applications in one place—replacing, for instance, stand-alone email software for many users.)

None of the software apps and websites of the “social media era” function in isolation. Instead, they participate in the larger ecology, which includes search engines, recommendation engines, blogging systems, RSS feeds, and other web technologies; inexpensive consumer electronic devices for capturing and accessing media (digital cameras, mobile phones, music players, video players, digital photo frames, internet enabled TVs); and technologies that enable transfer of media between devices, people, and the web (storage devices, wireless technologies such as Wi-Fi and WiMax, and communication standards such as USB and 4G). Without this ecology most web services and mobile apps would not be possible. Therefore, this ecology needs to be taken into account in any discussion of social networks and their software—as well as consumer-level content access and media development software designed to work with web-based media sharing sites. And while the particular elements and their relationship in this ecology are likely to change over time—for instance, almost all media content may eventually be available over computer networks; communication between devices may similarly become fully transparent; and the very rigid physical separation between people, devices they control, and “non-smart” passive space may become blurred—the very idea of a technological ecology consisting of a number of interacting parts which include software is not unlikely to go away any time soon. Thus, if one day I were to write a detailed account of social media, I would need to discuss consumer electronics, network architectures and protocols and other elements of this ecology as much as the social software itself.

Media after software

Any summary of a 100,000-word book of theoretical arguments can't cover all important points. However, I will risk this because I know that even an incomplete summary will still be useful for the readers. Here are some of the proposals developed in this book about the experiences and meanings of “media” for contemporary designers who create it using software applications, and for the users of interactive media applications and services:

- 1 The computer is not a new “medium”—it is the first “metamedium”: a combination of existing, new, and yet to be invented media. (This is the argument of Kay, which I take as my starting point.)
- 2 A “medium” (as it exists in software) is a combination of particular techniques for generation, editing and accessing content. (I use the generic term “access” as an alias to the longer list of terms—navigating, browsing, viewing, listening, reading, and interacting.) Softwarization virtualizes already existing techniques and adds many new ones. All these techniques together form the “computer metamedium.” Any single “medium” uses a subset of these.

New techniques and their variations are constantly being developed which changes the identity of each medium that uses them. For the users of popular commercial media software, a medium changes with each software release.

- 3 “What we identify by a conceptual inertia as “properties” of different mediums are actually *the properties of media software*—their interfaces, the tools, and the techniques they make possible for accessing, navigating, creating, modifying, publishing and sharing media documents.”
- 4 Following the same logic, “properties” of any media object are no longer fully defined by the content and formats of the files storing the information. They now also depend on the software used to access this object. For example, depending on whether the same image is accessed via a default media viewer, a consumer app for media access and

editing, or professional editing software such as Photoshop, its “properties” change significantly.

- 5 The techniques that make up the computer metamedium can be divided into two categories. General-purpose (i.e. “media-independent”) techniques are implemented to work in the similar way on all media types (for example: select, copy, search, filter, etc.) Media-specific techniques can only work on particular data structures (for example, you can increase amplitude of a sound track or reduce the number of vertices in a 3D shape, but not vice versa). Each software “medium” combines some media-independent and some media-specific techniques.
- 6 The idea of a data structure leads us to an alternative definition of a software medium. “As defined by application software and experienced by users, a ‘medium’ is a pairing of a particular data structure and the algorithms for creation, editing and viewing the content stored in this structure.”

From this perspective, each of the categories of media development software can be said to define its own “medium”—because the programs offered in each of these categories typically (but not always) share the same fundamental data structure. The examples that fit this are vector graphics editors, raster graphics editors, 2D animation and motion graphics software, 3D computer graphics software, sound editors, text processors, and HTML editors.

- 7 Following the first stage of the computer metamedium invention, we enter the next stage of media “hybridity” and “deep remixability.” “The unique properties and techniques of different media became software elements that can be combined together in previously impossible ways.” “Both the simulated and new media types—text, hypertext, still photographs, digital video, 2D animation, 3D animation, navigable 3D spaces, maps, location information—came to function as building blocks for many new media combinations.”

This condition is not the simple consequence of the universal digital code used for all media types. Instead, it is the result of the gradual development of interoperability

technologies including standard media file formats, import/export functions in applications, and network protocols.

- 8 The previous formulations lead us to view contemporary media development using a model of biological evolution and its concept of massive numbers of species that share common traits—away from the modern model of a small number of very different mediums with their unique languages. Instead of trying to place any particular project, app, or web service in some category selected from a small number, we can instead view it as a combination of the techniques selected from a very large pool. Some of these combinations occur more often; others may only occur once. The successful combinations become popular, leading to similar projects; and some may become design patterns used in numerous applications.

Software epistemology

One of the key ideas developed in this book is that the computer metamedium is characterized by “permanent extendibility.” New algorithms and techniques that work with common media data types and file formats can be invented at any time by anyone with the right skills. These inventions can be distributed instantly over the web, without a need for the large resources that were required in the twentieth century to introduce a new commercial media device with new functions or a new media format. Use of open source and free software licenses and web-based hosting services repositories such as GitHub encourages people to collectively expand existing software tools, which often leads to their rapid evolution.

The permanent extendibility of the computer metamedium has important consequences not only for how we create and interact with media, but also for the *techniques of knowledge in a “computerized society”* (“Knowledge in Computerized Societies” is the subtitle of the opening section of the celebrated 1979 book *The Postmodern Condition: A Report on Knowledge* by Jean-François Lyotard).

Turning everything into data, and using algorithms to analyze it changes what it means to know something. It creates new strategies

that together make up *software epistemology*. Epistemology is a branch of philosophy that asks questions such as what is knowledge, how it can be acquired, and to what extent a subject can be known. Digital code, data visualization, GIS, information retrieval, machine learning techniques, constantly increasing speed of processors and decreasing cost of storage, big data analytics technologies, social media, and other parts of the modern technological universe introduce new ways of acquiring knowledge, and in the process redefine what knowledge is.

For instance, it is always possible to invent new algorithms (or new ways to scale existing algorithms to analyze big data faster) that can analyze the already existing data in ways the previous algorithms could not. As a result, you can extract additional patterns and generate new information from the older, already analyzed data.

Algorithms and software applications that analyze images and video provide particularly striking examples of this capacity to generate additional information from the data years or even decades after it was recorded.

In *Blowup*, a 1966 film directed by Michelangelo Antonioni, the photographer who takes pictures of a couple kissing in the park uses the strategy of printing progressive enlargements of one area of the photograph, until a grainy close-up reveals a body in the grass and a man with a gun hiding in the trees.

During the time that this film was being created and unknown to its director, computer science researchers were already developing the new field of digital image processing, including algorithms for image enhancement such as sharpening of edges, increasing contrast, and reducing noise and blur. The early articles in the field show the blurry photographs taken by surveillance planes, which were sharpened by the algorithms. As I already explained earlier, today many of these techniques are built into all image manipulation software such as Photoshop, as well as in the firmware of digital cameras. They became essential to both consumer photography and commercial visual media, as every published photograph in mass media first goes through some software adjustments.

Contemporary DSLR models and high-end compact digital cameras can record images both in JPEG and Raw formats. With JPEG format, an image is compressed, which limits the possibilities for later extraction of additional information using software.

Raw format stores the unprocessed data from the camera's image sensor. The use of this format assumes that a photographer will later manipulate the photo in software to get the best results from the millions of pixels recorded by the camera. In his guide for the use of the two formats by photographers, William Porter explains: "Working with a Raw file, you'll have a greater chance of recovering highlights in overexposed clouds, rescuing detail from areas of shadow or darkness, and correcting white balance problems. You'll be able to minimize your images' noise and maximize their sharpness, with much greater finesse."⁶

This means that new software with better algorithms can generate new information from a photo in Raw format captured years earlier. (See the examples from Porter's article available online for a dramatic demonstration of this.⁷) In another example of software epistemology, in the demo presented at one SIGGRAPH annual conferences, a few film shots of 1940s' Hollywood film were manipulated in software to re-generate the same shots from a different point of view.

In visual effects production today, one of the most widely used operations is the algorithmic extraction of the position of the video camera that was used to capture a shot. This operation is called "motion tracking" and it exemplifies how information that is not directly available in the data can be inferred by algorithms. (The extracted camera position is used to insert computer graphics into the live action footage in the correct perspective.)

Another important type of software epistemology is *data fusion*—using data from different sources to create new knowledge that is not explicitly contained in any of them. For example, using the web sources, it is possible to create a comprehensive description of an individual by combining pieces of information from his/her various social media profiles and making deductions from them.

Combining separate media sources can also give additional meanings to each of the sources. Consider the technique of the automatic stitching of a number of separate photos into a single panorama, available in most digital cameras. Strictly speaking, the

⁶ William Porter, "Raw vs. JPEG: Which should you shoot and when?", techhive.com (September 19, 2012).

⁷ *Ibid.*

underlying algorithms do not add any new information to each of the images (i.e., their pixels are not modified). But since each image now is a part of the larger panorama, its meaning for a human observer changes.

The abilities to generate new information from the old data, fuse separate information sources together, and create new knowledge from old analog sources are just some techniques of software epistemology. In my future publications I am hoping to gradually describe other techniques—as I am teaching myself data mining and other algorithmic knowledge techniques of knowledge commonly used by contemporary software societies.

In the beginning of the book I asked: *what is media after software?* If artistic mediums were traditionally defined by the techniques and representational capacities of particular tools and machines (brushes, ink, paper, musical instruments, a printing press, a photo camera, a film camera, video cameras and editing equipment), what happens to this concept after most of these tools and techniques are simulated in a single software environment? In other words, what is a “medium” as defined by software applications used to create, edit, distribute, and access it?

While “media effects,” “media representation,” “media industry,” “media theory” and “media history” have been extensively discussed in large number of books and articles in a number of academic disciplines, this literature does not usually include analysis of software tools and platforms. In contrast, the vast universe of “how to” books, instructional videos and tutorials contains very little theory, because the goal of all these publications is practical instruction. (My search for “Photoshop” in amazon.com under books on August 12, 2013 returned 9,405 publications; the search for “After Effects” returned 1,201 results, and the search for “3ds Max” returned 1,972 results.) The aim of my book was to help bridge the gap between these two separate universes of theory and practice.

Following the question of what it means to create media with software took us on a long journey through a few decades. We did find some possible answers that I hope you have found interesting and provocative. But of course, since both this book and all my writing are directed first of all to media practitioners—professionals and students creating new software applications and tools, graphic designs, web designs, motion graphics, animations, films,

space designs, architecture, objects, devices, and digital art—you can do more than simply agree or disagree with my analysis. By inventing new techniques, or through the innovative application of existing techniques—and by finding new ways to represent the world, the human being, and the data, and new ways for people to connect, share, and collaborate—you can expand the boundaries of “media after software.”