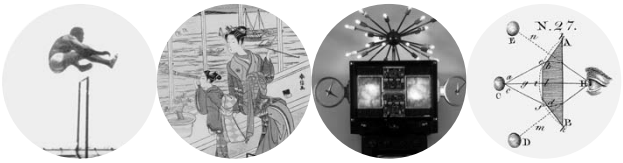
Media After Software

Lev Manovich

Journal of Visual Culture 2013 12: 30
DOI: 10.1177/1470412912470237

journal of visual culture



Media After Software

Lev Manovich

Abstract

While earlier reproduction technologies such as woodblock printing, moveable type printing, lithography, and photography represented media in ways accessible to bare senses, the media technologies of the late 19th century abandoned these formats in favor of an electrical signal. Simultaneously, they also introduced a fundamentally new dimension of media – interface (i.e. the ways to represent and control the signal). And this in its turn changed how media functions – its 'properties' were no longer solely contained in the data but were now also dependent on the interfaces provided by technology manufacturers. The shift to digital data and media software a hundred years later extends this principle further. With all types of data now encoded as sets of numbers, they can only be efficiently accessed by users via software applications. As a result, the 'properties' of digital media (how it can be edited, shared, and analyzed) are now defined by the particular software as opposed to solely being contained in the actual content (i.e. digital files).

Keywords

digital media • interface • media theory • software studies

We Have Never Been (Only) Digital

In the 1990s, a single term came to stand for the whole range of new technologies, new expressive and communicative possibilities, new forms of community and sociality that were emerging around computers and the internet. The term was 'digital'. It received its official seal of approval, so to speak, in 1996 when the director of MIT Media Lab Nicholas Negroponte

journal of visual culture [http://vcu.sagepub.com]
SAGE Publications (Los Angeles, London, New Delhi, Singapore and Washington DC)

Vol 12(1): 30–37 DOI 10.1177/1470412912470237

collected his *Wired* columns into the book that he named *Being Digital*. Many years later, the term 'digital' still dominates both popular and academic understanding of what new media is about.

When I did Google searches for 'digital', 'interactive', and 'multimedia' on 28 August 2009, the first search returned 757 million results; the other two only returned between 235 and 240 millions each. Doing searches on Google Scholar produced similar results: 10,800,000 for 'digital', 4,150,000 for 'web', 3,920,000 for 'software', 2,760,000 for 'interactive', 1,870,000 for 'multimedia'. Clearly, Negroponte was right: we have become digital.

I don't need to convince anybody today about the transformative effects the internet, the web, and other technological networks already had on human culture and society. However, what I do want to convince you about is the crucial role of another part of the computer revolution that has been discussed less. And yet, if we want to really understand the forms of contemporary media and also what 'media' means today, this part is crucial. The part in question is software.

None of the new media authoring and editing techniques we associate with computers are simply a result of media 'being digital'. The new ways of media access, distribution, analysis, generation, and manipulation all come from *software*. Which also means that they are the result of the particular choices made by individuals, companies, and consortiums who develop software – media authoring and editing applications, compression codecs, file formats, programming and scripting languages used to create interactive and dynamic media such as PHP and JavaScript. Some of these choices define general principles and protocols which govern modern software environments: for instance, 'cut' and 'paste' commands built into all software running under a Graphical User Interface and its newer versions (such as iPhone OS), or one-way hyperlinks as implemented in world wide web technology. Other choices are specific to particular types of software (for instance, illustration programs) or individual software packages.

If particular software techniques or interface metaphors which appear in one application – be it a desktop program, web application, or mobile app – become popular with its users, it may often soon appear in other apps. For example, after Flickr added tag clouds to its interface, they soon were added to numerous other web sites. The appearance of particular techniques in applications can also be traced to the economics of the software industry – for instance, when one software company buys another company, it may merge its existing package with the software from the company it bought. For instance, in 1995 Silicon Graphics bought two 3D computer graphics suites – Wavefront and Alias – and merged them into a new product Alias|Wavefront. Big companies such as Google and Facebook are periodically buying smaller companies and then add the software products these companies develop to their own offerings. Thus, one of Google's most popular applications, Google Earth, is based on software originally developed by Keyhole, Inc. acquired by Google in 2004.

Often, techniques developed for one purpose later migrate into another area, as happened when image processing techniques established in the second part of the 1950s for the analysis of reconnaissance photographs made their way into Photoshop in the late 1980s – now used to creatively modify images and to make photographs more 'artistic looking'.

All these software mutations and new species of software techniques are deeply social – they don't simply come from individual minds or from some 'essential' property of a digital computer or a computer network. They come from software developed by groups of people, marketed to large numbers of users, and then constantly refined and expanded to stay competitive in relation to other products in the same market category. (Google and Facebook update their code a few times day; GitHub, the popular software hosting services, updates its code dozens of times a day.)

In summary: the techniques, tools, and conventions of media software applications are not the result of a technological change from 'analog' to 'digital' media. The shift to digital enables the development of media authoring software – but it does not constrain the directions in which it already evolved and continues to evolve. They are the result of intellectual ideas by people who conceived of it in the first place (Ivan Sutherland, Douglas Engelbart, Alan Kay, etc.), the actual products created by software companies and open source communities, the cultural and social processes set up when many people and companies start using it, and software market forces and constraints.

This means that the terms 'digital media' and 'new media' do not capture very well the uniqueness of the 'digital revolution'. (I like the term 'media computing' – however, it is not used widely apart from some communities in computer science primarily in Europe). Why don't they work? Because all the new qualities of 'digital media' are not situated 'inside' the media objects. Rather, they all exist 'outside' – as commands and techniques of media viewers, authoring software, animation, compositing, and editing software, game engine software, wiki software, and all other software 'species'. While digital representation makes it possible for computers to work with images, text, sounds, and other media types, in principle it is the software that determines what we can do with them. So while *we are indeed 'being digital', the actual forms of this 'being' come from software.*

There Is Only Software

Accepting the centrality of software puts into question another fundamental concept of aesthetic and media theory – that of the 'properties of a medium'. What does it mean to refer to a 'digital medium' as having 'properties'? For example, is it meaningful to talk about unique properties of digital photographs, or electronic texts, or web sites, or digital maps?

The answer is no. Different types of digital content do not have any properties by themselves. *What as users we experience as properties of media content come from software used to create, edit, present, and access this content.*

This includes all media authoring and viewing application software made for both professionals and consumers, from Photoshop to your mobile web browser (it also includes custom software developed for particular products such as a DVD menu or an interface of an interactive kiosk). So whenever you think of 'properties' of digital media, you should always remember that this term means *software techniques defined to work on particular types of media ecologies, content and media data*. (Flickr's whole system for uploading, tagging, organizing, commenting and sharing images is the example of 'media ecology'; a raster 24-bit image stored in JPEG format is an example of a type of 'media data'.)

It is important to make it clear that I am not saying that today all the differences between different media types – continuous tone images, vector images, simple text, formatted text, 3D models, animations, video, maps, music, etc. – are completely determined by application software. Obviously, these media types have different representational and expressive capabilities; they can produce different emotional effects; they are processed by different sensors and networks of neurons in the brain; and they are likely to correspond to different types of mental processes and mental representations. These differences have been discussed for thousands of years – from ancient philosophy and classical aesthetic theory to modern art and contemporary neuroscience. What I am arguing is something else. On the one hand, *interactive software adds a new set of operations which can be applied to all these media types – which we as users experience as their new 'properties'*. (The examples include separation between data structure and its display, hyperlinking, visualization, and search interface.) On the other hand, *the 'properties' of a particular media type can vary dramatically depending on the software application used for its authoring and access*.

Let's go though one example in detail. As the example of media type, we will use a photograph. In the analog era, once a photograph was printed, all the information was 'fixed'. Looking at this photograph at home, in an exhibition, or in a book did not affect this information. Certainly, a photographer could produce a different print with a higher or lower contrast or use a different paper – but this resulted in a physically different object, i.e. a new photographic print that contained different information. (For example, some details were lost if the contrast was increased.)

So what happens with a digital photograph? We can take a photo with a dedicated digital camera or capture it with a mobile phone, or scan it from an old book. In every case, we end with a file that contains an array of pixels which hold color values, and a file header that specifies image dimensions, color profile, information about the camera and shot conditions such as exposure, and other metadata. In other words, we end up with what is normally called 'digital media' – a file containing numbers which represent the details of some scene or an object.

However, unless you are a programmer, you never directly deal with these numbers. Instead, most of us interact with digital media files via some

application software. And depending on which software you use, what you can do with a particular digital media file can change dramatically. MMS (multimedia messaging) software on your phone may simply display a photo sent by a friend – and allow us to forward it to somebody else but nothing else.

Free media viewers/players that run on desktops or over the web typically give you more functions. For instance, a desktop version of Google's Picasa 3.0 includes crop, auto color, red-eye reduction, variety of filters (soft focus, glow, etc.) and a number of other functions. It can also display the same photo as color or black and white – without modifying the actual digital media file.

Finally, if I open the same photo in Photoshop, I can do a lot more. I can instruct Photoshop to automatically replace some colors in a photo with others, make visible its linear structure by running edge detection filter, blur it in a dozen of different ways, composite it with another photo, and perform hundreds of other operations.

To summarize this discussion, let me make a bold statement. *There is no such thing as 'digital media'. There is only software – as applied to media (or 'content').)* Or, to put this differently: *for users who only interact with media content through application software, the 'properties' of digital media are defined by the particular software as opposed to solely being contained in the actual content (i.e. inside digital files).*

Data, senses, interface

'Digital media' is a result of the gradual development and accumulation of a large number of software techniques, algorithms, data structures, and interface conventions and metaphors. These techniques exist at different levels of generality ranging from a small number of very general (e.g. copy/paste) to thousands of very particular ones designed to do particular tasks – for example, algorithms used to generate natural-looking landscapes or software which can extract the camera position from live action footage in order to correctly align a 3D model when it is composited with this footage.

Because of the multiplicity and variety of these software techniques, it is unwise to try to reduce 'digital media' to a small set of new properties. Such reduction would only be possible if we could organize all these techniques hierarchically, seeing them as different applications of a few general principles. After thinking on and off about this for over 10 years (starting with my 1999 article 'Avant-Garde as Software' where I first tried to provide a taxonomy of these new techniques), I eventually came to the conclusion that any such hierarchy would only mislead us. The reason is that *all* these techniques *equally* change the identity of a single or multiple media type they can be applied to.

The fact that one technique may appear in many software packages designed to work with different media types (we can call them 'media independent'

techniques) while another technique may be specific to a particular type of media (we can call these techniques 'media specific') does not make the latter any less theoretically important than the former. For instance, because zoom function is present in word processors, media viewers, animation software, 3D modeling software, web browsers, etc., this does not make it more important than the algorithm designed to do only one particular thing in relation to one media type – for instance, a 'spheresize' command which modifies coordinates of all the points in a 3D polygonal model so it appears more spherical.

I don't think that we can qualitatively measure the practical effects on cultural production of both types of operations in this example to conclude that one is more radical than the other. Both operations change the media they act upon qualitatively, rather than quantitatively. They both add new qualities (or 'affordances') to media which it did not have before. A Word document which can be zoomed across multiple scales to reveal many pages at once has a different 'media identity' from one which cannot. Similarly, the ability to precisely sphere size a 3D model is a new way of working with spatial form which did not exist before 3D software.

In 'Avant-Garde as Software' (1999), I grouped all new techniques of digital media into four types based on what functions they support: access, generation, manipulation, and analysis. But even such simple differentiation appears problematic to me today – partly because of the evolution of software since 1999, which led to a gradual integration of these functions. For example, when a user selects a media file on his or her laptop, tablet or phone, the file automatically opens in a media player/viewer program. And today most media viewers and players (Windows Media Player, Apple's QuickTime Player, etc.) already offer some basic editing functions. Therefore, in practical terms today you can't simply 'access' media without automatically being offered some ways to 'modify' it. (To be clear, I am talking here about personal computers and mobile devices and not specialized hardware specifically designed to offer only access and prevent modification of commercial digital content – such as DVD players or MP3 players.)

How did we arrive at this new situation where, instead of looking at/hearing/reading content directly, most of us always experience content through the layer of applications? The seemingly obvious answer is the adoption of numerical code as the new universal intermediary. I call it intermediary because in order to make media accessible to our senses, it has to be analog – a traveling wave of oscillating pressure which we experience as sound, the voltage levels applied to the pixel elements of LCD which make them appear as different colors, different amounts of dyes deposited on paper by dye-sublimation printers, and so on. Such conversions from A to D (analog to digital) and D to A (digital to analog) are central for digital media functioning: for example, from the light waves to numbers stored in a file representing the image, and then back to the voltage levels controlling the display. Or, in another example, when we design an object to be printed on a 3D printer, an analog representation on the screen is translated by a

computer into a digital file that then drives the analog signals controlling the printer.

The two levels of encoding – first, a sampling of a continuous analog signal which results in its representation using a scale of discrete numbers (for example, 256 levels commonly used to represent grey tones in images), followed by a translation of this discrete representation into a binary numerical system – make 'media' incomprehensible for direct observation. The main reason for this is not the binary code per se (invented by the Indian scholar Pingala around 5th–2nd century BC) since it is possible to learn how to convert in your head a binary notation into a decimal one. The problem is that representing even one image digitally requires lots of numbers. For example, an image with HD resolution (1920 × 1080) contains 2,073,600 pixels, or 6,220,800 distinct RGB values – making it impossibly hard to comprehend the patterns such sets of numbers may represent if you examine these numbers directly. (In passing: because of these considerations, any digital image can be understand as information visualization – revealing patterns contained in its numerical representation.)

Because looking at such sets of numbers with our bare eyes is meaningless, we need to employ some technologies to translate them into analog representations acceptable to our senses. Most often, an image file is translated by digital hardware and software into an image appearing on our screen. However, a digital representation of one type of media can also be translated into another media type that is meaningful to our senses. For example, in audio-visual performances, software often uses video to drive sound, or reversely uses sound to generate abstract visuals. (Interestingly, the precursor to Thomas Edison's 1877 phonograph – the first device to record and reproduce sound – was Édouard-Léon Scott de Martinville's 1857 phonautograph that transcribed sound into a visual media. In other words, sound visualization was invented before sound recording and reproduction.)

From the beginning, technologies that generated and transmitted electromagnetic analog signals (e.g. a gramophone) included at least some controls for its modification such as changing signal amplitude. The first well-known electronic instrument invented by Leon Theremin in 1920 turned such controls into a new paradigm for music performance. A performer controlled amplitude (volume) and frequency (pitch) of a sound by moving his or her hands closer or further away from the two antennas.

Software significantly extends this principle by including more controls and more ways of representing the data. For example, I can choose to display this text I am writing now in Word as an outline, or select a 'print layout' which will show me boundaries of pages; I can choose to see footnotes or hide them; I can ask the application to automatically summarize the text; I can change different font families and sizes, and so on. Thus, while the actual data as it is represented and stored in a computer is no longer directly accessible to our senses, the new model of encoding and access has other significant advantages since the data can be formatted in a variety of ways.

This formatting can be changed interactively; it can be also stored with the data and recalled later.

We can articulate the relations between earlier electro-magnetic recording and reproduction technologies, which were developed in the last decades of the 19th century, and media software developed 100 years later (telephone: Bell, 1875; phonograph: Edison, 1878; television: Nipkow, 1884; radio: Fessenden, 1900). While previous reproduction technologies such as woodblock printing, moveable type printing, printmaking, lithography, and photography retained the original form of media, the media technologies of the late 19th century abandoned it in favor of an electrical signal. In other words, they introduced *coding* as a way to store and transmit media. Simultaneously, these technologies also introduced a fundamentally new layer of media – *interface*, i.e. the ways to represent ('format') and control the signal. And this in its turn changes how media functions – its 'properties' were no longer solely contained in the data but were now also dependent on the interface provided by technology manufacturers.

The shift to digital data and media software a hundred years later generalized this principle to all media. With all data types now encoded as sets of numbers, they can only be efficiently accessed by users via software applications which translate these numbers into sensory representations. The consequence of this is what we already discussed: all 'properties of digital media' are now defined by the particular software as opposed to solely being contained in the actual content, i.e. digital files. So what was already true for audio recording, radio, television, and video now also applies to text, images, and 3D objects and scenes.

In short: *media becomes software*.

Reference

Manovich L (1999) Avant-garde as software. In: Kovats S (ed.) *Media Revolutions*. Frankfurt: Campus Verlag. [http://manovich.net/articles.html]

Negroponte N (1996) *Being Digital*. Philadelphia, PA: Coronet Books.

Lev Manovich is the author of *Software Takes Command* (Continuum, 2013), *Soft Cinema: Navigating the Database* (MIT Press, 2005), and *The Language of New Media* (MIT Press, 2001) which is described as 'the most suggestive and broad ranging media history since Marshall McLuhan'. Manovich is a Professor at CUNY Graduate Center, a Director of the Software Studies Initiative, and a Visiting Professor at European Graduate School (EGS).

Email: manovich.lev@gmail.com