Executable movies: on the existence and propriety of networked images

Filmes executáveis: da existência e propriedade das imagens em rede

GABRIEL MENOTTI*

Universidade Federal do Espírito Santo, Departamento de Comunicação Social. Vitória-ES, Brasil

ABSTRACT

This paper explores the existence and propriety of moving images within computer networks. It means to show that computer-based images are executable, standing for running software systems as much as the applications that are used to play, edit, copy and transmit them. Thus, in computation, movie and apparatus become completely mingled together, developing a complex materiality. Analysing how computers enact and distribute moving images, I end up evoking the concept of code as a parameter that sets conditions for medial property. According to this parameter, different media might retain their specific identities even if they become equivalent surface effects of the same socio-technical system.

Keywords: Technicalimage, software studies, cinema, materiality, mediumspecificity

* Independent critic and curator. Holds a PhD in Media & Communications from Goldsmiths, University of London (2011) and another in Communication and Semiotics from PUC-SP (2012). Has worked as visiting lecturer at Middlesex University and Goldsmiths. Is currently a lecturer at the Social Communications Department at UFES. E-mail: gabriel.menotti@gmail.com

RESUMO

Este artigo explora a existência e propriedade das imagens em movimento em meio às redes de computador. Ele pretende demonstrar que imagens baseadas em computação são executáveis, representando processos de software da mesma forma que os aplicativos utilizados para reproduzi-las, editá-las, copiá-las e transmiti-las. Nessas condições, o filme e o dispositivo se tornam completamente misturados, desenvolvendo uma materialidade complexa. Evocando o conceito de código, a conclusão irá delinear como diferentes meios podem manter especificidades ainda que se tornem efeitos de superfície equivalentes do mesmo sistema sociotécnico.

Palavras-chave: Imagem técnica, software studies, cinema, materialidade, especificidades do meio

DOI: http://dx.doi.org/10.11606/issn.1982-8160.v9i1p243-263

Executable movies: on the existence and propriety of networked images

THE PROGRESSIVE DIGITIZATION of channels and information seems to be leading to a world in which no ontological distinction between media systems exists. According to theorists such as Friedrich Kittler (1999) and Lev Manovich (2008), in the new technological configuration all of these systems will be transformed into software layers of computer networks, which in turn will become "universal media machines" (Manovich, 2008: 30). This process of convergence, understood in terms of a dematerialization of media's underpinnings, seems particularly disturbing to the organization of cinema, a medium whose distinctive features have been traditionally based on the use of celluloid film as a physical support for the inscription of moving images. As film becomes obsolete and all kinds of images start to circulate primarily as patterns of binary data, is it still possible to differentiate cinema from other visual systems?

According to film scholar D. N. Rodowick, yes. Reflecting upon the "disappearance of the photographic ontology" (2007: i), Rodowick has stated that "while film disappears, cinema persists" (Ibid.). For him, it is the *young field* of cinema studies that has to change in order cope with the ways in which the traditional medium continues to exist within the present media ecology. One can reasonably assume that the perspective offered by cinema studies, as it approaches the medium's underpinnings in terms of the universal apparatus imagined by scholars such as Jean-Louis Baudry and Christian Metz, would overlook the technical complexity of computer networks. As a consequence, it would remain myopic to the ways in which these networks might express the cinematographic object.

In order to keep a critical hold over this object, Gertrud Koch suggests that cinema studies could either be transformed or merge with other fields of media scholarship (2009). This paper takes a step in the later direction, by attempting to connect cinema studies to even younger disciplines, more attentive to the materiality of processes of storage and transmission, such as the field of software studies. This emerging field is concerned with the interaction between software and culture that underpins "new representational and communication media" (Manovich, 2008: 4) and is therefore central to the constitution of cinema after computation. Drawing references from it, this paper means to explore the ways in which moving images are enacted and distributed through digital networks. In doing so, I hope to demonstrate that computer-based movies are *executable*, standing for running algorithms as much as the applications that are commonly used to play, edit, copy and transmit them.

Such demonstration will be accomplished through an analysis of the workings of computing mechanisms, supplemented with close examination of

And Then There Was Salsa (2010),[1] an audiovisual piece whose account provides exemplary expressions of the computer's mediatic qualities. Without necessarily focusing on the particularities of the piece's code, the paper means to show how its constitution as *software* blends the image into machine processes, making it dependent on different socio-technical and economic vectors. I believe that this combined approach will allow for a deeper understanding not only of the nature of digital movies, but also of the ways in which the specificities of cinema are maintained after media convergence, all the while being mobilized through new layers of operation and control.

1. A short video advertisement created by Goodby, Silverstein & Partners for a brand of tortilla sauce by Frito-Lay, published on the Vimeo website in February 2010.

PRINCIPLES OF COMPUTATION: EARLY COMPUTERS AND ALGORITHMIC ABSTRACTIONS

To understand the nature of the moving images that result from computation, one must first pierce through the main illusion maintained by digital technologies: that of the dematerialization of media. As Matthew Kirschenbaum has stated, "there is no computation without data's representation in a corresponding physical substratum" (2009: 27). This can be clearly perceived in the very earliest forerunners of the digital computer, such as the abacus. Dating from the BC era, this manual device organized calculus by the means of the movement of pebbles in a pre-defined grid. The correspondence between abstract and material operations was continued through the history of computing mechanisms until the Universal Turing machine, which N. Katherine Hayles characterizes as the "theoretical basis for modern computers" (2005: 176). The mathematician Alan Turing first mentioned this device in a paper called "On Computable Numbers, with an Application to the *Entscheidungsproblem*," published in 1936. In the text, Turing defines computable numbers as those whose "decimal can be written down by a machine" (1936), thus establishing the possibility of being inscribed as a precondition for their existence. Furthermore, the paper describes a universal machine able to "compute any computable sequence" (Ibid.). Albeit hypothetical, such mechanism seems physical throughout, consisting of a scanner supplied with segmented tape "analogue of paper." Computation occurs while this tape runs through the machine, which reads and writes symbols onto the discrete segments.

A few years later, working independently, the German engineer Konrad Zuse would find a technical solution similar to Turing's model which enabled him to create the first programmable, fully automatic computing machine, the *Z3* (in 1941). Z3's programmes were stored in a sort of punched tape that was none other than recycled 35mm film stock. Lev Manovich sees this fact as highly symbolic: for him, it represents the reduction of media "to their original condition

as information carrier, nothing less, nothing more" (2001: 25). Nonetheless, it is more likely that Zuse's reasons for adopting film were related to this material's physical affordances and availability. As pointed out by Andrés Burbano, film has certain qualities that favour the computer's mechanical operation. Firstly, its sprockets and frame division guarantee high accuracy to the step-by-step movement required for discrete (digital) calculus. Moreover, film can be bent, "creating 'loops' that would allow the machine to perform recursive operations" (Burbano, 2009: 9). A final, but perhaps more important detail, is that Zuse had easy access to this material, since his grandfather worked in the German film industry (Ibid.: 7).

The Z3's architecture and mode of operation make clear that there is no such thing as an immaterial dataset. Just like present-day digital computers, Zuse's machine operated according to the binary numeral system, meaning that the symbols it employed in data processing and storage were just 0s and 1s. However, these "symbols" were not the pure representation of abstract values that the computer "read" – they were an arrangement of physical structures, as constitutive of the computer mechanism as cogs are part of a gear system. 0s and 1s actually refer to 1) the presence or absence of holes in the film, which mechanically induced 2) the position of the relay switches that constituted the machine's processor, which in turn defined 3) the on-off state of lamps used as the computer's output display.

In the Z3 and other early computing machines, these arrangements were interpreted as bits, and then translated into mathematical values and operations, by specialized human agents called "computers" themselves (Hayles, 2005: 1). However, as electronic technologies evolved and the machines grew in complexity, the interpretation of data by the computer was integrated as part of its input and output structures. These processes of translation were internalized as layers of software abstraction, which code and decode binary patterns prior to human operation, translating these patterns into forms closer to ordinary symbolic systems (Ibid.: 108). In other words, they express the physical organization of the machine as information (such as numbers and text) that a human operator can actually make sense of. As primary examples of software abstraction, one can refer to the different programming frameworks (which aim to approximate the machine's syntax to that of everyday languages) and user interfaces (which aim to represent data according to useful metaphors). More specialized cases would be applications with a defined purpose, such as web browsers and movie editing suites.

Abstractions are necessary for computer-based media because they bridge the gap between the computer's unfathomable procedures (for example,

lighting up a complex sequence of coloured pixels on a screen) and its mediatic uses ("playing a movie"). However, as they do so, abstractions also divert the operation of the computer away from the actual processes of computation. As Friedrich Kittler has said, software hides the machine from its users (1995); it makes users overlook the very physicality of the computer, along with its particular kinetic and visual qualities, and confine themselves to metaphoric representations – as if the computer was the dynamic "desktop," with its neat icons and resizable windows, and not an electronic machine for information processing.

This incorporation of increasingly sophisticated abstractions to computer systems, coupled with the exponential increase of storage capacity and speed of transmission (Kirschenbaum, 2009: 34), are the main reasons why computer--based media *behaves* as if it was immaterial. However, at its core, even the most modern computer is a mechanism not so very different from the Z3 – or from an abacus, for that matter. In spite of their physical differences, they all operate according to the same principles.

What principles are these? First, that computation implies particular forms of *organized movement*. While this may be obvious in an abacus, in which the sliding of pebbles is a clear operation, it becomes predominant in the electronic computer, in which even stable data is motion-dependent. The binary patterns are not stored in a hard drive as fixed electromagnetic traces, in a way that is immediately apprehensible just as the holes in a punched tape. They are changes of voltage in the electric current flowing through this device, and therefore they only exist when the computer is running. According to Matthew Kirschenbaum's description,

> the read/write head [of the hard disk] measures reversals between magnetic fields rather than the actual charge of an individual magnetic dipole. In other words, [the hard disk] is a differential device – signification depends upon a change in the value of the signal being received rather than the substance of the signal itself (2009: 90).

The other fundamental principle to be inferred from the computer's structure is that these patterns of information stand not only for datasets, but also for the instructions of data processing. In the Turing machine, for instance, the symbols written on tape not only represented input and output values; they also controlled all of the formal procedures of calculation. In more complex digital computers, it is implied that "all code operations" – all software abstractions – always come down to the same thing: the aforementioned "signifiers of voltage differences" (Kittler, 1995). Therefore, computer-based media would entail no

strict division either between inscription and transmission, or between datasets and instructions: in the computer, everything is reduced to the continuing information of the mechanism. This fact can be summed up by Friedrich Kittler's idea that *there is no software*. In other words, that software is not something running in a computer – *it is the computer running in a particular way.*

What do these principles of operation tell us about the way computers enact moving images? First, the essential equivalence between datasets and instructions allows us to say that the movie file is software. More precisely, it is a pattern of abstract information: a series of formal procedures that organize the computer mechanism according to human systems of representation (Kittler, 1995). As such, the movie file is insufficient, since it is not able to control the operation of the whole system by itself alone. It must always interact with other algorithms (such as the operating system and the media player) in order to produce images. This means to say that the movie is not *in* the movie file – the *movie* results from the way the computer runs, as it translates and incorporates the instructions partly contained in the movie file. Down to its core, the file is just a collection of codified binary data, which "means" nothing. If the system does not "know the rules" needed to decodify it, it will not be able to assemble images from the file.

Of what do these rules consist? Besides the use of the right software, digital movie reproduction also requires the proper *codecs*. Codecs are algorithms for encoding and decoding audiovisual information: "they scale, reorder, decompose and reconstitute perceptible images and sounds so that they can get through information networks and electronic media" (Mackenzie, 2008: 48). Establishing a universal paradigm for the compression of information, codecs reduce drastically the size of a movie file, allowing its effective storage and transmission through digital means. J. D. Lasica states, for instance, that it is only because of the MPEG-2 codec that DVDs became a viable support for movies. This codec reduces by 97% the amount of data needed to store moving image information, making it possible to contain an entire feature film on one versatile disc of 4,7 gigabytes without any significant loss of quality (Lasica, 2005: 88).

One could say that codecs set standards of data organization, making it possible to run the same audiovisual information in the most diverse hardware. A movie codified in the popular h.264 format (a standard for video compression) can equally be played in Linux operating systems, in portable media players made in Taiwan, in the last generation of mobile phones and in the Apple TV device. Contrariwise, if the right codec is not installed, the movie cannot be viewed at all (Mackenzie, 2008: 48). Thus, codecs not only "influence the very texture, flow, and materiality of sounds and images" (Ibid.), but they are a

constitutive part of the movie visuals, as important as the movie data file. It could be even said that, when a movie is watched on a computer, what is being seen is the work of codecs unpacking and organizing binary data according to their own complex spatial logic (Ibid.: 51).

THE SYSTEM'S VISUAL PERFORMANCE VISUAL AND USER INTERACTION

To understand the nature of the images produced by computer interactions further, one must also consider the inherent visuality of the machine. It is almost redundant to say that the interfaces that supposedly frame and control the playback of a movie in a computer are software themselves. However, considering the interdependence between movie and apparatus, this fact has strong implications for the nature of the digital movie. It means that these interfaces are not fundamental, physical underpinnings: as much as a digital movie, they are rational organizations of the machine, resulting from the way the computer processes information. From this perspective, a media player window is no more of a structure than the movie playing "inside" of it is. Ignoring the subtleties of computer architecture, it might be said that both the movie and its frame – including the dashboard that allows its random navigation and even the operating system in which the media player is being executed – result from the same interaction of algorithms. All of these "layers" are produced concomitantly while the computer is running; thus everything that is on the screen is of the same nature as a real-time abstraction of the machine.

This means that the "movie" and its "interfaces" are equivalent as surface effects. Any distinction between the image that is inside a media player window and those that are outside of it (such as control buttons or a sliding timeline) does not arise from computation. Of course, these images are different, but not in an intrinsic way. What differentiates one from the other is not that they behave or react differently to user interaction, because they are not reacting in the first place – the computer is. On the contrary: what differentiates one image from another is that *the user* reacts differently to them. This separation between "movie" and "interface" comes from expectations the public has about the machine, which drives the way they engage with it. These expectations make the users overlook the fact that the movie is no more spectacular than the system in which it is being played.

User interfaces have particular visual qualities that are themselves mediatic. This might be hard to perceive because, in everyday computer operation, the act of viewing is just a parameter for the reflexive agency over the system. In these normal situations, the gaze is specialized and becomes part of the

machine. The operator is so immersed in an image that it becomes difficult to watch it: the *optical* dimension only matters while it is subjected to the *haptical* one. Not surprisingly, computer screens are still called *monitors*. The screen seldom exists in function of the mouse, the joysticks and the keyboard. The image seldom appears to make the manipulation of datasets possible; it is an input and output channel.

Nevertheless, when we distance ourselves from the operation of the machine, the sheer visuality of the computer is made evident. Such distance can be historical: once more, it is fruitful to look at early computer mechanisms to realise that the processes of computation have a strong visual dimension. For instance, one could mention the collateral images formed by the inscription of data in punched cards, or the mechanical animation of rotating drums created by the calculations of an analytical engine.[2] As stated above, the first computer operators had to interpret such patterns in order to turn them into data. Nowadays, the machine interprets data into visuals beforehand, and therefore allows even more complex operations. These visuals include not only the mere transformation of binary values into symbols that the users can readily manipulate (numbers, words, images) but a whole set of choreographies that the interfaces perform in order to be more appealing: applications create smooth waves on the screen as they pop up and twist when they are minimized; menus have adjustable transparency and cast shadows over one another; the desktop pulses with indications of weather forecast or simulated lava lamps.

2. The first programmable computer, conceived by Charles Babbage in 1837.

These "eyecandy" effects do not directly contribute to the system performance – i.e. to the amount of useful activity the computer can accomplish with the available resources. In fact, they do quite the opposite, as they consume processing power and memory that could be employed in the "proper" manipulation of datasets. Even so, they *do* favour computer operation because they provide different forms of feedback to the users, making the interaction with the machine more organic. Hence, this sort of *visual performances* of the system exists to foster its transparency, making its operation more seamless and dynamic. The main reason these performances are difficult to perceive is precisely because one of their functions is *to create the invisibility* of the machine.

Let's consider for instance the plastic qualities of Vimeo, one of the most popular online video hosting platforms. The playback interface it uses is customized to match the website's visual identity, with its solid colours and round edge geometry. Whenever a video page is loaded, a series of standard control buttons can be seen overlaying the player window. As the mouse pointer hovers over them, these buttons change colour, as if to indicate their responsiveness.

Once the buttons are "pressed" and the video starts playing, the interface fades away, cleaning up the view to the movie. These subtle animations are produced by computer operation as much as the video being shown inside the window. Nonetheless, they have been designed with the particular objective of denying their own visuality. The animations seem to make the interface less of an autonomous visual form and more of an operational aspect of the website's overall structure – less *moving images*, more *mechanism*.

Despite the impression created by the design of Vimeo's control buttons, there are no fundamental boundaries between images and interfaces in a computer system. This is clearly demonstrated in the piece *And Then There Was Salsa*, a short video advertisement created by Goodby, Silverstein & Partners for a brand of tortilla sauce by Frito-Lay, published on the website in February 2010. At first this piece seems to behave like any of the others to be found on Vimeo. Once the user presses the play button, the video starts running as expected, within the borders of the movie player window. It depicts a 3D-modelled flamenco dancer on a hill of vegetables. However, as this character swirls through the scenario, the animation escapes its regular frame and takes over the entire browser window. First, the layout explodes, covering the whole webpage with lush vegetation. Then the player window enlarges, while tomatoes, onions and jalapeños start flying all around it (fig. 01). The dancer slides across the background of the page and proceeds to slice Vimeo's logo just as she had sliced the cartoon vegetables, revealing that one image is no more "cinematographic" than the other. Thus what first appeared to be just an operational interface, contingent to the movie, becomes its most appealing part.

FIGURE 1 – *And Then There Was Salsa* shown on Vimeo

And Then There was Salsa literally mobilizes the characteristics of computer interfaces in favour of a visual spectacle. The piece brings into question the distinction between operational and spectacular images. Thinking along these lines, one might wonder if there really are any limits to the interaction possible with graphical interfaces and the spectation of digital movies. The engagement with a computer system regularly involves activities such as handling the mouse, pressing metaphoric buttons and focusing on certain portions of the screen. The way the user suppresses awareness of these activities does not seem to be any different from the way a moviegoer ignores his own situation in the theatre in favour of diegetic experience. In both cases, it is the engagement of the public that abstracts certain technical processes from the machine's performance, while making others relevant to the meaning and value of the work. Thus, while the computer mechanism does not separate spectacle from operation, its user does, and in that way organizes the experience of media technology.

THE CONSTITUTION OF THE IMAGE IN THE COMPUTER

Apart from the lack of a fundamental distinction between movies and interfaces, there is another, more important characteristic of digital movies that can be inferred from their nature as software: the fact that they depend on a supply of constant information from the system. In the first place, digital images are never properly *inscribed*. Even when they are supposedly "stored" in a hard drive, they only exist during the transmission of electricity through this mechanism (Kirschenbaum, 2009: 95). Moreover, since they are stored as *codified* datasets, they can only be displayed upon their real-time decodification by the machine. Therefore, even the most static images, while they are being shown on a computer, are a consequence of procedural interactions – the effect of the system's unprecedented activity.

The system activity defines the way movies are enacted by digital computation. Far from being reproduced or represented, movies literally result from the continuing operation of the machine. Again, this subtle difference can be better explained by referring to earlier, simpler computers. Let's make an analogy between a modern-day PC and the Turing Machine. At first glance, one might think that a movie playing on a PC would be equivalent to the symbols written on the Turing Machine's tape once it has finished its calculation. However, this is not the case. These symbols are the final outcome of the Machine operation. In the previous subsection it was demonstrated that an image cannot be detached from the performance of the computer in such a definitive way and still be considered computer graphics. In that sense, it would be more illuminating to compare a movie playing on a PC to the *tape* of the Turing Machine, or to

the complex patterns of movement that the tape undergoes, as it is erased and rewritten during the process of computation. Rather than a product of the computer, the image should be taken as a real-time index of it – as a hint of the electricity that flows from the power source, passes through the processing unit and ends up exciting the pixel grid of the screen.

Before becoming any form of representation, a digital movie is just a fleeting trace of the running computer. This fact constitutes a strong argument against the myth that a digital image never degrades, and that it can be copied and reproduced without any loss. If we accept that a digital image is this real-time index, then we must assume that the image is never preserved and cannot be copied at all. However, this is not because the digital representation misses the "literal spatial and temporal molding of the originating event" that Rodowick defends as being the causal force fundamental to the photographic image (2007: 11). Such representations do have an originating event, and one from which they cannot be disconnected, which is the computation. Conversely, it is precisely because of this material bond that digital images cannot be stabilized. After all, the substratum necessary for computation is not an infinitely reproducible mathematical construct, but a "messy world of matter and metal" (Kirschenbaum, 2009: 27). In other words, digital images are too material to be autonomous. Even less than photographic or electronic images, the digital ones cannot be separated from their circulation.

Thus, if digital technologies "propagate an illusion of immateriality" (Ibid.: 135), it is because they further intensify movie circulation, just as video did before them. Electronic transmission first promoted a dialogical regime that shortened the gap between movie production and consumption. Computer processing, in turn, effaces the very division between operations of inscription and transmission, merging the enactment and distribution of forms. It increases the rhythm and scale of circulation to the point that the nature of the movie as a process of information can be made apparent to human perception.

THE MOVIE THROUGH NETWORKS: THE PEER-TO-PEER MULTIPLICATION OF FORMS

Just as the storage and sheer display of an image in a local machine involves active processes of data decodification, so does the copying and transmission of a movie through computer networks. To understand this, one could consider the operation of peer-to-peer (p2p) file sharing, a model of data distribution that has been initially associated with piracy, but is increasingly becoming a sort of alternative standard for the film industry. As I have noted elsewhere, one of the reasons for the growing popularity of this model is the technical efficiency

with which it employs the dispersed structure of the Internet (Menotti, 2012). In p2p networks, users do not download files from an exclusive central server, but instead from one another – hence the name. In order to optimize network traffic, particular p2p protocols assure that singular files do not come entirely from one source. This means that when a movie is acquired by the means of p2p file sharing the data is actually transferred in pieces from different locations, and is only assembled as a coherent video at the end of the process.

Taking the distributed dynamic of p2p into account, artist Sven König and the !mediengruppe bitnik propose an analogy between the operation of codecs and that of data transmission. They suggest that, just as a movie running in a computer is not simply contained in the movie data, a "film" found on a file sharing environment is not a mere version of the original film. It is a completely new work, which results from a process described as

> the sum of [>1] the original film, [>2] the work of the mathematicians who laid the theoretical foundations for [>3] the programmers who designed the encoding software / the codec and [>4] the file sharer who finally uses all that software to intentionally make the [>5] film widely available. The processes behind [2] - [4] usually stay invisible, leading to the wrong assumption that [1] = [5] (!mediengruppe & König, 2007).

In that sense, we should assume that the invisible processes of information that produce computer images are not only internal to the machine, but also environmental, depending on the coordination of the transmission and preservation of audiovisual data in a broader infrastructure. According to this perspective, the obstacles for the complete autonomy of the image are even more plural, beginning with the high disposability of standards for data storage and codification (Usai, 1999: 44). As the technical milieu changes, digital movies have to be translated into new formats or become "hieroglyphs," as Paolo Cherchi Usai called them (Ibid.: 46). For that reason, Matthew Kirschenbaum describes the eventual fate of all digital objects as to

> inexorably be reduced to opaque code blocks, or BLOBs, as they become detached and drift away from their native software environments, and as those software environments themselves become distanced from the hardware running the operating systems that support and sustain them (2009: 234).

Hence, the preservation of digital objects depends on access to systems capable of hosting and interpreting them (Kirschenbaum, 2009: 186, 189). Such environmental reliance can be explained by referring again to *And Then There Was Salsa*. This piece was located in a specific video website, and it could not

be available anywhere else. Its attachment to Vimeo went beyond the space it took on its servers and the way it was promoted within the website's community of design-savvy users. In order to make the vegetables fly all over the browser, the movie had to appropriate the visual interface of the platform and, more importantly, to interact directly with its software structure. The filmmakers not only had to produce the animation in accordance to Vimeo aesthetics, but also to contact its webmasters in advance, in order to understand how the site works and be able to install specific algorithms into it. Of course, all of these definitions demand long-term business negotiations, and it is no surprise that the user account which owned the video had a badge of "sponsor" next to it.

As could have been expected, *And Then There Was Salsa* had the same tragic destiny of other digital objects, once the necessary environmental conditions were gone. The original movie can no longer be found online, being substituted by an error message that states that it was deleted from Vimeo on the 7 January 2011. What probably happened is that the contract between the advertising agency and the website expired, causing the page to be taken offline and thus effectively destroying the work. In that sense, the existence of a movie within computer networks seems to be extremely fragile, as it can be terminated at any moment, according to the decision of the network managers – a decision that might be taken for legal or political reasons, as well as business-related ones.

On the other hand, it might be argued that these movies subsist precisely by being radically dislocated and transformed. Challenging the idea that the transmission of digital objects is a purely technical undertaking, Kirschenbaum insists on the "fundamentally social" dimension of these objects, whose existence relies on particular cultural practices (2009: 21). The example he uses to illustrate this point is the preservation of the self-degenerative codework *Agrippa* by hackers who made versions of it and distributed them in underground BBS[3] forums (Ibid.: 218). In that sense, it could be argued that a digital movie is likewise "preserved" by being re-encoded and multiplied through different channels, while being converted to their particular image formats and resolutions. For instance, although the "sponsor" account that first published *And Then There Was Salsa* has disappeared, it does not mean that the movie has been completely wiped from the Internet. A number of versions of it can still be found on Vimeo or other video platforms such as YouTube, uploaded by users who probably do not have any relation to its original makers. Some of these versions are mere reproductions of the video, lacking the animations outside the player window that characterized it. Others, however, are screen captures that show the original video in the context of its animated webpage, providing a more comprehensive documentation of how it functioned (fig. 02).

3. This acronym stands for *Bulletin Board Systems*, systems that allowed people to connect their computers to one another through a terminal program. BBSs were popular in the 1990s, before the World Wide Web.

FIGURE 2 – A version of *And Then There Was Salsa* hosted on YouTube.

Such multiplications of the movie could be seen as its final development as a process of information, which interweaves the meaning and value of the piece into the very social fabric of the system, blurring the lines between media consumption and production. One could argue that this condition of existence is promoted by the way in which the public is embedded in the structure of computer networks. On the Internet, just as in a game of Chinese whispers, the audience is actively carrying out the process of cinematographic distribution. With this radical transformation of how the public is engaging with the underpinnings of the medium, the nature of movie circulation changes as well, foregrounding the relevance of distribution as a sort of collective enactment of the movie.

Therefore, digital technologies seem to embed the image even further within the medium's structure, allowing platforms of distribution to get within the movie and use it to propagate themselves. Movies acquired from p2p networks, for example, often carry a text file listing the release group responsible for its original upload, as well as the online directory to which the movie was first posted. This metadata is made in order to spread the names of such release groups and directories, ensuring their influence in the particular economy of reputation that motivates the file sharing scene (Lasica, 2005: 53-55). The pervasiveness of a website such as YouTube is even stronger. One might think that embedding the documentation of *And Then There Was Salsa* in another webpage would isolate it from the original context, but what happens is precisely the opposite. The embedded video becomes overlaid by YouTube's watermark

and advertisements, as well as by links to other works in its database. It is as if, through the embedded video, the whole of YouTube could infiltrate a different website. Being this invasive, the platform reinforces its superlative authority over the works it distributes – an actual form of control, better expressed by its capacity to ban user accounts, take videos offline and block access from certain geopolitical areas.

CODE AND THE CONDITIONS OF PROPERTY

The perspective of software studies demonstrates that processes of computation fuse movie and apparatus together. On the one hand, digital applications are operated by means of the very visual effects they supposedly produce. On the other, the image completely dissolves into the procedures of its storage and transmission. In such conditions, cinema can only exist by the means of the continuing interactions between physical and logical mechanisms, both among themselves and with human users, on both local and networked platforms. While this does not imply a dematerialization of the medium, it certainly makes it much more volatile. On what basis, therefore, can we still separate cinema from other media systems and practices that rely on computer technology?

In this final section, the answer to this question will be delineated by referring to the notion of *code*. Characterized as a standard or protocol of communication, the code seems to establish conditions for mediatic specificity in a computational environment. This explanation will draw from the wider historical perspective advocated by Friedrich Kittler, for whom codes "are not a peculiarity of computer technology and genetic engineering" (2008: 40). Kittler finds the roots of code in pre-Christian systems of command and communication that are operated by means of encryption. He characterizes this process in accordance with Wolfgang Coy's definition as a "mapping a finite set of symbols of an alphabet on to a suitable signal sequence" (Ibid.). Primary examples of these systems are forms of secret writing as old as the Roman Empire, used either by the government or by conspirators. Such ciphers are produced by reorganizing a message according to a given *key*, which works as a particular convention within the universal conventions of language. The circulation of an encrypted message is thus restricted to those who share this convention. Therefore, even if the physical carrier of the message is intercepted by someone unaware of the key, its meaning would not be disclosed. In that sense, it is fair to say that the message is not simply *contained* in the cipher – the message *occurs* when the cipher is *operated* through its key.

The cipher that supposedly contains the algorithmic procedures of a computer application is sometimes called its *source code*. This moniker is misleading

because it gives the impression that the code stands for the fundamental cause or essence of the application. On the contrary, source code is always conditioned by a sort of key: the rules of the programming architecture in which the application is supposed to run. Therefore, the real "source" of an application would be the complex series of software interactions that decipher its code. Nonetheless, according to the principles of computation described earlier, one must assume that these interactions are themselves abstractions of the machine's processes. When the computer operation comes down to these processes, not even the zeros and ones of machine language really exist – they are a mere description of the discrete states of the running mechanism.

Considered in this light, one can start to see the difference between the ways a Roman conspirator and a computer "decipher" code. Whereas the former translates code into a message he can understand and then performs its orders, the later performs the message by the means of its physical structure, incorporating the code before translating it into meaningful effects. This means that the machine does not truly understand the meaning of commands such as "go to" and "print"; it only responds to the way these commands affect its physical mechanisms. For that reason, the algorithmic logic described in a source code is as *superficial* as the images shown on a monitor screen, in the sense that they are equivalent abstractions of the machine's unfathomable operations. It therefore seems that code is not the fundamental cause of computer operations, but instead is another rationalization of processes that are latent within the machine. Put differently, codification would not be a genuine contingency for the operation of the computer, but another of its surface effects.

This fact seems to contradict what I have exposed so far. After all, if a movie is codified in a particular format, it cannot be played on a machine that does not have the proper codec installed. It might therefore be thought that code defines fundamental possibilities for movie circulation. Yet, how this assumption is already tied to a specific notion of the movie and of what its proper manifestations should be should not be ignored. Provided that there are physical means, a movie file in an unspecified format *can run* on a computer that does not have the corresponding codec installed; it just does not provoke the continuous sequence of images one would have expected. Nevertheless, the computer does process the movie as information. Even when the machine "does nothing" this does not mean that a million calculations have not been performed within it. However disappointing it may be for the user, an error message *is* a meaningful effect of these calculations. It stands for the way in which the computer exerts its material affordances, just as it does when a keyboard switch is closed or the processor overheats.

In that sense, more than defining fundamental possibilities for a movie to exist, code specifies the conventional means by which digital computers can be used in a "cinematographic" way. By establishing these conventions, code organizes technology in favour of what would be the proper circulation of a movie. In doing so, codification allows for a layer of medial operation, comprising a set of normal procedures and specific data standards. By employing such procedures, users do not have to worry about closing the right switches and lighting pixels up; they can just use the machine to play and create movies. In a similar way, data standards promote the uniformity of rendering routines – that is, of the way images are constituted by different computer mechanisms. Complying with these standards, a movie file can be handled by the most diverse systems in the same way.

This layer of medial operation allows for a common ground upon which movies can circulate according to the historically constituted norms of cinema. Moviemakers do not have to be concerned with computation; they do not even need to understand how the codification and decodification of data works. Thanks to codified standards, they are free to make cinema as they always did, employing applications whose interface simulate established practices of film production. Their work finishes where that of codecs starts: packing and unpacking bits into complex signifying arrangements.

In this way, code organizes computers in favour of movie circulation, outlining shared platforms for the production and consumption of cinematographic works. By coordinating the simulation of the medium's traditional apparatus and operations, code defines what is proper to them. Anything that is codified as a movie, and therefore can be operated as a movie, *is* a movie. Allowing for this purely arbitrary definition of the cinematographic object, the implications of code are equivalent to those of a *protocol*. Alexander Galloway once used the notion of protocol in order to describe "how control exists after decentralization," in a distributed network fitting the model of Deleuzian control societies (Galloway, 2004: 29). This idea could also be used to explain how the specificity of the medium persists in a post-mediatic digital system – in other words, how it is still possible to identify "cinema" when cinema is no different from other computer-processed bytes.

Drawing from computer engineering, Galloway characterizes protocol as "a set of recommendations and rules that outline specific technical standards" that is at the core of network computing (Ibid.: 6). By means of compliance with a protocol, participants are able to connect with one another and form a previously nonexistent network. In that respect, a protocol is a technology of inclusion (Ibid.: 147). However, from the way it is embedded into the system, a protocol is also a very powerful technology of regulation, which synthesises the

negotiation of flows structuring a network (Ibid.: 74-75). Therefore, protocol does not simply belong to the realm of *discourse*, but to that of *possibility* (Ibid.: 52-53), establishing conditions of presence *in* and *of* a given system. Since the sheer existence within the network depends on the acceptance of protocol, there can be no resistance to it: "opposing protocol is like opposing gravity" (Ibid.: 147).

The particular subject of Galloway's investigation is the Internet, the global network of networks. In rough terms, the Internet works as an agreement that both distributes information indiscriminately and regulates this distribution hierarchically. This shows that the freedom of universal connection depends on the submission to protocolar control. One should not think of this control as intrinsically harmful. Without it, the network would lose its coherence: "if the Internet were truly rhizomatic, […] it would resist the deep, meaningful uses that people make of it everyday" (Ibid.: 64). In that sense, one of the main functions of protocol is to sew the fragmented architecture of the network into an intelligible platform such as the Web, which users can experience intuitively. Likewise, by the means of data standards, computational processes are sewn into cohesive cinematographic operations, setting the conditions for movie circulation in the new technological regime.

In that way, medial parameters for the engagement with digital systems are established. As computers are turned into the medium of all media as prophesised by Kittler (2010: 225), these parameters become necessary in order to situate cinema within itself. Approaching pure specification, code defines what is proper to the medium – what ultimately belongs to its circuit. Within digital technologies, what is not codified as a movie *is not a movie*. Inasmuch as it may *look like a movie*, it cannot *circulate as such*: an operational system would not identify it with the proper icon; a DVD player would not be able to run it; a film festival would never accept its submission.

An even more critical issue is how these dynamics of codification blend the question of what is proper to the medium to that of *to whom the medium is a property*. As digital technologies make the existence of the movie contingent upon video codecs and platforms of online distribution, they subject cinema to the patents of the former as well as to the policies of the later. In that sense, computer networks create much more invasive layers that allow for greater economic and legal control over media, but which nonetheless remain hidden under the pretence of neutral structures. In order to understand how these layers are affecting the continuing development of cinema, and how their prerogatives can be challenged or displaced, it seems necessary to make a constant effort to reach out beyond the screen, employing methods that illuminate the complex bias of technological development. M

WORKS CITED

!MEDDIENGRUPPE Bitnik & KÖNIG, Sven. what is the characteristic structure of p2p films? *Download Finished – FAQ*. 2007. Available at: <http://www.download-finished.com/faq.html>. Access in: 14 Aug. 2011.

BURBANO, Andrés. Between punched film and the first computers, the work of Konrad Zuse. *Re: Live Conference 2009*. Melbourne: The University of Melbourne & Victorian College of the Arts and Music, 2009.

GALLOWAY, Alexander. *Protocol*: How Control Exists After Decentralization. Cambridge, Mass: MIT Press, 2004.

HAYLES, N. Katherine. *My Mother was a Computer*. Chicago: University of Chicago Press, 2005.

KIRSCHENBAUM, Matthew. *Mechanisms*: New Media and Forensic Imagination. Cambridge, Mass: MIT, 2009.

KITTLER, Friedrich. There is no Software. *CTheory*, 18 Oct. 1995. Available at: <http://goo.gl/saM3m>. Access in: 14 Aug. 2011.

_____. *Gramophone, Film, Typewriter*. Stanford: Stanford University Press, 1999.

_____. Code. In: FULLER, Matthew (ed.). *Software Studies*: A Lexicon. Cambridge, Mass: MIT Press, 2008. p. 40-47.

_____. *Optical Media*. Oxford: Polity, 2010.

KOCH, Gertrud. Carnivore or Chameleon: The Fate of Cinema Studies. *Critical Inquiry*, v. 35, n. 4, p. 918-928, 2009. DOI: http://www.jstor.org/stable/10.1086/599582

LASICA, J. D. *Darknet*: Hollywood's war against the Digital Generation. Nova Jersey: John Wiley & Sons, 2005.

MACKENZIE, Adrian. Codecs. In: FULLER, Matthew (ed.). *Software Studies*: A Lexicon. Cambridge, Mass: MIT Press, 2008. p. 48-54.

MANOVICH, Lev. *The Language of New Media*. Cambridge, Mass: MIT Press, 2001.

_____. *Software Takes Command*. 2008. Available at: <http://goo.gl/D7fde>. Access in: 14 Ago. 2011.

MENOTTI, Gabriel. Distribuición digital de películas: compromiso tecnológico, transgresión institucional y experiencia mediática em El estreno en línea de Steal this Film II. *L'Atalante Revista de estudios cinematográficos*. Espanha, n. 13, p. 12-19, 2012.

RODOWICK, D. N. *The Virtual Life of Film*. Londres: Harvard University Press, 2007.

TURING, Alan. On Computable Numbers, With an Application to the Entscheidungs problem. *Proceedings of the London Mathematical Society*. Londres, n. 42 (1), 1936. Available at: <http://goo.gl/AWM4Y>. Access in: 14 Dec. 2011. DOI: 10.1112/plms/s2-42.1.230

USAI, Paolo Cherchi. A ModelImage, IV. Decay Cinema: The Art and Aesthetics of Moving Image Destruction. *Stanford Humanities Review*, Stanford, n. 7.2, p. xiv-49, 1999.

This text was received at 10 September 2013 and accepted at 04 November 2014.