Generation Flash

author: Lev Manovich

year: 2002

This essay that consists of a number of self-contained segments looks at the phenomenon of Flash graphics on the Web that attracted a lot of creative energy in the last few years. More than just a result of a particular software / hardware situation (low bandwidth leading to the use of vector graphics), Flash aesthetics exemplifies cultural sensibility of a new generation. [1] This generation does not care if their work is called art or design. This generation is no longer is interested in "media critique" which preoccupied media artists of the last two decades; instead it is engaged in software critique. This generation writes its own software code to create their own cultural systems, instead of using samples of commercial media. [2] The result is the new modernism of data visualizations, vector nets, pixel-thin grids and arrows: Bauhaus design in the service of information design. Instead the Baroque assault of commercial media, Flash generation serves us the modernist aesthetics and rationality of software. Information design is used as tool to make sense of reality while programming becomes a tool of empowerment. [3]

Turntable and Flash Remixing

for www.whitneybiennial.com

Turntable is a web-based software that allows the user to mix in real-time up to 6 different Flash animations, in addition manipulating color palette, size of individual animations and other parameters. For www.whitneybiennial.com, the participating artists were asked to submit short Flash animations that were exhibited on the site both separately and as part of Turntable remixes. Some remixes consisted of animations of the same artists while others used animations by different artists.

It has become a cliché to announce that “we live in remix culture.” Yes, we do. But is it possible to go beyond this simple statement of fact? For instances, can we distinguish between different kinds of remix aesthetics? What is the relationship between our remixes made with electronic and computer tools and such earlier forms as collage and montage? What are the similarities and differences between audio remixes and visual remixes?

Think loop. The basic building block of an electronic soundtrack, the loop also conquered surprisingly strong position in contemporary visual culture. Left to their own devices, Flash animations, QuickTime movies, the characters in computer games loop endlessly - until the human user intervenes by clicking. As I have shown elsewhere, all nineteenth century pre-cinematic visual devices also relied on loops. Throughout the nineteenth century, these loops kept getting longer and longer - eventually turning into a feature narrative… Today, we witness the opposite movement – artists sampling short segments of feature films or TV shows, arranging them as loops, and exhibiting these loops as “video installations.” The loop thus becomes the new default method to “critique” media culture, replacing a still photograph of post-modern critique of the 1980s. At the same time, it also replaces the still photograph as the new index of the real: since everybody knows that a still photography can be digitally manipulated, a short moving sequence arranged in a loop becomes a better way to represent reality - for the time being.

Think Internet. What was referred in post-modern times as quoting, appropriation, and pastiche no longer needs any special name. Now this is simply the basic logic of cultural production: download images, code, shapes, scripts, etc.; modify them, and then paste the new works online - send them into circulation. (Note: with Internet, the always-existing loop of cultural production runs much faster: a new trend or style may spread overnight like a plague.) When I ask my students to create their own images by making photographs or by shooting video, they have a revelation: images do not have to come from Internet! Shall I also reveal to them that images do not have to come from a technological device that record reality – that instead they can be drawn or painted?

Think image. Compare it to sound. It seems possible to layer many many sounds and tracks together while maintaining legibility. The result just keeps getting more complex, more interesting. Vision seems to be working differently. Of course, commercial images we see every day on TV and in cinema are often made from layers as well, sometimes as many as thousands – but these layers work together to create a single illusionistic (or super-illusionistic) space. In other words, they are not being heard as separate sounds. When we start mixing arbitrary images together, we quickly destroy any meaning. (If you need proof, just go and play with the classic "The Digital Landfill.") [4] How many separate image tracks can be mixed together before the composite becomes nothing but noise? Six seems to be a good number – which is exactly the number of image tracks one can load onto Turntable.

Think sample versus the whole work. If we are indeed living in a remix culture, does it still make sense to create whole works – if these works will be taken apart and turned into samples by others anyway? Indeed, why painstakingly adjust separate tracks of Director movie or After Effects composition getting it just right if the "public" will "open source" them into their individual tracks for their own use using some free software? Of course, the answer is yes: we still need art. We still want to say something about the world and our lives in it; we still need our own "mirror standing in the middle of a dirty road," as Stendahl called art in the nineteenth century. Yet we also need to accept that for others our work will be just a set of samples, or maybe just one sample. Turntable is the visual software that makes this new aesthetic condition painfully obvious. It invites us to play with the dialectic of the sample and the composite, of our own works and the works of others. Welcome to visual remixing Flash style.

Think Turntable.

Art, Media Art, and Software Art

Recently "software art" has emerged as the new dynamic area of new media arts. Flash's ActionScript, Director's Lingo, Perl, MAX, JavaScript, Java, C++, and other programming and scripting languages are the medium of choice of a steadily increasing number of young artists. Thematically, software art often deals with data visualization; other areas of creative activity include the tools for online collaborative performance / composition (Keystroke), DJ/VJ software, and alternatives to / critiques of commercial software (Autoillustrator), especially the browsers (early classics like Netomat, Web Stalker, and many others since then). Often, artists create not singular works, but software environments open for others to use (such as Alex Galloway's Carnivore.) Stylistically, many works implicitly reference visual modernism (John Simon seems to be the only one so far to weave modernist references in his works explicitly).

Suddenly, programming is cool. Suddenly, the techniques and imagery that for two decades were associated with SIGGRAPH geek ness and were considered bad taste – visual output of mathematical functions, particle systems, RGB color palette – are welcomed on the plasma screens of the gallery walls. It is no longer *October* and *Wallpaper* but Flash and Director manuals that are the required read for any serious young artist.

Of course, from its early days in 1960s computer artists have always written their own software. In fact, until the middle of the 1980s, writing own software or at least using special very high-end programming languages designed by others (such as Zgrass) was the only way to do computer art. [5] So what is new about the recently emerged phenomenon of software art? Is it necessary?

Let's distinguish between three figures: an artist; a media artist; and a software artist.

A romantic/modernist artist (the nineteenth century and the first half of the twentieth century) is a genius who creates from scratch, imposing the phantoms of his imagination on the world.

Next, we have the new figure of a media artist (the 1960s – the 1980s) that corresponds to the period of post-modernism. Of course, modernist artists also used media recording technologies such as photography and film, but they treated these technologies similar to other artistic tools: as means to create an original and subjective view of the world. In contrast, post-modern media artists accept the impossibility of an original, unmediated vision of reality; their subject matter is not reality itself, but representation of reality by media, and the world of media itself. Therefore, these media artists not only use media technologies as tools, but they also use the content of commercial media. A typical strategy of a media artist is to re-photograph a newspaper photograph, or to re-edit a segment of TV show, or to isolate a scene from a Hollywood film / TV shows and turn it into a loop (from Nam June Paik and Dara Birnbaum to Douglas Gordon, Paul Pffefer, Jennifer and Kevin McCoy). Of course, a media artist does not have to use commercial media technologies (photography, film, video, new media) – s/he can also use other media, from oil paint to printing to sculpture.

The media artist is a parasite who lives at the expense of the commercial media – the result of collective craftsmanship of highly skilled people. In addition, an artist who samples from / subverts / pokes at commercial media can ultimately never compete with it. Instead of a feature film, we get a single scene; instead of a complex computer game with playability, narrative, AI, etc., we get just a critique of its iconography.

Thirty years of media art and post-modernism have inevitably led to a reaction. We are tired of always taking existing media as a starting point. We are tired of being always secondary, always reacting to what already exists.

Enter a software artist – the new romantic. Instead of working exclusively with commercial media – and instead of using commercial software – software artist marks his/her mark on the world by writing the original code. This act of code writing itself is very important, regardless of what this code actually does at the end.

A software artist re-uses the language of modernist abstraction and design – lines and geometric shapes, mathematically generated curves and outlined color fields – to get away from figuration in general, and cinematographic language of commercial media in particular. Instead of photographs and clips of films and TV, we get lines and abstract compositions. In short, instead of QuickTime, we use Flash. Instead of computer as a media machine – a vision being heavily promoted by computer industry (and most clearly articulated by Apple who promotes a MAC as a "digital hub" for other media recording / playing devices), we go back to computer as a programming machine.

Programming liberates art from being secondary to commercial media. The similar reason may be behind the recent popularity of "sound art." While commercial media now uses every possible visual style, commercial sound environments still have not appropriated all of sound space. While rock and roll, hip-hop, and techno have already become standard elevator music (at least in more hip elevators such as the Hudson Hotel in NYC), it seems that the rhythm-less regions of sound space are still untouched – at least for now.

UTOPIA in Shockwave

UTOPIA is a Shockwave project by Futurefarmers for Tirana Biennale 01 Internet section.

Futurefarmers: Amy Franceschini and Sascha Merg, http://nutrishnia.org/level/

UTOPIA is playful and deceitful - because it pretends to be more innocent, more simple, and more light than it actually is. At first glance it can be taken for something made for children - or for adults whose references are not Karl Marx, Sigmund Freud, Rem Koolhaas, and Philip Stark, but text messaging, gnuttela, retro Atari graphics, and nettime. This is the new generation that emerged in the 1990s. In contrast to visual and media artists of the 1960s-1980s, whose main target was media - ads, cinema, television - the new generation does not waste its energy on media critique. Instead of bashing commercial media environment, it creates its own: Web sites, mixes, software tools, furniture, clothes, digital video, Flash / Shockwave animations and interactives.

The new sensibility, which Utopia exemplifies so well, is soft, elegant, restrained, and smart. This is the new software intelligentsia. Look at the thin low-contrast lines of UTOPIA, praystation.com, and so many Flash projects included in Tirana Biennale 01. If images of the previous generations of media artists, from Nam June Paik to Barbara Krueger, were screaming, trying to compete with the intensity of the commercial media, the new data artists such as Franceschini/Merg whisper in our ears. In contrast to media's arrogance, they offer us intelligence. In contrast to media stream of endless repeated icons and sound bytes, they offer us small and economical systems: stylized nature, ecology, or the game / music generator / Lego-like parade in UTOPIA.

Futurefarmers are among the few Flash/Schockwave masters who use their skills for social rather than simply a formal end. Their project theyrule.net is a great example of how smart programming and smart graphics can be used politically. Instead of presenting a packaged political message, it gives us data and the tools to analyze it. It knows that we are intelligent enough to draw the right conclusion. This is the new rhetoric of interactivity: we get convinced not by listening / watching a prepared message but by actively working with the data: reorganizing it, uncovering the connections, becoming aware of correlations.

UTOPIA does not have explicit political content; instead it presents its message through a visual allegory. Like *SimCity* and similar sims, the program presents us with a whole miniature world which runs according to its own system of rules. (All the animation in UTOPIA is result of code execution – nothing is hand animated.) The cosmogony of this world reflects our new understanding of our own planet - post Cold War, Internet, ecology, Gaia, and globalization. Notice the thin barely visible lines that connect the actors and the blocks. (This is the same device used in theyrule.net.) In the universe of UTOPIA, everything is interconnected, and each action of an individual actor affects the system as a whole. Intellectually, we know that this is how our Earth functions ecologically and economically - but UTOPIA represents this on a scale we can grasp perceptually.

The lines also serve another purpose. Despite CNN, Greenpeace, the glass roof of Berlin's Reichstag and other institutions and devices working to make the functioning of modern societies transparent to their citizens, most of it is not visible. This is not only because we don't know the motives behind this or that Government policy or because advertising and PR constantly work to make things appear differently from what they really are – the societies' functioning is not visible in a literal sense. For instance, we don't know where the cells are which make our cell phones work; we don't know the layout of private financial network that circle the Earth; we don't know what companies are located in a building we pass every day on a way to work; and so on. But in UTOPIA, we do know – because the links are made visible. UTOPIA is Utopia because it is a society where cause and effect connection are rendered visible and comprehensible. The program re-writes Marxism as vector graphics; it substitutes the figure of "connections" for the old figure of "unveiling."

UTOPIA is serious business behind its playful façade – but it is not all business. Drawing on our current fascination with computer games and interactive image-sound software, UTOPIA is a visual and intellectual delight, UTOPIA draws on the current fascination with computer games and interactive image-sound software. It is Tetris that meets Marx that meets data mining that meets the club dance floor. It is a game for the new generation that know that the world is a network, that the media is not worth taking very seriously, and that programming can be used as a political tool.

The Unbearable Lightness of FLASH

Tirana Biennale 01 Internet section www.electronicorphanage.com/biennale was organized by Miltos Manetas / Electronic Orphanage. The exhibition consisted of a few dozen projects by Web designers and artists, many of whom work in Flash or Schockwave. Manetas commissioned me, Peter Lunenfeld, and Norman Klein to write the analysis of the show. This text is my contribution; many ideas in it developed out of the conversations the three of us had about the works in the show. The names in parentheses below refer to the artists in the show; go to the show site to see their projects.

Biology

Flash artists are big on biological references. Abstract plants, minimalist creatures, or simply clouds of pixels dance in patterns which to a human eye signal "life" (Geoff Stearns: deconcept.com, Vitaly Leokumovich: unclickable.com, Danny Hobart: dannyhobart.com; uncontrol.com). Often we see self-regenerating systems. But this is not life as it naturally developed on Earth; rather, it looks like something we are likely to witness in some biotech laboratory where biology is put in the service of industrial production. We see hyper accelerated regeneration and evolution. We see complex systems emerging before our eyes: millions of years of evolution are compressed into a few seconds.

There is another feature that distinguishes life a la Flash from real life: the non-existence of death. Biological organisms and systems are born, they develop, and eventually they die. In short, they have teleology. But in Flash projects life works differently: since these projects are loops, there is no death. Life just keeps running forever – more precisely, until your computer maintains Net connection.

Amplification: Flash aesthetics and Computer Games

Abstract ecosystems in Flash projects have another characteristic that makes playing so pleasurable (Joel Fox). They brilliantly use the power of the computer to amplify user's actions. This power puts a computer in line with other magical devices; not accidentally, the most obvious place to see it is in games, although it is also at work in all of our interactions with a computer. For instance, when you tell Mario to step to the left by moving a joystick, this initiates a small delightful narrative: Mario comes across a hill; he starts climbing the hill; the hill turns to be too steep; Mario slides back onto the ground; Mario gets up, all shaking. None of these actions required anything from us; all we had to do is just to move the joystick once. The computer program amplifies our single action, expanding it into a narrative sequence.

Historically, computer games were always a step ahead from the general human computer interface. In the 1960s and 1970s users communicated with a computer using non-graphical interfaces: entering the program onto a stack of punch cards, typing on a command line, and so on. In contrast since their beginnings in the late 1950s, computer games adopted interactive graphical interface – something that only came to personal computers in the 1980s.

Similarly, today's games already use what many computer scientists think will be the next paradigm in HCI: active amplification of user's actions. In the future, we are told, agent programs would watch our interactions with a computer, notice the patterns, and then automate many tasks we do regularly, from backing up the data at regular intervals to filtering and answering our email. The computer would also monitor our behavior and attention level, adjusting its behavior accordingly: speeding up, slowing down, and so on. In some ways this new paradigm is already at work in some applications: for instance, Internet browser offers us the list of sites relevant to the topic we are searching on; Microsoft Office Assistant trying to guess when we need help. However, there is a crucial problem with moving to such active amplification across the whole of HCI. The more power we delegate to a computer, the more we lose control over what it is doing. How do we know that the agent program identified a correct pattern in our daily use of email? How do we know that a commerce agent we send on the Web to negotiate with other agents the lowest price for a product was not corrupted by them? In short, how do we know that a computer amplified our actions correctly?

Computer games are games, and the worst that may happen is that we lose. Therefore, active amplification is present in practically every game: Mario embarking on mini-narratives of its own with a single move of a joystick; troops conducting complex military maneuvers while you directly control only their leader in Rainbow Six; Lora Craft executing whole acrobatic sequences with a press of a keyboard key. (Note that in "normal" games this amplification does not exist: when you move a single figure on a chessboard, this is all that happens; your move does not initiate a sequence of steps.)

Flash projects heavily use active amplification. It gives many projects the magical feeling. Often we are confronted with an empty screen, but a single click brings to life a whole universe: abstract particle systems, plant-like outlines, or a population of minimalist creatures. The user as a God controlling the universe is something we also often encounter in computer games; but Flash projects also give us the pleasure of creating the universe from scratch.

The active amplification is not the only feature Flash projects share with games. More generally, as Peter Lunenfeld suggested, computer games are for Flash generation what movies were for Warhol. Cinema and TV colonized the unconscious of the previous generations of media artists who continue to use the gallery as their therapy coach, spilling bits and pieces of their childhood media archives in public (for instance, Douglas Gordon). Flash artists are less obsessed with commercial time-based media. Instead, their iconography, temporal rhythms, and interaction aesthetics come from games (Mike Clavert: mikeclavert.com). Sometimes the user participation is needed for the Flash game to work; sometimes the game just plays itself (UTOPIA by futurefarmers.com; dextro.org).

Flash versus Net Art

Tirana Biennale 01 Internet exhibition: this title is deeply ironic. The exhibition did not include any projects from Albania, or any other post-communist East European country for that matter. This was quite different from many early net art exhibitions of the middle of the 1990s whose stars came from the East: Vuk Cosic, Alexei Shulgin, Olia Lialina. 1990s net art was the first international art movement since the 1960s that included east Europe in a big way. Prague, Ljubljana, Riga, and Moscow counted as much as Amsterdam, Berlin, and New York. Equally including artists from the West and the East, net art perfectly corresponded to the economic and social utopia of a new post Cold War world of the 1990s.

Now this utopia is over. The power structure of the global Empire has become clear, and the demographics of Tirana Biennale 01 Internet section reflected this perfectly. Many artists included in Tirana Biennale 01 Internet exhibition work in key IT regions of the world: San Francisco (Silicon Valley), New York (Silicon Alley), and Northern Europe.

What happened? In the mid-1990s, net art relied on simple HTML that run well on both fast and slow connections – and this is enabled active participation of the artists from the East. But the subsequent colonization of the Web by multimedia formats – Flash, Shockwave, QuickTime, and so on – restored the traditional West/East power structure. Now Web art requires fast Internet connections for both the artist and the audiences. With its slow connections, East is out of the game. The Utopia is over; welcome to the Empire.

(Tirana Biennale 01 did include one artist from China who contributed a beautiful animation of martial arts fighters. But we never found who he was. All we knew about him was his email address: zhu_zhq@sohu.com. Maybe he did not even live in China.)

Generation FLASH: FAQ

After I posted the preceding segments on popular mailing lists dealing with new media art and cyberculture (rhizome.org and nettime.org), I received lots of responses. Here are my answers to two most common questions which appeared in a number of responses.

Question:

Is not “soft modernism” you describe simply a result of particular technological limitations of multimedia on the Net? You seem to mistake the particular features of Flash designed to deliver animation over the narrow bandwidth for a larger zeitgeist.

Answer:

Now that the new release of Flash (Flash MX) allows for import and streaming of video, it is possible that soon "Flash generation" / "soft modernism" aesthetics will leave Flash sites. This is fine. My concern in this essay is *not* with Flash software and its limitations/capabilities per se, but with the new sensibility that during the last couple of years manifested in many Flash projects. In other words, I am interested in "generation Flash" that is quite different from Flash software/format.

Therefore, the number of people who after reading my text accused me of confusing a technical standard with aesthetics missed my argument. The vector-oriented look of "soft modernism" is not simply a result of narrow bandwidth or a nostalgia for 1960s design - it *always* happens when people begin to generate graphics through programming and discover that they can use simple equations, etc. This is also why "soft modernism" of Flash projects and other software artists replays, sometimes in amazing detail, the aesthetics of early computer art (1950s-1970s) when people were only able to create images and animations through programming.

Question:

There is no reason software art cannot use representation images or any other form. Why do you associate software art with non-representational, abstract vector-based graphics?

Answer:

Of course, software artists can use representational images or any other “conventional” form or media. It was not accidental that soon after his arrival at Xerox PARC in the 1970s, Alan Kay and his associates created a paint program and an animation program, alongside with overlapping windows, icons, Smalltalk, and other principles of modern interactive graphical computing. The abilities to manipulate and generate media are not after-thoughts to a modern computer - they are central to its identity as a "personal dynamic medium" (Alan Kay.) To put his differently: computer is a simulation machine, and as such it can and should be used to simulate other media.

So, I have nothing against software artists using/creating media, but I hope that "Flash generation" will extend its programming work to representational media! In other words, if in the early 1970s the paint program and the animation program were revolutionary in changing people idea about a computer away from computation and towards a (creative) medium, after almost two decades of menu-based media manipulation programs and the use of computers as media distribution machine (greatly accelerated by World Wide Web), a little programming can be quite revolutionary! In short, we now are so used to think of a computer as a "personal dynamic medium," that we need to remind ourselves and others that it is also a programmable machine.

Now, think about how programming has been used so far to create/use still images, animation, and film/video. There are three trajectories that can be traced historically. One trajectory extends from the earliest works of computer art - the films by the Whitney made with an analog computer already in the mid-1950s (who were the students of Oscar Fischinger and thus represent a direct link with the early twentieth century modernism) - to today's "soft modernism" of Flash projects and data visualization artworks. In other words, this is the use of programming to generate and control abstract images.

The second trajectory begins in the 1980s when Hollywood and TV designers started to use computer-generated imagery (CGI). Now, programming was put in the service of traditional cinematic realism. Particle systems, formal grammars, AI, and other software techniques became the means to generate flying bats, hilly landscapes, ocean waves, explosions, alien creatures, and other figurative elements integrated in a photorealistic universe of a narrative film.

What about using algorithms not simply to generate figurative elements of a narrative but to control the whole fictional universe? This is the third trajectory: programming in computer games (1960-). Here algorithms may control the narrative events, the behavior of characters, camera movement, and other characteristics of the game world - all in real time. Unfortunately, as we all know, aesthetically revolutionary computer and player driven game worlds feature formula-driven content that makes even a bad Hollywood film appear original and inspiring by comparison. (*Grand Theft Auto 3* is no exception here - despite its breakthroughs in simulating a more compelling and open universe.)

I think this brief survey shows that there is still an untouched space completely open for experimentation and creative research - using programming to generate and/or control figurative/fictional media. For instance, in the case of a movie, programming can be used to generate characters on the fly, to composite in real-time characters shot against a blue screen with backgrounds, to control the sequence of scenes, to apply filters to any scene in real-time, to combine pre-recorded scene with on the imagery generated on the fly, to have characters interact with the viewer, etc., etc. In short, programming can be used to control *any* aspect of a fictional media work.

Of course, once in a while one encounters projects moving in this direction at places like SIGGRAPH or ISEA, but they are typically research demos created in universities that do not reach culture at large. Of course, you can object that having an algorithmically controlled complex fictional universe requires the kind of programming investment only possible in a commercial game company or in a university. After all, this is not the same as writing a script that draws a few lines that keep moving in response to user input…yes, but why our fictional/figurative works have to follow the formulas of commercial media? If one accepts that the characters do not have to be "photorealistic," that the fictional world does not have to be exclusively three-dimensional, that chance and randomness can co-exist with narrative logic, or that stick figures can co-exist with 3-D characters and video footage, etc., programming figuration / fiction becomes less formidable. In short, while I welcome programming Flash, I think it is much more challenging to program QuickTime.

Postscript: On the Lightness of Flash

When I first visited the most famous Flash site – praystation.net – I was struck by the lightness of its graphics. More quiet than a whisper, more elegant than Dior or Chanel, more minimal than 1960s minimalist sculptures of Judd, more subdued than the winter landscape in heavy fog, the site pushed the contrast scale to the limits of legibility. The similar lightness and restrain can be found in many projects included in Biennale 01 show.

Again, the contrast with screaming graphics of commercial media and the media art of the previous generations is obvious.

The lightness of Flash can be thought of as a visual equivalent of electronic ambient music. Every line and every pixel counts. Flash appeals to our visual intelligence - and cognitive intelligence. After the century of RGB color which begun with Matisse and ended with aggressive spreads of Wired, we are asked to start over, to begin from scratch. Flash generation invites us to undergo a visual cleansing – this is why we see a monochrome palette, white and light gray. It uses neo-minimalism as a pill to cure us from post-modernism. In Flash, the rationality of modernism is combined with the rationality of programming and the affect of computer games to create the new aesthetics of lightness, curiosity, and intelligence. Make sure your browser has the right plug-in: welcome to generation Flash.

I am not advocating a revival of modernism. Of course, we don't want to simply replay Mondrian and Klee on computer screens. The task of the new generation is to integrate the two key aesthetic paradigms of the twentieth century: (1) belief in science and rationality, emphasis on efficiency and basic forms, idealism, and heroic spirit of modernism; (2) skepticism, interest in “marginality” and “complexity,” deconstructive strategies, baroque opaqueness, and excess of post-modernism (1960s-). At this point all the features of the second paradigm became tired clichés. Therefore, a partial return to modernism is not a bad first step, as long as it is just a first step towards developing the new aesthetics for the new age.

Of course, this aesthetics should also fully engage with the difficult questions of globalization. The *remix culture* we are living now is not only engaged in remixing all previous cultural forms and texts of but also in remixing various features which come from what used to be call national cultures as well as from already existing remixes between immigrant populations and their “host” cultures. The solution offered by multinational conglomerates – a composite which takes certain signifiers from a few national cultures – for instance, French idea of elegance, Japanese manga iconography, “cool Britannia” references, and so on, and integrates it all into a rather bland and monolithic text which is then being send back to all the places around the world – is obviously not a satisfactory solution. (It reminds me of Soviet-style centralized economy when the all the output of collective farms was sent to the center where it was decided how it was distributed nationally.) Luckily, numerous remixes which follow different logics are being explored around the world by musicians, theatre groups, dancers, designers, architects, and so on. Nobody knows what will emerge from this global cultural laboratory – and this is what makes out times so interesting.

Although most of my arguments in this book are about visual culture and visual aesthetics, it is relevant at this point to evoke a different practice. Music historically has been the artistic field that was always been ahead of other fields in using computers to enable new aesthetic paradigms. The whole practice of popular electronic music in the last three decades is a testament to how empowering new technologies are in welding new complex and rich remixes between different cultures, styles, and sensibilities. Without electronic and computing technologies – from a turntable and a tape recorder to peer-to-peer file sharing networks and music synthesis software running on a regular laptop, most of this culture would never come to be. The field of electronic sound (which pretty much means most sounds today) with its multitude voices and a real bottom-up, "emergent" logic, is a powerful alternative to the "top-down" cultural composites sold by global media conglomerates around the world. Let us hope that other artists and designers in other fields will follow music lead in using a computer to enable similarly rich remix cultures.

References:

[1] This article is about "Flash Generation" and not about the Web sites made with Flash software. Many of the sites which inspired me to think of "Flash aesthetics" are not necessarily made with Flash; they use Shockwave, DHTML, Quicktime, and other Web multimedia formats. Thus, the qualities I describe below as specific to "Flash aesthetics" are not unique to Flash sites.

[2] For instance, the work of Lisa Jevbratt, John Simon, and Golan Levin.

[3] “Generation Flash” incorporates revised versions of the texts commissioned for www.whitneybiennial.com and http://www.electronicorphanage.com/biennale. Both exhibitions were organized by Miltos Manetas / Electronic Orphanage. “On UTOPIA” was commissioned by Futurefarmers.

[4] See http://www.potatoland.org/landfill/

[5] After GUI-based applications such as Hypercard, Director, Photoshop and others became commonplace, many computer artists continued to do their own programming: writing custom code to control an interactive installation, programming in LINGO an interactive multimedia work, etc. This was not referred to as software art; it was taken for granted that even in the age of GUI-based applications a really serious artistic engagement with computers requires getting one’s hands dirty in code.