directed by Hugo de Sancto Caro (Hugues de Saint-Cher), a Dominican friar later made a cardinal.

12. In an intriguing if eccentric book, *Symmetry, Causality, Mind,* computer vision researcher Michael Leyton proposes that "time reversal" is the sole task undertaken by intelligence. He sees vision as a two-fold problem: first, the eye must reverse the formation of the optical image incident on the retina (a classic "inverse problem") to identify objects and the spatial relations between them; secondly, the mind must "reverse the formation of the environment"—it must adduce reasons why these objects should be where they are, and should have the forms they have, and it must ascertain as best it can the intentions or implications of these things with respect to its own needs and goals. Only then can it be in a position to decide what, if anything, to do about the situation.

13. "Noise-corrupted" is synonymous with "massively convolved with impractically many broadly diffused and attenuated traces of events that we happen not to be interested in right now."

14. "Drawn together"—see Bruno Latour's essay "Drawing Things Together" for an account of broadly analogous issues in the social production of knowledge.

15. Readers with a taste for such deliberations might enjoy following the elegant turns Paul Valery's curiosity takes in his essay, "Man and the Seashell."

■

Glitch

Olga Goriunova and Alexei Shulgin

This term is usually identified as jargon, used in electronic industries and services, among programmers, circuit-bending practitioners, gamers, media artists, and designers. In electrical systems, a glitch is a short-lived error in a system or machine. A glitch appears as a defect (a voltage-change or signal of the wrong duration—a change of input) in an electrical circuit. Thus, a glitch is a short-term deviation from a correct value and as such the term can also describe hardware malfunctions. The outcome of a glitch is not predictable.

When applied to software, the meaning of glitch is slightly altered. A glitch is an unpredictable change in the system's behavior, when *something obviously goes wrong*.

Glitch is often used as a synonym for bug; but not for error. An error might produce a glitch but might not lead to a perceivable malfunction of a system. Errors in software are usually structured as: syntax errors (grammatical errors in a program), logic errors (error in an algorithm), and exception errors (arising from unexpected conditions and events).

Glitches have become an integral part of computer culture and some phenomena are perceived as glitches although they are not glitches in technical terms. Artifacts that look like glitches do not always result from an error. What users might perceive as "glitchy" can arise from a normally working function of a program. Sometimes these might originate from technical limitations, such as low image-processing speed or low bandwidth when displaying video. For example, the codecs of some video-conferencing software, such as CU-Seeme,[1] visibly "pixelize" the image, allowing the compression of parts of the images that remain static over different frames when, for instance, the transfer speed drops.

To comply with the customary usage of "glitch" we propose to think of glitches as resulting from error, though in reality it might be difficult or impossible to distinguish whether the particular glitch is planned or results from a problem. To understand the roles glitches play in culture, knowing their origin is not of primary importance. Understanding glitches as erroneous brings more to a comprehension of their role than trying to give a clear definition that would include or subordinate encoded glitches and glitches as malfunctions.

Glitches are usually regarded as marginal. In reality, glitches can be claimed to be a manifestation of genuine software aesthetics. Let us look at machine aesthetics as formed by functionality and dysfunctionality, and then proceed to the concept of glitches as computing's aesthetic core, as marks of (dys)functions, (re)actions and (e)motions that are worked out in human-computer assemblages.

Computers do not have a recognizable or significant aesthetic that possesses some kind of authenticity and completeness. It is commonplace that the aesthetics of software are largely adopted from other spheres, media, and conventions. Thus, the desktop is a metaphor for a writing table, icons descend from labels or images of objects, while the command line interface is inherited from telegraph, teletype, and typewriter.

The aesthetics of computers that developed over a few decades from the early 1950s to the early 1980s, when they were first introduced to the public and on to the current time (consisting of dynamic menus, mouse, pointer, direct manipulation of objects on the screen, buttons, system sounds, human

computer interaction models) are, in our opinion, not rich and self-sufficient enough to be called the aesthetic of the computer.

On top of that the current aesthetic of software is not complete; it does not work very well as it does not contribute enough to the computer's user-friendliness. Besides, it is a widely acknowledged problem that the customary information design principles of arranging computer data, derived from earlier conventions (such as the treelike folder structure), result in users having problem, with data archiving and the memorization of document names and locations.

Historically, the shape, style, and decoration of every new technology has been introduced in a manner owing much to the aesthetics and thinking customary of the time. Thus, when mechanism had not yet replaced naturalism as means of framing reality, Lewis Mumford argues, mechanisms were introduced with organic symbols. For instance, a typical eighteenth century automaton, "the clockwork Venus," consisted of a female mannequin resting on top of a clockwork mechanism.[2] As technology developed further, some genuine machine aesthetics were born, primarily derived from machine functionality. And it was their functionality that some avant-garde movements of the twentieth century admired in the machine. For instance, among the Russian avant-garde movements of the beginning of the twentieth century (e.g., Cubo-Futurism, Abstractionism, Rayonism, Suprematism) artists such as Mayakovsky, Gontcharova, Kandinsky, Larionov, and Malevich poeticized new machines for their speed, energy, and dynamics. The methods they used to depict movement, light, power, and speed could be regarded aesthetically as grandparents of some of today's glitches (certain correlation of color mass; unlimited diversity of colors, lines and forms; repeating geometrical structures, figures, lines, dots, etc.).

Rationalism and the precision of technical creation inspired many. Thus, Meyerhold writes: "Arts should be based on scientific grounds."[3] Russian constructivists such as Tatlin established a compositional organization based on the kinetics of simple objects and complex ideas of movement—rotating inner mechanisms and open structure, using "real" materials—all intended to function for utilitarian use. Punin writes of Tatlin's Tower: "Beneath our eyes there is being solved the most complex problem of culture: utilitarian form becomes pure creative form."[4]

Functional machines, primarily built by engineers, established strong aesthetic principles that have defined technological design for years. Functional elements are later used as nonfunctional design elements that are appreciated

as "beautiful" by users not least due to the cultural memory of their origin. For instance, the curved part of the wing over the tire of some car models reproduces the guards used in horse-driven vehicles and early automobiles to protect users and vehicle from dust and to affix lights onto. It does not carry any advance in function, but is used in automobile design as a recognizable and nostalgic element.

Today, the functionality of the computer is concealed inside the gray/white/beige box that covers the cards, slots, motherboard, and wires. In modding[5] these parts are reimagined as elements of visual richness that convey a symbolism. Hardware elements are aestheticized: Users might install neon lights, weird jumbo fans and colorful wires into a transparent computer case or even build an entirely new one from scratch. Electronic boards jutting out at 90 degree angles and architectures of twisted wire are widely used, as in cinema and design, to represent *technical substances*.

By contrast, the way data is presented on a hard drive is not human-readable. It is stored in different segments of the disk and reassembled each time the documents are retrieved according to a plan kept as a separate file. Software functionality here is invisible and an interface is needed to use the machine. Modern software almost always conceals its functionality behind the window. It provides us instead with images such as a page flying from one folder to another, an hourglass, or that of a gray line gradually being filled with color.

There are moments in the history of computer technology that are rich in computer functionality producing distinct aesthetics. At such times, computer functionality reveals itself through technological limitations. Bottlenecks, such as processor speed, screen resolution, color depth, or network bandwidth—4-bit, 8-bit music, 16-color pixelized visuals, slow rendering, compressed image and video with artifacts—create an authentic computer aesthetics, that is, the aesthetics of low-tech today.

There are vast contemporary 8-bit music communities (such as Micromusic .net), based entirely on producing music on emulators or surviving models of the early home computers of the 1980s, such as Atari or Commodore. Alongside producing sine waves, the sound chips of such computers attempted to simulate preexisting musical reality: guitar, percussion, piano. Imperfect and restricted, the chips could only produce idiosyncratic, funny and easy to recognize sounds which were far from the originals. Scarcity of means encouraged a special aesthetics of musical low-tech: of coolness, romanticism and imperfection. People making 8-bit music nowadays relate back to their childhoods'

favorite toys, memories that are shared by many people. Returning to a genuine computer aesthetics of obsolete technology is not a question of individual choice, but has the quality of a communal, social decision.

Functionality, as a characteristic of established machine aesthetics is always chased by dysfunctionality (if not preceded by it). Functional machines, robots, mechanized people (from Judaism's Golem,[6] Frankenstein's monster[7]) to the rebellious computers of the twentieth century) are interpreted as alien to human nature, sooner or later becoming "evil" as they stop functioning correctly. Thus, the dysfunctional mind, conduct, and vision become human, compelling, sincere, meaningful, revelatory. As aesthetic principles, chance, unplanned action, and uncommon behaviors were already central to European and Russian literature of the nineteenth century in the work of writers such as Balzac, Flaubert and Dostoyevsky.

In the technological era, society became organized according to the logic of machines, conveyor belt principles, "rationally" based discrimination theories, and war technology, with an increase in fear, frustration, refusal, and protest. As a response, errors, inconsistencies of vision, of method, and of behavior become popular modernist artistic methods used in Dadaism, Surrealism, and other art movements. One of Surrealism's declared predecessors, the Comte de Lautréamont, provided us with the lasting phrase that something could be as "beautiful as the chance encounter of a sewing machine and an umbrella on a dissection table."[8] The introduction of chance, "hasard," (fr.), subconsciousness, and irrationality into art and life was seen as being both opposed to and deeply embedded in rationality and functionality.

Dysfunctional machines are not only those that are broken (images and figures of crashed cars and other mass produced imperfections figure in the aesthetics of Fluxus and Pop Art); they are also those that do not comply with the general logic of machines, by acting irrationally and sometimes even turning into humans. Thus, at the end of the Soviet movie *Adventures of Electronic Boy* (1977), a robotic boy starts crying and this emotion symbolizes that he has become human.

A glitch is a singular dysfunctional event that allows insight beyond the customary, omnipresent, and alien computer aesthetics. A glitch is a mess that is a moment, a possibility to glance at software's inner structure, whether it is a mechanism of data compression or HTML code. Although a glitch does not reveal the true functionality of the computer, it shows the ghostly conventionality of the forms by which digital spaces are organized.

Glitches are produced by error and are usually not intended by humans. As a not-entirely human-produced reality, its elements are not one-hundred percent compatible with customary human logic, visual, sound, or behavioral conventions of organizing and acting in space. Aesthetically some glitches might inherit from avant-garde currents, but are not directly a product of the latter (figure 8). Avant-garde artists inspired or disgusted by technology and its societal influence have created a range of artistic responses, the aesthetics of which today's glitches strangely seem to comply with. A glitch reminds us of our cultural experience at the same time as developing it by suggesting new aesthetic forms.

A glitch is stunning. It appears as a temporary replacement of some boring conventional surface; as a crazy and dangerous momentum (Will the computer come back to "normal"? Will data be lost?) that breaks the expected flow. A glitch is the loss of control. When the computer does the unexpected and goes beyond the borders of the commonplace, changes the context, acts as if it is not logical but profoundly irrational, behaves not in the way technology should,

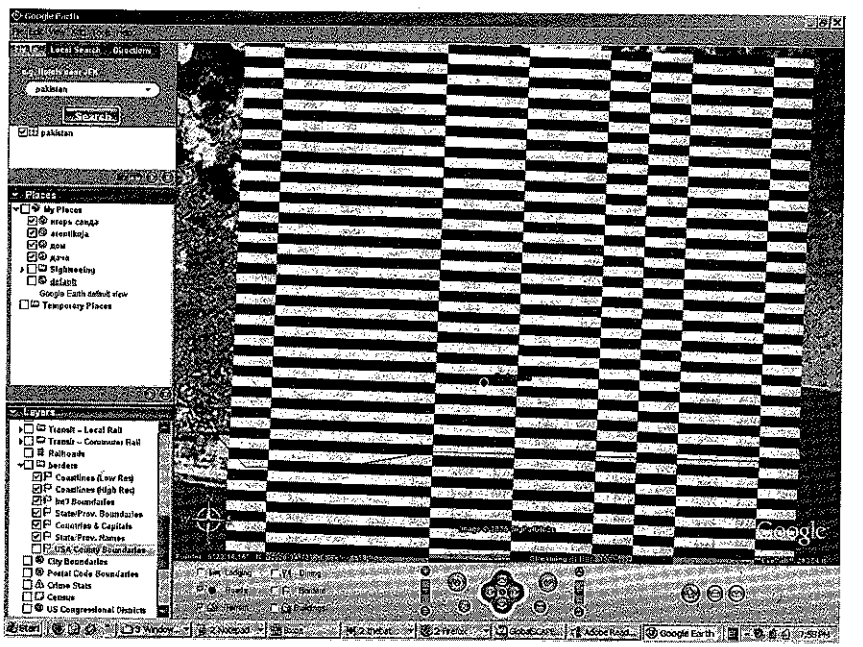

Figure 8 Glitch on *Google Earth*, 2006.

it releases the tension and hatred of the user toward an ever-functional but uncomfortable machine.

> Error sets free the irrational potential and works out the fundamental concepts and forces that bind people and machines. An error [is] a sign of the absence of an ideal functionality, whether it be understood in the technical, social or economic sense.[9]

As with every new aesthetic form, glitches are compelling for artists and designers as well as regular users. Glitches are an important realm in electronic and digital arts. Some artists focus on finding, saving, developing, and conceptualizing glitches, and glitches form entire currents in sonic arts and creative music making. For example, the Dutch-Belgian group Jodi are known for their attention to all kinds of computer visual manifestations that go beyond well-known interfaces. It's enough only to look at their web-page http://wwwwwwwww.jodi.org to get a sense of their style (figure 9). On http://text.jodi.org a user browses through an endless sequence of pages that are obviously

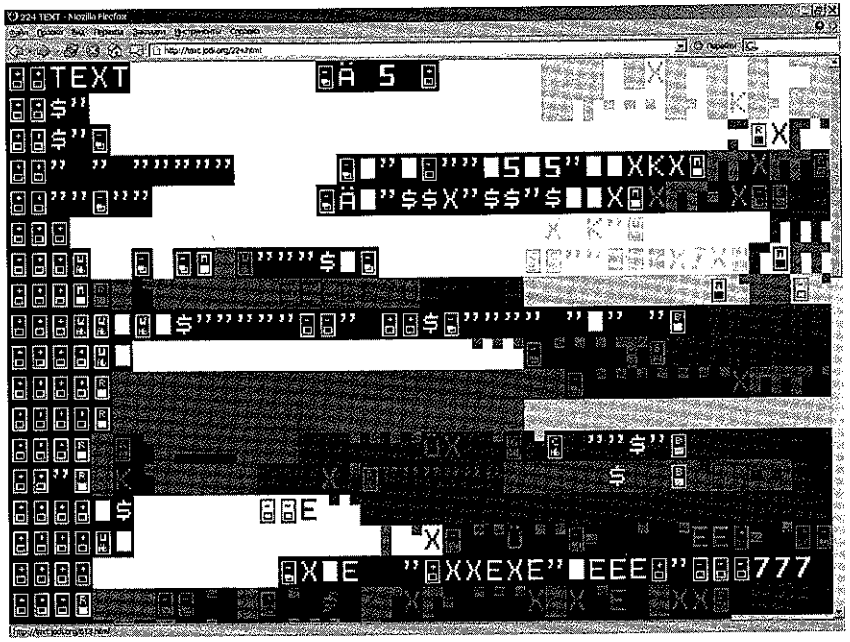

Figure 9 JODI, http://text.jodi.org.

of computer origin, and appear to be both meaningless and fascinatingly beautiful.

Video gamers practice glitching (exploiting bugs in games).[10] Game modifications by Jodi, such as *Untitled Game,*[11] as well as by other artists, such as Joan Leandre's *(Retroyou) R/C* and *NostalG*[12] are achieved by altering parts of the code of existing games (figure 10). The resulting games range from absurd environments in which cars can be driven, but with a distinct tendency to sometimes fly into outer space, to messy visual environments one can hardly navigate, but which reveal dazzling digital aesthetic qualities.

In his *aPpRoPiRaTe!* (figure 11) Sven Koenig exploits a bug found in a video player that makes a video compression algorithm display itself.[13] By deleting or modifying key frames (an encoded movie does not contain all full frames but a few key frames, the rest of the frames are saved as differences between key frames) he manages to modify the entire film without much effort. As a result

Figure 10 Jean Leandre, *(Retroyou) R/C.*

Figure 11 Stefan Koenig, *aPpRoPiRaTe!*.

we get excitingly distorted yet recognizable variants of videos popular in file exchange networks, where such algorithms are widely used. And, of course, with this much work already done for them in advance, we'll see the power of the new aesthetics of the glitch used in commercial products very soon.

Notes

1. Traces of CU-SeeMe can be found through http://archive.org by searching for http://cu-seeme.com.

2. Lewis Mumford, *Technics and Civilization*, 52–55.

3. Vsevolod Meyerhold, "Artist of the Future," in *Hermitage,* no. 6, 10.

4. Nikolay Punin, *The Memorial to the Third International,* 5.

5. See "case modification" in Wikipedia: http://en.wikipedia.org/wiki/Case_modification/.

6. For an excellent account of Golem, see: http://en.wikipedia.org/wiki/Golem/.

7. Mary Shelley, *Frankenstein.*

8. Lautréamont, *Les chants de Maldoror,* Russian edition, 55.

9. Pit Schultz, "Jodi as a Software Culture." in Tilman Baumgarten, ed. *Install.exe,* Christoph Merian Verlag.

10. See "glitch" in Wikipedia: http://en.wikipedia.org/wiki/Glitch/.

11. JODI, http://wwwwwwwww.jodi.org/.

12. Joan Leandre (Retroyou), R/C and NostalG, http://www.retroyou.org/ and http://runme.org/project/+SOFTSFRAGILE/.

13. Sven Koenig, *aPpRoPiRaTe!*, http://popmodernism.org/appropirate/.

Import/Export

Lev Manovich

Although "import"/"export" commands appear in most modern media authoring and editing software running under GUI, at first sight they do not seem to be very important for understanding software culture. You are not authoring new media or modifying media objects or accessing information across the globe, as in web browsing. All these commands allow you to do is to move data around between different applications. In other words, they make data created in one application compatible with other applications. And that does not look so glamorous.

But think again. What is the largest part of the economy of the greater Los Angeles area? It is not entertainment—from movie production to museums and everything in between accounts for only around 15 percent. It turns out that the largest part is import/export business, accounting for over 60 percent. A commonly invoked characteristic of globalization is greater connectivity—places, systems, countries, organizations, etc., becoming connected in more and more ways. And connectivity can only happen if you have certain level of compatibility: between business codes and procedures, between shipping technologies, between network protocols, and so on.