Peter Shultz
Affordances of Computer Art
Media History and Theory, Fall 2005
Holly Willis

While the computer can be used to simulate many things — photography, film editing, animation — its power lies in using the computers unique abilities. Using the computer to create art affords the artist certain possibilities, specifically: databases of materials and algorithmic tools for using those databases. While the computer makes databases more useable, it is the algorithmic tools that lead to results uniquely available with the computer. After looking at how three general areas use these affordances, I will discuss how these tools can be pushed even further.

Algorithmic tools are series of instructions, rules, and guides that instruct the computer what to do with the information in the database. Pre-computer, it was possible for an artist to create a set of potential objects/forms/techniques/edits and then manually go through them, personally making decisions about what to include or not. While this could have been guided by particular rules, the artist was doing the selection herself and therefore intimately involved. What the computer affords is for the artist to first create the rules and then allow the computer to run through them. Not only does this add to the power of the rules (by removing the artist at the point of individual decisions) but it allows for much more complex or purely random rules to be written (see below).

Philosophically it could be argued that all computer graphic art (CG Art) is algorithmic. The computer is given a set of instructions and then left to render them according to the user's settings. Vastly more interesting, though, is not the mechanical work the computer is doing but the creative work. Creative algorithms do more then render, they create objects.

The work of Gilles Tran shows us some possibilities and highlights some the characteristic of art made with the computer. Tran is a CG Art hobbyist who works as an engineer in the animal feed industry in France. Though not a professional, his work is well known in the CG community and is available as posters and postcards online (The123d.com 2004). (While a feed engineer might seem an odd choice, his working process and well document website made him a good choice.) *Family* (2003) is an image of a family living in the apparent bowls of some industrial complex. Pipes, fans, chains all create a frame for a scene of domestic life. The pipes and other non-family elements where all created from an algorithm. Tran set the general parameters and then let the computer determine the specific placement of each piece drawn from a database of forms (Tran, Gilles 2004). Tran uses algorithms like this in many of his works, sometimes subtly (the windows in *Wet Bird* 2000), and some times not *(the Cathedral* (1998), *the Scarecrow* (2000) (Tran, Gilles 2004)) What is unique to this art compared to more traditional fantasy painting is the ability to give up choices and allow the computer to create major portions of the scene.

Tran's work is of particular interest because of the tools he uses. Until recently he worked using POV-Ray, a program without an interface (Tran, Gilles 2004). The user creates scenes by writing text files, not unlike html, and sending those to the render

(Povray.org. 2005). This working process means his images are composed of either pre-existing models (made by others), simple shapes created in code, or objects created from scripts (algorithms.) Since the scenes are created by writing text files, they lend themselves well to cutting and pasting. *Main Street* is a sort of retrospective. The scene is filled with objects Tran created for other projects, taken out of context and placed together (Tran, Gilles 2004).

Also of note is Tran's practice of making several versions of any one scene. *Family* alone has a day version, a night version, and a 360-degree panoramic version. *Wet Bird* similarly can be bought as a poster in three distinctly different versions. Tran, who is also a life long painter, relates these versions to painting's traditions of working in series (The123d.com 2004). Recently he has been working with the same scene and rendering it, with major changes, over 30 different ways (Tran, Gilles 2004). But this kind of versioning is another example of the mutable image discussed by Peter Lunenfeld.

Could these images have been made without computers? Yes. It is hard to imagine any still image that couldn't be made without the computer. The point isn't could, but rather would, these kinds of images have been made before the computer?

Soft Cinema (2002) by Lev Manovich explores the ideas of algorithms and databases to the moving image. *Soft Cinema* is an installation piece made up of a computer that is selecting what the viewer sees and the equipment to project it. While *Soft Cinema* could be considered a system, it is experienced by it's viewers as a time based piece. Manovich began by creating a database of 4 hours of video and animation, 3 hours of narration, and 5 hours of music. He then composed four narrative stories, each broken into several scenes. When one watches the piece, one of the narratives plays out.

A portion of the screen tells you what story and scene you are currently in, and there is a text scroll on the screen recounting the written narrative. The rest of the screen is generated by an algorithm. That algorithm composes the screen layout, selects which clips to use for the next scene and plays them out in sequence (Manovich 2002).

Manovich's direct control of the story only extends to what he has written and the creation of the algorithm. The same algorithm could make different choices when ran successively. Manovich can also change the algorithms between showings of the piece. The clips in the database are tagged by content, style, color and so on. The writer of the algorithms (not necessarily Manovich) creates rules around the tags. Manovich points out that some things in the database where mislabeled, so even the most tightly scripted set of rules will not return the exact results the author might expect. This process is in contrast to using a video editing program where the results are always known (Manovich 2002).

Manovich sees his work as a response to a world where places are made of several layers, and information is given to us in many simultaneous forms. His computer generated layouts specifically reference cable news screens or websites, with their multiple simultaneous flows (Manovich 2002). I see his work as a response to what can be created from a system where all information is encapsulated into a tagged packages and reconfigured at will. *Soft Cinema's* message and form are both dependent on the computer for their creation.

Soft Cinema is intentionally different each time a viewer watches it. Moreover, the database the feeds it changes and new algorithms written for different installations of it. It has no set state, though there is a dvd and book available (Manovich 2002). Like the work of Gilles Tran, "Soft Cinema" has a strong mutable quality.

The most obvious place to look for the effect of computers is in computer specific media. For my final example I will focus on computer games, specifically *Dragon Quest, Sim City,* and *Grand Theft Auto*. While not yet considered to be art, computer games can provide insight into the potential the computer has for working with databases and algorithms.

Dragon Quest was published in Japan in 1986 and came to America has *Dragon Warrior* in 1989 (Dragon Quest Shrine 2004). It is a fairly simple game built for the Nintendo Entertainment System. While some might argue that text based games like *Ultima* were the first database games, since you are basically just moving through and array of text, *Dragon Quest* is a stronger example. The game world is presented as a vast 2D map made up of a grid of squares representing mountains, streams, fields, etc. Not visually represented, but present at each grid point, is a second set of data about the types and frequencies of monster attacks that happen there. While the story of the game is told and controlled by scripted, linear events that the player can not alter, the bulk of the game play is about wandering around the world/grid/database searching for monsters to fight. In this way each play session and each player experience is unique. This kind of game design, where there is a broad world defined by a database and the player negotiates it by traveling around, was popular for role playing games and adventure games but slowly went away — until it was resurrected by the *Grand Theft Auto* game series.

Grand Theft Auto and the large number of sequels that still follow from it are, in concept, exactly like *Dragon Quest*. The player navigates a wide ranging world of objects and hidden attributes. To move forward the player must complete steps in a linear storyline. The fact that *Grand Theft Auto* has rocked the gaming world, and that anyone

who uses the same technique is accused of stealing its design, is a sign of how much the industry has moved away from this form since *Dragon Quest*. What feels so revolutionary about GTA is that with the rise of 3D games and first person shooters, linear, pre-scripted gaming has become the norm.

While both Dragon Quest and GTA are a somewhat mutable experience, neither of them allow the player to stray in a way that changes the story, or cause major change in the game world. Also both games create a databases and allow the player to play in them but neither uses an algorithm to pull from that database and change the players experience.

There are a few, obscure games that have used algorithms and databases in a more robust way. *Way of the Samurai* (2002) for the Playstation 2 and its sequel is the only good example I could find. In *Way of the Samurai* a world is created that only lasts three days. There is no particular plot or acts that must be committed. Instead the player wanders the game area choosing to engage, or not, in various battles-conversations-quests. The game contains a vast database of causes and effects and nearly every decision the player takes affects the world. I am relying on an IGN review for this information (Smith, David 2002).

Sim City series (1989 plus many sequels) is different from any of the previous game examples. Instead of being presented with a completed game world database and then exploring it, the player begins with a world that only has environmental information and then must fill it with their own database entries of objects. The user has a pool of potential objects, from streets to zones of housing to airports, and must construct a working city from nothing. Each of the items the player places in the world both responds

to overriding algorithms already in the world and creates it's own localized effects. *Sim City* has no story, other than what the player writes for themselves, and no specific goals. The algorithms in *Sim City* can be thought of as a force that the player pushes and pulls against in a quest to reach there own, self selected goals.

Sim City moves beyond the idea of a separate database and algorithm, with the algorithm making selections from the database. Instead it is an example of an object oriented programming (OOP) paradigm. OOP varies from traditional programming in that there is a database of objects and those objects have rules for how they interact with each other, instead of there being one primary program that makes all the decisions. In *Sim City* each of the zone the player paints down affects the ones around them.

Object oriented programming could be applied to the previous examples. For instance, if *Soft Cinema* was re-imagined as a OOP project, instead of there being a separate algorithm that controls each scene, each clip would be given it's own personality and desires and those traits would interact with the other clips to create the final piece.

By using computers to their fullest, artist have the ability to create art that would not be possible without them. Specifically by creating and using databases and algorithms they have the ability to respond to our time which is highly mediated by computers and databases.

Dragon Quest Shrine. 2004. *Dragon Quest* [online]. Available from the World Wide Web: (http://www.dqshrine.com/dq/)

Manovich, Lev. 2002. *Soft Cinema* [online]. Available from the World wide Web: (http://www.manovich.net/cinema_future/toc.htm)

Manovich, Lev. 2002. *Soft Cinema: concepts | extended version | 1200* words [online]. Available from the World Wide Web: (http://www.manovich.net/cinema_future/sc_concepts_full.html)

Povray.org. 2005. *POV-Ray - The Persistence of Vision Raytracer* [online]. Available from the World Wide Web: (http://www.povray.org/)

Smith, David. 2002. *Way Of The Samurai: Acquire's new samurai sim doesn't look like much, but possesses hidden depths* [online]. Available from the World Wide Web: (http://ps2.ign.com/articles/361/361421p1.html)

The123d.com. 2004. *The123d interviews Gilles Tran* [online]. Available from the World Wide Web: (http://www.the123d.com/interviews/gilles_tran/gilles_tran01.shtml)

Tran, Gilles. 2004. *List of images* [online]. Available from the World Wide Web: (http://www.oyonale.com/ldc/english/imagelist.htm)

Tran, Gilles. 2004. *Family* [online]. Available from the World Wide Web: (http://www.oyonale.com/ldc/english/family_night.htm)

Tran, Gilles. 2004. *Variations* [online]. Available from the World Wide Web: (http://www.oyonale.com/variations/index.php)