11. game technology for serious applications

immersion does not require illusion but involvement

learning objectives

After reading this chapter you should have an idea how to approach the development of a moderately complex game, and you should also be able to discuss the notion of immersion and argue why using game technology is relevant for serious applications.

Game playing is fundamental to human life. Not only for entertainment, but also to acquire the necessary skills for survival. Game playing can take a variety of forms, but nowadays the dominant game paradigm is undoubtedly the interactive video game, to be played on a multimedia-enhanced PC or game console. Currently, games are being (re) discovered in the academic field, another serious areas of society, as excellent means for both the transfer of knowledge and, perhaps more importantly, for attitude change.

In this chapter we will look at the various issues in developing a game, and more specifically, in section 11.2 at requirements for a promotional game for our faculty and the issues that came up when giving a masterclass game development for high school students using this game. Finally, we will sketch the history of immersive systems, in particular panoramas, in section 11.3, and we will discuss how immersion is to be realized in a game context.

11.1 constructing a game

game playing

> According to Huizinga: "in the game we confront a function of the living creature which cannot be determined either biologically or logically".

From Kress and van Leeuwen (1996)

visual culture

> Games are an increasingly important element in our visual culture.

game programming

- gameplay programming
- game engine programming

2

elements of game design

pre-production, production, post-production

plan/blueprint

In the medium of game creation, we can capture fun in two areas:

fun

- in the general flow of the game experience and
- in the individual moments during a playing session.

of a certain player's unique experience

what is a game?

> Let's begin with a simple definition of what a game might be: a game is a series of processes that takes a player to a result, Schuytema (2007).

interesting decisions, skill aspect of a game

Let's postulate a slightly more robust definition for an interactive electronic game, Schuytema (2007):

interactive electronic game

> A game is a play activity comprised of a series of actions and decisions, constrained by rules and the game world, moving towards an end condition. The rules and the game world are delivered by electronic media and controlled by a digital program. The rules and game world exist to create interesting situations to challenge and oppose the player. The player's actions, his decisions, excitement, and chances, really, his journey, all comprise the "soul of play". It is the richness of context, the challenge, excitement, and fun of a player's journey, and not simply the attainment of the end condition that determines the success of the game.

varoom thung blam

battle

- confrontation on well-established area
- delimited in space/time
- audience/participants who judge victory/loss

game design team

design team

- managers
- producers
- programmers
- testers
- designers

game software architecture

From Sherrod (2006):

game engine

- rendering system – 2D/3D graphics
- input system – user interaction
- sound system – ambient and re-active
- physics system – for the blockbusters
- animation system – motion of objects and characters
- artificial intelligence system – for real challenge(s)

success factors

a brief history of games

From Sanchez-Crespo Dalmau (2004)

a brief history of game programming

- phase i: before space war – hardwired
- phase ii: spacewar on atari – console with game
- phase iii: game console and PC – seperate game development
- phase iv: shakedown and consolidation – player code in data files
- phase v: advent of the game engine – user level design
- phase vi: the handheld revolution – the GameBoy
- phase vii: the cellular phenomenon – larger installed user base
- phase viii: multiplayer games – from MUD to Everquest

remarks:

1) tennis for two William Higinbotham Brookhaven National Labs New York,
1950'
2) Steve Russell, 1961, MIT, spacewar, two player game on Digital PDP-1
3) Atari VCS, Apple II, IBM PC (Dos)
4) Donkeykong, Pacman -¿ Nintendo
5) Doom -¿ Valve Halflife
6) Gameboy with well-established collection of game
7) NTT Docomo I-Mode, Samurai Romanesque
8) MUD (1979), MULE (1983), Ultima/Everquest 1600 hours/year

near future: a truly cinematic gaming experience

3

Screens from Samurai Romanesque.

4

example(s) – *samurai romanesque*

Samurai Romanesque, available on Japan's NIT DoCoMo packet-switched i-mode network, is an example of a mobile game with a large following. This massive multi-player game is developed by the japanese game developer Dwango. It runs on the Java 2 platform Micro Edition (J2ME). Players take, as we read in Krikke (2003) a virtual journey through 15-th century Japan, engage other players in real-time battles, visit historical towns and villages, practice the art of Zen, engage in romances and even can have children. This massive multiplayer role-playing game can accomodate up to half a million players, and is accounted to be a huge success in Japan. *Samurai Romanesque* is an example of a mobile game incorporating features such as position awareness, player history, chatting, and effective graphics. In Krikke (2003), it is further explained how the technology with which the game is implemented positions itself in the *battle for mobile cyberspace*.

research direction(s)– *serious games*

play, learn, become[1]

> Serious games and simulations are poised for a second revolution. Today's children, our workforce and scientists are increasingly playing, learning, and inventing in visually intensive "virtual" environments. In our increasingly experiential economy, immersive educational and training solutions are needed to advance the workforce of tomorrow. Game-based learning and technologies meet this challenge.

peace maker[2]

> Q: With the lion's share of strategy games on the market being devoted to ending a conflict through violence, why was it important to you to emphasize the need for a peaceful solution?
>
> A: When we started to work on the project and looked around at other video games, we encountered the notion that war is much more challenging and

[1] www.virtualheroes.com
[2] seriousgamessource.com/features/feature_071806_peacemaker.php

conflict is essential to engage players. Many people we talked to perceived peacemaking as mere negotiations, where a group of diplomats sit at a table for lengthy discussions and sign agreements. We tried to shed light on what we see as the other side of peacemaking how challenging it is for a leader to gain trust and understanding in the face of constant violence. How difficult it is to execute concessions, while your own population is under stress or feeling despair. In a sense, peacemaking can be more complicated, sophisticated and rewarding than war making, and it is a message that we would like to convey to young adults, the future generation of leaders.

black/white greyscale sepia thermal

5

11.2 game @ VU

In june 2005 we started with the development of a game, nicknamed VU-Life 2, using the Half-Life 2 SDK. We acquired a Cybercafe license for Half-Life 2, with 15 seats, because we would like to gain experience with using a state-of-the-art game engine, and we were impressed by the graphic capabilities of the Half-Life 2 Source game engine.

After some first explorations, we set ourselves the goal:

- to develop a game that could be used for promoting our institute, and
- to prepare a masterclass game development for high-school students.

Our first ideas concerning a game included a game in which the subject chases a target, a game where the subject has to escape, and an adventure game. In the end we decided for a less ambitious target, namely to develop a game which gives the subject information about our institute, by exploring a realistic game environment, representing part of our faculty. As an incentive, a simple puzzle was included which gives the subject information on how to obtain a 'hidden treasure', to be found in a specific location in the game environment. See section 2 for more information on this.

fig. 1: VU-Life 2 – opening screen

With only about eight months time, we decided to do a feasibility study first, to gain experience with the Half-Life 2 SDK technology, and to determine whether our requirements for the game and the masterclass could be met.

For the VU-Life 2 game, we can summarize our requirements as follows:

- the game must provide information about the faculty of sciences of the VU,
- the game environment must be realistic and sufficiently complex, and
- the interaction must be of a non-aggressive, non-violent, nature.

The last requirement has to do with the fact that the VU is by its origin a Christian university, so that anny agressive or violent interaction could hardly be considered to be an appropriate theme for a promotional game.

For the masterclass, we stated the following requirements:

- it must be suitable for beginners, in particular high school students,
- it must explain basic texture manipulation, and
- offer templates for modifying a game level, and finally
- there must be a simple (easy to understand) manual.

The format for a masterclass for high-school students at our institute is three times two hours of instruction. The goal is to attract (more) students for the exact sciences. However, if the masterclass would be too complex, we would run the risk to chase potential students away, which would be highly counter-productive.

In this paper we will report our experiences in developing the VU-Life 2 game and the associated masterclass[3]. The structure of this paper is as follows. In section 2 we will give an impression of the VU-Life 2 game by presenting a typical usage scenario. In the sections that follow, we will discuss the technical issues encountered in developing the VU-Life 2 game, and the assignments for

[3] www.cs.vu.nl/∼eliens/game

the masterclass. In section 4, we will moreover describe the documentation we developed for the masterclass. In section 5 we will discuss the lessons we learned and in particular our experiences in presenting the masterclass to high-school students. And finally, we will draw our conclusions by giving a summary of our efforts and indicating our plans for the future.

VU-Life 2 – the game

To give an impression of the game and how we used the Source game engine and the associated Half-Life 2 SDK, let's start with a typical game scenario, illustrated with a walkthrough.

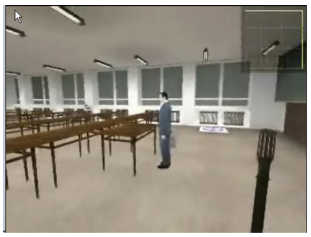


fig. 2(a) lecture room


(b) lecture room


(c) student office

When starting VU-Life 2, the player is positioned somewhere in the game environment, such as a lecture room, fig. 1(a). In the front left corner of the lecture room, middle right of fig. 1(a), there is a place marked as an information spot. The information spot corresponds with one of the nine in the top right of the screen. The player is expected to detect this correspondence by exploring the game environment. The nine squares together form a puzzle, indicating, when all squares are filled, where the hidden treasure can be found. In other words, when the player visits all the nine information spots contained in the game environment, the player has solved the puzzle and may proceed to obtain the hidden treasure.

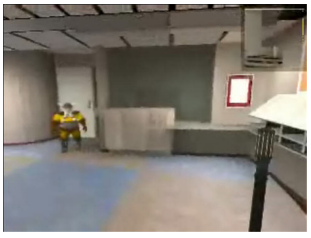
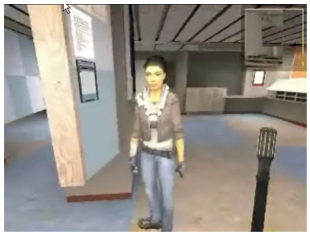


fig. 3(a) student office


(b) student office


(c) student office

To visit all the information spots, the player has to explore the game environment, including another lecture room, fig. 1(b), the student administration office, figs. 1(c) and 2, and the student dining room, fig. 3. While exploring the game environment, the player may read information about the curriculum, meet other students, fig. 2, and encounter potentially dangerous individuals, fig. 3(b).

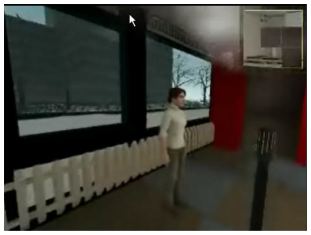


fig. 4(a) restaurant (b) restaurant (c) restaurant

As illustrated in figs. 1-3, the puzzle squares will gradually become filled, and when complete, the combined puzzle squares will indicate the location of the hidden treasure, which is the 7th row of chairs of the lecture room in fig. 1(b).

Despite the fact that we intended to create a non-violent game, we must admit that the hidden treasure actually consists of obtaining the power to use weapons. Form our observations, and this was exactly what motivated us to include this feature, the use of weapons proved to be a most enjoyable aspect for the high school students playing the VU-Life 2 game, in particular when allowed to play in multi-user mode.

using the Half-Life 2 SDK – technical issues

The VU-Life 2 team had no prior experience with the Half-Life 2 Source SDK. Therefore we started by exploring three aspects of the Source SDK: level design with the Hammer editor, making game modifications, and importing (custom) models into Half-Life 2. During the exploration of these aspects we came across various technical issues, which we will discuss below.

level design First, we made various smaller levels. Each level was compiled and tested seperately so that it worked fine as a standalone level. The idea was to combine them, that is to create one large world containing the smaller levels. However, the initial coupling caused several compiling errors. After analyzing the errors, some important restrictons for building (large) levels became clear.

In the second part of the level compilation process called VVIS, a visibility tree of the level is made. This tree is used to tell the renderer what to draw from a given (player) viewpoint in the level. The amount of used brushes (the default shapes for creating a level) determine the size of the visibility tree. The bigger the tree, the longer VVIS will take to build the visibility tree at compile time and the more work the renderer has to determine what to draw at runtime. Therefore, the standard brushes should only be used for basic level structure. All other brushes that do not contribute to defining the basic level structure should be tied to so-called *func_detail* entities. This makes VVIS ignore them so that they do not contribute to the visibility tree, thus saving compiling and rendering time.

In addition, there is a (hardcoded) maximum to the number of vertices/faces you can use for a level. Each brush-based entity contributes to the number of

vertices used. It is possible, however, to reduce the number of vertices used by converting brush-based objects to entities. This is done outside of the Hammer level editor with the use of 3D modelling software and the appropriate conversion tools.

With the above mentioned restrictions in mind we were able to create a relatively large level that more or less realistically represents the faculty of exact sciences of the VU campus. The key locations are, as partially illustrated in figs. 2-4, restaurant (fig. 4), lecture room S111 (fig. 2(a)), lecture room KC159 (fig. 2(b)), student office (figs. 2(c) and 3), multimedia room S353 (not shown).

To give an impression of the overall size of the *VU.vmf* game level, as map information we obtained 6464 solids, 41725 faces, 849 point entities, 1363 solid entities, and 129 unique textures, requiring in total a texture memory of 67918851 bytes (66.33 MB).

game modifications Since a multi-user environment was required. we chose to modify the Half-Life 2 Deathmatch source code, The biggest challenge for modifying the code was finding out how to implement the features for VU-Life 2. To this end, relevant code fragments were carefully studied in order to find out how the code is structured and works. Furthermore, by experimenting, it was possible to get the features working. Below is a list of features for the VU-Life 2 Mod.

- *Player properties* – Players start out immortal, meaning that they cannot "die" while exploring the world. Furthermore, continuous sprinting is enabled, which allows the player to walk around faster.
- *Puzzle HUD* – When the player starts out, the puzzle HUD is the only HUD element displayed.
- *Puzzle setter* – Allows puzzle parts to be displayed on the puzzle HUD.
- *Weapon enabler* – Allows weapons to be enabled/disabled for the player. Enabling the weapons also enables damage, and swithes from the puzzle HUD to the default Half-Life 2 HUD, which displays weapon and damage information along with a crosshair.

importing models Getting a model into the Half-Life 2 environment requires two steps:

- The model must be exported to the custom Valve format smd
- The model must be compiled from *smd* to *mdl* format

The first step required finding the correct plugin that allowed a conversion to the *smd* format. The second step required using Valve tool *studiomdl* and defining a *qc* file, which is used to specify properties for the compiled model. The default Valve tool *studiomdl.exe* proved to be difficult to work with, because it requires a lot of parameters have to be set. By using the StudioMDL 2.0 GUI, compiling the *smd* file was very easy. It sets the appropriate parameters, allowing the user to focus on the compiling of the model.

the masterclass – instruction and assignments

The masterclass consisted of three sessions, two hours each. In the first session, the (high school) students were given an overview and general instructions on how to accomplish the assignments, and were then set to play the VU-Life 2 game.

The assignments, as already indicated in section 1, were:

1. to modify an existing game level by applying different textures,
2. to create objects within an existing game level, and
3. (for advanced students only) to create a new level.

More complex assignments, such as creading a Mod, were considered to be outside of the scope of this masterclass.

The overview and instructions given in the first session included:

- an overview of the history of games,
- a general introduction on modelling characters and objects,
- the use of the Hammer editor, and finally,
- an explanation of the assignments.

The history of games encompassed historic landmarks such as Pong, Tetris and The Sims, as well as a brief discussion of current games like Worlds of Warcraft, and Half-Life 2.

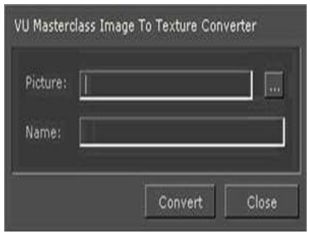
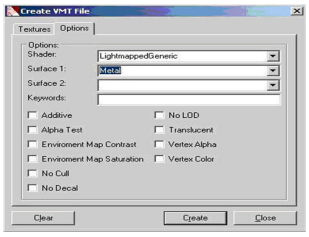
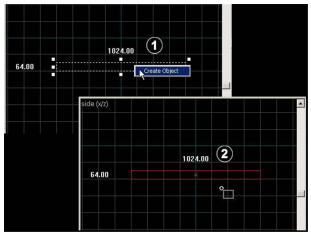


fig. 5(a) converter



(b) VMT tool



(c) camera

In the introduction on modelling an overview was given of the major tools, like Maya and 3DSMax, as well as a brief explanation of notions such as vectors, polygons, textures, lights, and skeleton-based animation.

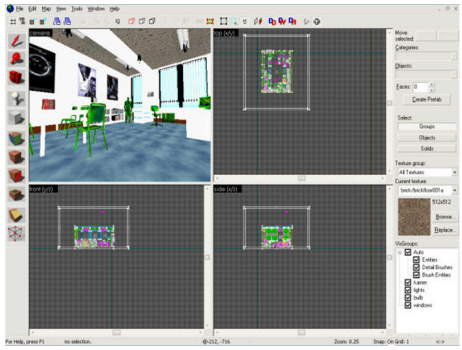
Both the explanation of the use of the Hammer and the assigments were explicitly meant as a preparation for session two, in which the students started working on their assignments.

In addition to the oral overview and instructions, the students were given a manual, that was made available in paper as well as online, to prepare themselves for the assignments. The homework for the second session was to make pictures suitable for the application as textures in the *masterclass room*, which is depicted in fig. 5(a).

To allow the students to easily apply their textures, a texture conversion tool, fig. 4(a), was offered, that converts and image file into a texture for a particular location in the game level based on keywords, e.g. *mc_floor* for the texture on the

floor of the *multimedia room*. Alternatively the students could use the VMT-Edit tool, fig. 4(b), and apply the texture using the Hammer editor, figs. 4(c) and 5.

fig. 6(a) masterclass room



(b) room in Hammer editor

The introduction on how to use the Hammer editor covered the basic tools, including the

- *block tool* – for creating simple object,
- *selection tool* – to select objects for texturing,
- *entity tool* – to select dynamic or interactive objects, and the
- *texture tool* – to apply textures to an object;

as well as how to compile a level into a map ready for play, including an explanation of the BSP (world), VIS (visibility), and RAD (radiosity) components.

The students were explicitly told that the assignments did not involve any programming, creating game AI, or modelling. (To learn these aspects of game development, they were simply adviced to sign up for our curriculum.) Instead, we told them, use your phantasy and be creative!

lessons learned

In the second session, the high school students started working with great fervour, see fig. 7.

Somewhat surprisingly, all students worked directly from the (paper) manual, rather than consulting the online documentation, or the help function with the tool.

fig. 7: masterclass at work

In retrospect, what appeared to be the main difficulty in developing the masterclass was to create challenging assignments for every skill level. In our case, the basic skill level (modifying textures of a template level) allowed the high school students to start immediately. By having optional advanced assignments like creating your own objects, you can keep all students interested, since there are assignments to match the various skill levels.

competition To stimulate the participants in their creativity, we awarded the best result, according to our judgement, with a VU-Life 2 T-shirt and a CD with Half-Life 2. The results varied from a music chamber, a space environment, a *Matrix* inspired room, and a messy study room. We awarded the *Matrix* room with the first prize, since it looked, although not very original, the most coherent.

example(s) – *dead media*

civilisation

> Media are special cases within the history of civilisation. They have contributed there share to the gigantic rubbish heaps that cover the face of our planet or to the mobile junk that zips through outer space.

dead media project

> Together with like-minded people, in 1995, Bruce Sterling started a mailinglist (at that time still an attractive option) to collect *obsolete software*. This list was soon expanded to collect *dead ideas*, or *dead artifacts*, and systems from the *history of technical media*: inventions that appeared suddenly and disappeared just as quickly, which dead-ended and were never developed further; models that never left the drawing board; or actual products that were bought and used and subsequently vanished into thin air.

machines can die

> Sterling's project confronted burgeoning phantasies about the immortality of machines with the simple facticity of a continuously growing list of things that have become defunct.

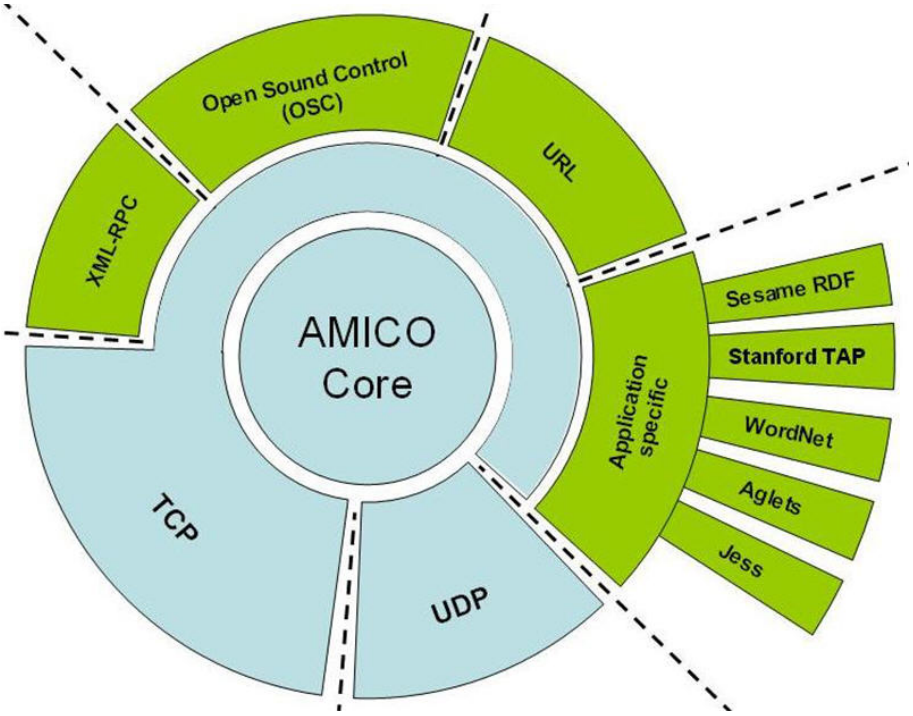
machines can die

> Once again, romantic notions of technology and of death were closely intertwined in the *Dead Media* Project.

research direction(s)– *open source game engine components*

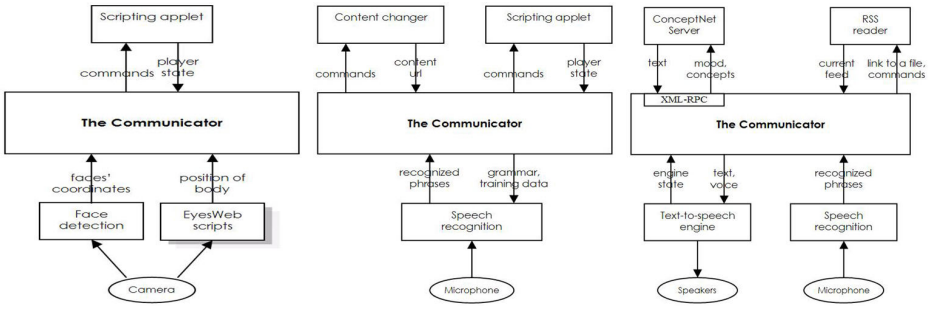
Delta3D[4]

[4] www.delta.org



6

multi-modal interaction – AMICO



camera input voice input news reader

7

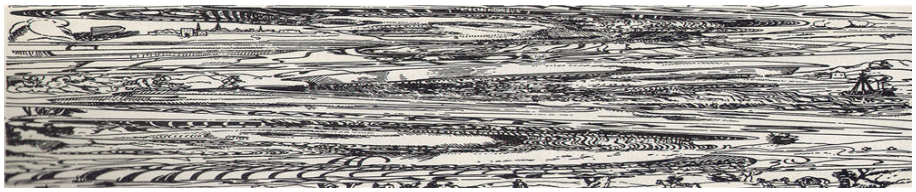
11.3 immersion is not illusion

perspectives The notion of perspective is an interesting notion itself, since it describes both

- the organisation of the image, as well as
- the (optimal) point of view of the viewer.

This intricate relation between viewer and image, dependent on perspective, implies that when looked at from the 'wrong point of view', there will be a distortion of the image. The 'normal' perspective, as we know it, is the 'central' perspective. However, there are variants of perspective that force the viewer into an abnormal point of view, as for example with anamorphismes.

In a multi-dimesnional space often a change of perspective, that is a change of point of view, suffices for the correction of a reducing or distorting projection. Just image how a plane is projected as a line on an orthogonal surface, and a line as a point.



From Kress and van Leeuwen (1996):

realism

> documentary modality of black and white realism ...

www.lichtensteinfoundation.org/

8

visual grammar

> Grammar goes beyond formal rules of correctness. It is a means of representing *patterns of experience*. It enables human beings to build a mental picture of reality, to make sense of their experience of what goes on around them and inside them.

immersion versus illusion

analogon of reality

> Certainly, the image is not the reality but at least it is its perfect *analogon* and it is exactly this analogical perfection which, to our common sense, photography. This can be seen as the special status of the photographic image, it is a message without a code. Roland Barthes, cited from Kress and van Leeuwen (1996), p. 23

immersion

> The concept of immersion when implemented as an artwork surrenders most of the essential properties of an artwork.
>
> Grau (2003), p. 319

properties of artwork(s)

- form
- structure
- function
- processuality
- statement

virtual reality

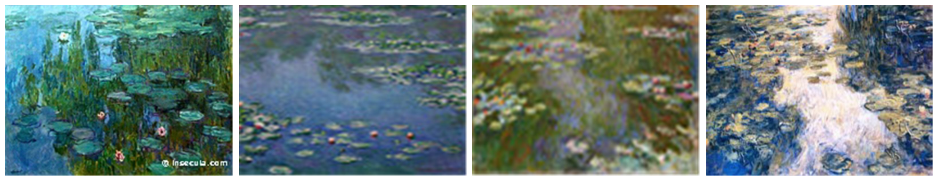
> The idea of *virtual reality* only appears to be without a history: in fact, it rests firmly on historic art traditions, which belongs to a discontinuous movement of seeking illusionary image spaces.

ecstatic transport

> Using contemporary image techniques, immersive art very often visualizes elements that can best be described as Dionysian: ecstatic transport and exhilaration.

collective memory

> It is an apparent feature of the concept of immersion that it engages with the spatial and pictorial concentration of the awareness of one's own people, the formation of collective identity through powerful images that occupy the function of memory.



9

new media

> A consequence of the constitutive function of artistic-illusionary utopias for the inception of new media of illusion is that the media are both a part of the history of culture and of technology.

10

realism versus naturalism

realism

> A *realism* is produced by a particular group as an effect of the complex of practices which define and constitute that group.

naturalism

> Each realism has its *naturalism*, that is a realism that is a definition of what counts as real, a set of criteria for the real, and it will find its expression in the *right*, the best, the most *natural* form of representing that kind of reality, be it a photograph or a diagram.

dominant paradigm

> The dominant standard by which we judge visual realism (and hence *visual modality*) remains for the moment, naturalism as conventionally understood, *photorealism*.

Kress and van Leeuwen (1996): *realism is a social construct*

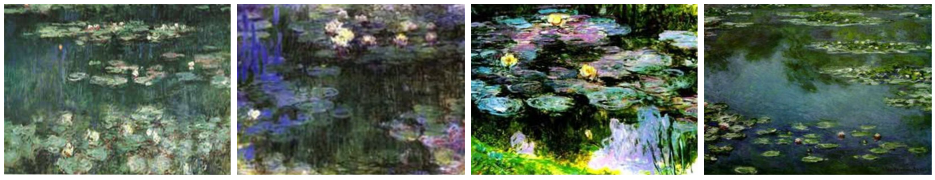
involvement– relationship(s) with application(s)

Sidney Fels, UIU04 keynote, *designing intimate experience*

Observation: "people form relationships with objects external to their own self".

aesthetics of interaction

- response – object disembodied from self
- control – self embodies object
- reflection – self disembodied from object
- belonging – object embodies self



11

example(s) – *Monet's Nympheas*

perso.orange.fr/art-deco.france/nympheas.htm

About Monet's Waterlilies Panorama in Giverny, from Grau (2003):

mass medium

> Thus, one year after Monet's death and fifty years after his *Impression soleil levant*, a late example of modern art reached the changed artistic landscape of the 1920's, transported in a derivative of *the* mass medium for images in the 19th century.

research directions– *information art*

See Wilson (2002).

Quote from MIT Press, Leonardo series:

cultural convergence

> *The cultural convergence of art, science, and technology provides ample opportunity for artists to challenge the very notion of how art is produced and to call into question its subject matter and its meaning in society.*

from Grau (2003).

tele-presence

- notions of artificial life
- fusion with (infinite) virtual image worlds
- transformation of self into digital data

human aspirations

> *Telepresence also combines the contents of three archetypal areas of human aspirations: automation, virtual illusion and metaphysical views of the self.*

cybergnosis

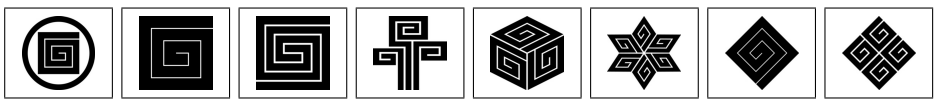
> What is being preached is the phantasm of union in a global net community, cybergnosis, salvation through technology, disembodied as a post-biological scattering of data that lives forever.

zealots

> What we observe are hyperzealots of a new technoreligion running wild, zapping, excerpting and floating in cyberspace.

aesthetics

> Since the eighteenth century, aesthetic theories have regarded *distance* as a constitutive element of reflection, self-discovery and the experience of art and nature.



questions

game technology for serious applications

1. (*) What are the elementary steps in game development? Discuss the role technology plays in determining the game development project trajectory.

concepts

2. What phases can you distinguish in the actual development of a game?
3. Give arguments pro and con the use of a game engine.
4. Discuss the notion of immersion, and explain why immersion does not necessarily imply illusion.

technology

5. Characterize the built-in functionality that comes with a game engine.
6. Give a characterization of the tools that come with the Source Half Life 2 SDK.
7. Give a brief description of the history of immersive environments and application.
8. Discuss, on a suitable level of abstraction, the immersive features of games.

projects & further reading

As a project, develop a non-violent game using the Source SDK. For example, you may develop an application that gives a community of users access their personal collections of photographs.

One interesting feature to explore is the use of narratives, that is a kind of guided tour that gives a user an overview of the collection of photographs by means of a story, taking (in other words) the user by the hand in navigating the gane space.

For further reading I suggest, apart from the manuals and learning materials that come with the Source SDK, books on game development such as Luna (2003). XXX

the artwork

1. *digital beauties* – taken from Wiedermann (2002).
2. Masereel, social realist works
3. Roy Lichtenstein, 1962
4. Masereel, social realist works
5. images from *Samurai Romanesque*, see section 1.3
6. HalfLife 2 shader programming
7. VU-Life 2 – opening screen
8. VU-Life 2 – screenshots
9. VU-Life 2 – screenshots
10. VU-Life 2 – screenshots
11. VU-Life 2 – tools
12. VU-Life 2 – tools

13. VU-Life 2 – masterclass
14. diagram AMICO core
15. diagram AMICO applications
16. Roy Lichtenstein, 1962, Stillives
17. Monet, Nympheas
18. Monet, Nympheas
19. Monet, Nympheas
20. signs – abstract, van Rooijen (2003), p. 146, 147.