

Game Programming in relation to Graphics by Raphael Mun

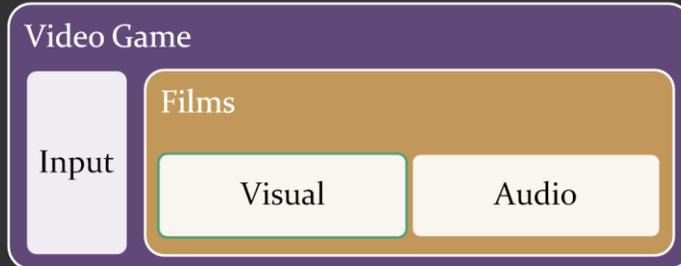
Quick Overview

- The Fundamentals of Media and Games
- A Bit of Game History
- The “Bad”s: What NOT To Do
- Graphics in Game Development
- Fitting Pieces into a Masterpiece
- The Compromise View
- Back to the Basics

Quick Overview:
Games on the whole
and where does graphics fit in?

THREE LESSONS!

The Fun-damentals



“It’s like a movie... with interaction!”

Breakdown of parts of a game

- Visual
- Audio
- Input

In comparison to movies, it adds the element of interaction
a.k.a. "Gameplay"

- Camera movement, player control, World Reaction

What this means... is Visual and Audio must be supported by input

The Past, The Present, ...

Historical Past of Game Development

Historical view of development

- What were the "best" games?

Aesthetics + Interaction at its best, even at 2D

Pac-Man, TMNT, Space Invaders, etc...

“The Classics”



Pac-man, Mario, Duck Hunt, Tetris... What did they do well?



All Three: Visual, Audio, and Input were superb! Now to explain...



How is PacMan visually stunning at low-resolution with just a couple of colors? Take the example of the famous painting of a black dot on a white background. It might not be the most realistic, but you're sure to remember it. Similarly, pacman is one of the most widely-recognized video game characters in the world and it is NOT because all of those people have played the game, although many people have and it became very popular.

Bubble Bobble

THE THEME SONG!!!!
+ 2 Player Co-Op
+ lots and LOTS of color



Bubble Bobble on the other hand was in fact visually stunning at the time with so many colors. It played well and of course the theme song stuck in your head for the rest of the day.

A Couple of Others

The Games

- Mario
- Castlevania
- Duck Hunt
- PONG

The Main Fun-damental

- Music
- Prettiness
- The Light-gun!
- SoundFX

Mario: Everyone knows the theme and the mustache

Castlevania: Eye-Candy... and awesome gameplay

Duck Hunt: It had a gun!!!!

PONG: *Poink*, do-do-do,*Poink*

So The Formula For A Good Video Game Is...

THERE IS NONE

There is no single formula for the 'best game' due to its subjective grading. Each game is considered as one whole product rather than its various features. This can be good or bad.

It can be good because it means that you have just as good of a chance of making the 'best game' as anyone else because there can be multiple 'best games'.

However, it also means that one feature will never make a game great. (IMPORTANT!) At the very least, one of the three fun-damentals must be good and the others should not detract away.

Games Like...



Myst – hardly any input, but the mouse input facilitates the effect

You Don't Know Jack – hardly any images! But the text and audio is fantastic and the input never intrudes.

Games Like...



Lesson #1: If any of the three had detracted from the game, it could have ruined the game...

case in point

The "Bad"s: What
NOT To Do

The Visual Disaster

Drake of the 99 Dragons



Not only did this game have a terrible title with a character that looks more like Frankenstein than any dragon, there are artifacts you can tell everywhere. The game that was trying to be Matrix-like received a 1.6-Abysmal rating.

The “Bad”s: What
NOT To Do

The Audio Disaster

Revolution X



Originally a light-gun arcade game with music, it was ported over to the SNES with terrible, looping music and repetitive enemies, and described in *Electronics Gaming Monthly* as the “worst second of your life repeated forever” and “biblically horrific” ...



One of the possibly better implementation of PacMan in 3D, people ended up playing from the mini-map on the top-left. Additional dimensions do not necessarily make a game better, as demonstrated in several other games such as Bubble Bobble 3D... Where's the joystick and how are the PacMen supposed to move when pushed left or right? More complexity of gameplay can occur from transitioning from 2D to 3D. Many games have fallen to this trap.

Graphics in Game Development

- Visual Side-Kick?
- Basic Building Blocks
 - Images
 - Polygons
 - Lights & Materials...

So where does Graphics come in?

- Visual support!!!

Graphics should be at the least, non-intrusive to a game
no graphics could be better than bad graphics
like... listening to a good soundtrack of a movie v.s.

watching a terrible movie with a good soundtrack

- What we have learned so far in graphics applying to games
- basic building blocks of images, polygons, lighting
- these are the materials to build a scene, the rest is art
- 'cel-shading', 'per-pixel lighting', normal-mapping, etc. are just catch-phrases, imagine all of these together = BAD!!!!
a realistic and non-photorealistic superbly terrible game...

Components of a Game Program

- 2D
- 3D
- Physics
- Music & Sound
- Input
- AI
- Networking
-

Taking a look at the components of programming a game...

- 2D
- 3D
- Physics
- Music & Sound
- Input
- AI
- Networking
- list goes on...

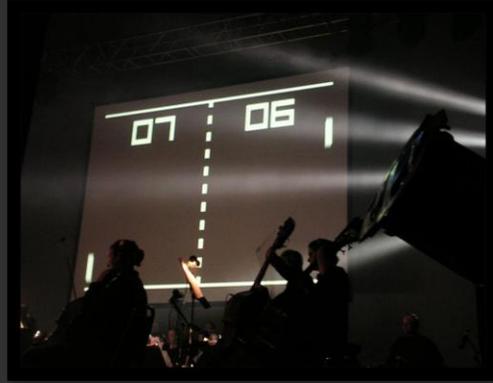
So from here, graphics is closely linked to several of the components, but it consists mainly of the 2D and 3D Rendering.

Event-driven programming is crucial because the programs are too complex to be written all at once, each part needs to be independent and tested as such.

The 3D part is what's been mainly focused in class, but a realistic looking car doesn't make a good racing game, although a crappy looking one definitely can break good a racing game, so the biggest part in game programming is compromising.

The Compromise View

What the heck is it?



Theme of compromise:

We've run into this problem multiple times already in the course in which you often need to compromise performance for quality and vice versa, clearly exemplified in raytracing. Programs can take a longer time to process to produce good-looking results, so with regards to computer graphics in video games, the key is stay real-time and either making the rest of the game better or not doing them harm.

For example, if you are programming a racing game, you might compromise on the texture quality of the surroundings and allow higher-resolution textures on your cars since that is what the player will be looking at most of the time. And the objects surrounding the player will just need to 'look' real so shiny objects will only need to look as if they're reflecting the surroundings and you might optimize around that. We implemented a version of this already when we did the skybox, which puts some limitations on the camera in relation to the sky and the horizons but the cost is low and the result is quite acceptable.

Most of good game programming is balancing this result to cost ratio, along with the time investment needed for such features as well. "How much will a feature contribute to the quality of the game?"

The Compromise View

What the heck is it?

**Lesson #2: It's about the "looks-pretty-good"s
(Looks Pretty and Looks Good [enough])**



Theme of compromise:

We've run into this problem multiple times already in the course in which you often need to compromise performance for quality and vice versa, clearly exemplified in raytracing. Programs can take a longer time to process to produce good-looking results, so with regards to computer graphics in video games, the key is stay real-time and either making the rest of the game better or not doing them harm.

For example, if you are programming a racing game, you might compromise on the texture quality of the surroundings and allow higher-resolution textures on your cars since that is what the player will be looking at most of the time. And the objects surrounding the player will just need to 'look' real so shiny objects will only need to look as if they're reflecting the surroundings and you might optimize around that. We implemented a version of this already when we did the skybox, which puts some limitations on the camera in relation to the sky and the horizons but the cost is low and the result is quite acceptable.

Most of good game programming is balancing this result to cost ratio, along with the time investment needed for such features as well. "How much will a feature contribute to the quality of the game?"

Lesson #2: It's about the "looks-pretty-good"s
(Looks Pretty and Looks Good [enough])

Common Compromises

- LoD (Level of Detail)
- A.I. Updates
- Framerate Lock
- Input Polling
- Billboarding
- Genre-Specific
-

Some more common 'compromised features' of games: (Questions Welcome! more detail on the chalkboard here...)

level of detail - Objects far away will use smaller (mipmapped) textures and lower poly-count versions of objects, and possibly not even animate

A.I. updating - characters can often appear to look as if they are changing by setting a state-machine based on time, similar to other procedural methods. So only characters within a region around the player can be active and mostly independent from each other with limited processing.

Framerate Lock - drawing to your screen only at certain framerates is one of the most common ways to speed up your processing, leaving you with free processing time. Also one of the reasons that event-driven programming is more necessary. Rendering is usually one of the most time consuming processes so this is one of the first things to do on an engine to optimize.

Input Polling - Games don't need to check for input since there is a response time involved between something on the screen and the person's reaction to it, so input keys are often polled at some millisecond intervals instead.

Billboarding - Many complex objects like trees have a symmetric property that allows them to be represented with a just a few textured quads. This significantly reduces the polygon count, but too many billboards drawn on top of each other can increase the fillrate.

Genre-specific optimizations - Often there are significant compromises you can make depending on the type of game you are making. For instance, a quadtree can be much quicker than an octree and more appropriate if a game takes place outdoors. Or for final-fantasy-esque RPG games, there are no physics involved and you can instead invest in content and story creation.

Back to the Basics

- The 3 Fun-damentals
- Feature Compromising
- Playability
 - Iterations!



Back to Basics...

Games make the Visual And Audio into a dynamic experience but the struggle is in making that experience as seamless as possible while creating a masterpiece and/or keeping it fun.

Blizzard is known for making 'good' games but the real secret is in their "always-late" releases. Iterating over each review of your game will bound to make it better each time!

Back to the Basics

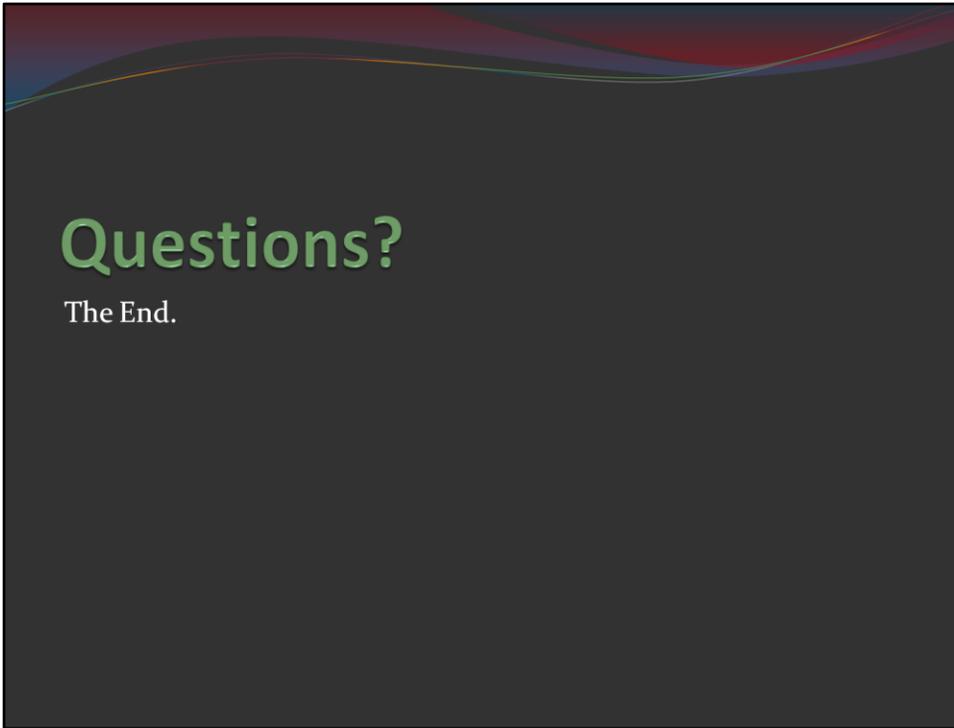
- The 3 Fun-damentals
- Feature Compromising
- Playability
 - Iterations!

Lesson #3: If working on a game for 5 years doesn't tire you of the game, it certainly won't for the ones voluntarily playing.



Lesson #3: If working on a game for 5 years doesn't tire you of the game, it certainly won't for the ones voluntarily playing.

Meaning... not that you should work on a game for 5 years, but if you feel that your game isn't fun at the very basic prototypes, go back to the drawing board!



Questions Answered Here.

Resources

- <http://www.screenhead.com/reviews/6-more-video-games-that-should-be-movies/>
- http://www.theinventory.org/classic_games.gif
- <http://www.geekrant.org/files/2006/pacman-hide.png>
- <http://www.klov.com/images/11/118124206555.png>
- <http://games.multimedia.cx/wp-content/uploads/myst-1.jpg>
- http://www.shacknews.com/extras/ydkj_retro/070322_ydkj_vol2_10.jpg
- http://www.gamespot.com/xbox/action/drake/images.html?om_act=convert&om_clk=gsimage&tag=images;all
- http://en.wikipedia.org/wiki/Video_games_notable_for_negative_reception
- <http://www.x-arcade.com/service/Newsletter/2005/Pacifi3d.jpg>
- http://www.geekshow.us/2006/08/wizard_world_06_video_games_list.html
- <http://blogs.sun.com/kevin/resource/starcraft2.jpg>