

12 Final Projects

Final project ground rules

Group size: 2 to 5 students

- choose your own groups
- expected scope is larger with more people

Charter: make a simple game with cool graphics

- game play should be simple — not the emphasis here
- graphics has to tackle significant challenges
- generally must be 3D (but talk to me if you have cool 2D ideas)

Deliverables

- project proposal, around A7
- milestone presentation, near end of classes
- final project presentation, during final exam time

What makes for interesting graphics?

Rendering

- fancy materials
- translucency
- procedural textures
- environment illumination

Animation

- skinning + morph targets
- collision detection
- particle system smoke, fire, explosions
- procedurally animated water, wind, etc.

What makes for interesting graphics?

Modeling

- subdivision surfaces
- voxelized terrain
- procedural models (plants, terrain, cities, ...)

Imaging

- bloom, lens flare (camera or eye)
- HDR tone mapping

Complexity management

- frustum culling, occlusion culling
- level-of-detail management

Overlap with other projects

In general, it's OK with me to build on your own earlier or concurrent work

- but you need to talk to me about it!

You have to disclose overlaps

- work that comes from projects you did for other courses (e.g. in 4620)
- work that comes from personal projects you did before this course
- work shared with concurrent projects for other courses (e.g. co-projects with 4152 or 5643)
 - in this case need to talk with **both** instructors!
- submitting overlapping work without saying anything is dishonest

Final Project Proposal

2-page description of game

- the “story board” equivalent
- say what constitutes the technical “meat”
- tentative schedule with allocation of team-members to tasks

Major areas of focus

- one primary, one secondary; larger groups: 2 primary, 2 secondary
- e.g. primary rendering, secondary animation or modeling

Project requirements

Must go significantly beyond PAs

- combine multiple techniques in interesting ways
- implement significant new techniques not in PAs

Quality product expected:

- nicely polished imagery
- principled methods
- correct implementations (with test results to prove it!)
- how you achieve results is as important as the results themselves

Code Base

Pick whatever code base you want

- Build on codebase from 5625 or 4620 (recommended)
- Start from scratch (probably bad idea)
- C++ (if you are confident)
- Pyglet, WebGL, ... (for the independent-minded)
- no game engines (talk to me about the line between graphics library and game engine)

Resources

Get models off the web

- do not spend all your time trying to model 1 person or 1 object.

GPU Gems 1, 2, 3 for ideas

- these are on NVidia developer pages

Articles referenced in lecture

Akenine-Möller et al.

NVidia and AMD demos and examples

Modeling

Procedural terrain renderer

- Blend textures based on height and normal
- Create grassy, rocky, snowy/rocky, and snowy/ice regions

Terrain view-dependent Level-of-Detail

Nice outdoor rendering

- build on sun-sky model in PAs
- achieve nice lighting and shadows for trees, ground, water, etc.

Subdivision surfaces (PA5)

- View-dependent level-of-detail

Performance based optimizations require good evaluation

High-Level Game Ideas

- Adventure Game
 - Maze-like setting
 - Might require collision detection
- Pinball
- 2D game behavior with 3D graphics is OK

High-Level Game Ideas

- **Terrain games**
 - Requires real-time terrain mesh that supports deformations
 - Projectile/explosion animation
- **Role-playing Game**
 - An action-oriented RPG might be interesting.
 - Visually interesting scenery, spells, etc.
- **Space Flight Simulator**
 - May require some view-culling
 - Ample opportunities to use particle systems

High-Level Game Ideas

- **First Person Shooter**
 - Some spatial hierarchy (BSP), collision detection...
- **Other feature ideas**
 - Feel free to implement wild and crazy effects, as long as you can explain to us why the effect on screen is the intended result and not a bug!

Game Mechanics (Slides by Walker White)

Actions

- What the player does
- Examples:
 - Move
 - Jump
 - Shoot
- Should NOT be your focus

Interactions

- What the state of the world is
- Examples:
 - Collisions
 - Restitution/Destruction
 - Visibility
- Should be your focus

Goal: Take a principle from computer graphics and implement a **single** interesting game mechanic

Other Game Design Concepts

(Slides by Walker White)

Objectives

- What the player wants to do
- Examples:
 - Reach an exit door
 - Kill/tag an enemy
 - Outrace/outlast an enemy
- Keep this simple!
 - Reach an exit
 - Tag a (dumb) opponent

Challenges

- Makes the objective difficult
- Examples:
 - Maze environments
 - Enemy speed
 - Enemy AI
- Also keep this simple!
 - Keep AI to simple visibility
 - Well designed mazes with a timer can be fun

Game Ideas

(Slides by Walker White)

- Stealth games
 - Simple visibility
 - Shadows (“visibility” = speed * shadows)
- Maze games
 - Reflection to swap between worlds
 - Shadows and lighting change geometry
 - Particle systems as moving hazards
 - Finite element modeling for destructible terrain
- Tag/Chase games are maze+enemy