

University of North Texas

I. Parberry, E. Carson, J. Nunn, J. Scheinberg, J. Cole

Why an Educational Game Engine?

- Public domain game engines exist
- Selection of a game engine is a major decision that can make or break a game programming class
- Students need an engine that is flexible, extensible, stable, and well-documented

Production Code vs. Instructional Code

- Existing game engines are *production code*, code that is designed to work, and nothing else
- An instructional game engine should be written to be easy to understand and modify, especially to relatively inexperienced students
- It should obey the educational principle "Proceed from the known into the unknown"

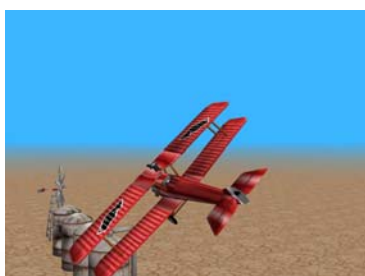
Incremental Development

- Develop a game engine as a sequence of demos, each built on its predecessor
- Each demo showcases a new feature demonstrated in rudimentary form, leaving room for students to enhance
- The trick is getting it *complex enough* to convey the fundamental principles, yet *simple enough* for students to understand
- Principle already proven at UNT with over a decade's experience teaching game programming using incremental development with *Ned's Turkey Farm*, resulting in 37 alumni in the game industry



Introducing SAGE

- Simple Academic Game Engine using incremental development
- Based on DirectX 9.0 and the toolkit from *3D Math Primer for Graphics and Game Development*



Model Format

- A major issue for artist-programmer communication
- File format converters exist, both proprietary and public domain, but over a decade's experience teaching game programming at the University of North Texas has shown that none will function adequately in an academic environment
- The problem is finding a file format that artists can *write* using standard art tools such as Maya, 3D Studio Max, Lightwave, and programmers can *read and parse* in C++.
- SAGE will use S3D, simple 3D format from *3D Math Primer for Graphics and Game Development*
- Text format easily readable by programmers
- Plug-in provided for Maya

Deliverables

- A sequence of 7 game demos, each building from its predecessor
- Visual C++ project files
- Code and content (art, sound)
- Doxygen generated documentation for the demos
- Microsoft Word generated tutorial for each demo
- Code will be released on an open source license
- Package will be posted on the LARC website and in the Microsoft Developer Network Academic Alliance Curriculum Repository

Planned Demos

1. Model importation and display
2. Terrain generation and rendering
3. Game engine architecture, including object management, input with DirectInput
4. Shaders using HLSL
5. Collision detection using axially-aligned bounding boxes
6. Particle engine for explosions, fire, and water
7. Sound using DirectSound

Releases

- SAGE project start June 27, 2005
- Pre-beta release of engine August 2005
- Demo beta releases planned monthly starting September 2005
- Final release June 2006

For More Information

Project website: <http://larc.csci.unt.edu/sage/>

Source code download: <http://larc.csci.unt.edu/sage/download/>