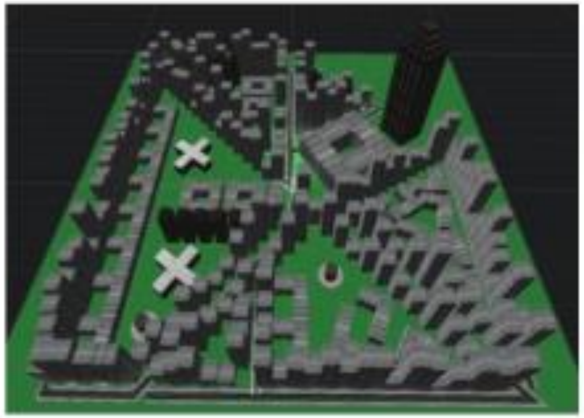
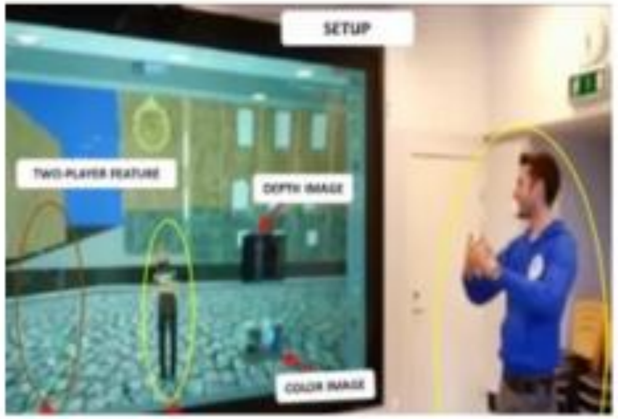
DD3336/DH2650





http://www.csc.kth.se/~chpeters/DD3336/

MIDDLEWARE
A Brief Overview

Christopher Peters

HPCViz, KTH Royal Institute of Technology, Sweden

chpeters@kth.se

http://www.csc.kth.se/~chpeters/

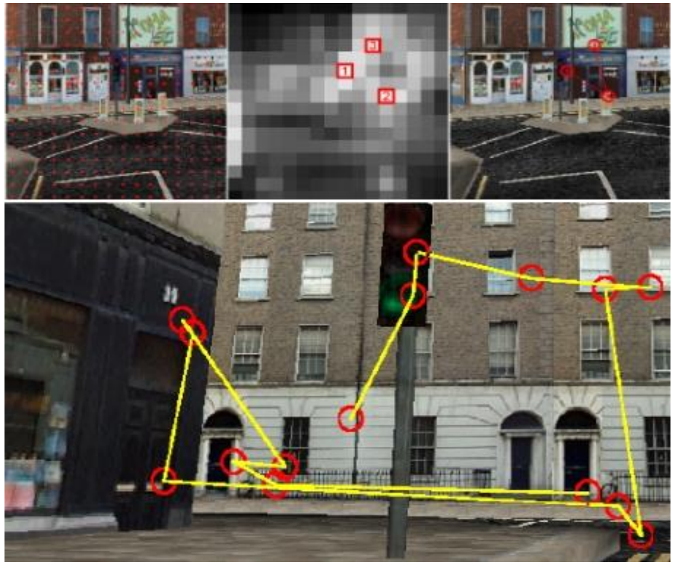
Who am I?

- Christopher Peters
  - email: chpeters@kth.se
  - https://www.kth.se/profile/chpeters/
- Unofficial Irish Doom E1M1 champ., ~1994
- Havok, 1999
- Associate Prof. (Docent)
- Research:

Computer graphics & animation

Game technologies

Perceptual computing



PhD, Visual Attention for Animating Characters

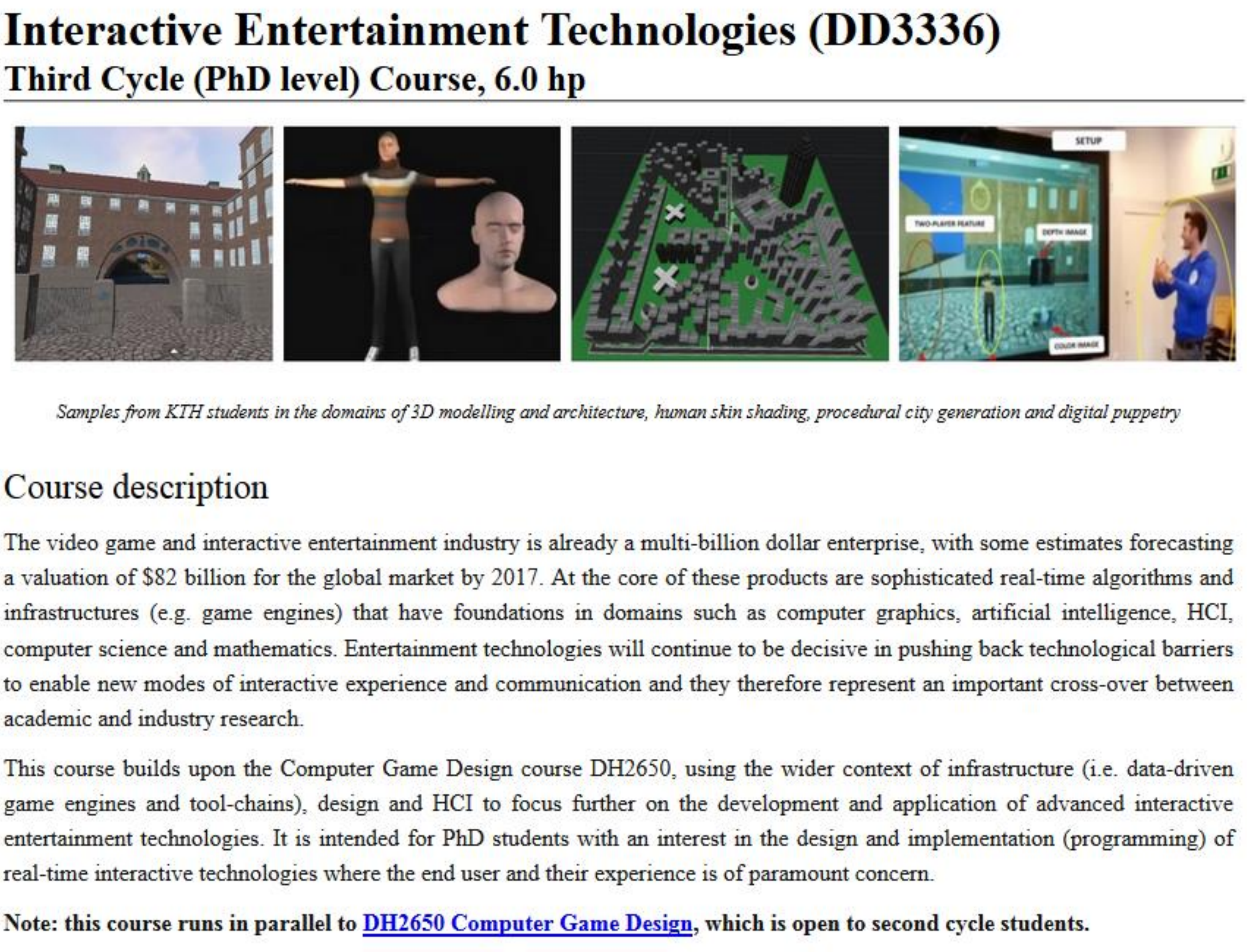
Metropolis Project, Trinity College Dublin

My Teaching

- **DD3336**, Interactive Entertainment Technologies (PhD level)
- **DT2350**, Human Perception for Information Technology
- **DH2323**, Computer Graphics and Interaction
- **DH2320**, Introduction to Visualization and Graphics
- **DD1354**, Models and Simulation

- Visualization (VIC) Studio

4K screen, Oculus Rift, eye-trackers, etc



Interactive Entertainment Technologies (DD3336)

Third Cycle (PhD level) Course, 6.0 hp

Samples from KTH students in the domains of 3D modelling and architecture, human skin shading, procedural city generation and digital puppetry

Course description

The video game and interactive entertainment industry is already a multi-billion dollar enterprise, with some estimates forecasting a valuation of $82 billion for the global market by 2017. At the core of these products are sophisticated real-time algorithms and infrastructures (e.g. game engines) that have foundations in domains such as computer graphics, artificial intelligence, HCI, computer science and mathematics. Entertainment technologies will continue to be decisive in pushing back technological barriers to enable new modes of interactive experience and communication and they therefore represent an important cross-over between academic and industry research.

This course builds upon the Computer Game Design course DH2650, using the wider context of infrastructure (i.e. data-driven game engines and tool-chains), design and HCI to focus further on the development and application of advanced interactive entertainment technologies. It is intended for PhD students with an interest in the design and implementation (programming) of real-time interactive technologies where the end user and their experience is of paramount concern.

Note: this course runs in parallel to DH2650 Computer Game Design, which is open to second cycle students.

My Teaching

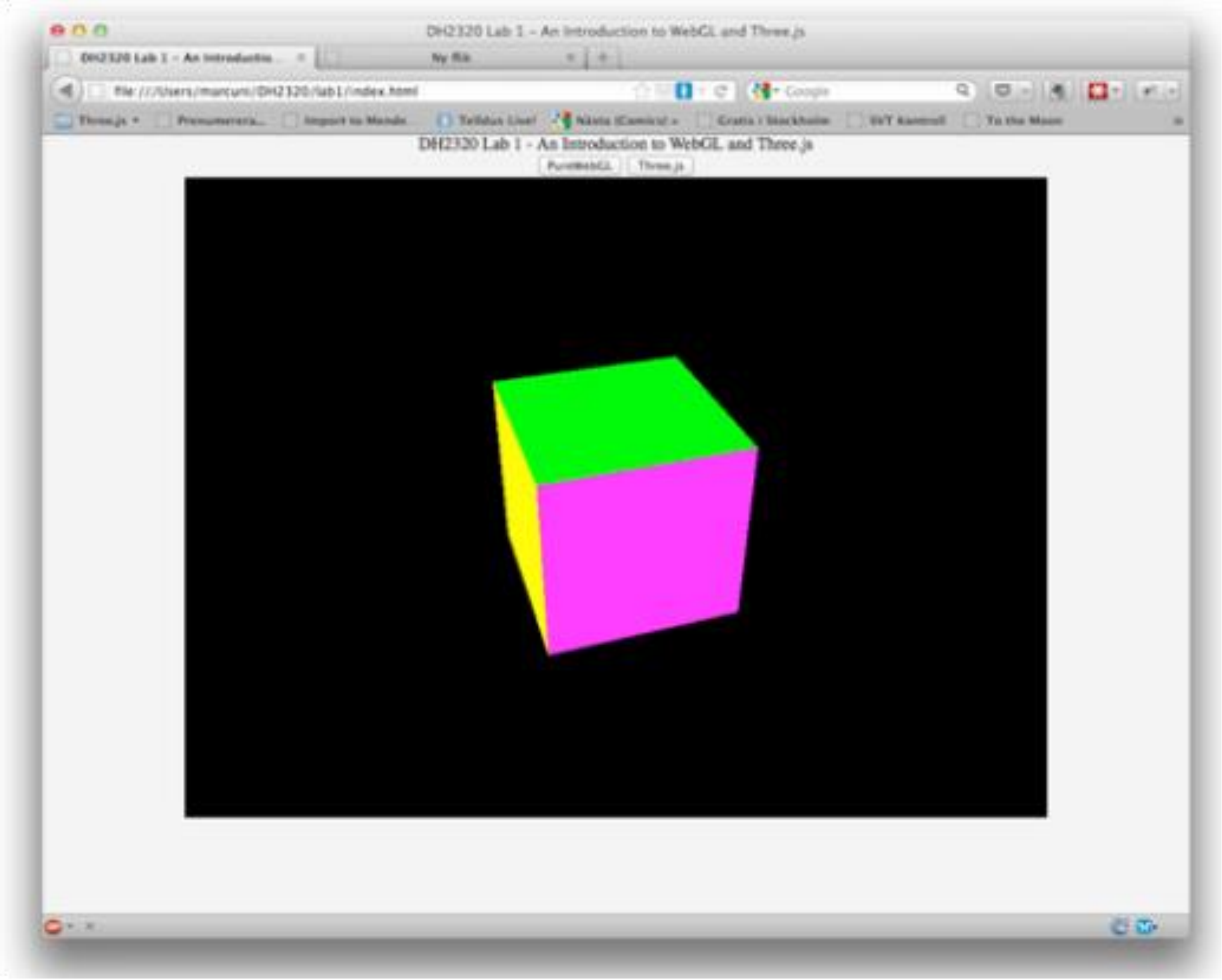
- **DD3336**, Interactive Entertainment Technologies (PhD level)
- **DT2350**, Human Perception for Information Technology
- **DH2323**, Computer Graphics and Interaction (P4)
- **DH2320**, Introduction to Visualization and Graphics
- **DD1354**, Models and Simulation (P3)

- Visualization (VIC) Studio

4K screen, Oculus Rift, eye-trackers, etc

Today's Question

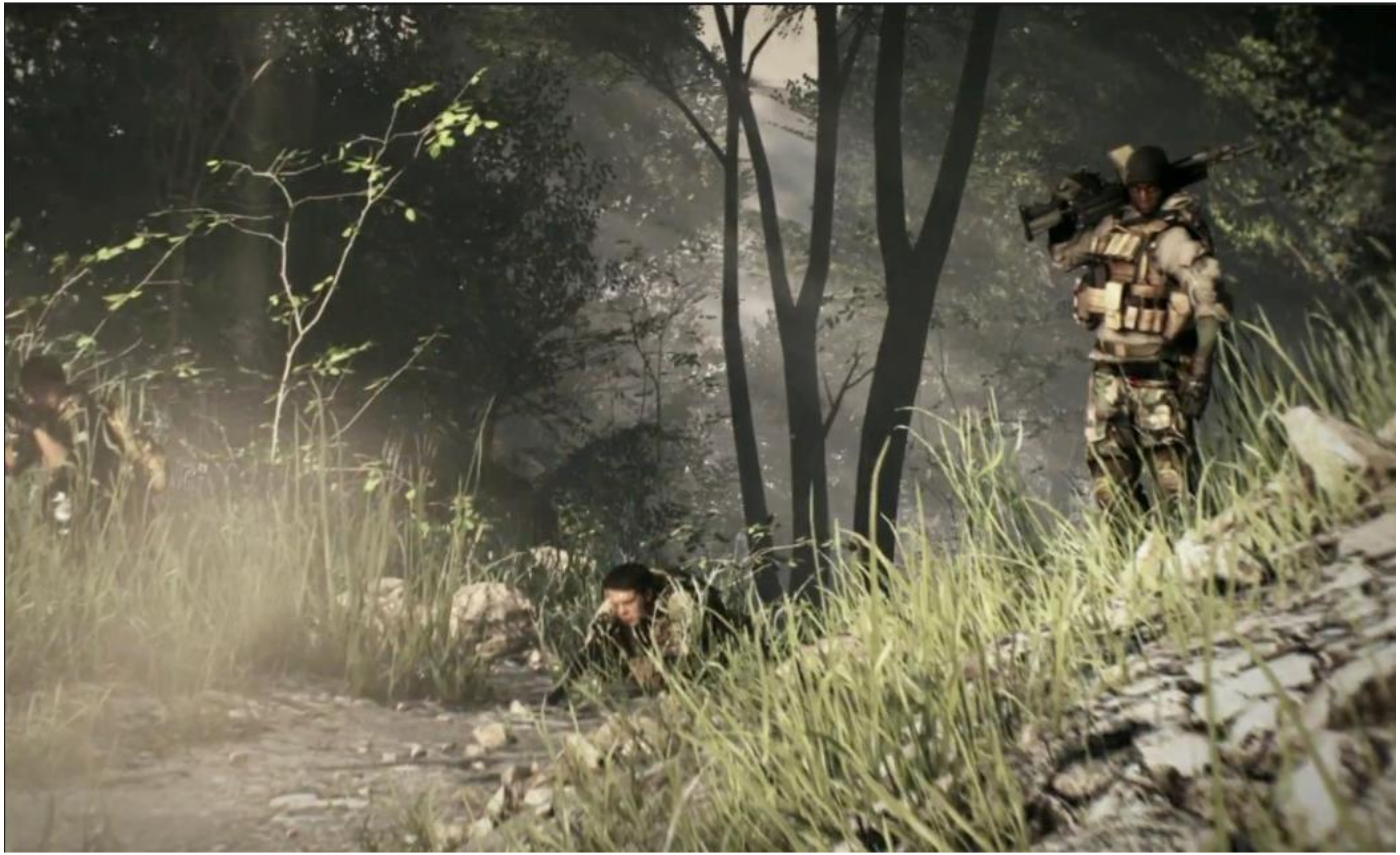
How to go from *this*…



DH2320 Lab 1 - An Introduction to WebGL and Three.js

Today’s Question

…To *this*?



Battlefield 4, DICE

Game Complexity

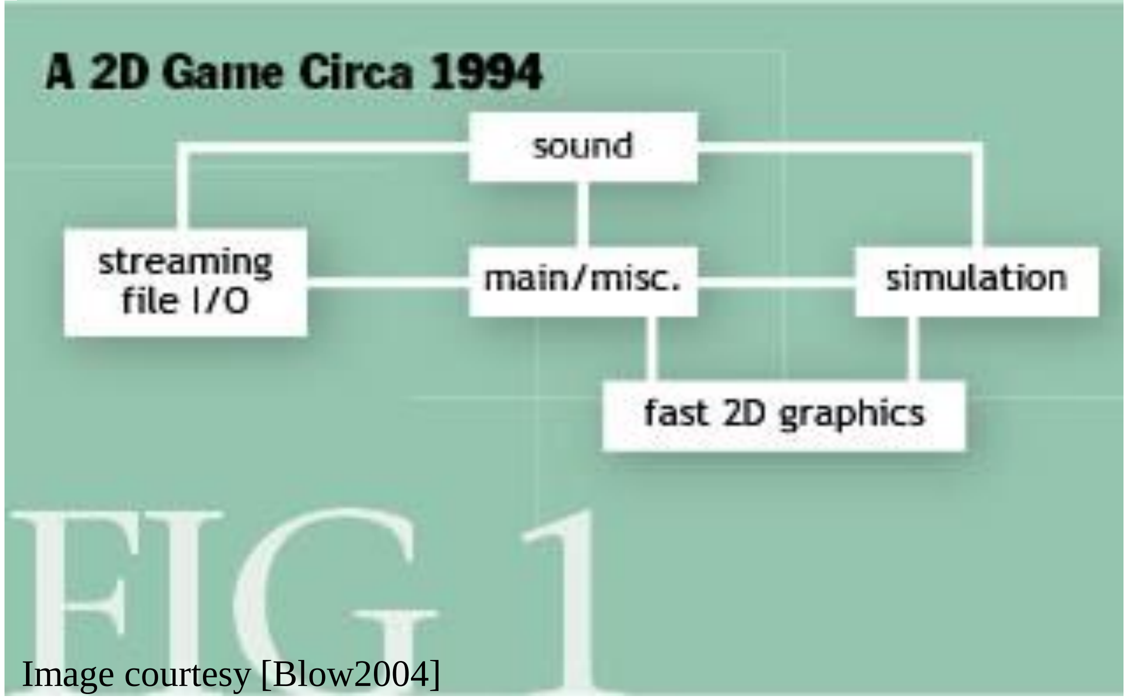


Image courtesy [Blow2004]

Blow, J. (2004).Game Development: Harder than you think, ACM Queue 1(10)

Game Complexity

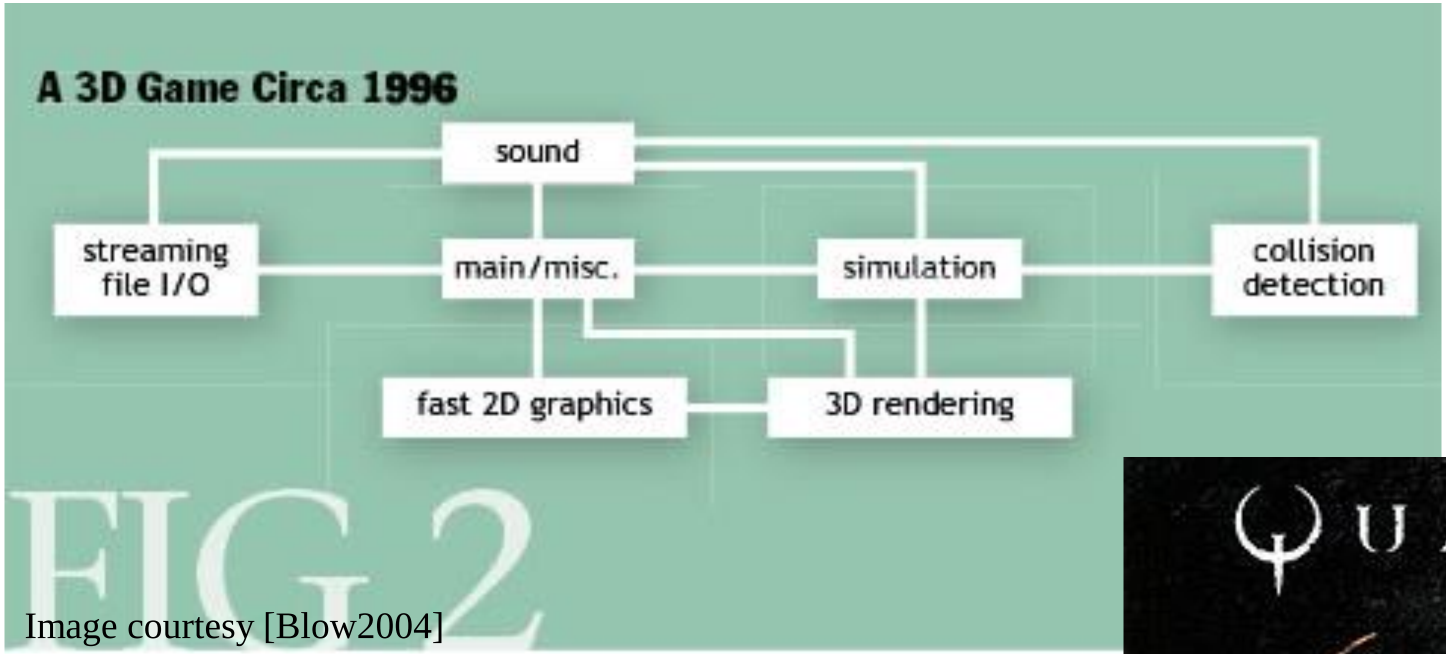
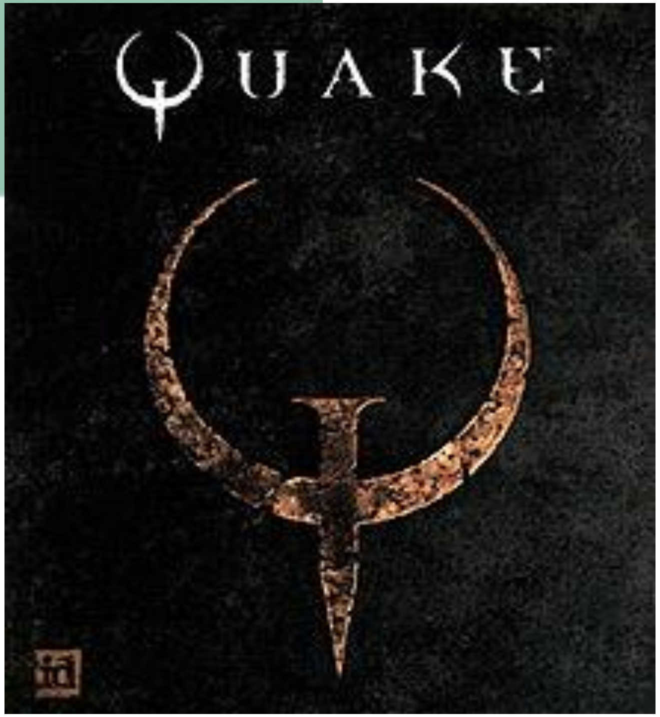


Image courtesy [Blow2004]



Game Complexity

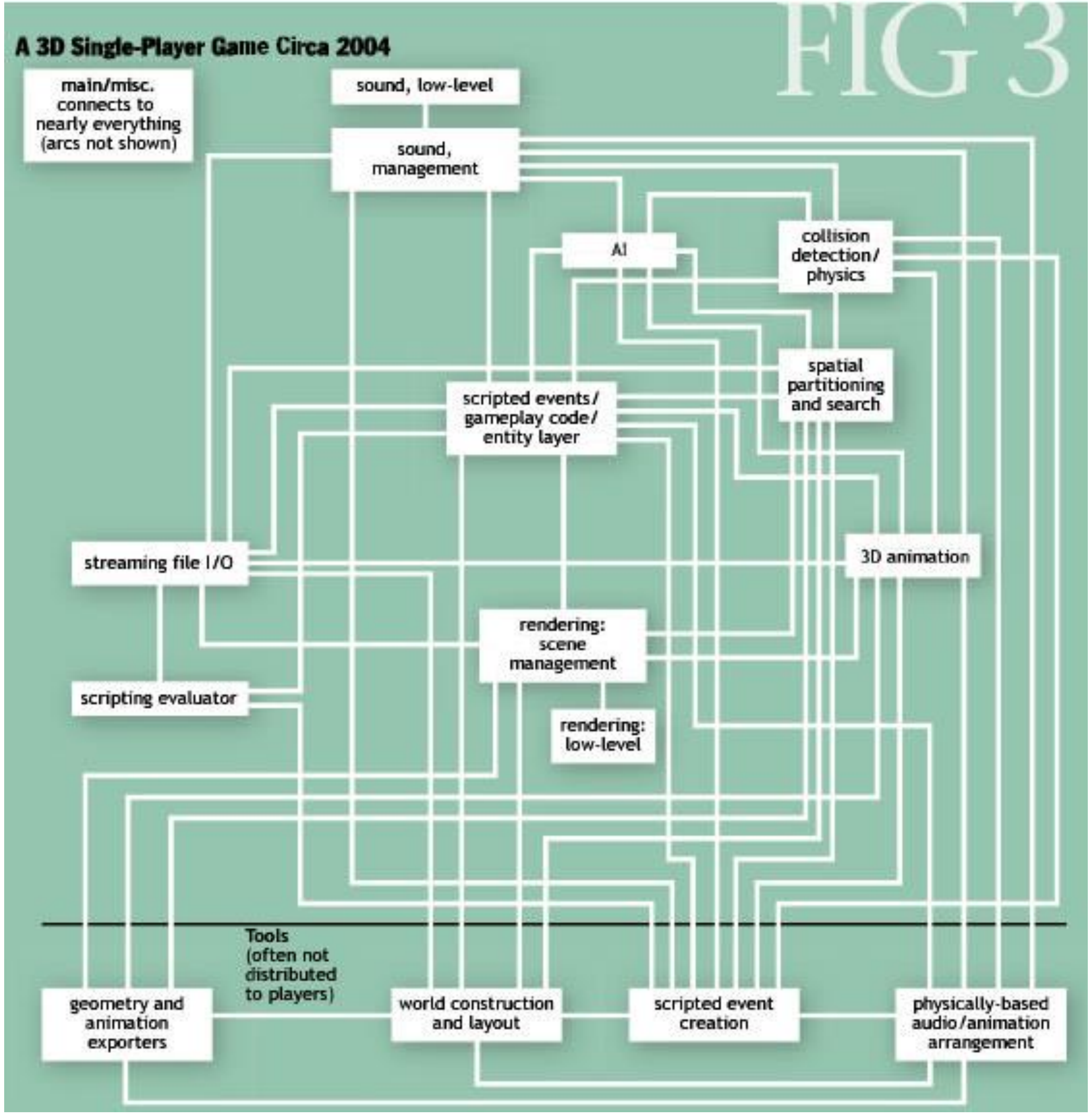


FIG 3
A 3D Single-Player Game Circa 2004
main/misc. connects to nearly everything (arcs not shown)
sound, low-level
sound, management
AI
collision detection/ physics
spatial partitioning and search
scripted events/ gameplay code/ entity layer
streaming file I/O
3D animation
rendering: scene management
scripting evaluator
rendering: low-level
Tools (often not distributed to players)
geometry and animation exporters
world construction and layout
scripted event creation
physically-based audio/animation arrangement

Image courtesy [Blow2004]

Game Complexity: 2004++

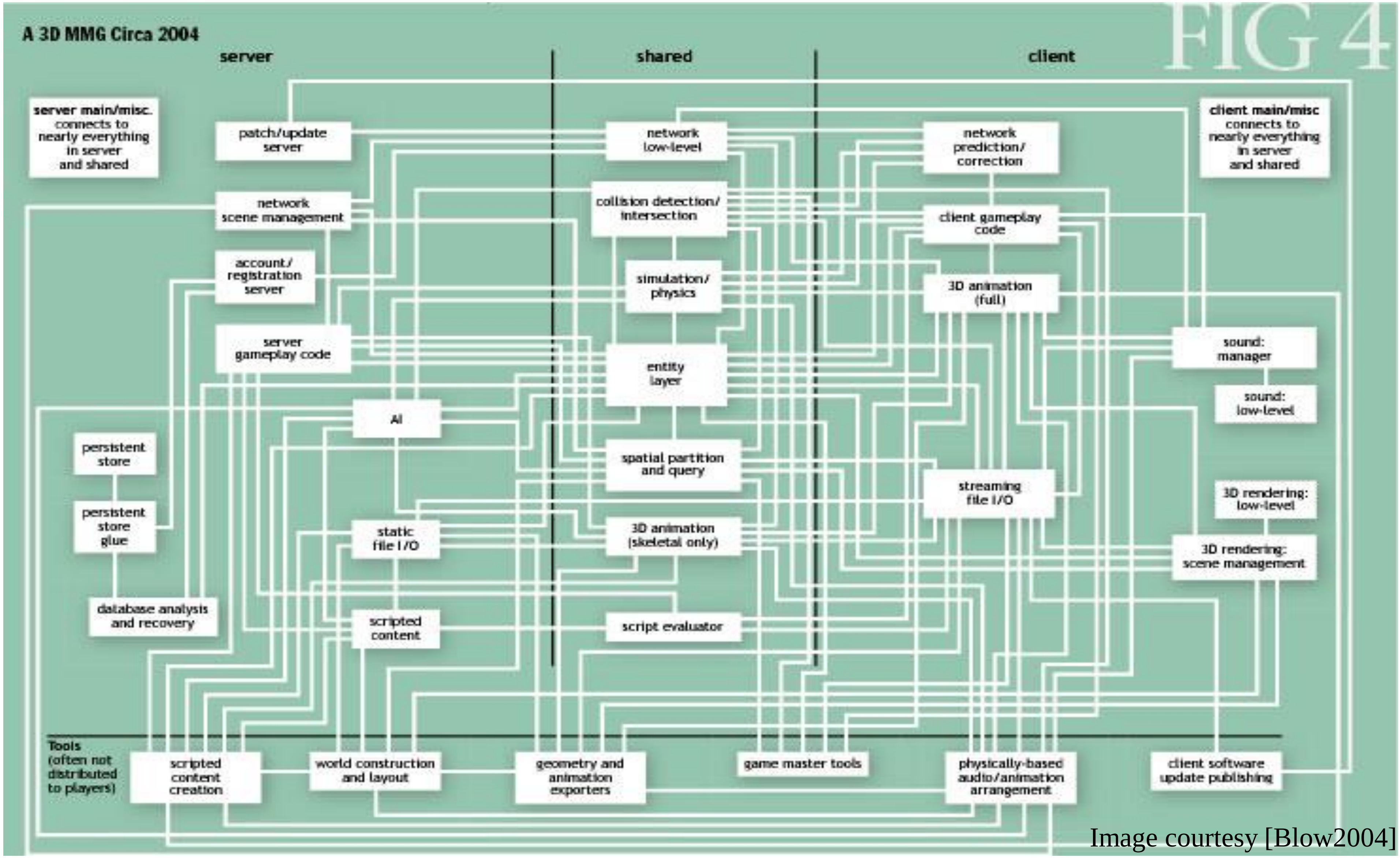
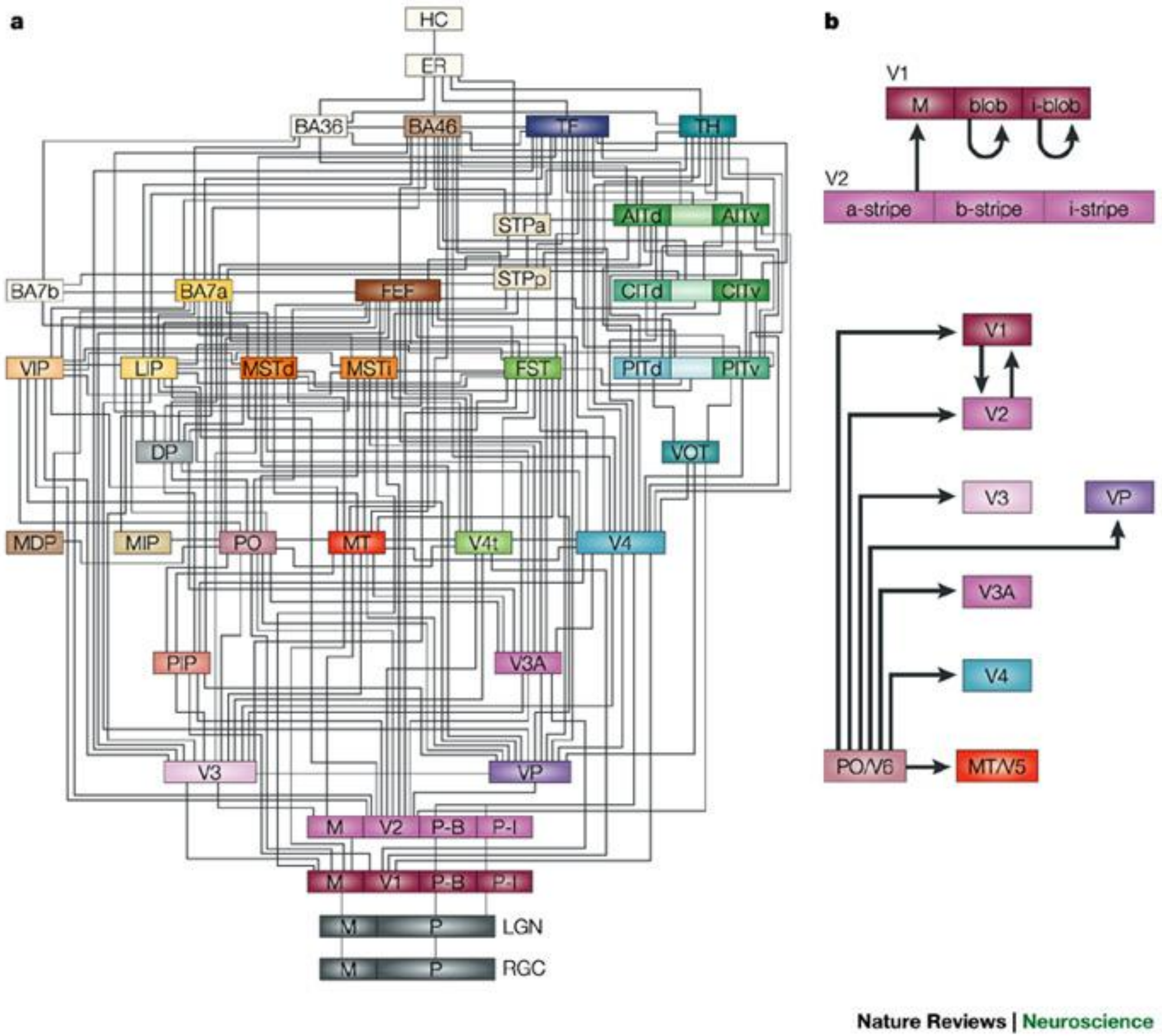

A 3D MMG Circa 2004
FIG 4
server
shared
client
server main/misc. connects to nearly everything in server and shared
patch/update server
network scene management
account/ registration server
server gameplay code
AI
persistent store
persistent store glue
static file I/O
database analysis and recovery
scripted content
network low-level
collision detection/ intersection
simulation/ physics
entity layer
spatial partition and query
3D animation (skeletal only)
script evaluator
network prediction/ correction
client gameplay code
3D animation (full)
streaming file I/O
client main/misc connects to nearly everything in server and shared
sound: manager
sound: low-level
3D rendering: low-level
3D rendering: scene management
Tools (often not distributed to players)
scripted content creation
world construction and layout
geometry and animation exporters
game master tools
physically-based audio/animation arrangement
client software update publishing

Image courtesy [Blow2004]

Game Complexity >2014?



Rees, Kreiman and Koch, Neural correlates of consciousness in humans,
Nature Reviews Neuroscience 3, 261-270, 2002

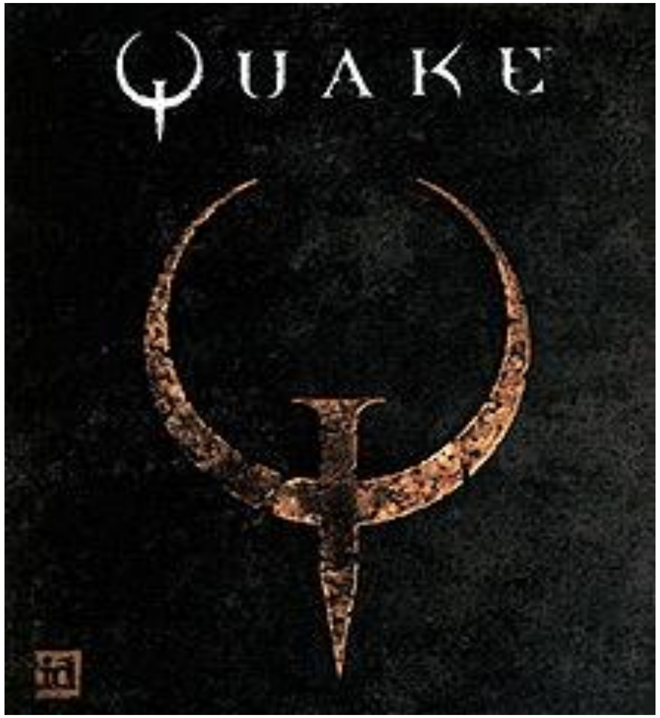
Q.

Do you have to *reinvent the wheel*?

A.

No: Reuse game components

Game Components



Game Components

Game Components

Core Code – Engines and middleware

(from programmers)

+

Tools

(from programmers)

+

Content

(from artists, designers, sound engineers...)

Game Components

Core Code – Engines and middleware
(from programmers)

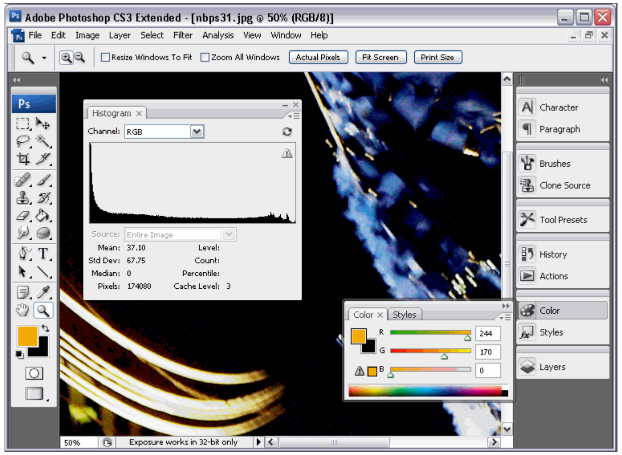
+

Tools
(from programmers)

+

Content
(from artists, designers, sound engineers...)

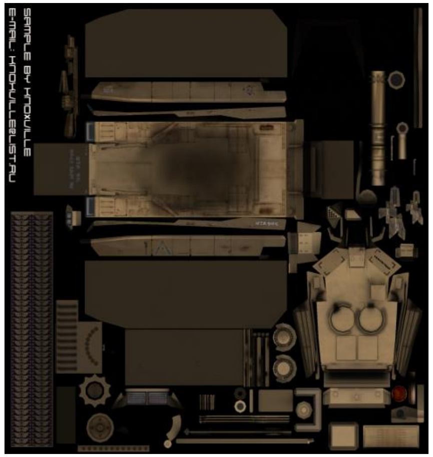
A Typical Chain



2D paint package

3D modelling package

Real-time engine

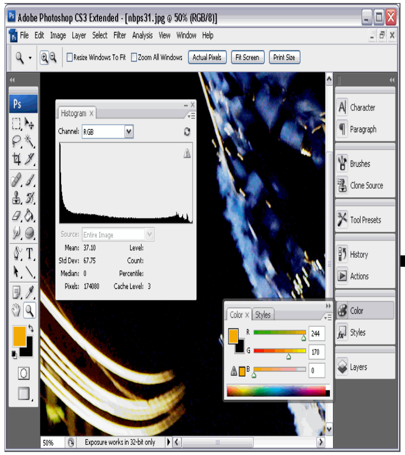


2D textures

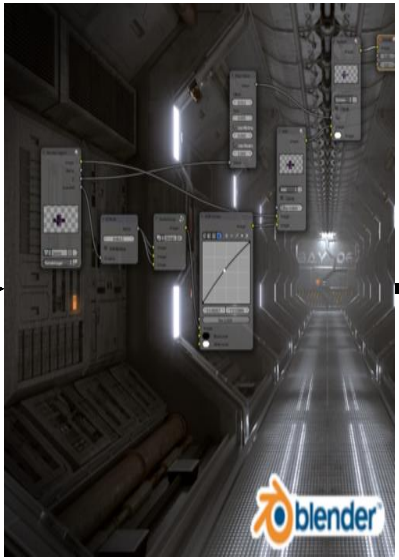
3D models and animations

Real-time rendering, animation and interaction

A Typical Chain

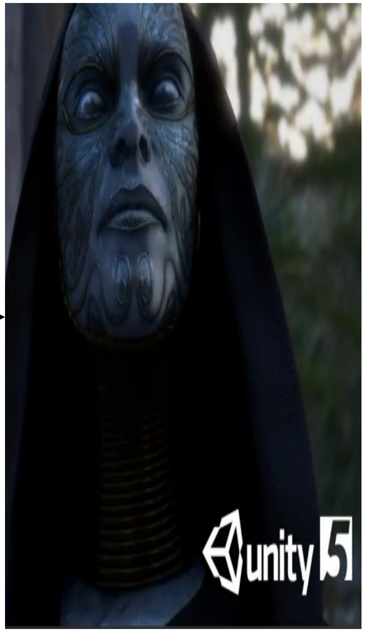

Adobe Photoshop CS3 Extended - [nbps31.jpg @ 50% (RGB/8)]
File Edit Image Layer Select Filter Analysis View Window Help
Histogram
Character
Paragraph
Brushes
Clone Source
Tool Presets
History
Actions
Color
Styles
Layers

2D paint package

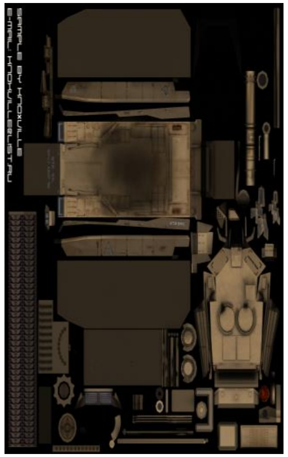


3D modelling package

Middleware



Real-time engine



2D textures

3D models and animations

Plugins

APIs

SDKs

Engines

Real-time rendering, animation and interaction

No Reinventing the Wheel

- API (Application Programmers Interface)

Software that interfaces with operating systems, libraries and services

- SDK (Software Development Kit)

Collection of libraries, API’s and tools made available for programming

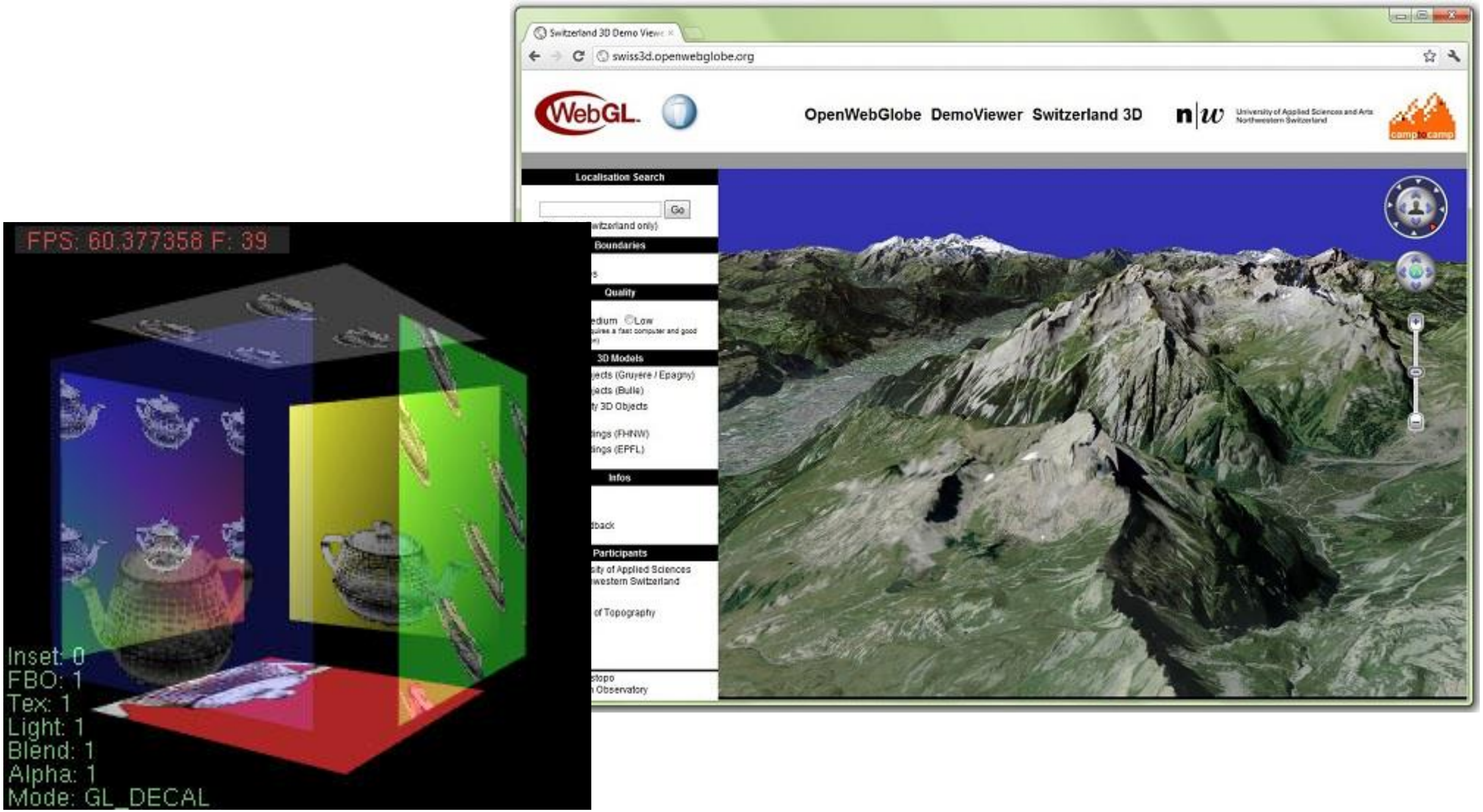
- Graphics/rendering engine

Takes care of rendering activities – visible surface determination, shading etc.

- Game engine

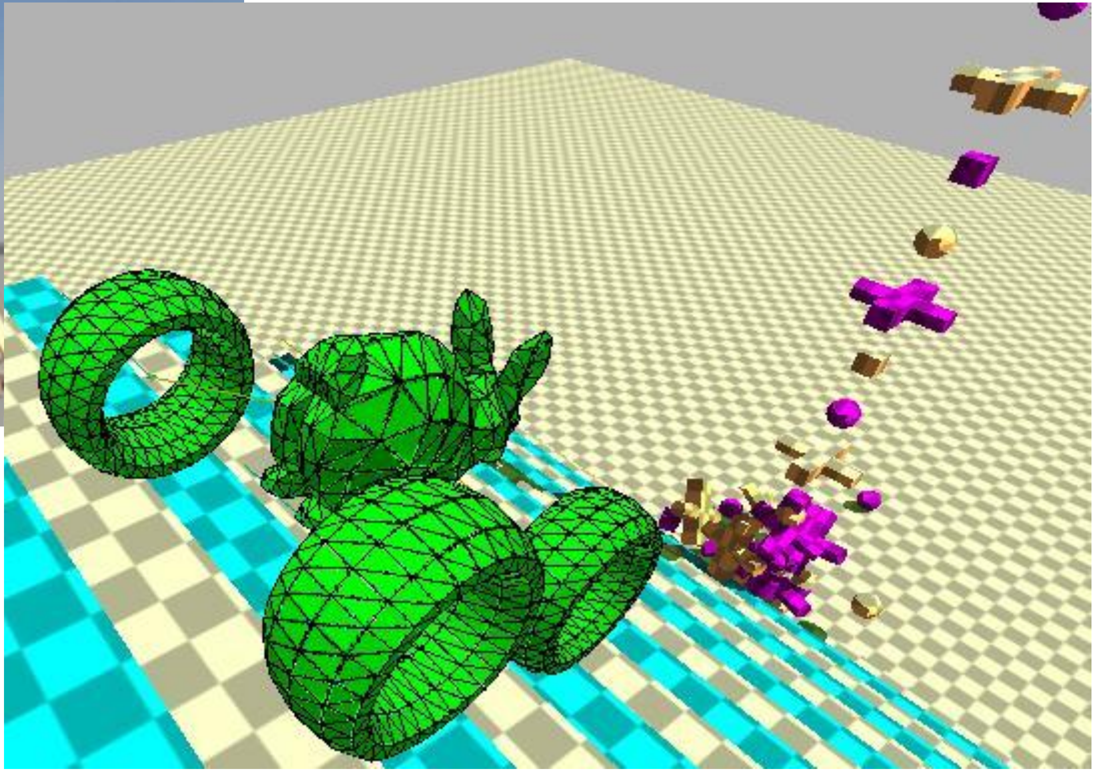
Integrated game development kit encompassing graphics and many other game-related aspects …

Example: OpenGL



Example: Bullet Physics SDK

http://www.youtube.com/watch?v=J9HaT23b-xc



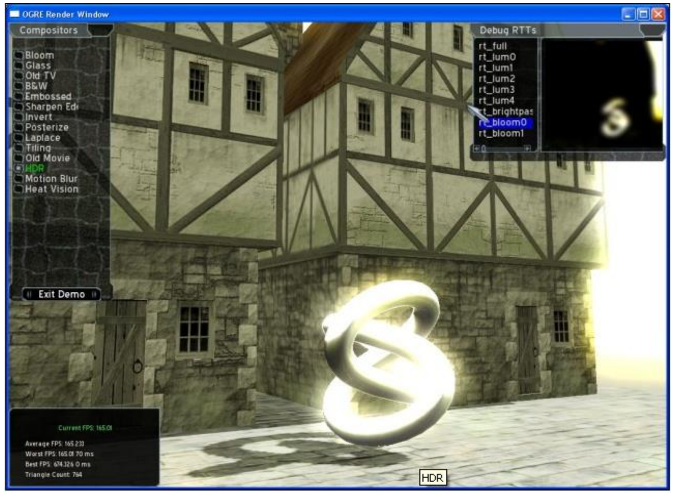
Example: OGRE

Object-oriented Graphics Rendering Engine

OGRE is primarily a **graphics engine** http://www.ogre3d.org/

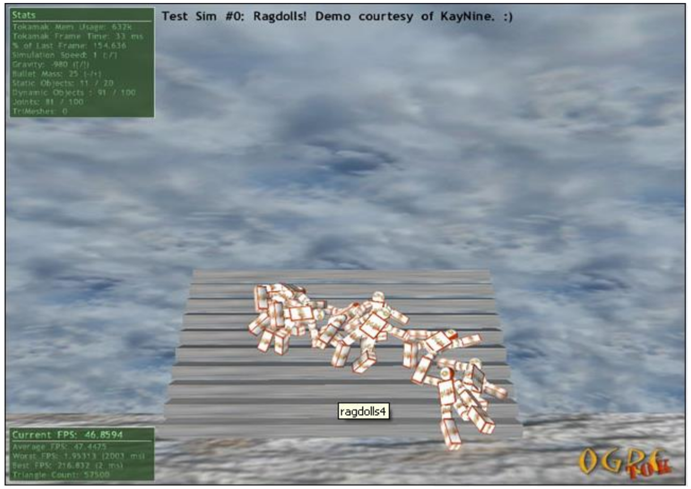
Does not concentrate on sound, AI, networking, collision, physics (but often available as add-ons...)

Ambient occlusion, parallax mapping, soft shadows, etc



```
Current FPS: 165.01
Average FPS: 165.233
Worst FPS: 165.01 70 ms
Best FPS: 674.326 0 ms
Triangle Count: 764
```

Ambient occlusion, soft shadows and parallax mapping

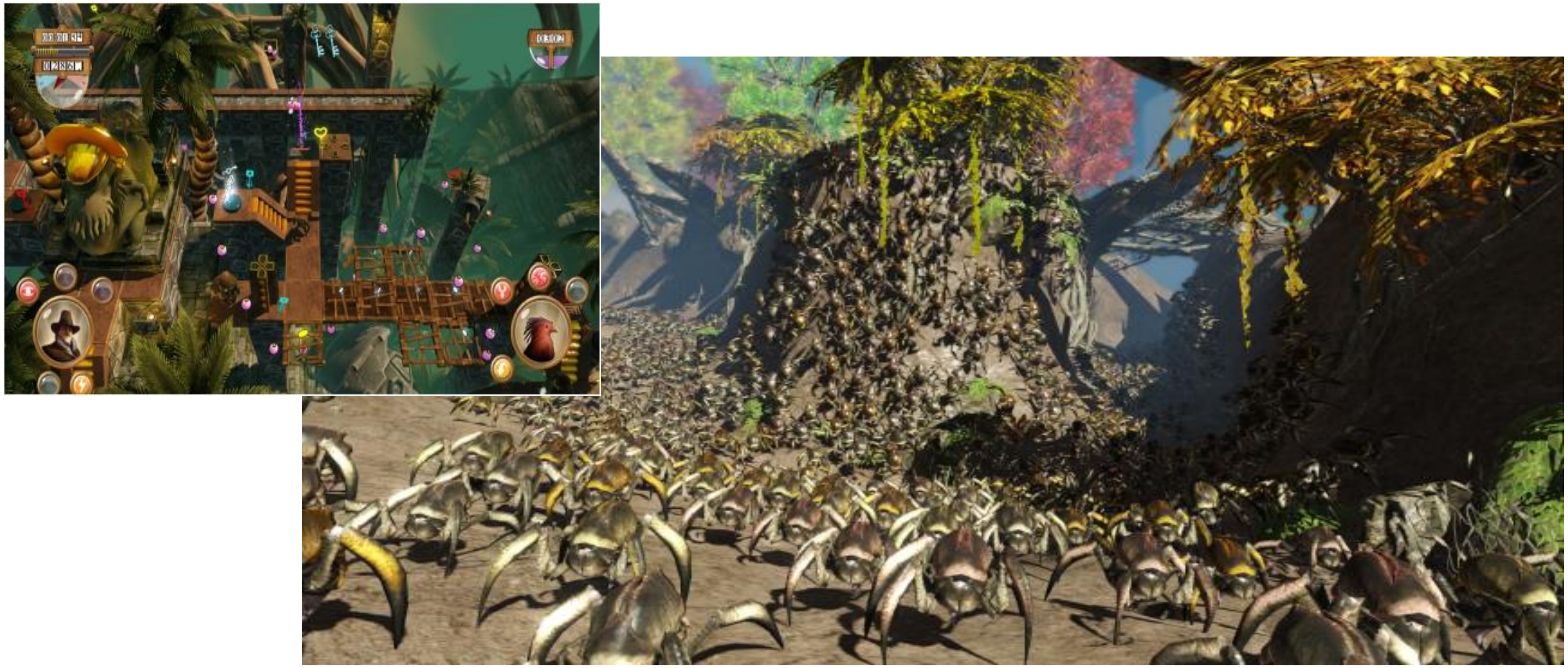


Test Sim #0: Ragdolls! Demo courtesy of KayNine. :)

```
Stats
Tokamak Mem Usage: 632k
Tokamak Frame Time: 33 ms
% of Last Frame: 154.636
Simulation Speed: 1 (;/)
Gravity: -980 ([/])
Bullet Mass: 25 (-/+)
Static Objects: 11 / 20
Dynamic Objects : 91 / 100
Joints: 81 / 100
TriMeshes: 0
```

```
Current FPS: 46.8594
Average FPS: 47.4475
Worst FPS: 1.95313 (2003 ms)
Best FPS: 216.832 (2 ms)
Triangle Count: 57500
```

Example: Bitsquid



Bitsquid is primarily a **games engine** http://www.bitsquid.se

Graphics *and...*

Sound, AI, networking, collision, physics, particles, animation, lighting, scripting, mobile support, etc

The Old/Hard Way (1)

Integrate your own components…

Choose a graphics library

- OpenGL

Add in peripherals

- GUI (Crazy Eddie)
- Model loading (AssImp)
- Physics (Bullet)
- Networking (Raknet)

Code your own shaders, etc

The Old/Hard Way (2)

- Integrate your own components
- Choose a graphics *engine*
  - Ogre
- Add in peripherals
  - GUI (CrazyEddie)
  - Networking (Raknet)
  - Add your own specialised algorithms

More feasible: no need to create a full graphics engine

Component **integration** is still an issue

The Modern Way

Just use a game engine already…

Modern engines deliver great off-the-shelf features

Great for rapid prototyping (GameMaker)

For **specialised features**, you will still need to make your own plugins, libraries, SDKs, etc the old/hard way

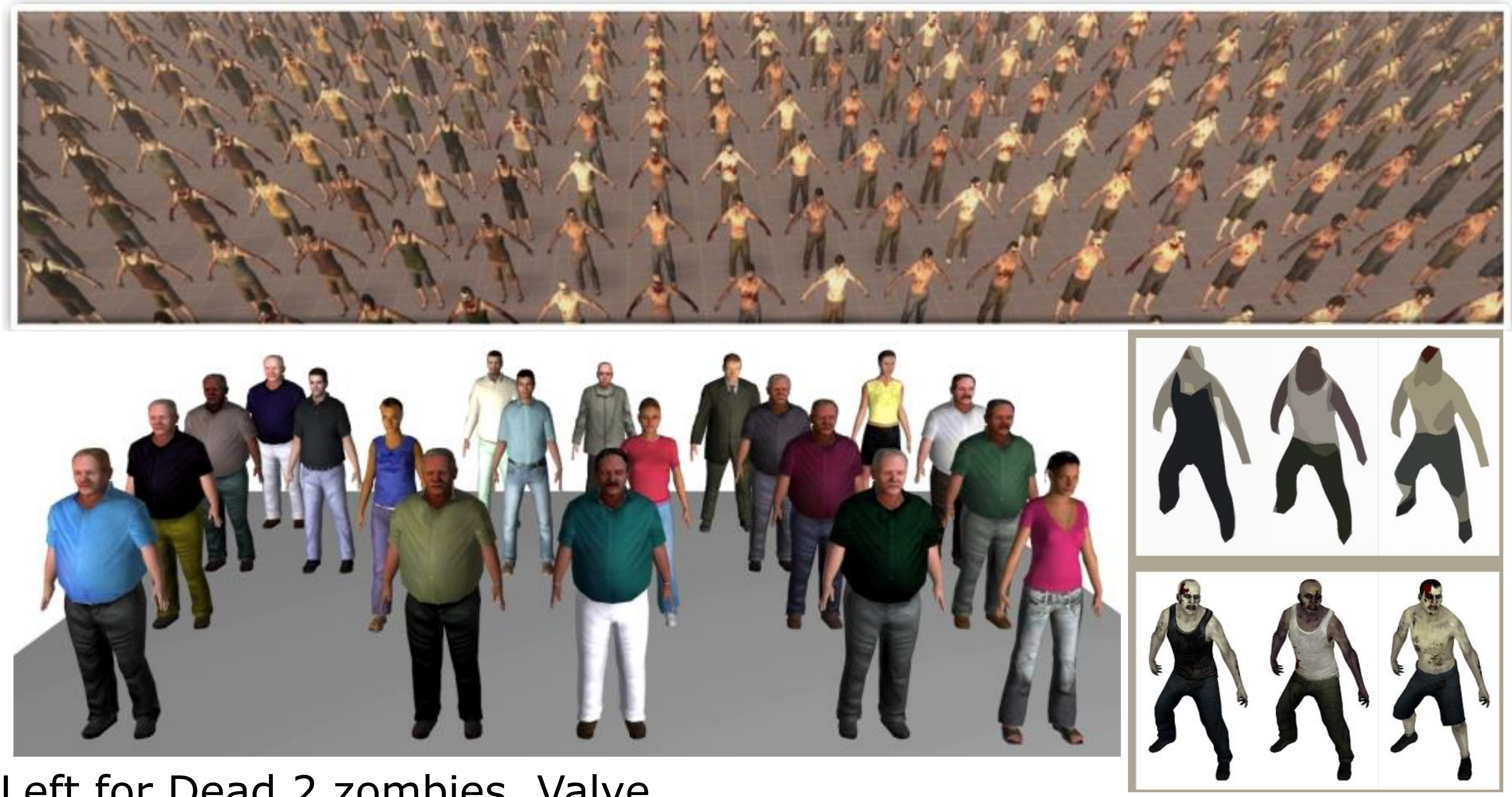
- E.g. C++ dll, DD3336, ACM Siggraph

Example: Crowd Rendering

- Specialist game technology
  - Not an ‘out of the box’ feature (yet)
- Rendering challenges
  - Real-time operation -> Imposters
  - Representation and variety of appearance -> Perception

Eye-posters:
https://kth.box.com/s/pxup6er92amskuxocu25

Crowd Variety and Perception



Left for Dead 2 zombies, Valve

Clone Attack! Perception of Crowd Variety

Rachel McDonnell, Micheal Larkin, Simon Dobbyn, Steven Collins, Carol O'Sullivan,
ACM Transactions on Graphics (SIGGRAPH 2008), 27, (3), 26:1 - 26:8, 2008

Middleware Considerations

Check licenses and costs (first!)

Developer support

- There’s a reason for it…
- Forum activity

Integration issues with your engine

- Source code access sometimes critical

Tool and engine support

- Blender/Maya/Max integration

Middleware Landscape

Ranked:

Physics (big!)

AI Navigation

Virtual characters

Weather rendering

Real-time global illumination

Occlusion culling

Trees and foilage*

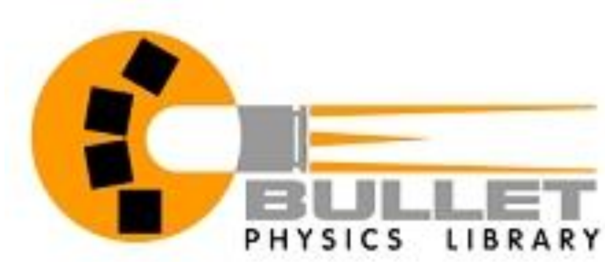
Miscellaneous (networking, UI)

Physics

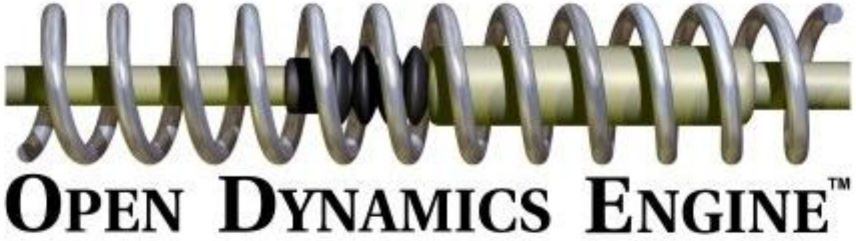
Mainly rigid-body and cloth simulation
Some fluids

Havok

Bullet



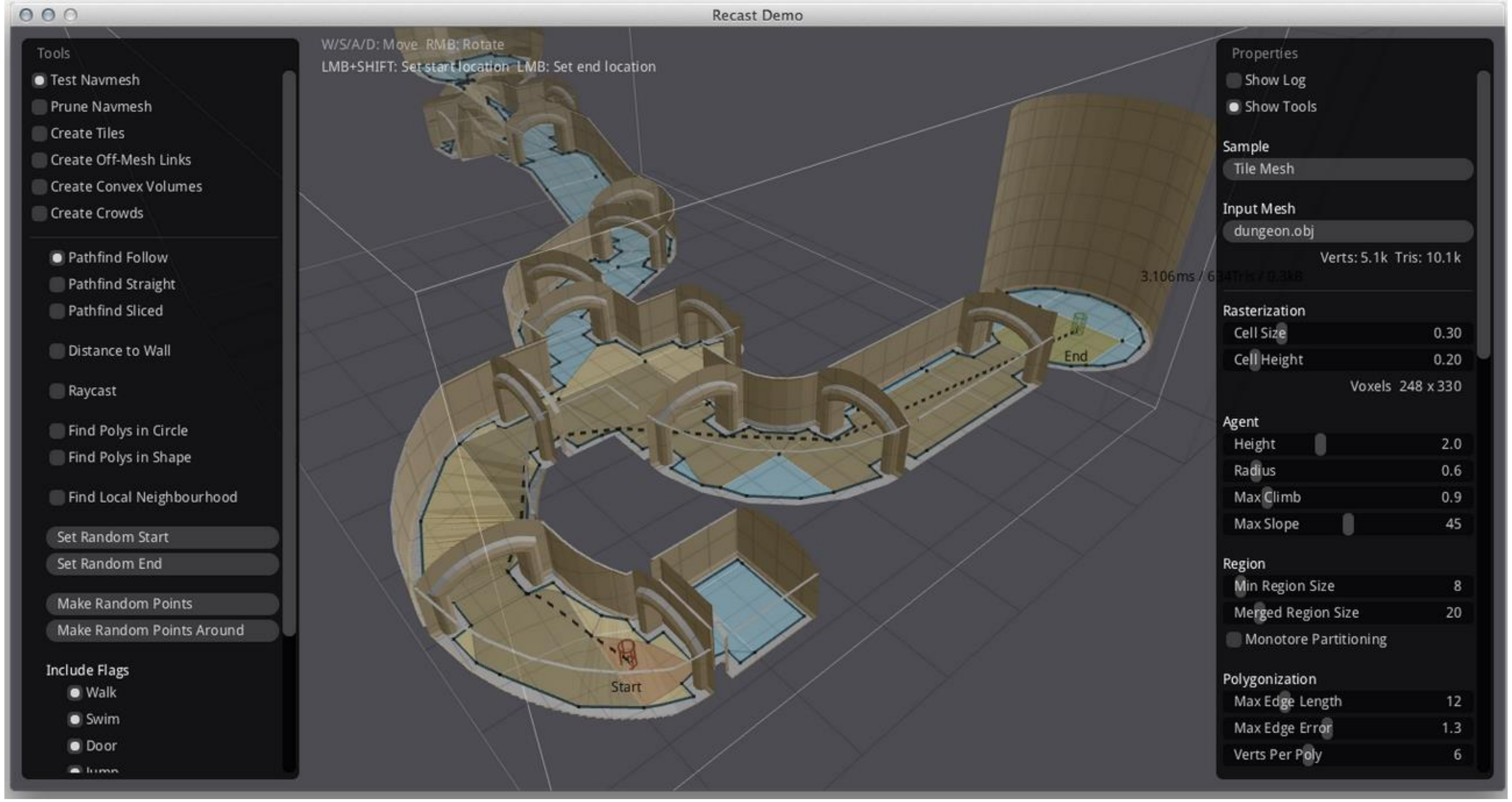
ODE



AI Navigation

Recast (nav meshes), Detour (pathfinding and spatial reasoning), MIT license

https://www.youtube.com/watch?v=XyfLSocd9ec


Recast Demo
W/S/A/D: Move RMB: Rotate
LMB+SHIFT: Set start location LMB: Set end location
End
Start

Tools

● Test Navmesh
○ Prune Navmesh
○ Create Tiles
○ Create Off-Mesh Links
○ Create Convex Volumes
○ Create Crowds

● Pathfind Follow
○ Pathfind Straight
○ Pathfind Sliced

○ Distance to Wall

○ Raycast

○ Find Polys in Circle
○ Find Polys in Shape

○ Find Local Neighbourhood

Set Random Start
Set Random End

Make Random Points
Make Random Points Around

Include Flags
● Walk
● Swim
● Door
● Jump

Properties

○ Show Log
● Show Tools

Sample
Tile Mesh

Input Mesh
dungeon.obj
Verts: 5.1k Tris: 10.1k

Rasterization
Cell Size 0.30
Cell Height 0.20
Voxels 248 x 330

Agent
Height 2.0
Radius 0.6
Max Climb 0.9
Max Slope 45

Region
Min Region Size 8
Merged Region Size 20
○ Monotore Partitioning

Polygonization
Max Edge Length 12
Max Edge Error 1.3
Verts Per Poly 6

Real-time Global Illumination

Geomerics Enlighten

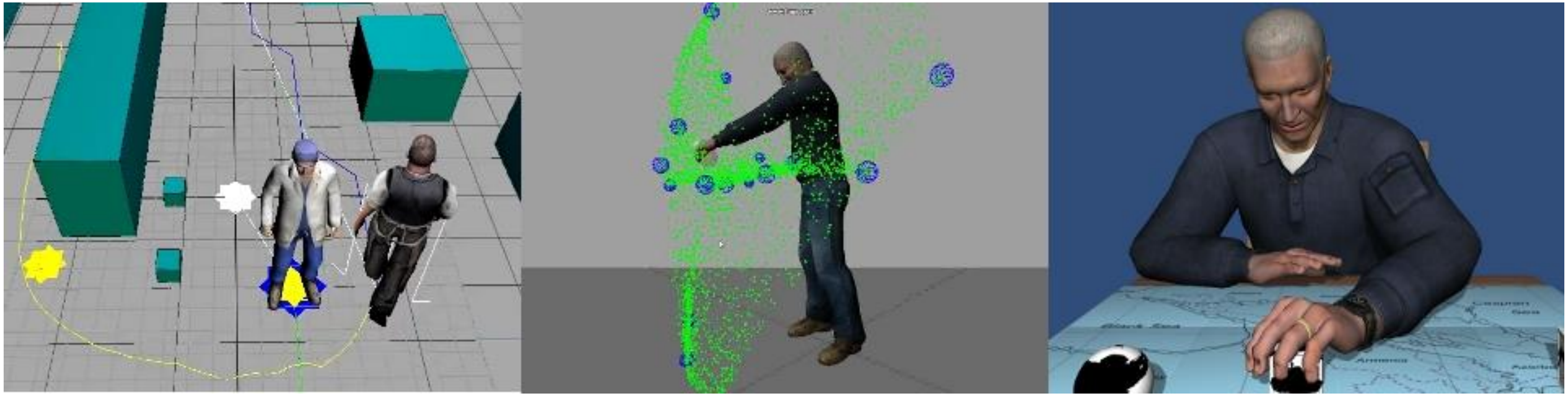
http://www.geomerics.com/enlighten/

Face Animation

http://www.facefx.com/content/english-un-declaration-human-rights

http://www.facefx.com/

Intelligent Virtual Characters



Smartbody, LGPL

http://smartbody.ict.usc.edu/video

Sky and Weather

Simul TrueSky, £150

http://simul.co/truesky/truesky-alpha-for-unity/

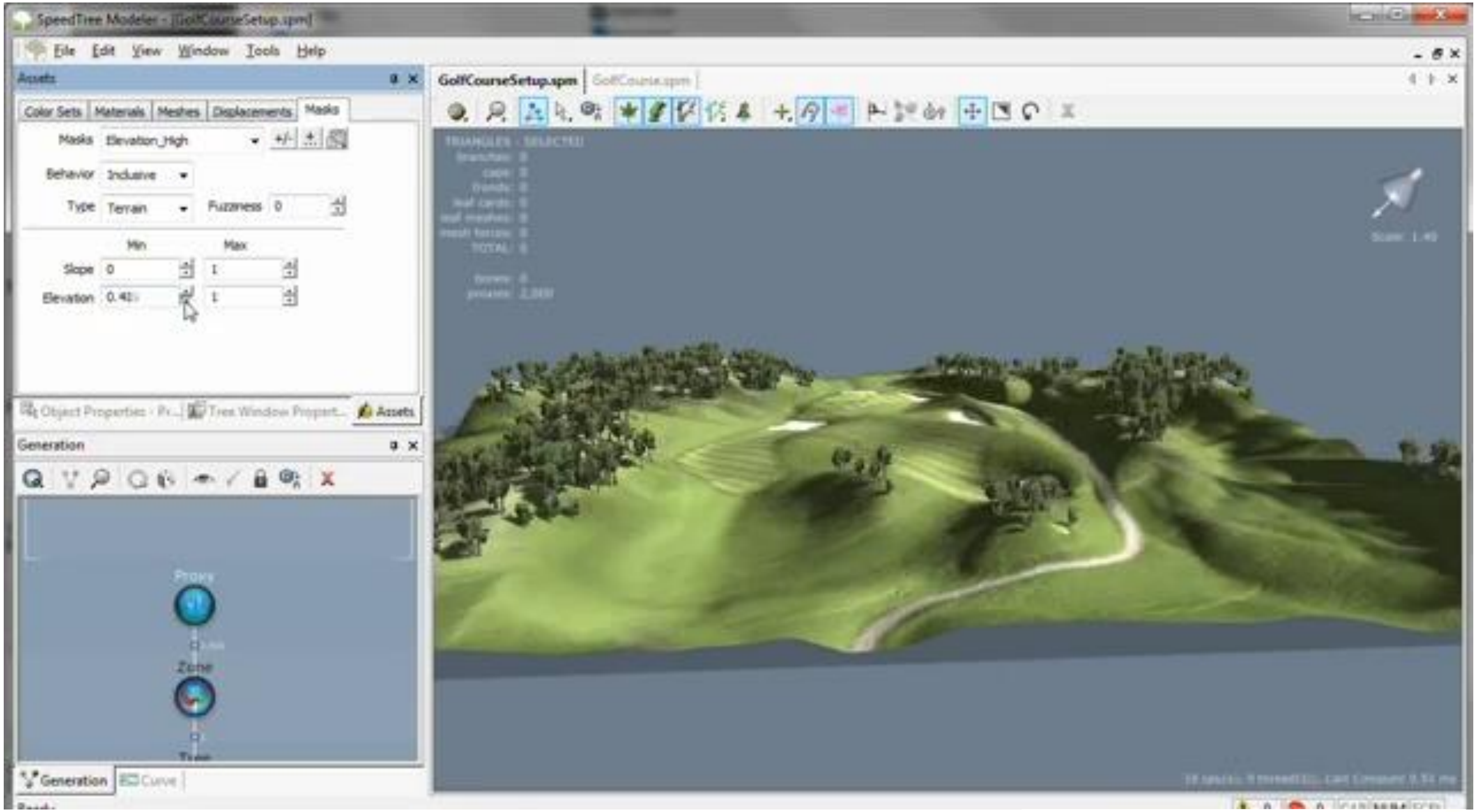
Sky and Oceans

Sundog
Software

Silverlining and Triton ($100 per seat)

http://sundog-soft.com/sds/

Trees and Foilage


SpeedTree Modeler - [GolfCourseSetup.spm]
File Edit View Window Tools Help
Assets
Color Sets Materials Meshes Displacements Masks
Min
Max
Object Properties Tree Window Propert... Assets
Generation
Proxy
Zone
Tree
Generation Curve
GolfCourseSetup.spm GolfCourse.spm
TRIANGLES - SELECTED

Masks Elevation_High

Behavior Inclusive

Type Terrain Fuzziness 0

Slope 0 1

Elevation 0.41 1

Speed Tree

http://www.speedtree.com/

http://www.youtube.com/watch?v=r18c7QlWLBQ

In General

Many other libraries, SDKs and tools out there

Have not really covered low level libraries for programming and math (vital!)

The landscape shifts quite quickly

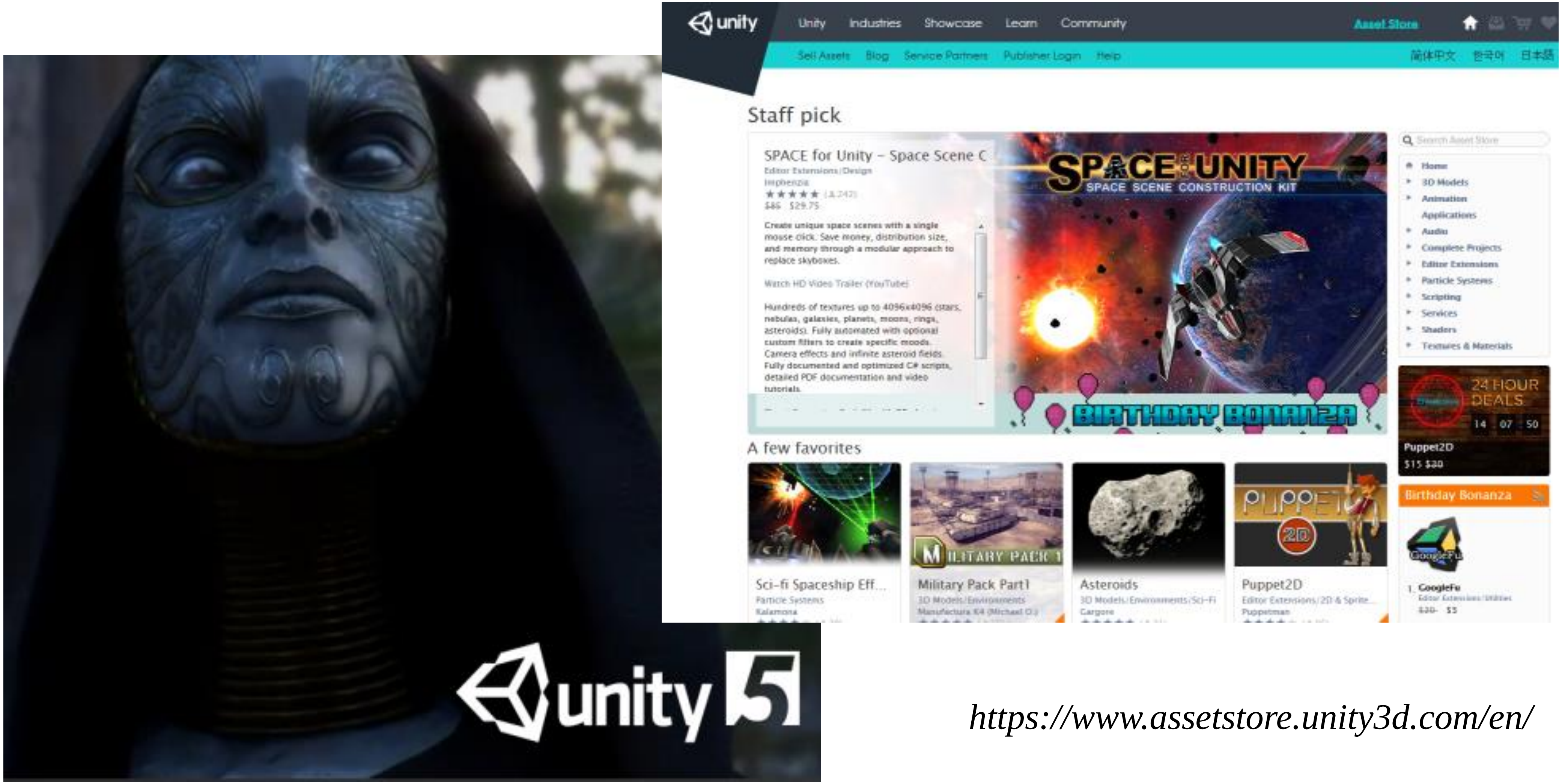
Many engine specific libraries available

For a slightly different (older) cross section of middleware, see:

Anderson and Peters, **No More Reinventing the Virtual Wheel: Middleware for Use in Computer Games and Interactive Computer Graphics Education, Eurographics**

Using Unity?

To the Asset Store Immediately!



Staff pick

SPACE for Unity – Space Scene C
Editor Extensions/Design
Imphenzia
★★★★★ (1,242)
~~$85~~ $29.75

Create unique space scenes with a single mouse click. Save money, distribution size, and memory through a modular approach to replace skyboxes.

Watch HD Video Trailer (YouTube)

Hundreds of textures up to 4096x4096 (stars, nebulas, galaxies, planets, moons, rings, asteroids). Fully automated with optional custom filters to create specific moods. Camera effects and infinite asteroid fields. Fully documented and optimized C# scripts, detailed PDF documentation and video tutorials.

A few favorites

Sci-fi Spaceship Eff...	Military Pack Part1	Asteroids	Puppet2D
Particle Systems	3D Models/Environments	3D Models/Environments/Sci-Fi	Editor Extensions/2D & Sprite...
Kalamona	Manufactura K4 (Michael O.)	Cargore	Puppetman

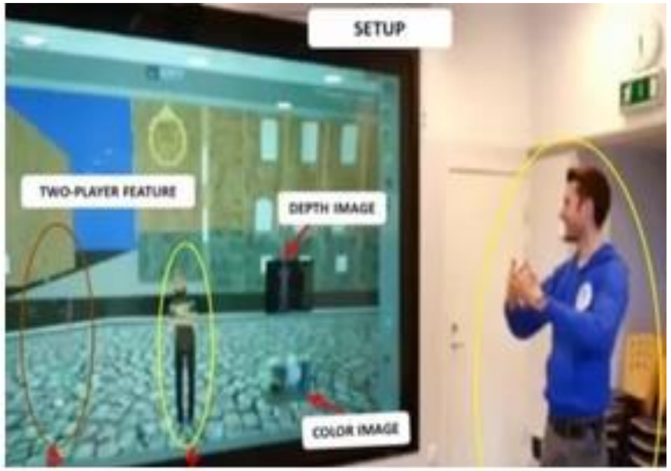
Puppet2D
$15 ~~$20~~

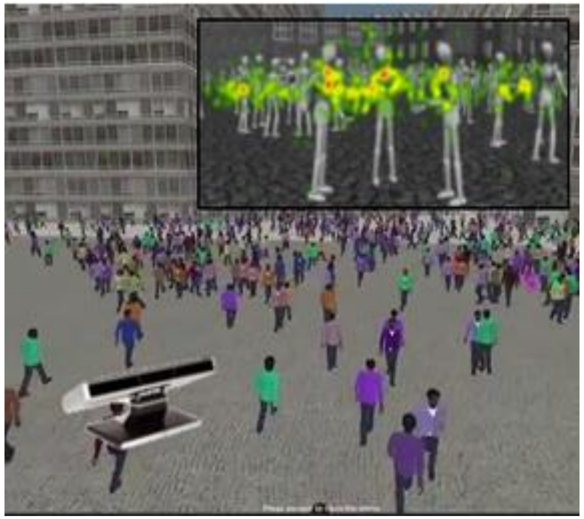
1. GoogleFu
Editor Extensions/Utilities
~~$20~~ $5

https://www.assetstore.unity3d.com/en/

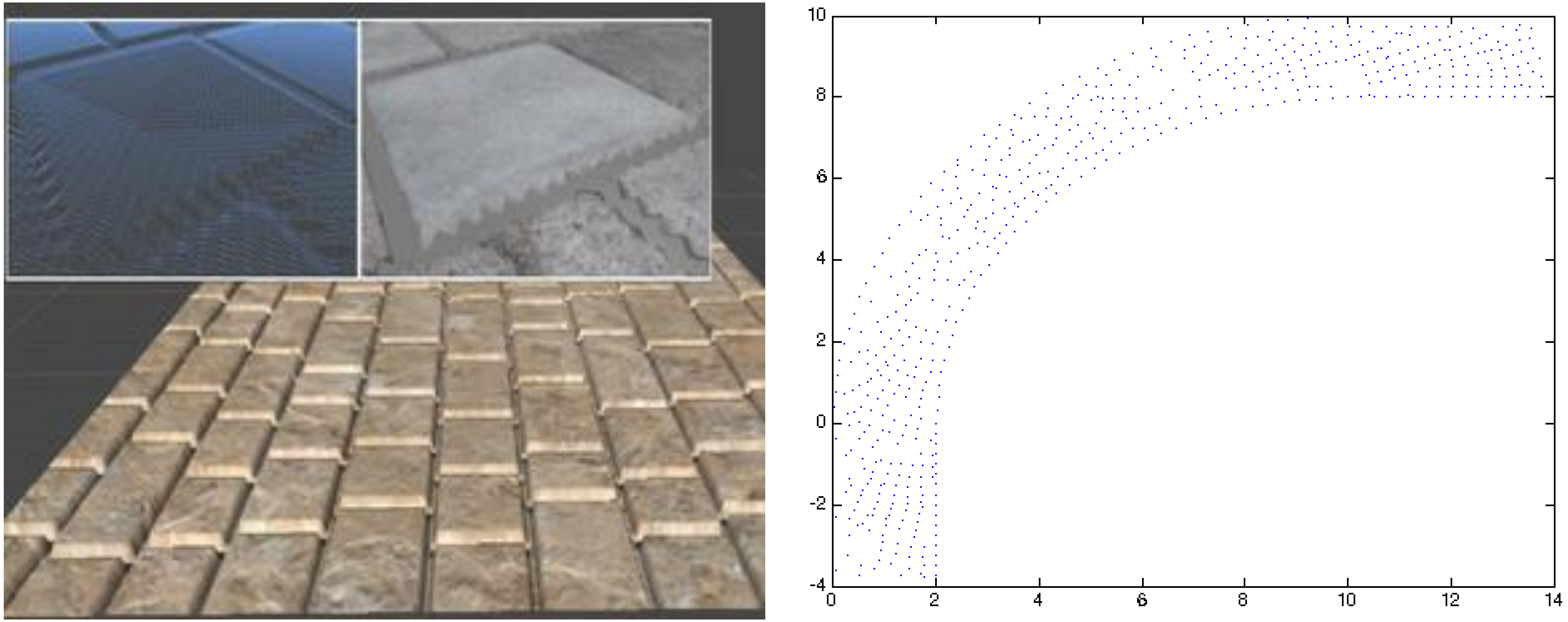
Postscript: Student Projects

http://www.csc.kth.se/~chpeters/projects.html





Example



Procedural footpath generation using a texture synthesis approach
Yang Zhou, see: http://projectfharlanghn.wordpress.com/

Remember

- **Aim to create portfolio work**

- **DD3336**, Interactive Entertainment Technologies (PhD level)
- **DH2413**, Advanced Graphics and Interaction
- **DT2350**, Human Perception for Information Technology
- **DH2323**, Computer Graphics and Interaction
- **DH2320**, Introduction to Visualization and Graphics
- **DD1354**, Models and Simulation

- Visualization (VIC) Studio