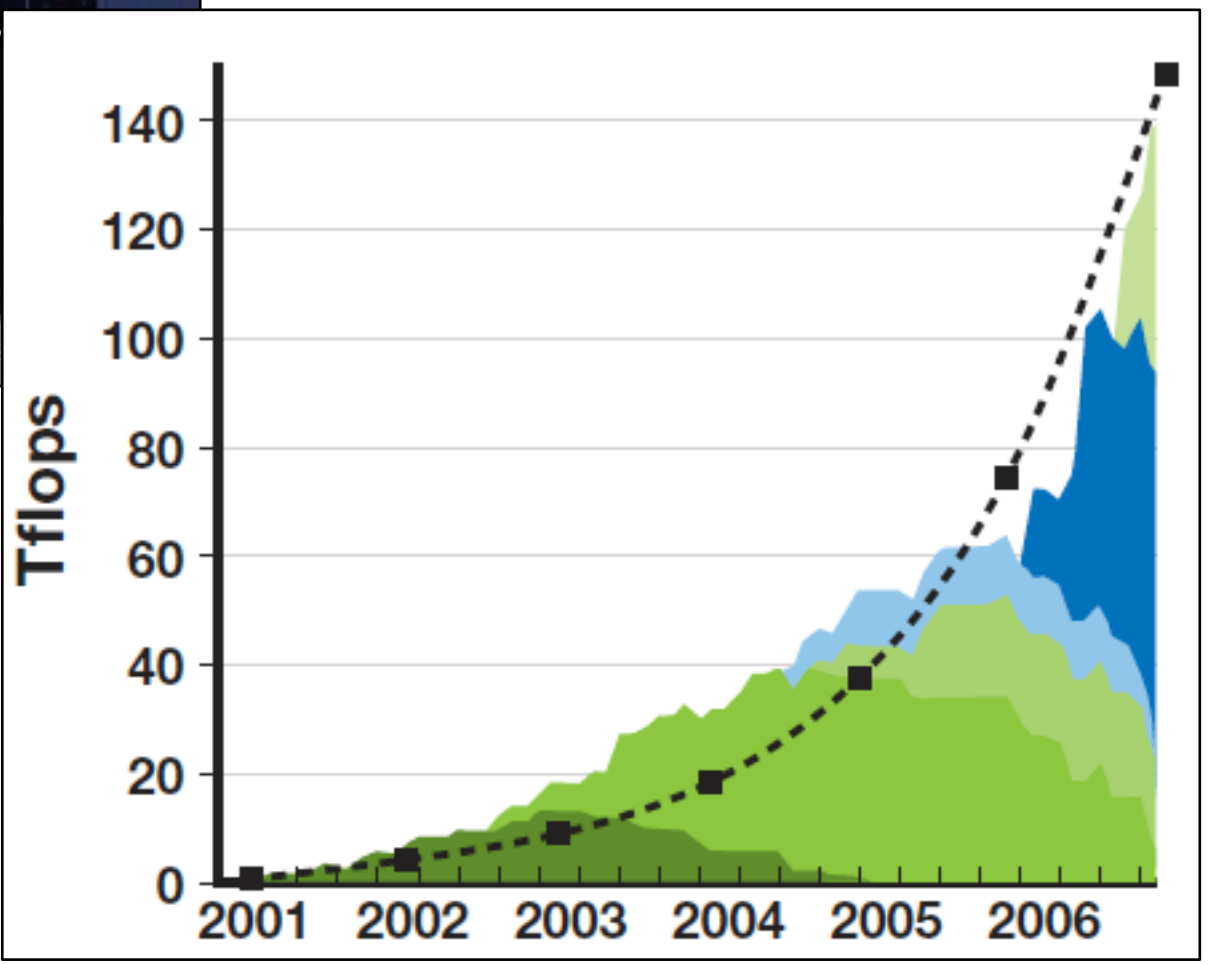
CEDEC 2008
Gems et al.

Nakamoto, Hiroshi
Borndigital. Inc

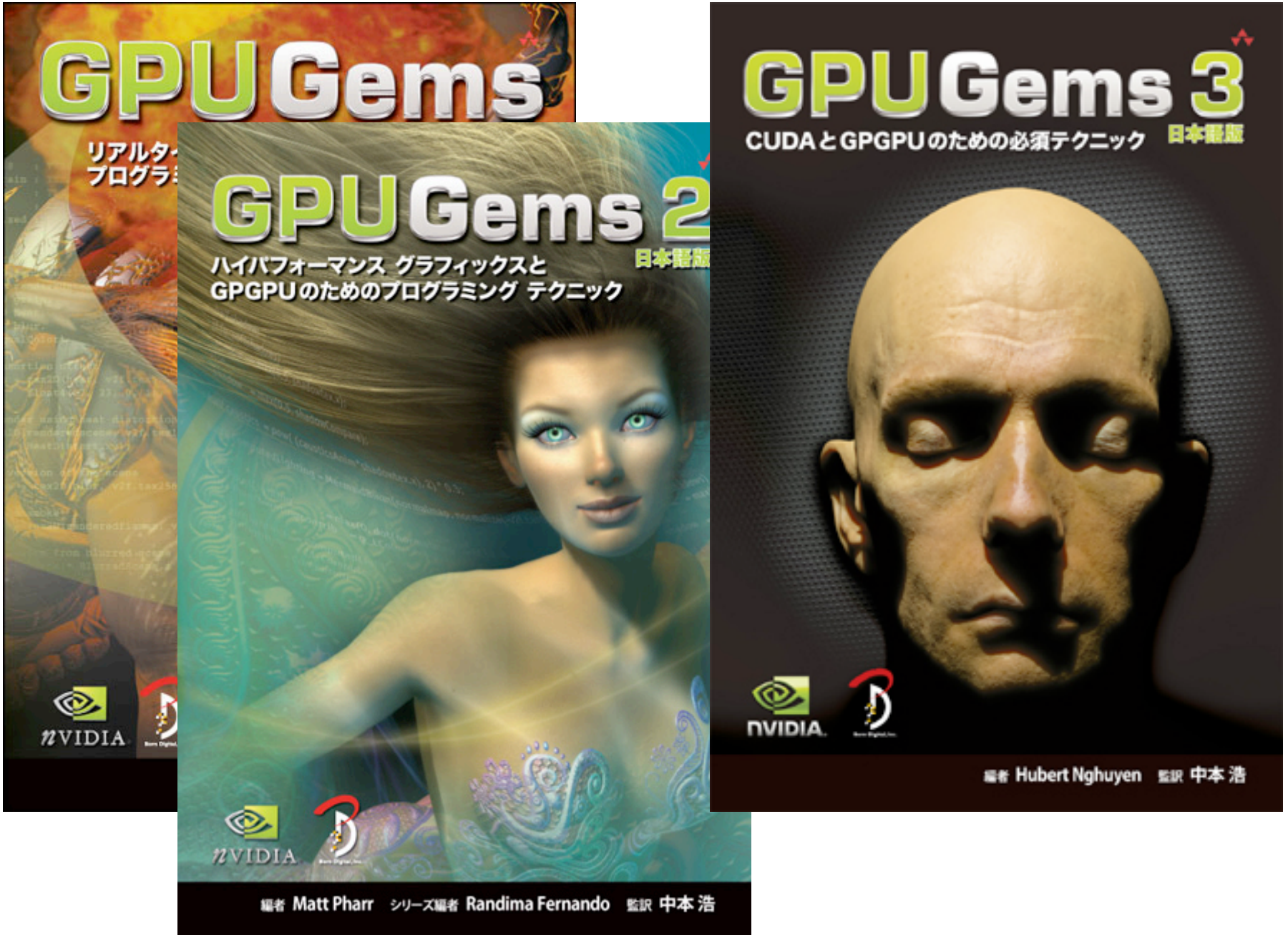
Computing World



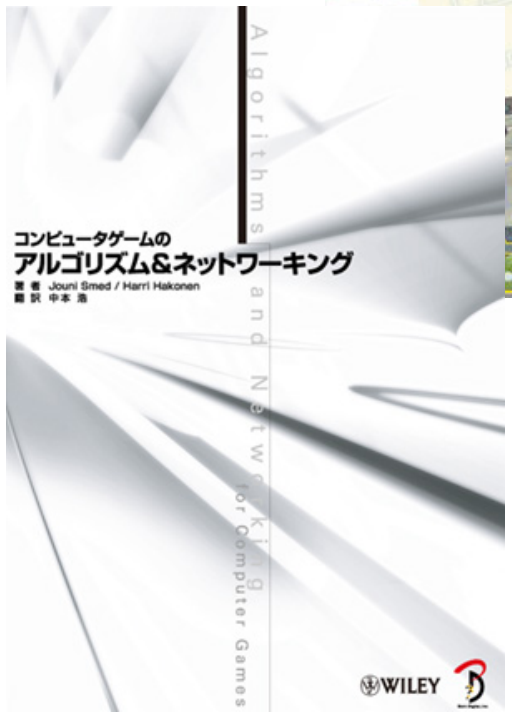
Personal Programming World

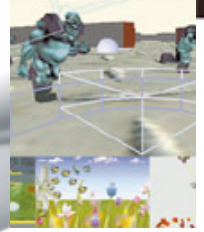
Quod scripsi, scripsi.


GAME PROGRAMMING Gems 6
THOMSON DELMAR LEARNING
編者 Michael Dickheiser
翻訳 川西裕幸 中本浩

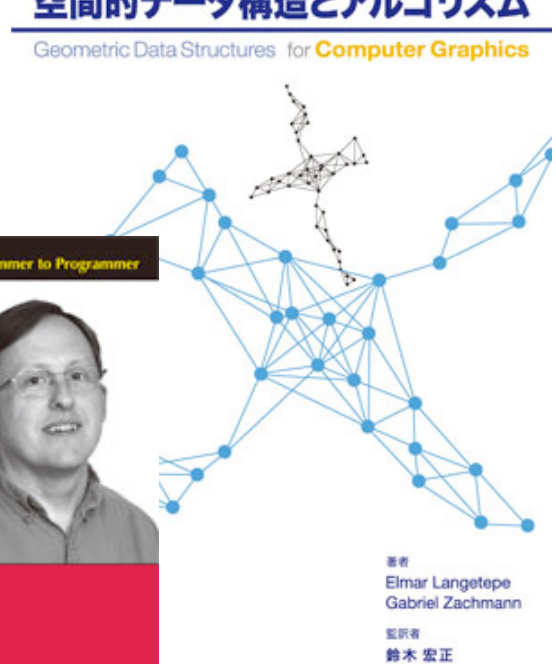

GPU Gems
GPU Gems 2
ハイパフォーマンス グラフィックスと
GPGPUのためのプログラミング テクニック
日本語版
編者 Matt Pharr シリーズ編者 Randima Fernando 監訳 中本 浩
GPU Gems 3
CUDAとGPGPUのための必須テクニック
日本語版
編者 Hubert Nghuyen 監訳 中本 浩


RUDY
RUCKER
ソフトウェア工学とコンピュータゲーム
ソフトウェア工学と
コンピュータゲーム
Software Engineering and Computer Games
• UML
• Patterns
• Windows
• OpenGL


コンピュータゲームの
アルゴリズム&ネットワーキング
著者 Jouni Smed / Harri Hakonen
翻訳 中本 浩
Algorithms and Networking for Computer Games
WILEY

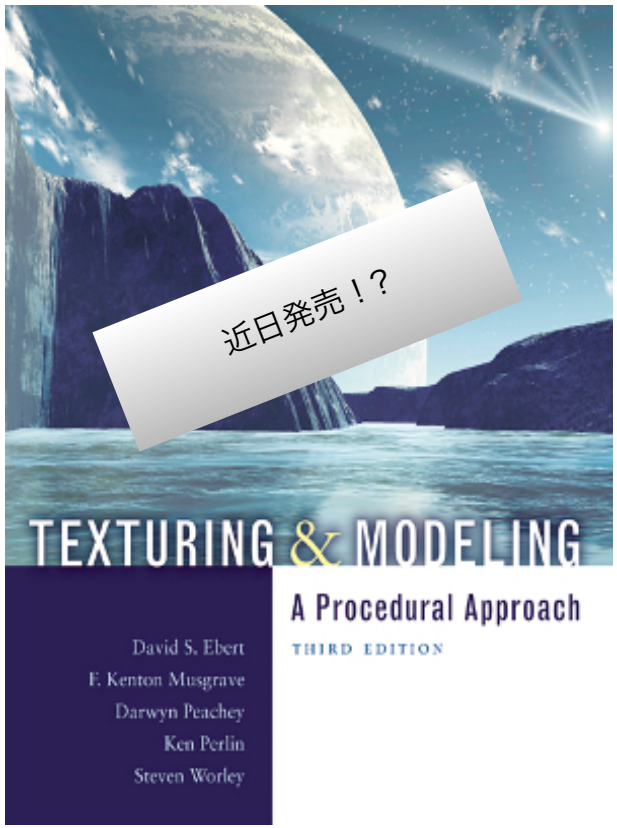


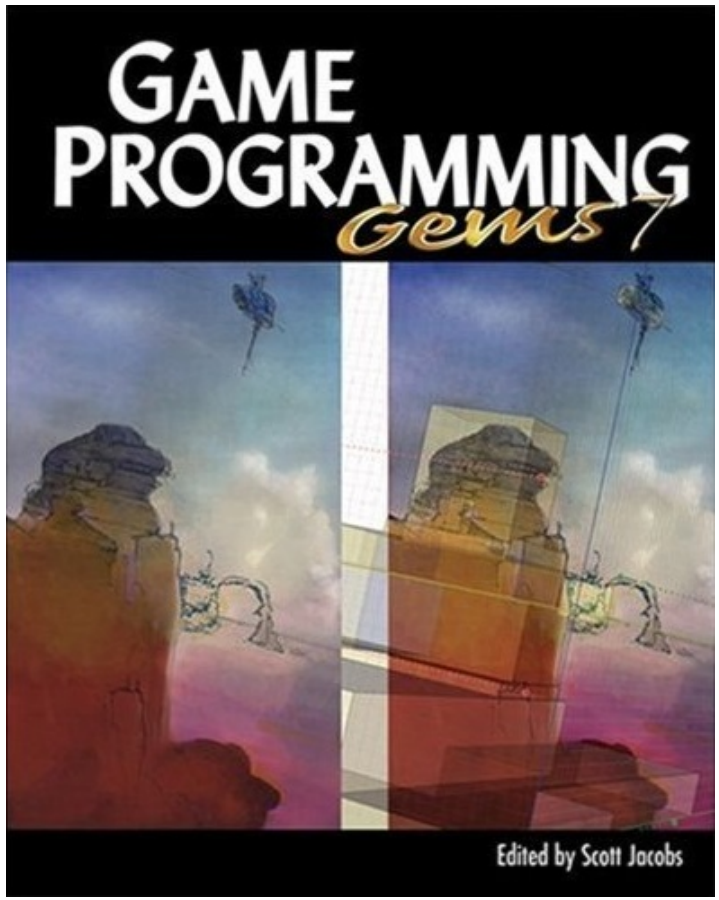

Programmer to Programmer
Professional
Python
フレームワーク
DjangoとTurboGearsで実現する
Web2.0プログラミング
Dana Moore, Raymond Budd, William Wright 著　中本 浩 監訳
wrox


空間的データ構造とアルゴリズム
Geometric Data Structures for Computer Graphics
著者
Elmar Langetepe
Gabriel Zachmann
監訳者
鈴木 宏正

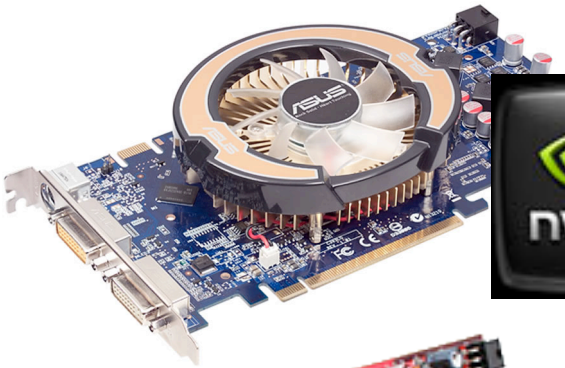

Cg
The Cg Tutorial
日本語版
プログラム可能な
リアルタイムグラフィックス完全ガイド
NVIDIA
The Cg Tutorial 日本語版
FREE
Real-Time Shader Programming 日本語版
Real-Time
Shader
Programming 日本語版
DIRECTX 9.0 対応
著者 Ron Fosner 監修 川西裕幸 翻訳 中本浩

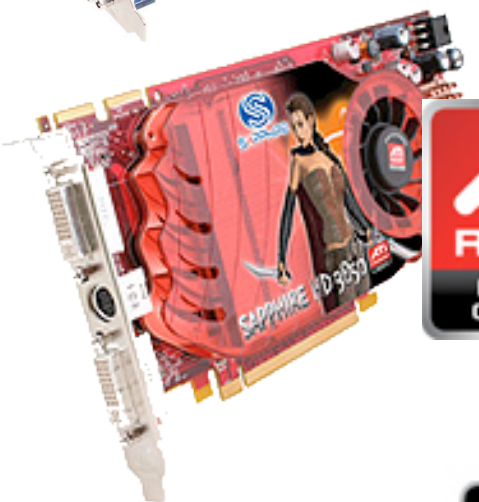

リアルタイム
レンダリング
REAL-TIME RENDERING
Second Edition
第2版
著者 Tomas Akenine-Möller/Eric Haines
翻訳 川西 裕幸
A K Peters
Natick, Massachusetts

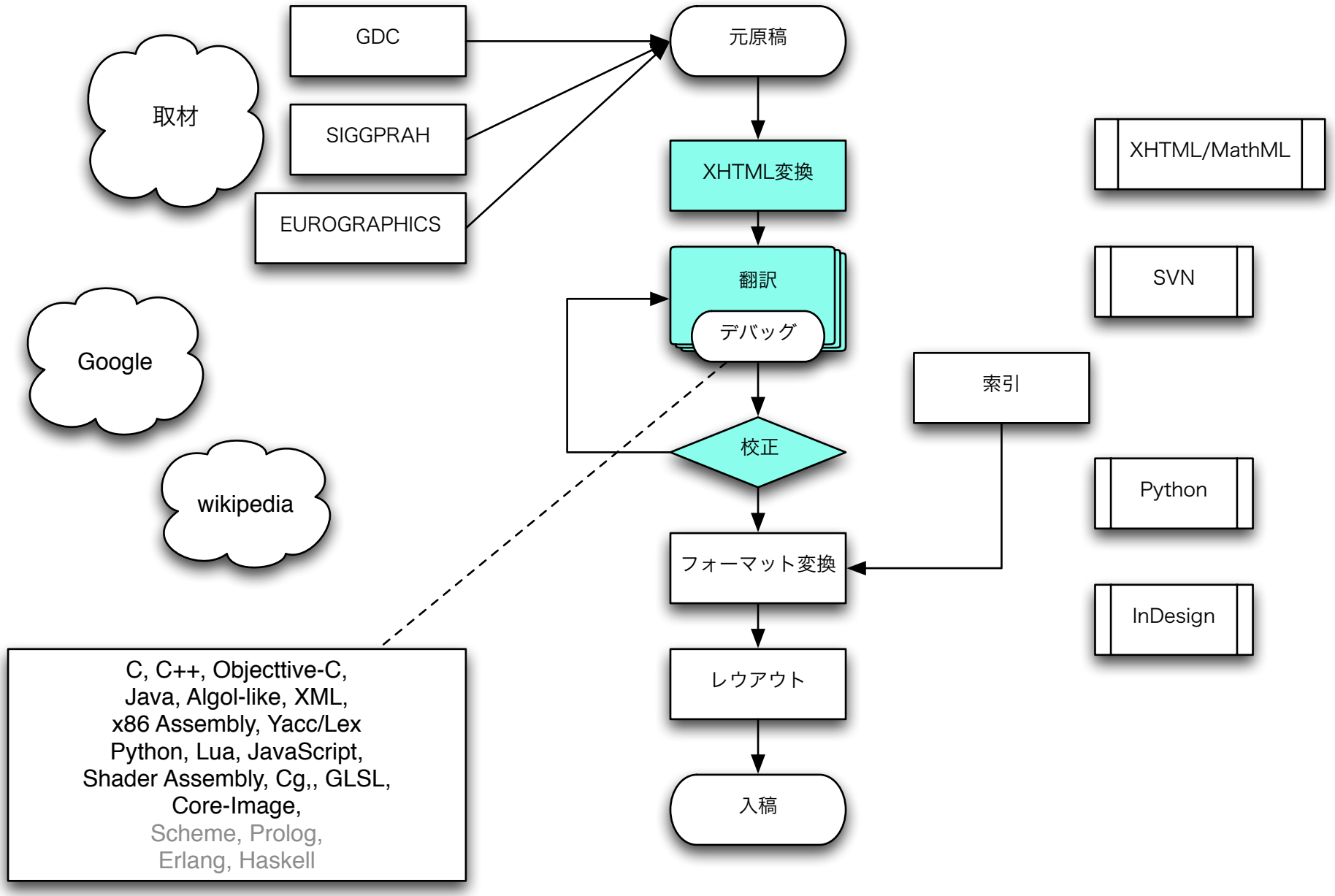

近日発売！？
TEXTURING & MODELING
A Procedural Approach
THIRD EDITION
David S. Ebert
F. Kenton Musgrave
Darwyn Peachey
Ken Perlin
Steven Worley



鋭意翻訳中！！






GDC
SIGGPRAH
EUROGRAPHICS
取材
元原稿
XHTML変換
翻訳
デバッグ
校正
フォーマット変換
レウアウト
入稿
XHTML/MathML
SVN
索引
Python
InDesign
Google
wikipedia
C, C++, Objecttive-C,
Java, Algol-like, XML,
x86 Assembly, Yacc/Lex
Python, Lua, JavaScript,
Shader Assembly, Cg,, GLSL,
Core-Image,
Scheme, Prolog,
Erlang, Haskell

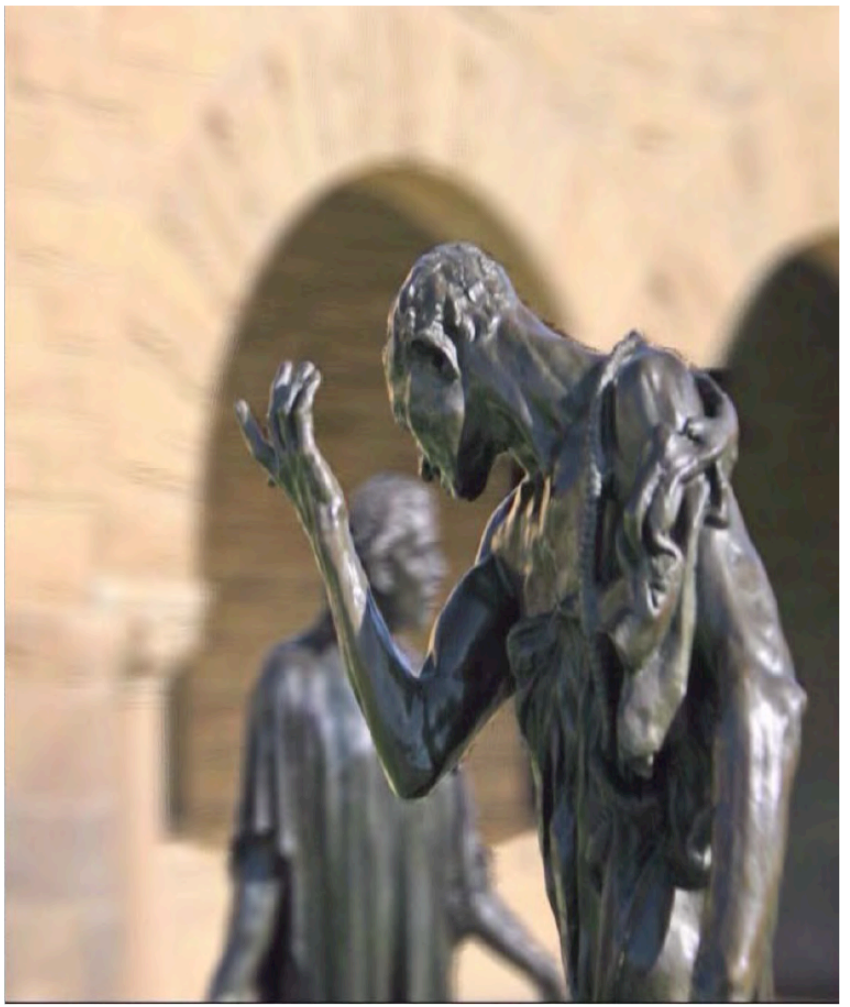
2008
Crete, Greece
April 14- 18

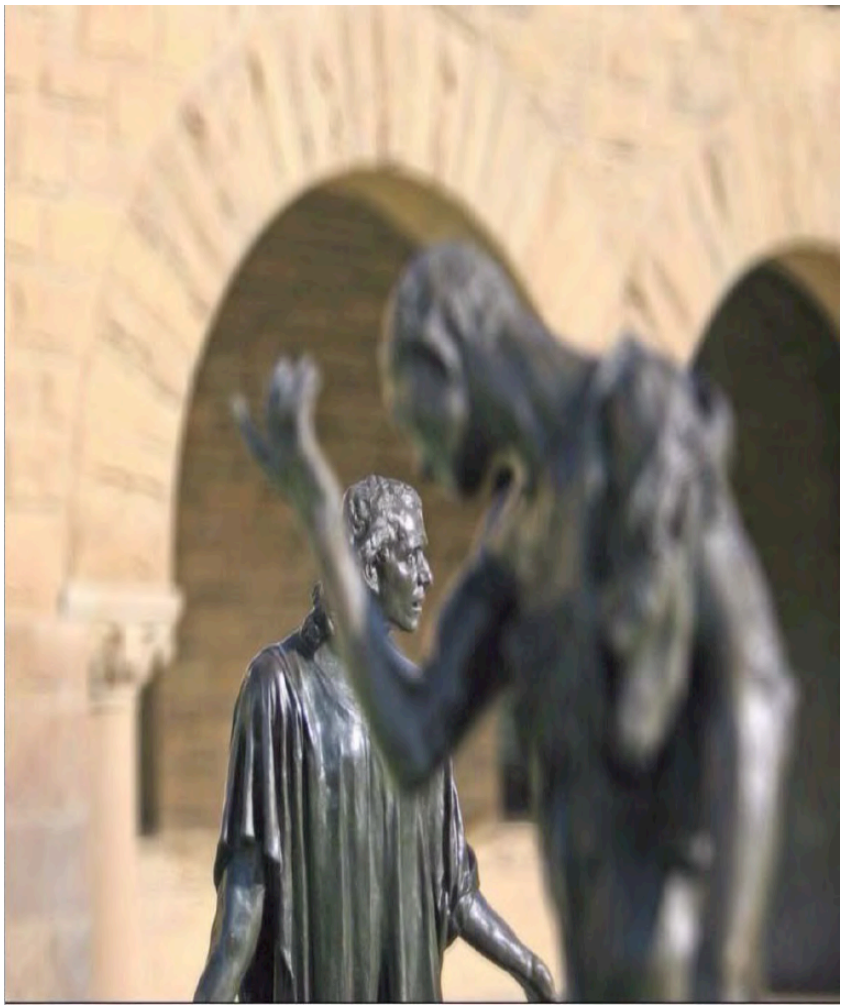
EUROGRAPHICS '08


29
30


ΑΝΑΧΩΡΗΣΕΙΣ / DEPARTURES
ΑΡ. ΠΤΗΣΗΣ FLIGHT NO.
ΠΡΟΟΡΙΣΜΟΣ DESTINATION
ΜΕΣΩ VIA
ΕΛΕΓΚΤΗΡΙΟ COUNTER
ΩΡΑ TIME
ΝΕΑ ΩΡΑ NEW TIME
ΕΞΟΔΟΣ GATE
ΠΑΡΑΤΗΡΗΣΕΙΣ REMARKS
A3 317 Athens 1500 Cnld
A3 4118 Bordeaux 1600 Cnld
A3 4901 Thessaloniki 1600 Check
A3 323 Athens 28 1720 1900 Check
IZ 094 Tel Aviv Larnaca 1730
CY 468 Rhodes 23 1750 9 Board
A3 4202 Rhodes 1815 1900
SEH 182 Kos 31 1830 Check
ZU 5022 Tel Aviv 2 1900 Check
TO 3503 Paris 33 1905 Check

(a) (b) (c)





Efficient Cache Replacement Using the Age and Cost Metrics

Colt "MainRoach" McAnlis
Microsoft Ensemble Studios
cmcanlis@ensemblestudios.com

In memory-constrained game environments, custom media caches are used to amplify the amount of data in a scene, while leaving a smaller memory footprint than containing the entire media in memory at once. The most difficult aspect of using a cache system is identifying the proper victim page to vacate when the cache fills to its upper bounds. As cache misses occur, the choice of page-replacement algorithm is essential—this choice is directly linked to the performance and efficiency of hardware memory usage for your game. A bad algorithm will often destroy the performance of your title, whereas a well implemented algorithm will enhance the quality of your game by a significant factor, without affecting performance. Popular cache-replacement algorithms, such as LRU, work well for their intended environment, but often struggle in situations that require more data to make accurate victim page identifications. This gem presents the Age and Cost metrics to be used as values in constructing the cache-replacement algorithm that best fits your game's needs.

Overview

When data is requested from main memory, operating systems will pull the data into a temporary area of memory (called a *cache*), which can be accessed at a faster speed than main memory. The cache itself is a predefined size, segmented into smaller sets

```
<h1>Efficient Cache Replacement Using the Age and
Cost Metrics</h1>
<p class="author">Colt “MainRoach” McAnlis<br/
>Microsoft Ensemble Studios</p>
<p class="email">cmcanlis@ensemblestudios.com</p>
<p class="dc">In memory-constrained game
environments, custom media caches are used to amplify
the amount of data in a scene, while leaving a
smaller memory footprint than containing the entire
media in memory at once. The most difficult aspect of
using a cache system is identifying the proper victim
page to vacate when the cache fills to its upper
bounds. As cache misses occur, the choice of page-
replacement algorithm is essential— this choice is
directly linked to the performance and efficiency of
hardware memory usage for your game. A bad algorithm
will often destroy the performance of your title,
whereas a well implemented algorithm will enhance the
quality of your game by a significant factor, without
affecting performance. Popular cache-replacement
algorithms, such as LRU, work well for their intended
environment, but often struggle in situations that
require more data to make accurate victim page
identifications. This gem presents the Age and Cost
metrics to be used as values in constructing the
cache-replacement algorithm that best fits your
game’s needs.</p>
<h2>Overview</h2>
<p class="ni">When data is requested from main
memory, operating systems will pull the data into a
temporary area of memory (called a <em>cache</
em>), ...</p>
```

Efficient Cache Replacement Using the Age and Cost Metrics

Colt “MainRoach” McAnlis
Microsoft Ensemble Studios

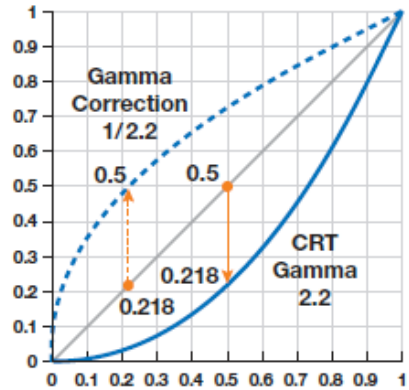
cmcanlis@ensemblestudios.com

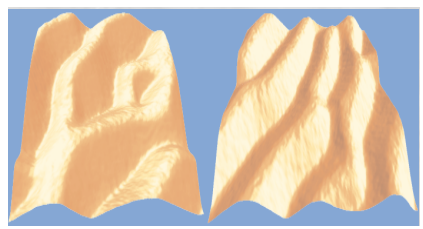
In memory-constrained game environments, custom media caches are used to amplify the amount of data in a scene, while leaving a smaller memory footprint than containing the entire media in memory at once. The most difficult aspect of using a cache system is identifying the proper victim page to vacate when the cache fills to its upper bounds. As cache misses occur, the choice of page-replacement algorithm is essential— this choice is directly linked to the performance and efficiency of hardware memory usage for your game. A bad algorithm will often destroy the performance of your title, whereas a well implemented algorithm will enhance the quality of your game by a significant factor, without affecting performance. Popular cache-replacement algorithms, such as LRU, work well for their intended environment, but often struggle in situations that require more data to make accurate victim page identifications. This gem presents the Age and Cost metrics to be used as values in constructing the cache-replacement algorithm that best fits your game's needs.

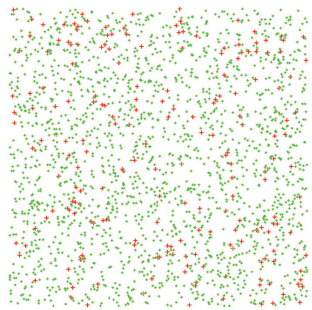
Overview

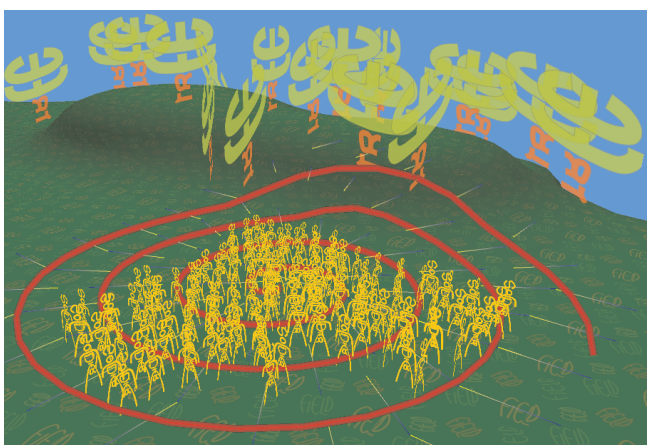
```
<h1>AgeとCost基準を使う効率的なキャッシュ置換</h1>
<p class="author">Colt “MainRoach” McAnlis<br/
>Microsoft Ensemble Studios</p>
<p class="email">cmcanlis@ensemblestudios.com</
p>
<p class="dc">メモリが制約されたゲーム環境では、カスタム
メディア キャッシュを使って、メディア全体を一度にメモリ中に
持つよりも小さなメモリ フットプリントを残しながら、シーン
中のデータの量を増幅する。キャッシュ システムの使用で最も
難しい側面が、キャッシュがその上限まで一杯になったときに明
け渡す、適切な犠牲ページの識別だ。キャッシュ ミスが発生す
るときには、ページ置換アルゴリズムの選択が非常に重要で、こ
の選択がゲームでのハードウェア メモリの使い方の性能と効率
に直接連関する。悪いアルゴリズムはタイトルの性能をしばしば
損なう一方、良い実装のアルゴリズムは性能に影響を与えること
なく、ゲームの品質を何倍も強化する。LRUなどのよく知られた
キャッシュ置換アルゴリズムは、その意図した環境では上手く動
作するが、正確な犠牲ページ識別を行うために必要なデータが多
い状況では、しばしば四苦八苦する。このGemでは、自分のゲー
ムのニーズに最適なキャッシュ置換アルゴリズムの構築で値とし
て使う、AgeとCost基準を紹介する。</p>
<h2>概要</h2>
<p class="ni">メイン メモリからデータが要求されると
き、OSはメイン メモリよりも高速にアクセスできるメモリの一
時領域<em>キャッシュ</em>と呼ばれる）、...</p>
```

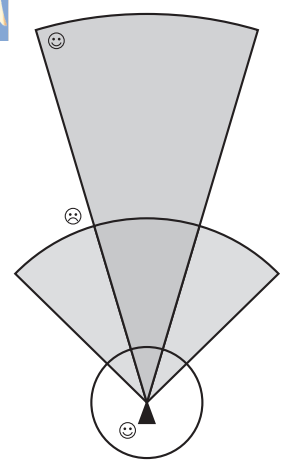
```
<UNICODE-WIN>
<Version:4.0><FeatureSet:InDesign-
Japanese><ParaStyle:h1>AgeとCost基準を使う効率的な
キャッシュ置換
<ParaStyle:p.author>Colt “MainRoach” McAnlis
Microsoft Ensemble Studios
<ParaStyle:p.email>cmcanlis@ensemblestudios.com
<ParaStyle:p.dc>メモリが制約されたゲーム環境では、カスタム
メディア キャッシュを使って、メディア全体を一度にメモリ中に持
つよりも小さなメモリ フットプリントを残しながら、シーン中の
データの量を増幅する。キャッシュ システムの使用で最も難しい側
面が、キャッシュがその上限まで一杯になったときに明け渡す、適
切な犠牲ページの識別だ。キャッシュ ミスが発生するときには、
ページ置換アルゴリズムの選択が非常に重要で、この選択がゲーム
でのハードウェア メモリの使い方の性能と効率に直接連関する。悪
いアルゴリズムはタイトルの性能をしばしば損なう一方、良い実装
のアルゴリズムは性能に影響を与えることなく、ゲームの品質を何
倍も強化する。LRUなどのよく知られたキャッシュ置換アルゴリズ
ムは、その意図した環境では上手く動作するが、正確な犠牲ページ
識別を行うために必要なデータが多い状況では、しばしば四苦八苦
する。このGemでは、自分のゲームのニーズに最適なキャッシュ置換
アルゴリズムの構築で値として使う、AgeとCost基準を紹介する。
<ParaStyle:h2>概要
<ParaStyle:p.ni>メイン メモリからデータが要求されると
き、OSはメイン メモリよりも高速にアクセスできるメモリの一時
領域（<CharStyle:em>キャッシュ<CharStyle:>と呼ばれる）...
```

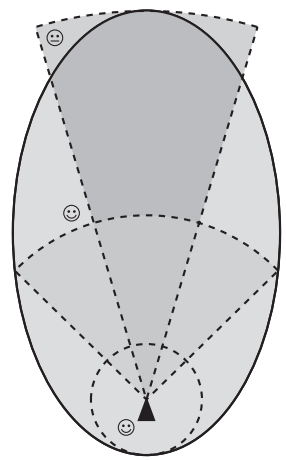

Gamma
Correction
1/2.2
0.5
0.5
0.218
0.218
CRT
Gamma
2.2
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1












小松建三
線形代数
千一夜物語
数学アレルギーには
線形代数が効く
科学技術立国をめざす国民が数学嫌いでは困ります!!
中学校の数学知識があれば大丈夫,
本書で数学アレルギーを治そう!

ジオメトリ

geometry

画像効果

Image Effects

光と影

Lights & Shadows

物理シミュレーション

Physical Sumulation

レンダリング

Rendering

GPUコンピューティング

GPU Computing

GPUによる複雑な手続き的地形の生成
アニメーション付き群衆のレンダリング
DirectX 10ブレンドシェイプ：限界を突き破る
次世代SpeedTreeのレンダリング
汎用適応型メッシュ精緻化
GPUで生成する手続き的な木の風アニメーション
GPU上でのメタボールの点ベースの可視化

真性インポスタ
GPU上での法線マップ焼き付け
高速なオフスクリーン パーティクル
線形であることの重要性
GPU上でのベクトル アートのレンダリング
色による物体検出：リアルタイム ビデオ画像処理にGPUを使う
後処理効果としてのモーション ブラー
実用的な後処理被写界深度

面積和分散影マップ
大域照明によるインタラクティブ シネマチック リライティング
プログラム可能なGPU上の平行分割影マップ
階層隠蔽カリングとジオメトリ シェーダを使う効率的で頑丈な影ボリューム
高品質アンビエント隠蔽
後処理としてのボリューム光散乱

GPU上のリアルタイム剛体シミュレーション
3D流体のリアルタイム シミュレーションとレンダリング
CUDAによる高速なN体シミュレーション
CUDAによる広域衝突検出
CUDAによる衝突検出のためのLCPアルゴリズム
4面体の1パスGPUスキャン変換を使う符号付き距離場

現実的でリアルタイムな肌のレンダリングのための高度なテクニック
プレイ可能ユニバーサル キャプチャ
Crysisの植生の手続き的アニメーションとシェーディング
堅牢な複数スペキュラ反射と屈折
レリーフ マッピング用の緩和コーン ステッピング
Tabula Rasaの遅延シェーディング
GPUベースの重要度サンプリング

GPU上の高速ウィルス シグネチャ照合
GPU上のAES暗号化と解読
CUDAによる効率的な乱数生成と応用
CUDAによる地球の地下の探査
CUDAによる並列プレフィックス和（スキャン）
ガウス関数の漸進的計算
ジオメトリ シェーダを使うコンパクトで可変長の
GPUフィードバック

Game Programming Gems 7

一般プログラミング
数学と物理
人工知能
オーディオ
グラフィックス
ネットワークとマルチプレイヤー
スクリプトとデータ駆動型システム

一般プログラミング

AgeとCost基準を使う効率的なキャッシュ置換
高性能ヒープ アロケータ
Webcamでプレイするビデオ ゲームのためのオプティカル フロー
マルチプラットフォーム スレッディング エンジンの設計と実装
蜂とゲーマーに： 六角形タイルの扱い方
セル オートマトンに基づくリアルタイム戦略ゲームへのスケッチ ベースのインターフェイス
一人称シューディング ゲーム用の足ナビゲーション テクニック
遅延関数呼び出し起動システム
マルチスレッド ジョブと依存性システム
高度なデバッグ テクニック

数学と物理

乱数の生成
ゲームのための高速な汎用レイ問合せ
最遠特徴マップによる高速な剛体衝突検出
投影空間による幾何学計算の精度の改善
XenoCollide：複雑な衝突を簡単に
変換セマンティックを使う効率的な衝突検出
三角スプライン
ガウス乱数を現実的に変化する投射物の経路に使う

AI

ビヘイビア クローニングで面白いエージェントを作り出す
リアリスティックで統合されたエージェント感知モデルの設計
AIアルゴリズムの複雑さの管理：ジェネリック プログラミング アプローチ
態度のすべて：意見、評判、NPCの個性の基礎的要素
プレイヤー トレースとインタラクティブ プレイヤー グラフによるゲームにおける知性の理解
目標指向プラン融合
A*を超えて：IDA*とフリンジ探索

オーディオ

プログラム可能グラフィックス ハードウェアによる音声信号処理
MultiStream―次世代オーディオ エンジンを書く技術
注意深く聞こう、おそらく二度とこれを聴くことはない
ゲームの音声環境から繰り返しを取り除き、サウンド デザインへの新しいアプローチを論じる
リアルタイム オーディオ エフェクトの適用
コンテキスト駆動型、階層化ミキシング

グラフィックス

高度な粒子堆積
チープな話：動的リアルタイム リップシンク
骨格アニメーションの累積誤差を減らす
粗い材質の拡散光のシェーディングのための代替モデル
高性能再分割サーフェイス
放射基底関数テクスチャによるレリーフ インポスタのアニメーション
SM1.1以降のクリップマッピング
高度なデカール システム
屋外地形レンダリング用の大きなテクスチャのマッピング
グラフタル インポスタによるアート ベースのレンダリング

ネットワークとマルチプレイヤー

ゲーム世界の同期の高レベル抽象化
オンライン ゲームでの認証
スマート パケット スニファによるゲーム ネットワークのデバッグ

スクリプトとデータ駆動型システム

自動Luaバインディング システム
内観を使うC++オブジェクトのデータベースへの直列化
データポート
アーティストをサポートしよう：シェーダをエンジンに加える
PythonのASTと踊る

Ipsa scientia potestas est.


Optical Flow
Graftal
Generic Programming
Seismic Processing
Attitude, Valence, Potency
Challenge Hash Authentification
Asian Option
Augmented Reality
Circle of Confusion
Constructive Solid Geometry
Distance Field
Interactive Player Graph
Relief Mapping
Windows Error Reporting
Catmull-Clark Subdivision
Lua
Polynomial Texture Method
Texture Atlas
Wavelet
Marching Cube
Fringe Search
Spherical Harmonics
Monte-Carlo Simulation
Mersennne Twister
Support Mapping
Python
Trigonometric Spline
Abstract Syntax Tree
Prefix Sum
Hough Transform
AES
Design Pattern
Quarternion
Navier-Stokes Equation
Smoothed Particle Hydrodynamics
Nyquist Limitation
SQL
Viseme & Phoneme
Packet Sniffer
Language Binding
Summed-Area Variance Shadow Map
Shader Model


オプティカル フロー
グラフタル
ジェネリック プログラミング
地震探査
態度, 価数, 効力
チャレンジ ハッシュ認証
アジア オプション
強化現実
錯乱円
構成立体幾何
距離場
インタラクティブ プレイヤー グラフ
レリーフ マッピング
Windows Error Reporting
Catmull-Clark 再分割
Lua
多項式テクスチャ法
テクスチャ アトラス
ウェーブレット
マーチングキューブ
フリンジ探索
球面調和
モンテカルロ シミュレーション
メルセンヌツイスター
サポート写像
プレフィックス和
Python
三角スプライン
抽象構文木
ハフ変換
AES
デザインパターン
クォータニオン
ナビエ-ストークス方程式
平滑化粒子流体力学
ナイキスト限界
SQL
口形素 & 音素
パケットスニファ
言語バインディング
面積和分散影マップ
シェーダモデル

Story

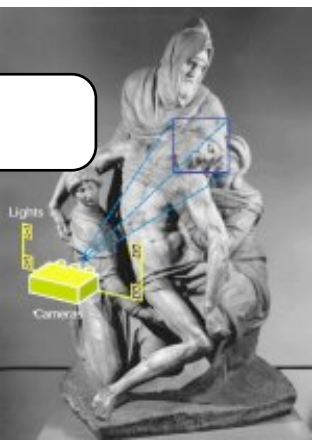
Networking

CAD

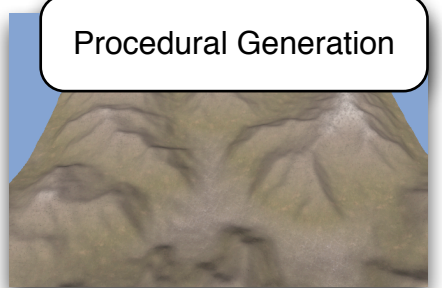
HD Video

Massive Data

SCAN



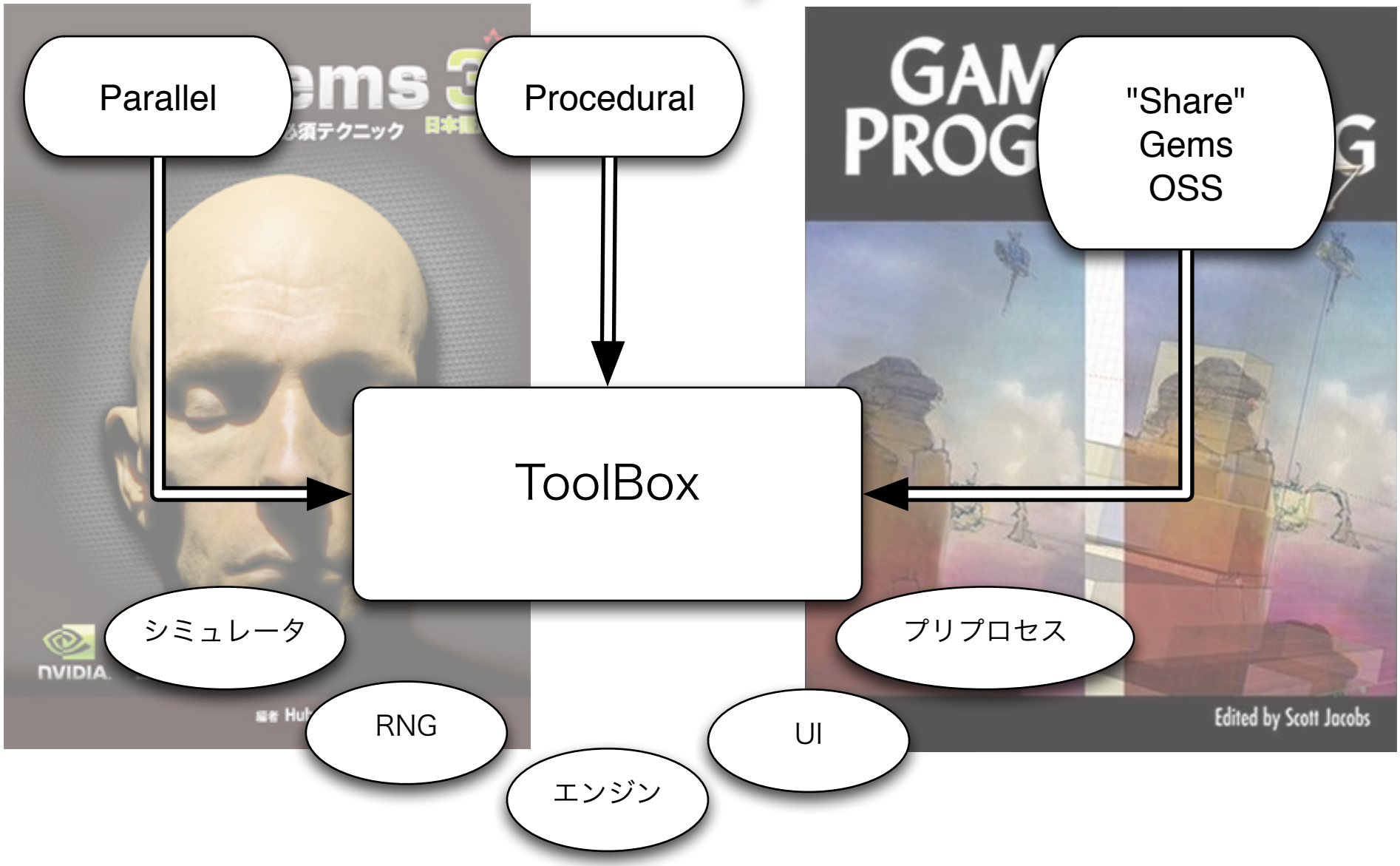
Procedural Generation

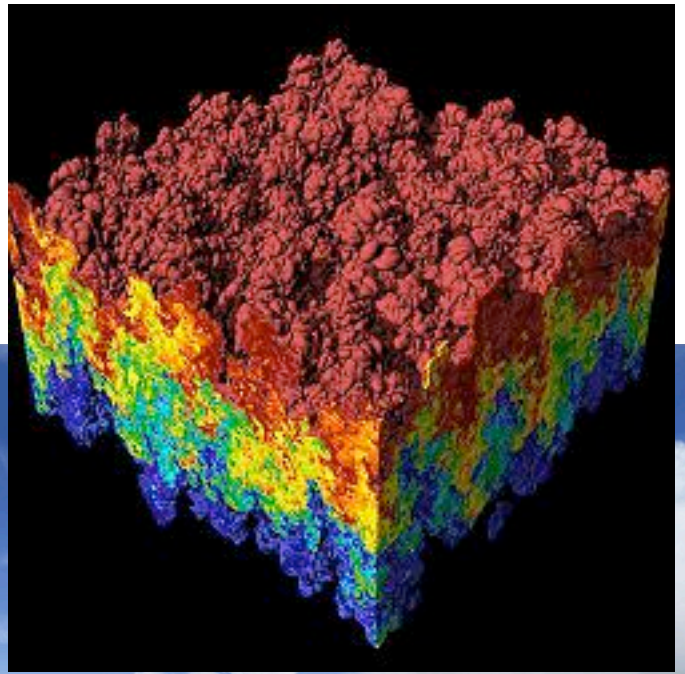


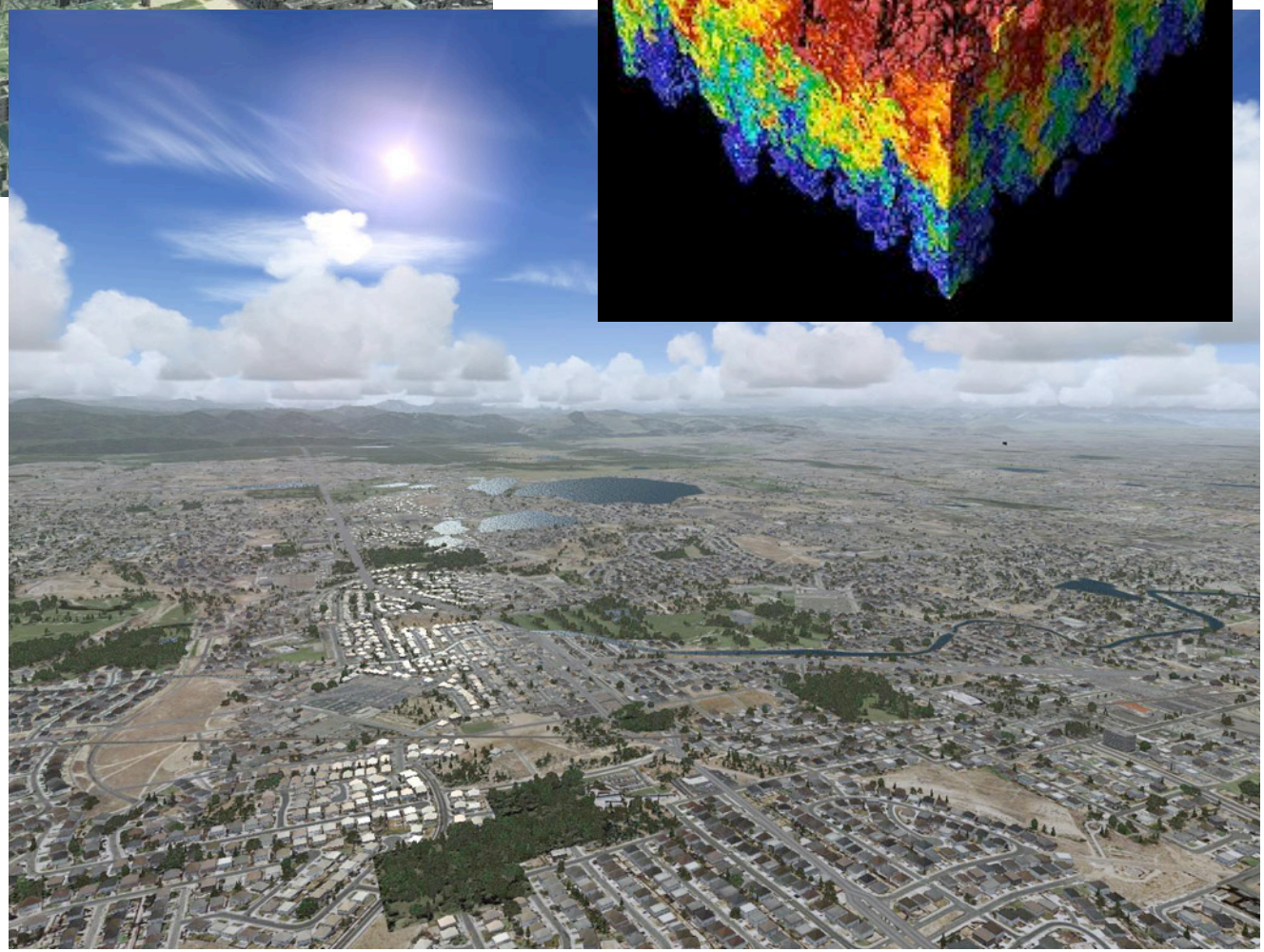
Photo

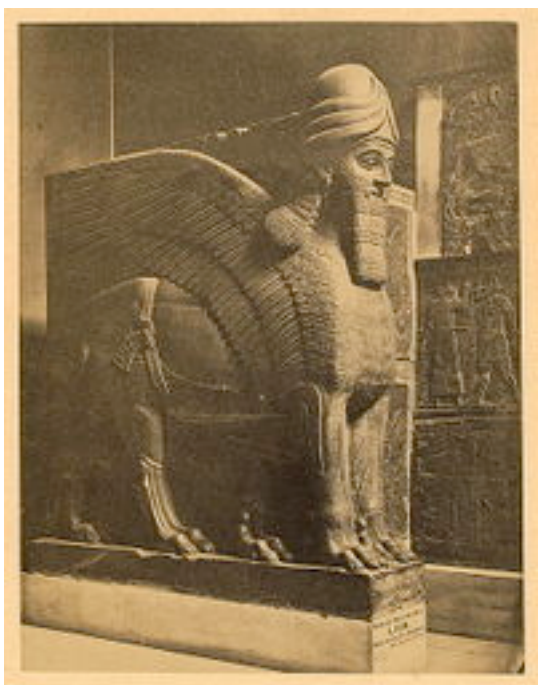
Game

Story


Parallel
Procedural
"Share"
Gems
OSS
ToolBox
シミュレータ
RNG
エンジン
UI
プリプロセス
NVIDIA
Edited by Scott Jacobs

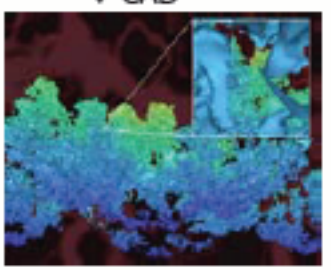




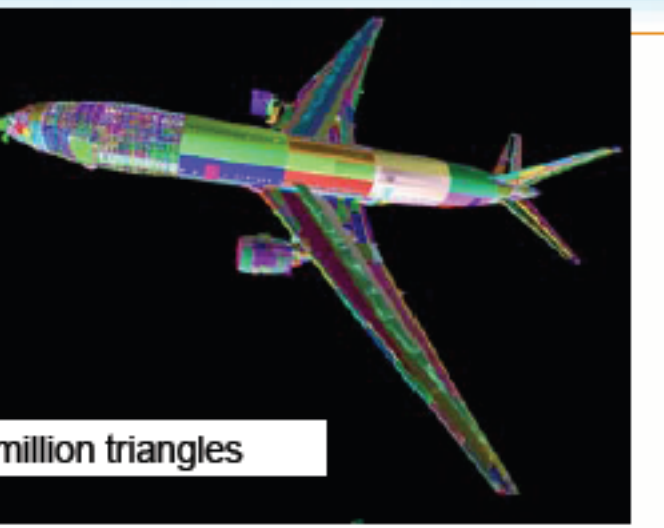


Massive models

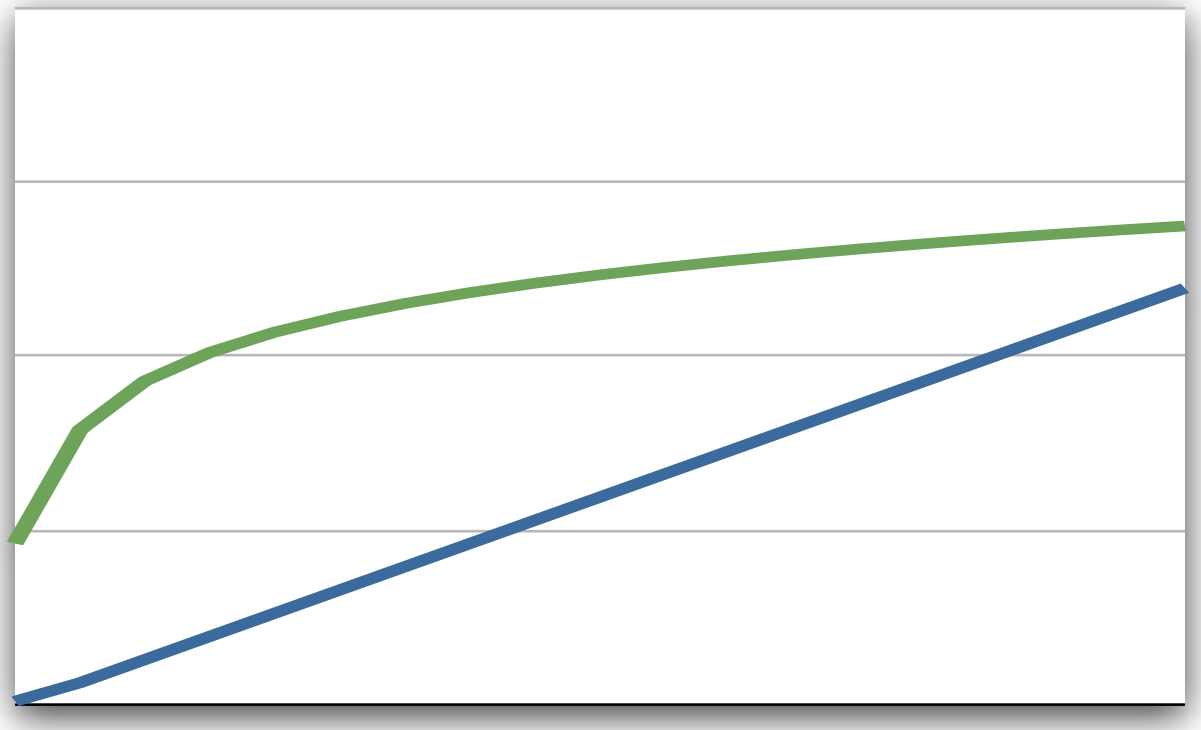
- *Model*: geometric representation of object
- Many sources:
  - Scientific simulation
  - Scanned objects
  - CAD



Boeing 777

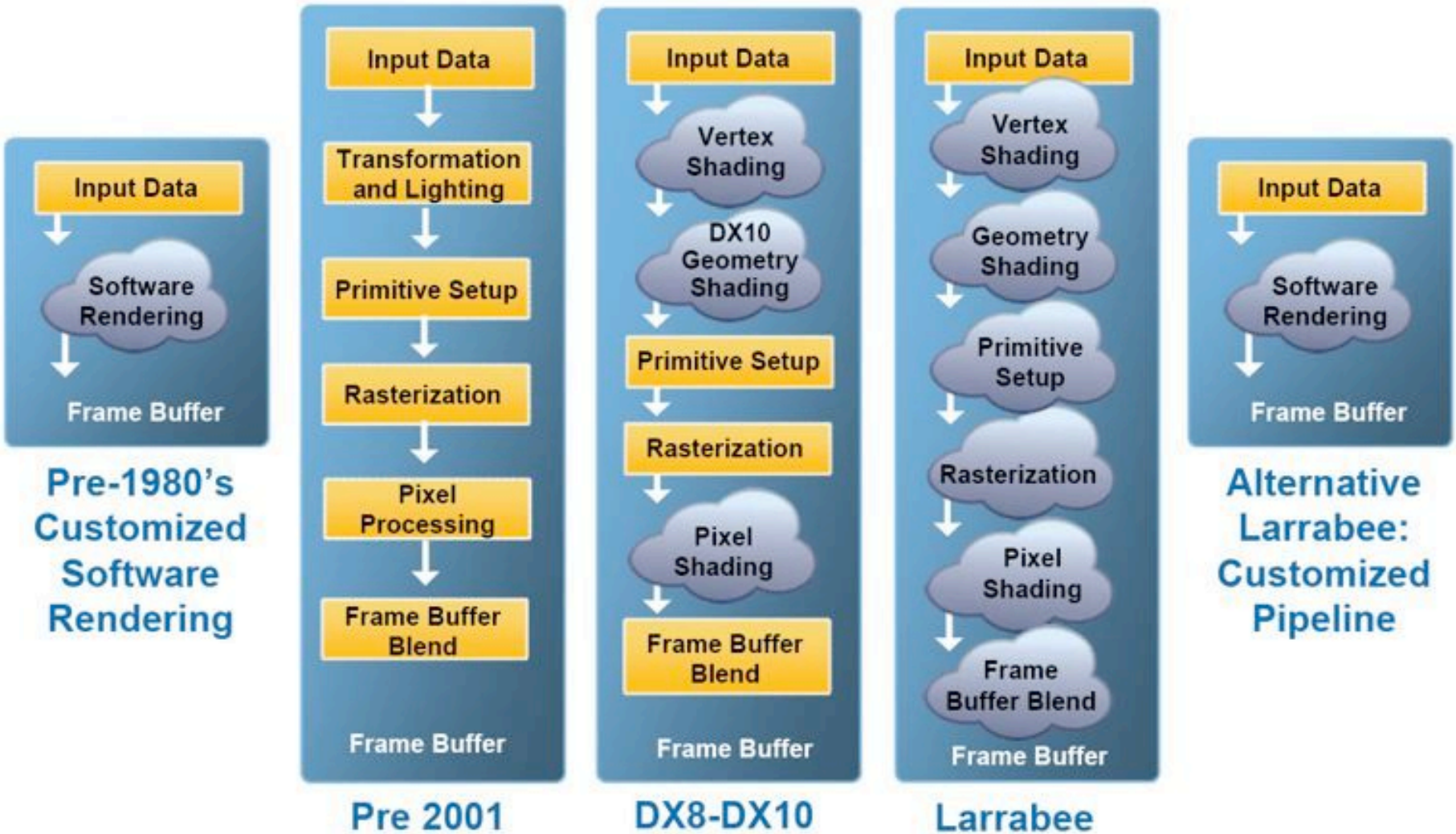


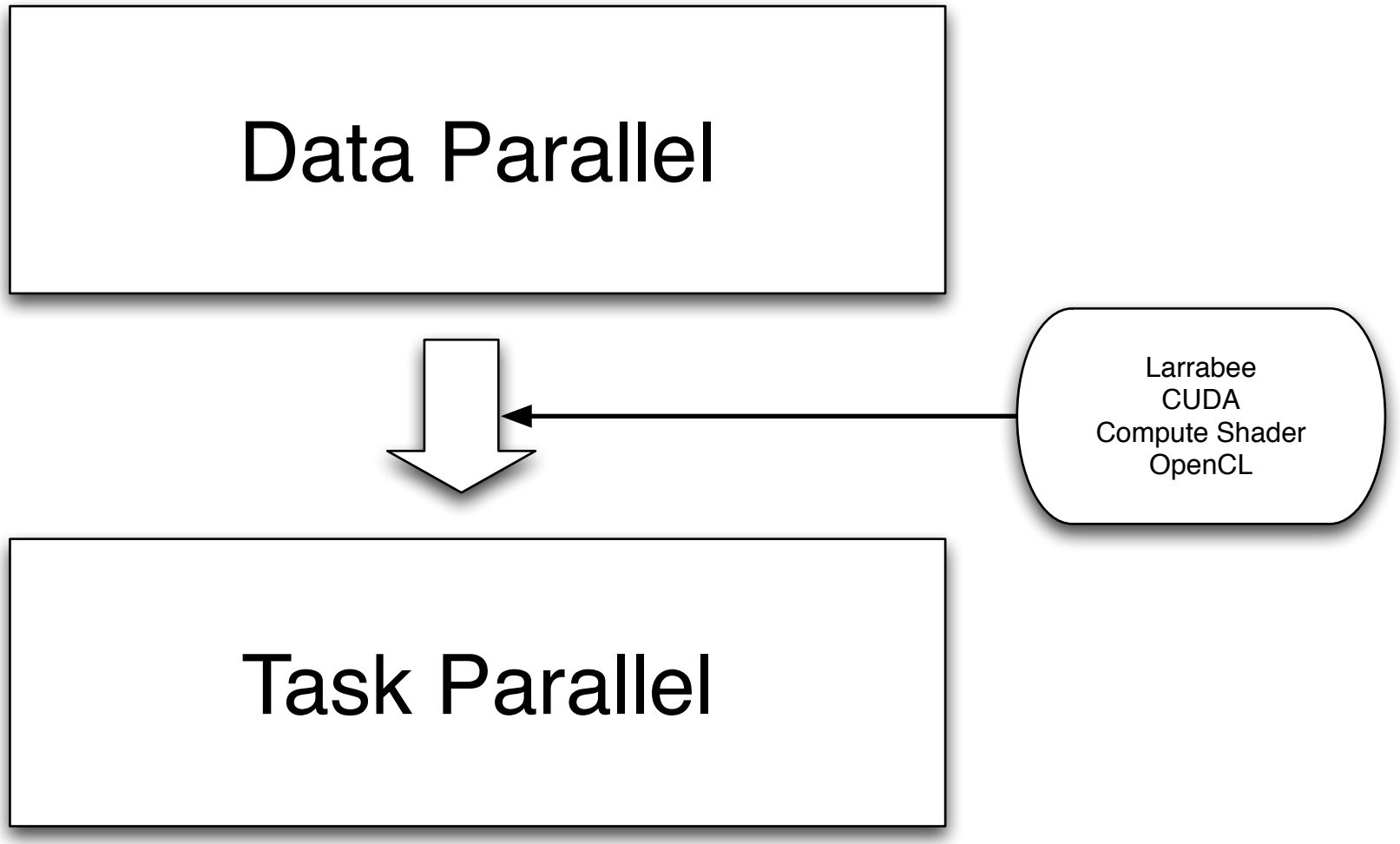
Ray Casting (Tracing) vs. Rasterlization



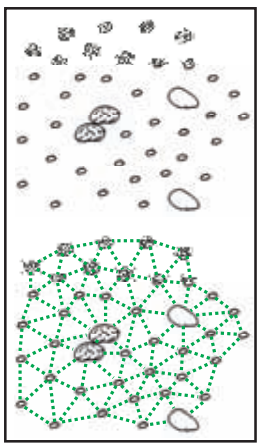
Logarithmic vs. Linear

Graphics Rendering Pipelines

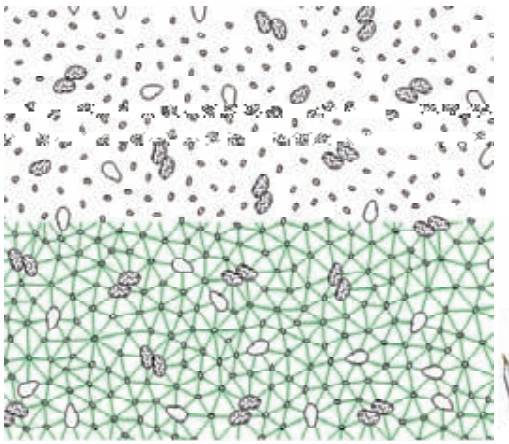



Data Parallel
Larrabee
CUDA
Compute Shader
OpenCL
Task Parallel

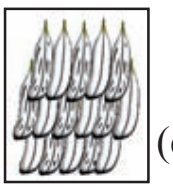
An Example-based Procedural System for Element Arrangement (Eurograph 2008)



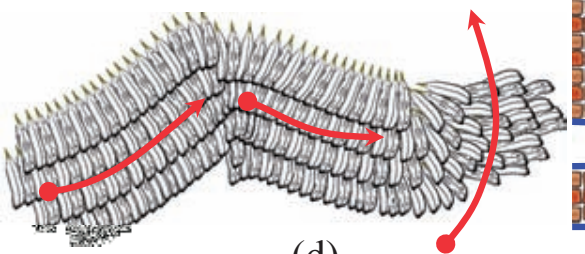
(a)



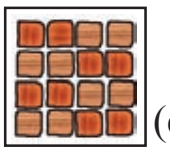
(b)



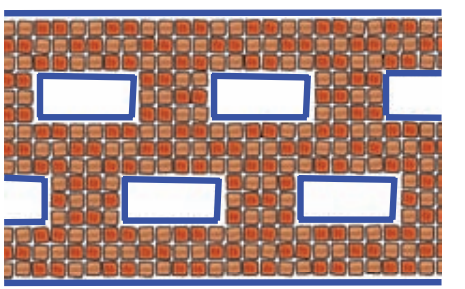
(c)



(d)

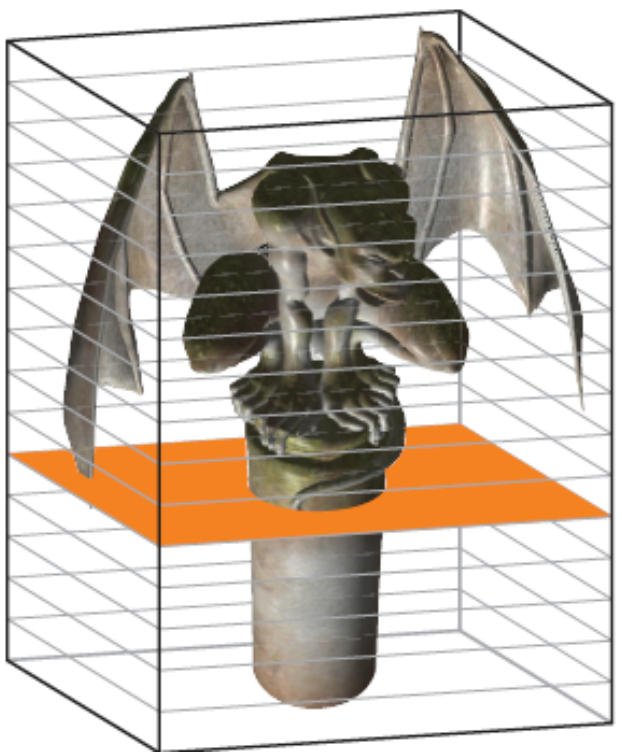


(e)

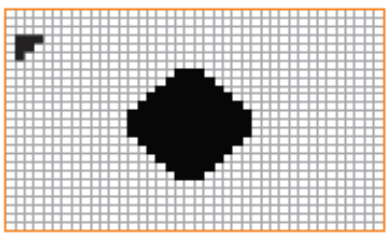


(f)

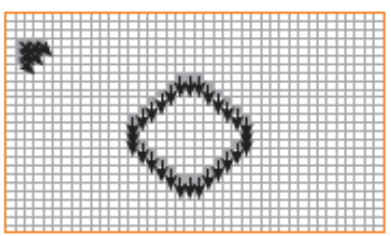
GPU Gems 3

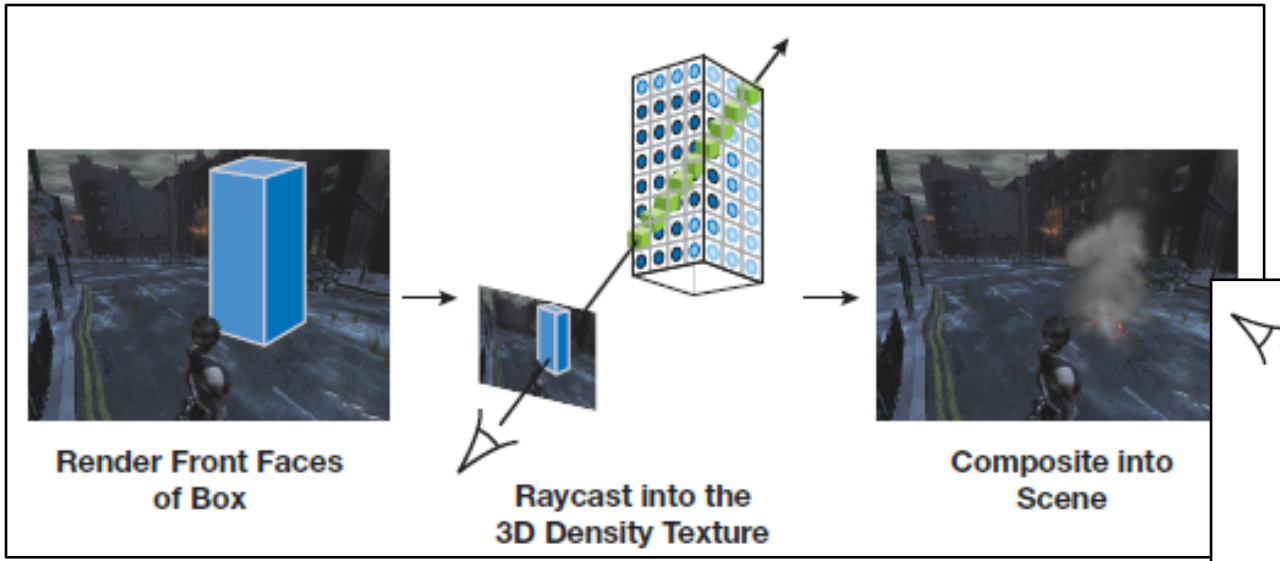


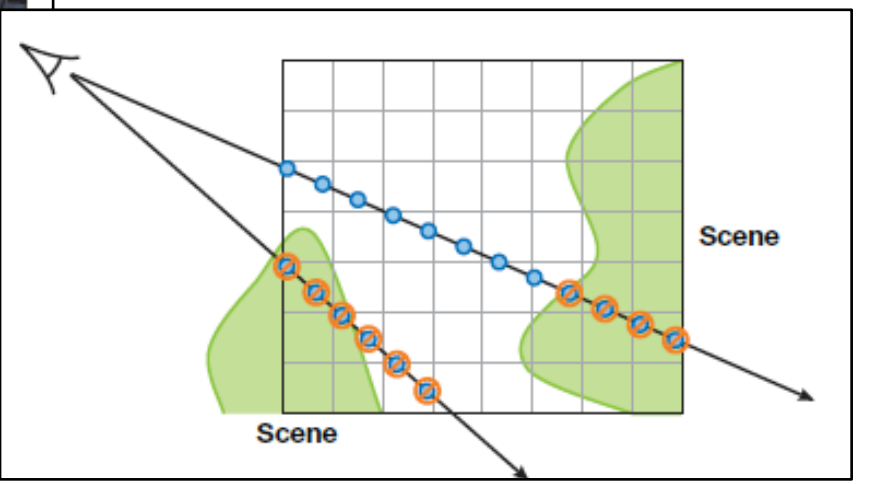
Inside – Outside Texture



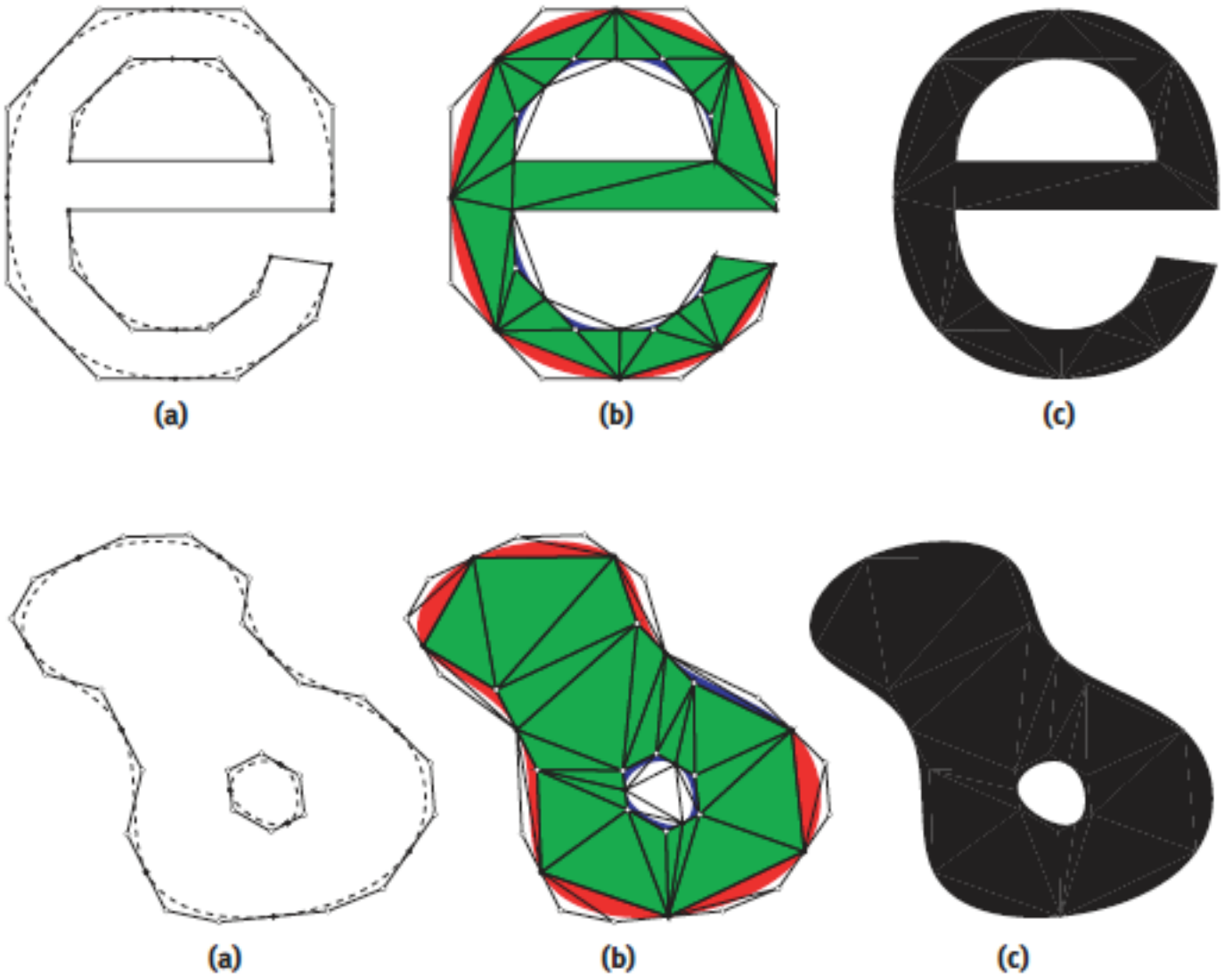
Velocity Texture



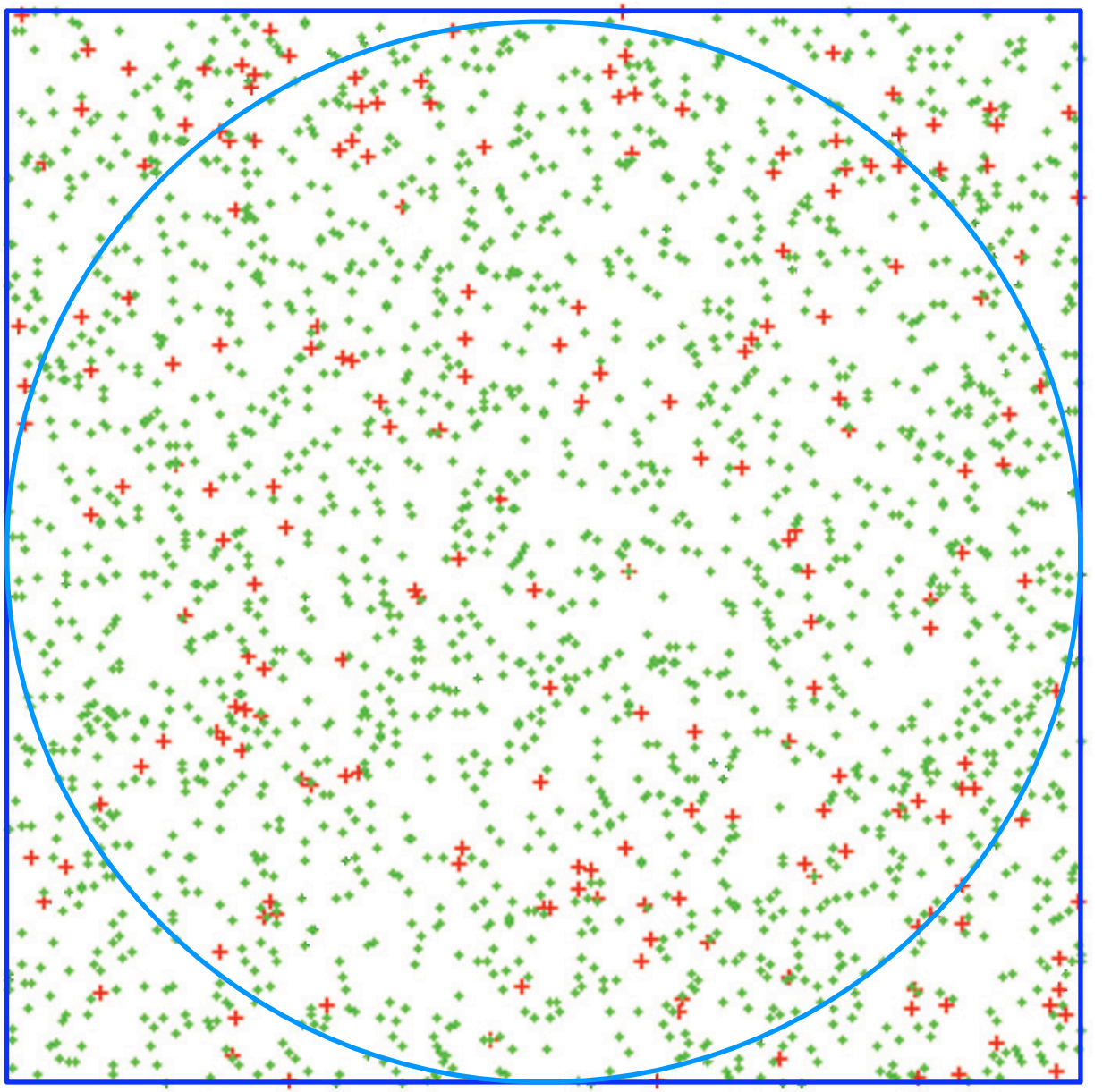

Render Front Faces
of Box
Raycast into the
3D Density Texture
Composite into
Scene

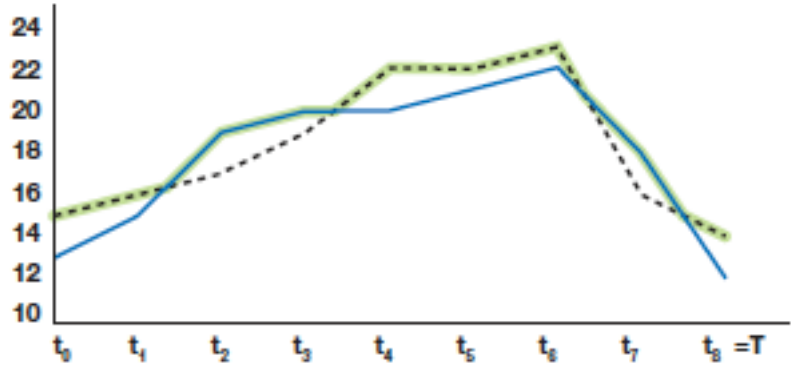

Scene
Scene

GPU Gems 3:



GPU Gems 3: CUDA



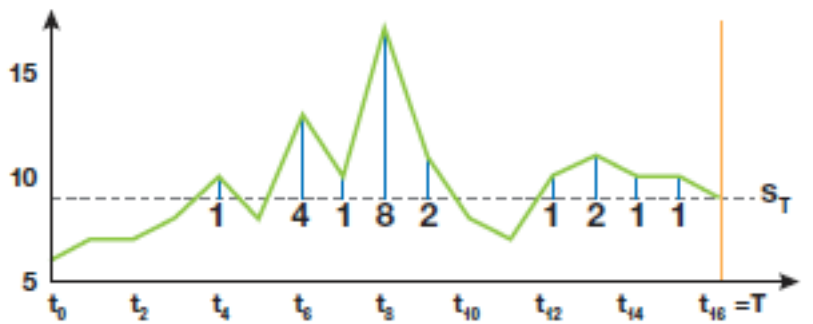


a_t	15	16	17	19	22	22	23	16	14
b_t	13	15	19	20	20	21	22	18	12
s_t	15	16	19	20	22	22	23	18	14

$$S = (s1+s2+ \dots +s_T)/T = (15+16+ \dots +14)/8 = 21.125$$

$$P = \max(S-s_T,0) = \max(21.125-14,0) = 7.125$$

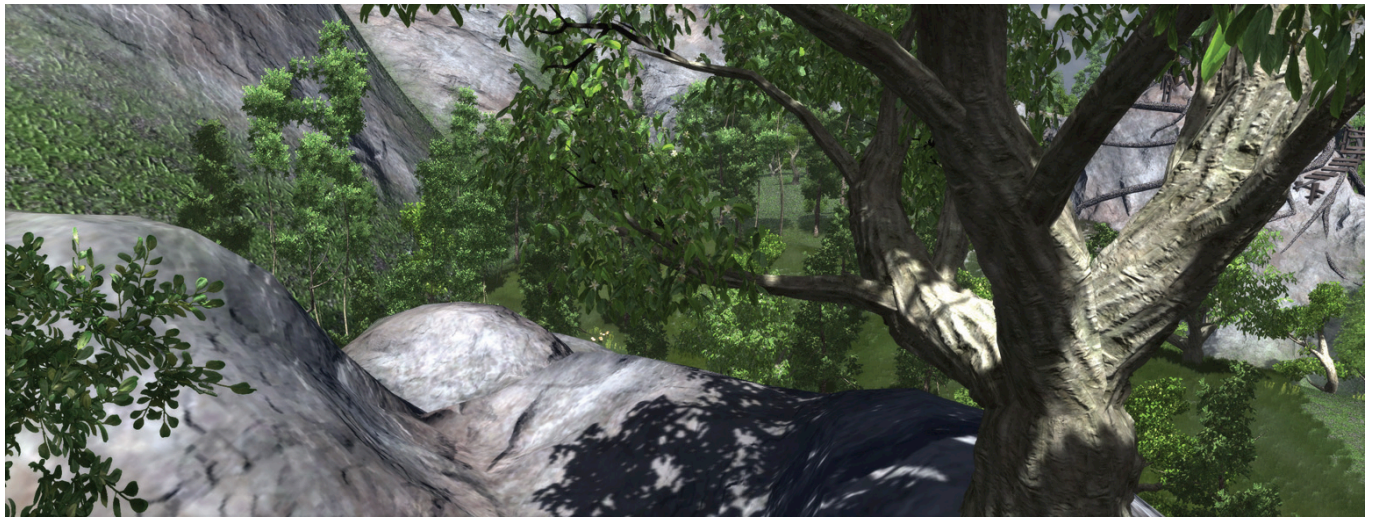
Figure 37-4. Example of One Run of the Asian Basket Simulation

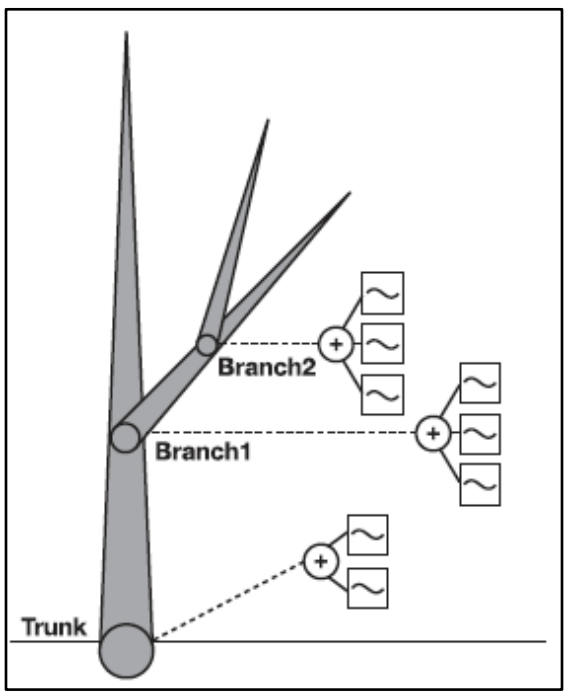


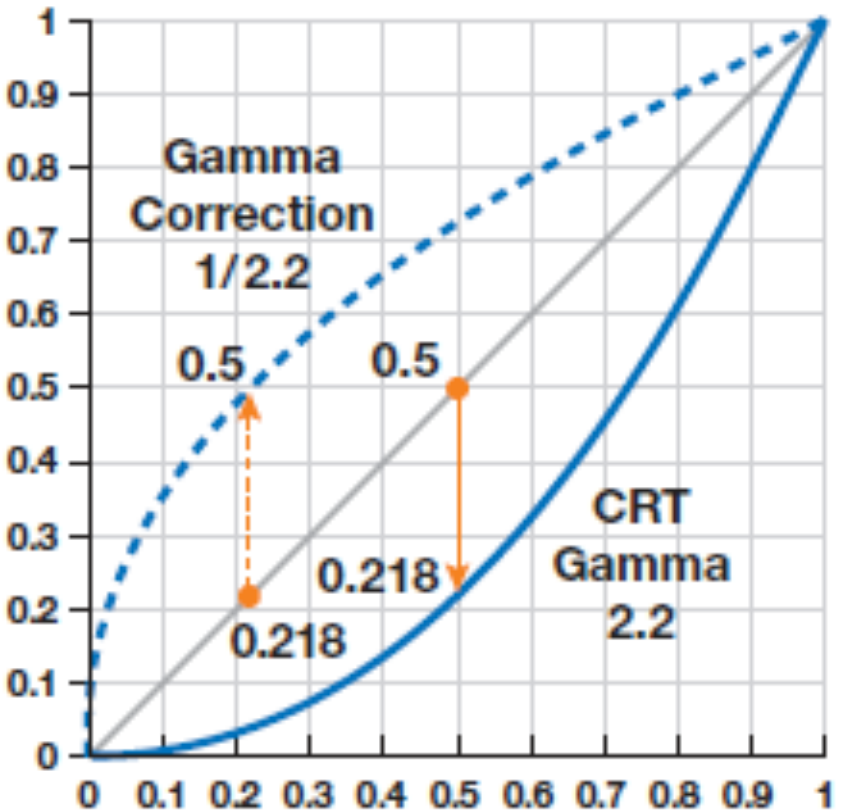
$$P = 1+4+1+8+ \dots +1 = 21$$

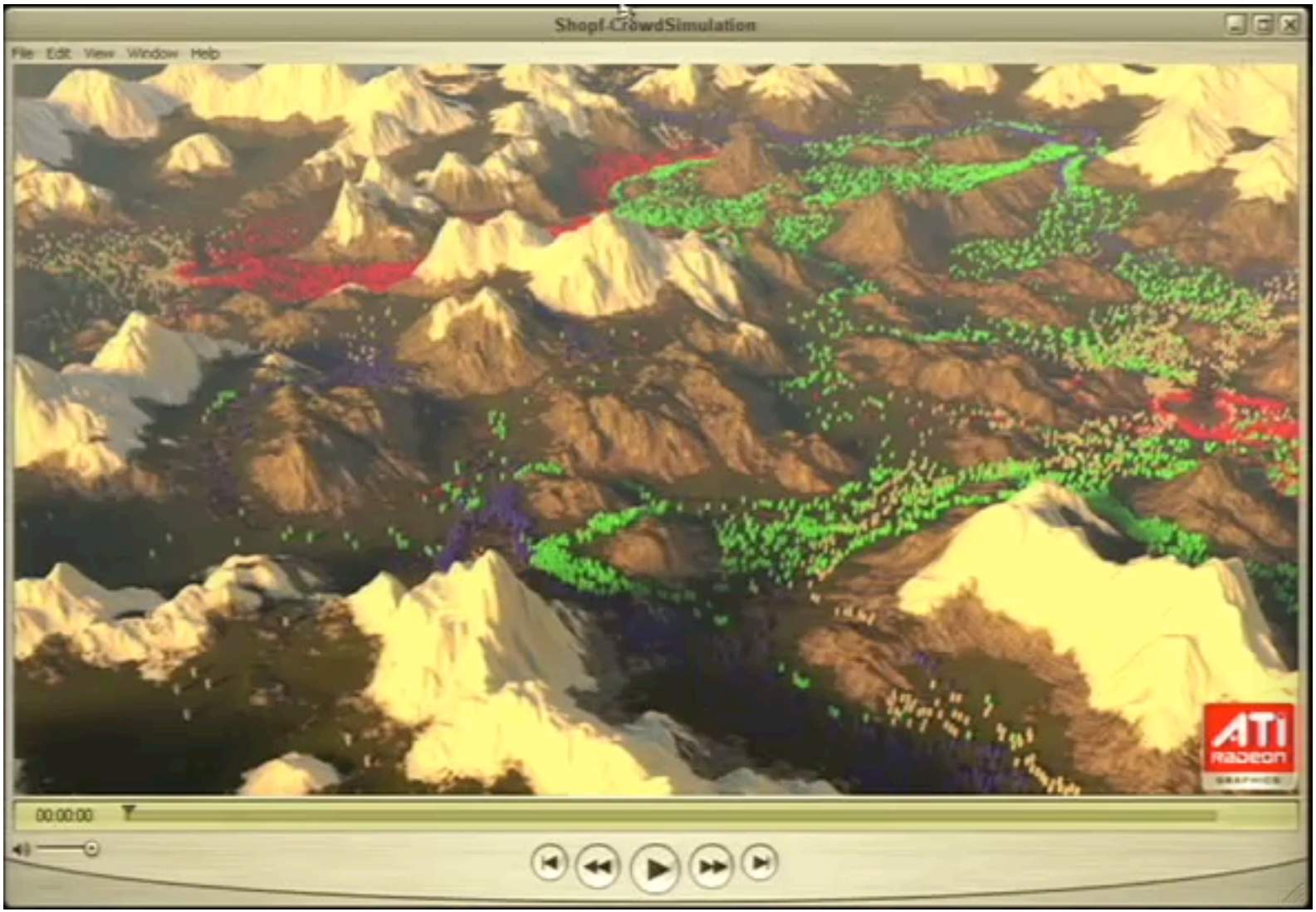
Figure 37-5. Movement of the Asset

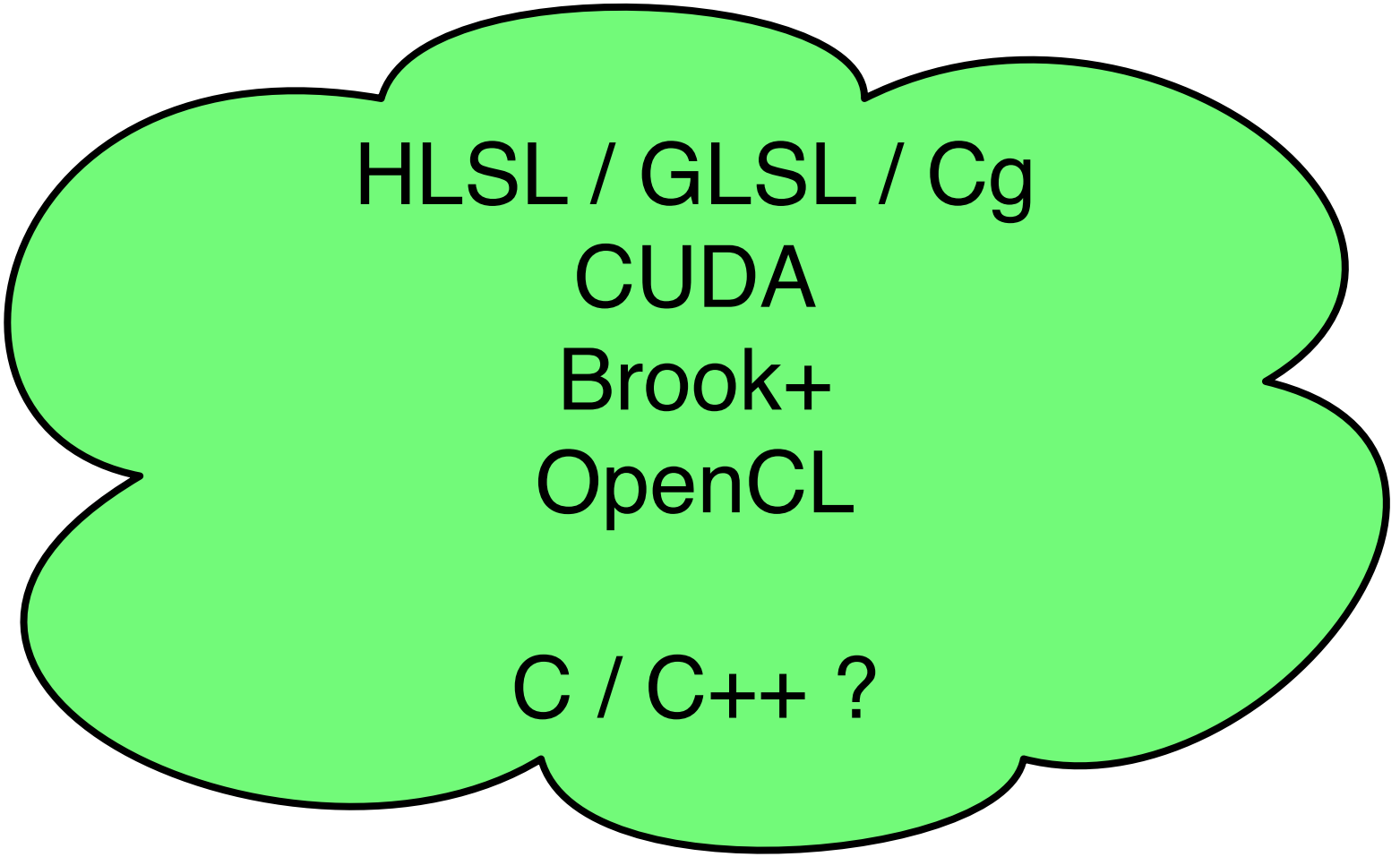
GPU Gems 3：3、6、16章

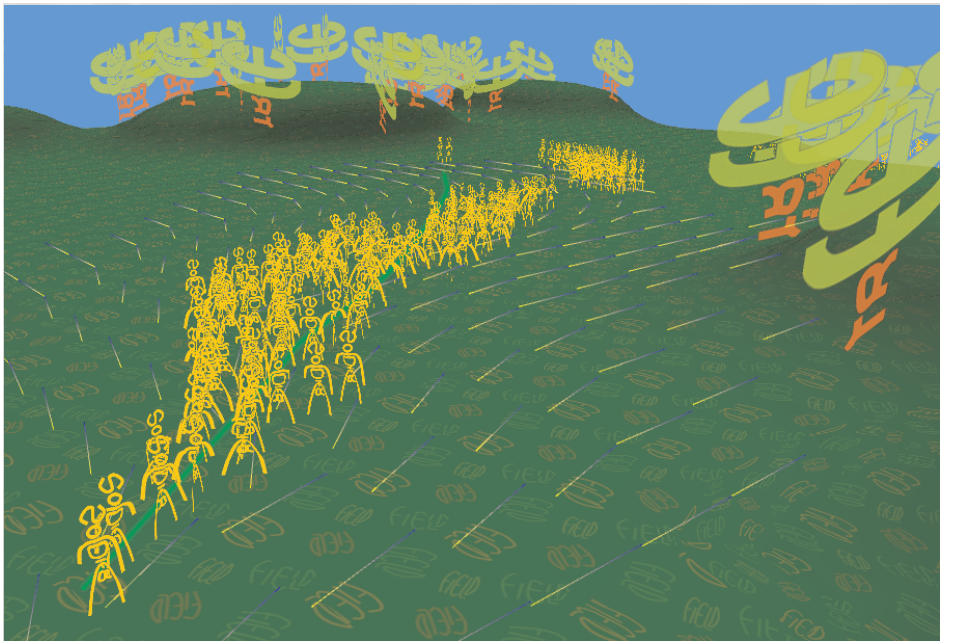


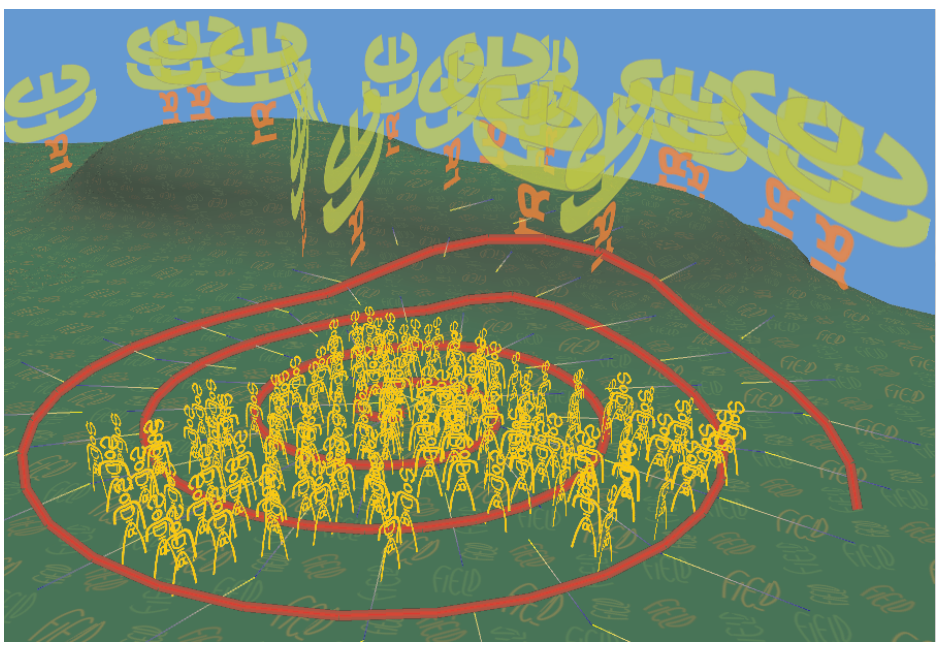


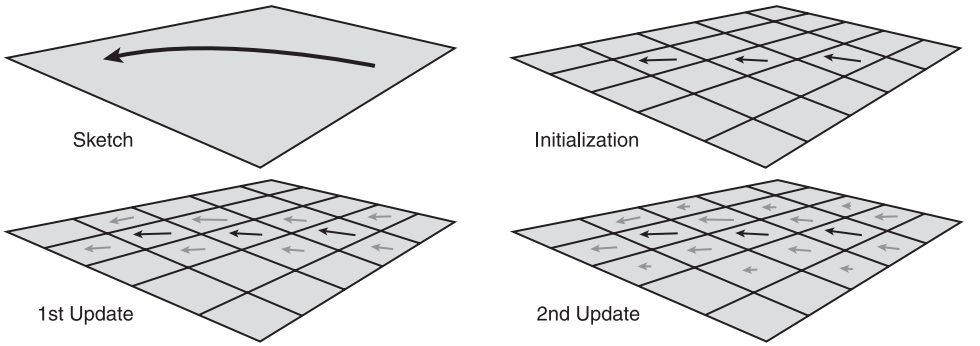

Gamma
Correction
1/2.2
0.5
0.5
0.218
0.218
CRT
Gamma
2.2
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

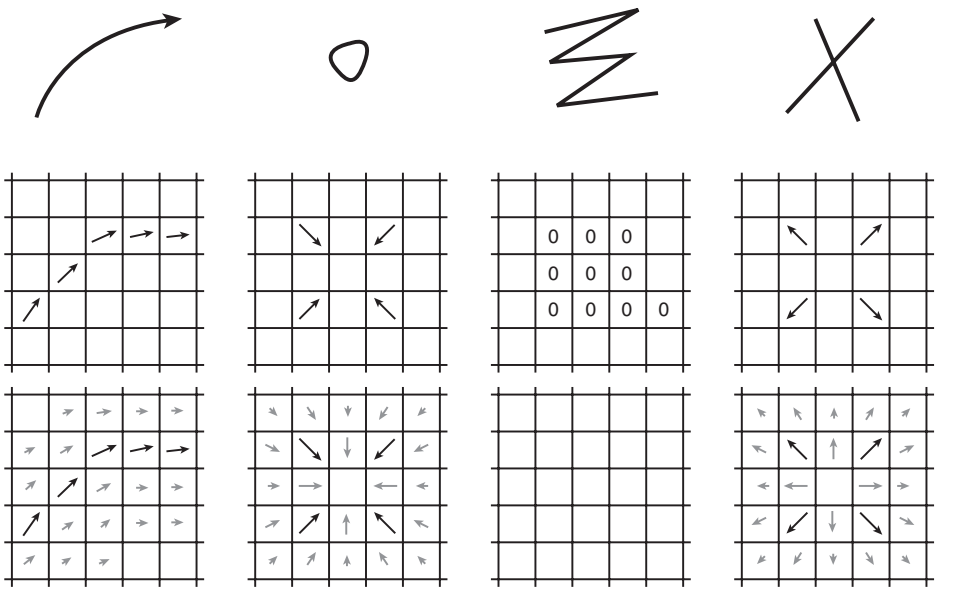

Shopf-CrowdSimulation
File Edit View Window Help
00:00:00
ATi
RADEON

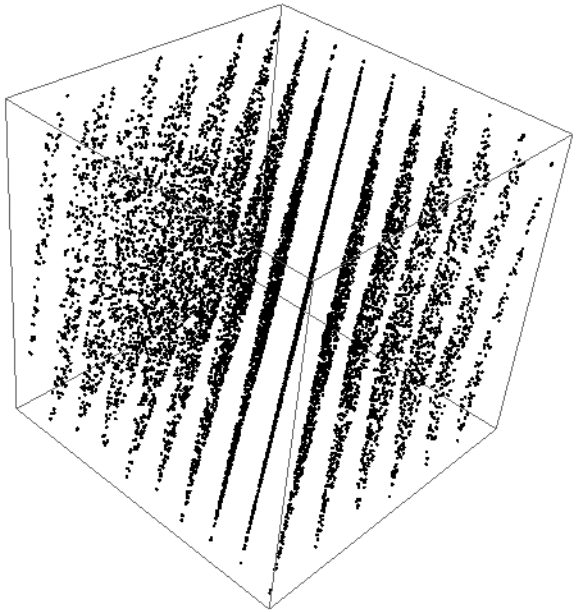

HLSL / GLSL / Cg
CUDA
Brook+
OpenCL
C / C++ ?

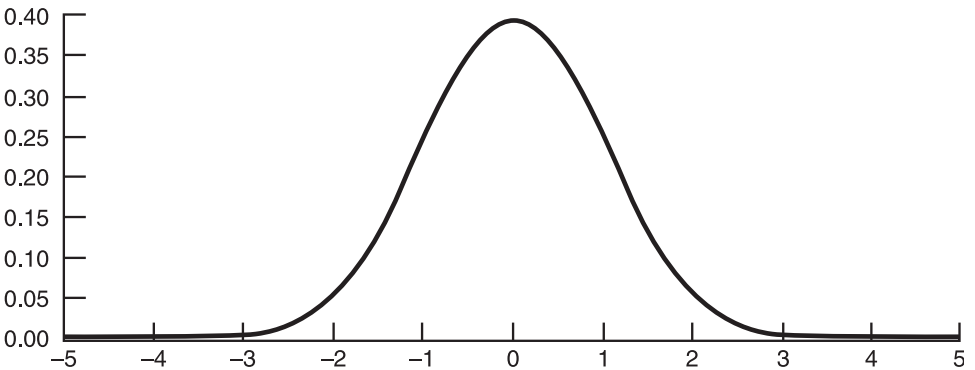


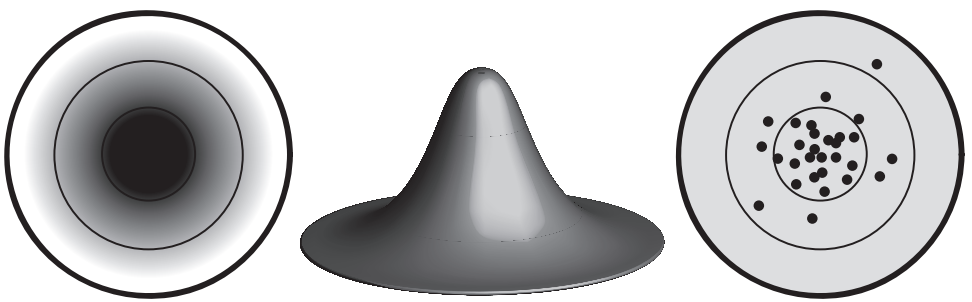


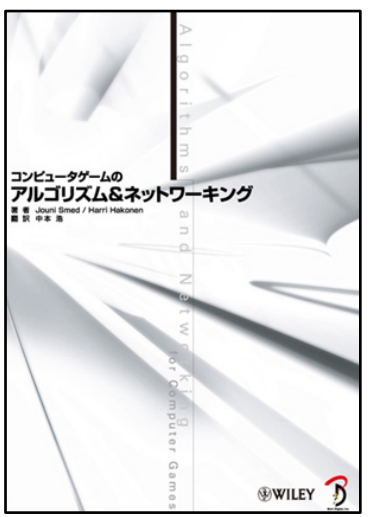

Sketch
Initialization
1st Update
2nd Update

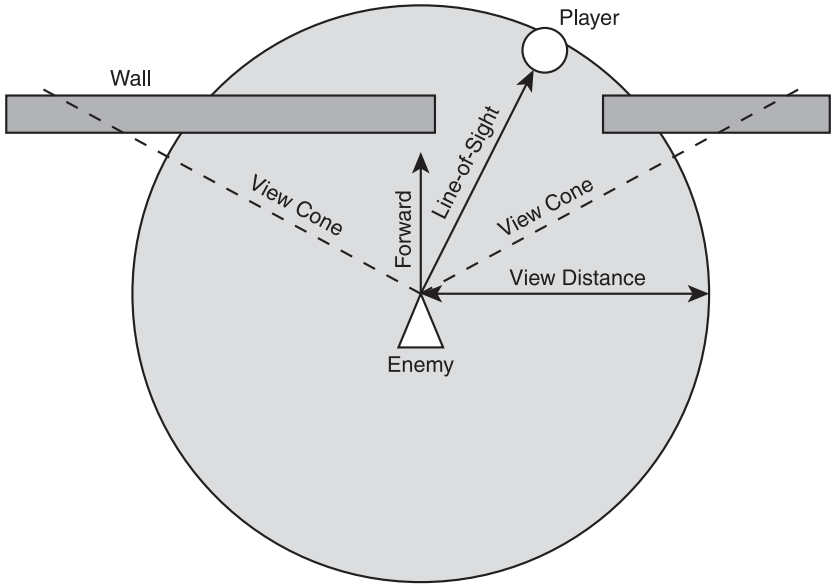

0 0 0
0 0 0
0 0 0 0

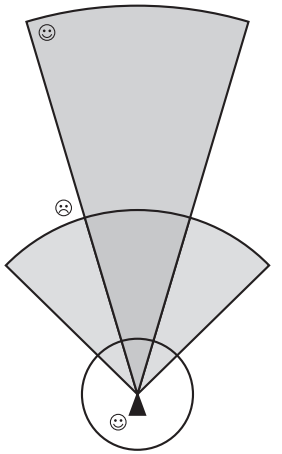


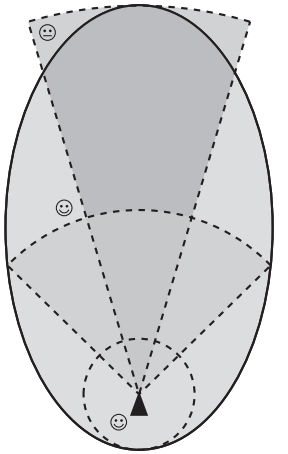

0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
–5
–4
–3
–2
–1
0
1
2
3
4
5

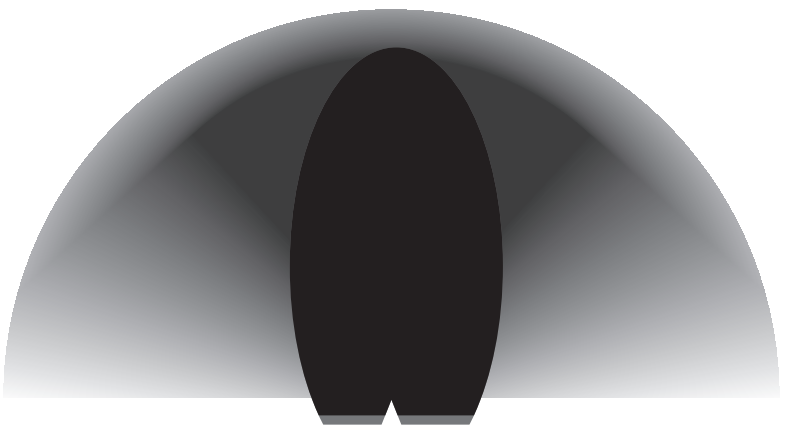


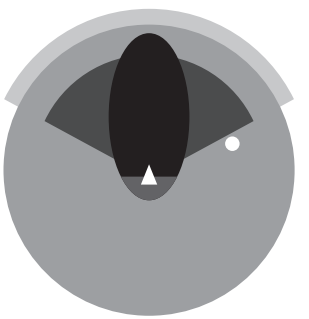

コンピュータゲームの
アルゴリズム&ネットワーキング
著者 Jouni Smed / Harri Hakonen
翻訳 中本 浩
Algorithms and Networking for Computer Games
WILEY





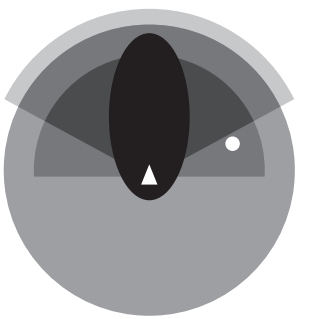




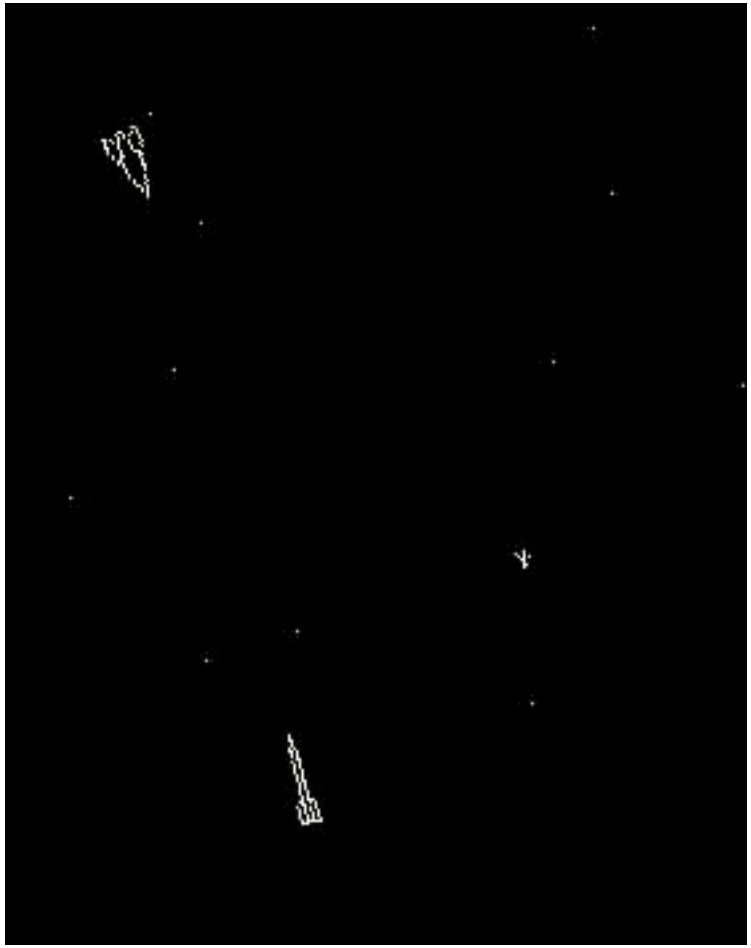


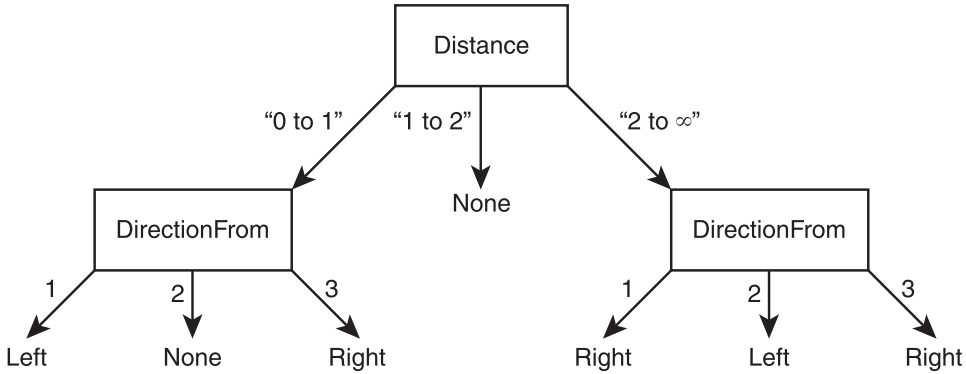
Take Max of Senses

Add Vision and Hearing

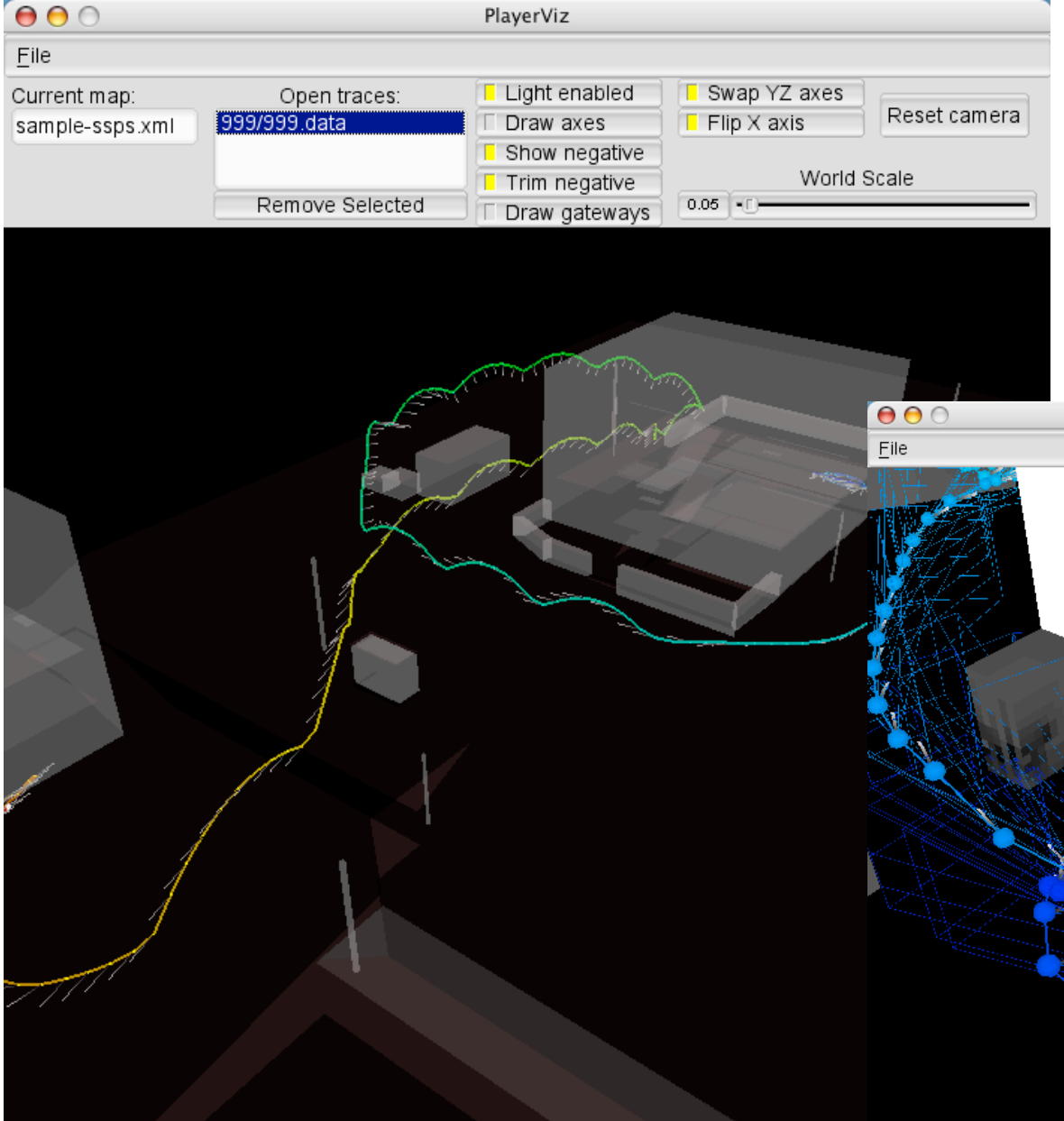


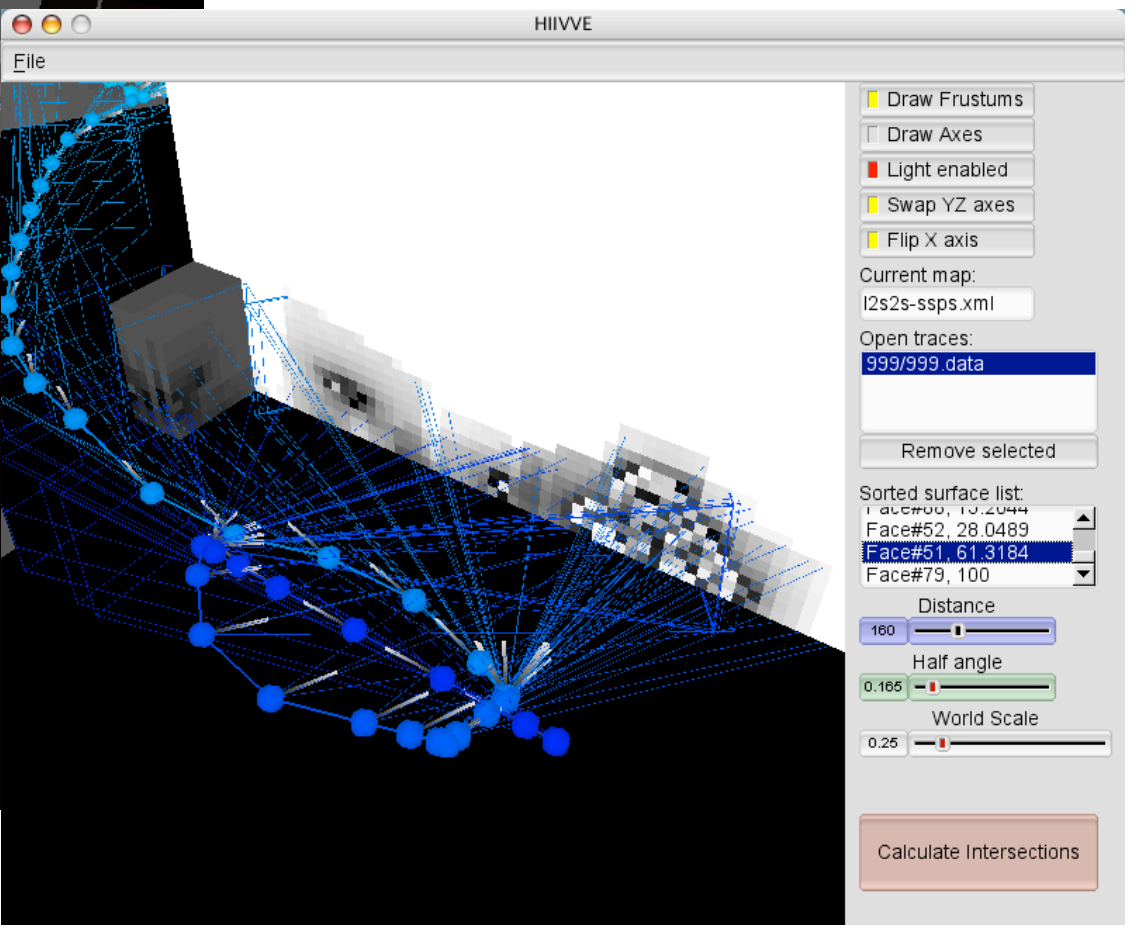
Add Hearing Proportionally





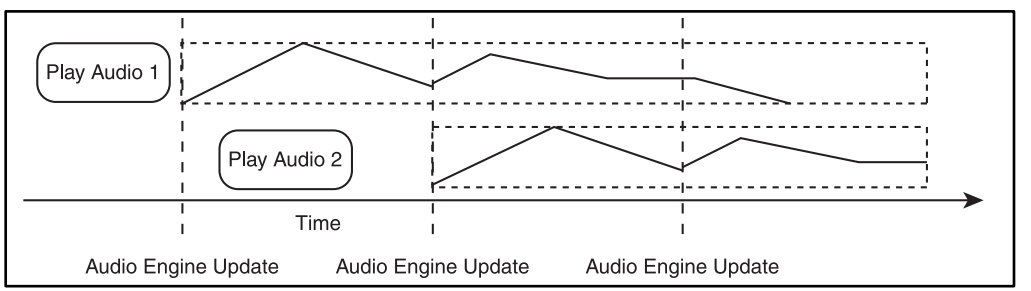

```
function turn
 if GameState.Distance == "0 to 1" then --Left branch
   if GameState.DirectionFrom == 1 then
     Ship.turn ("LEFT")
   elseif GameState.DirectionFrom == 2 then
     Ship.turn ("NONE")
   elseif GameState.DirectionFrom == 3 then
     Ship.turn ("RIGHT")
   else
     Ship.turn ("NONE")
   end
 elseif GameState.Distance == "1 to 2" then --Center branch
   Ship.turn ("NONE")
 elseif GameState.Distance == "2 to inf" then --Right branch
   if GameState.DirectionFrom == 1 then
     Ship.turn ("RIGHT")
   elseif GameState.DirectionFrom == 2 then
     Ship.turn ("LEFT")
   elseif GameState.DirectionFrom == 3 then
     Ship.turn ("RIGHT")
   else
  ...
```

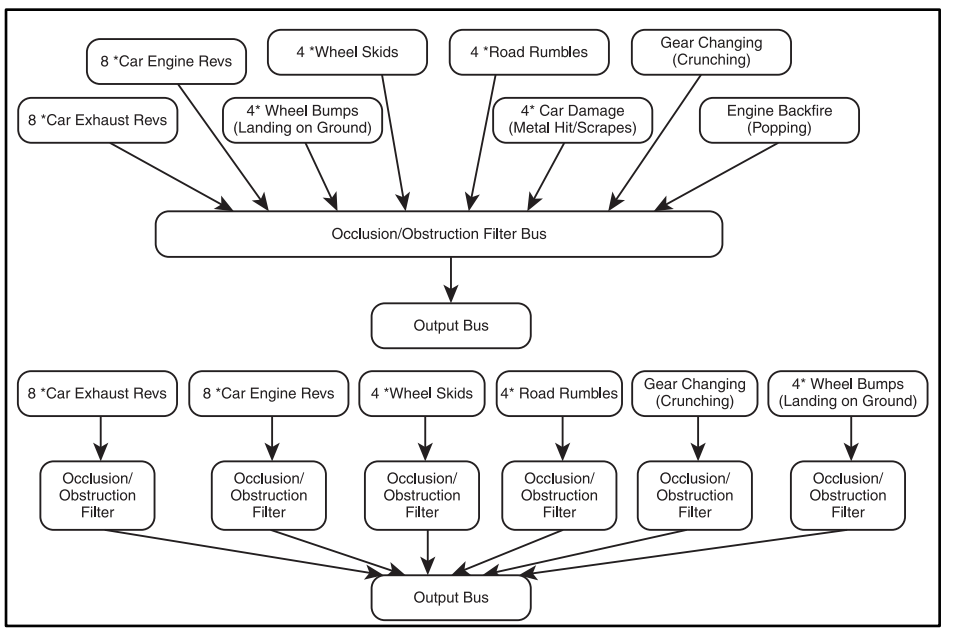


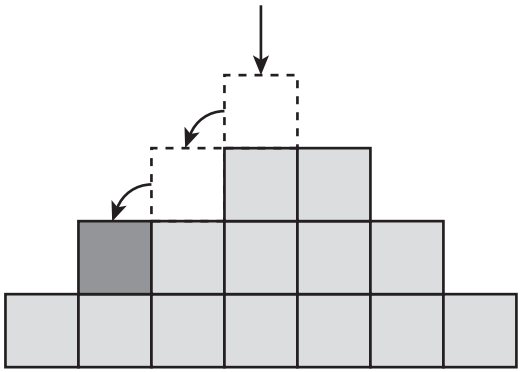


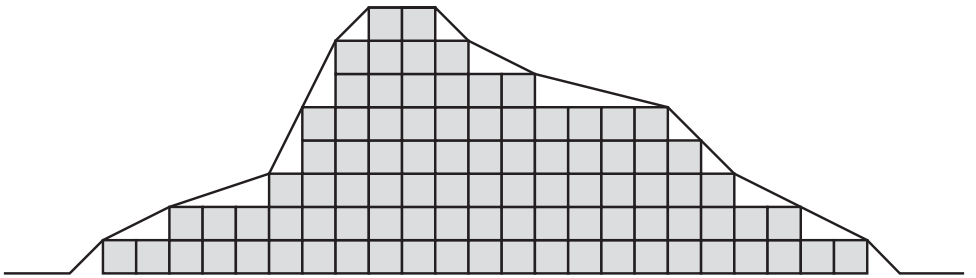
http://playground.uncc.edu/
GameIntelligenceGroup/Projects/CGUL/

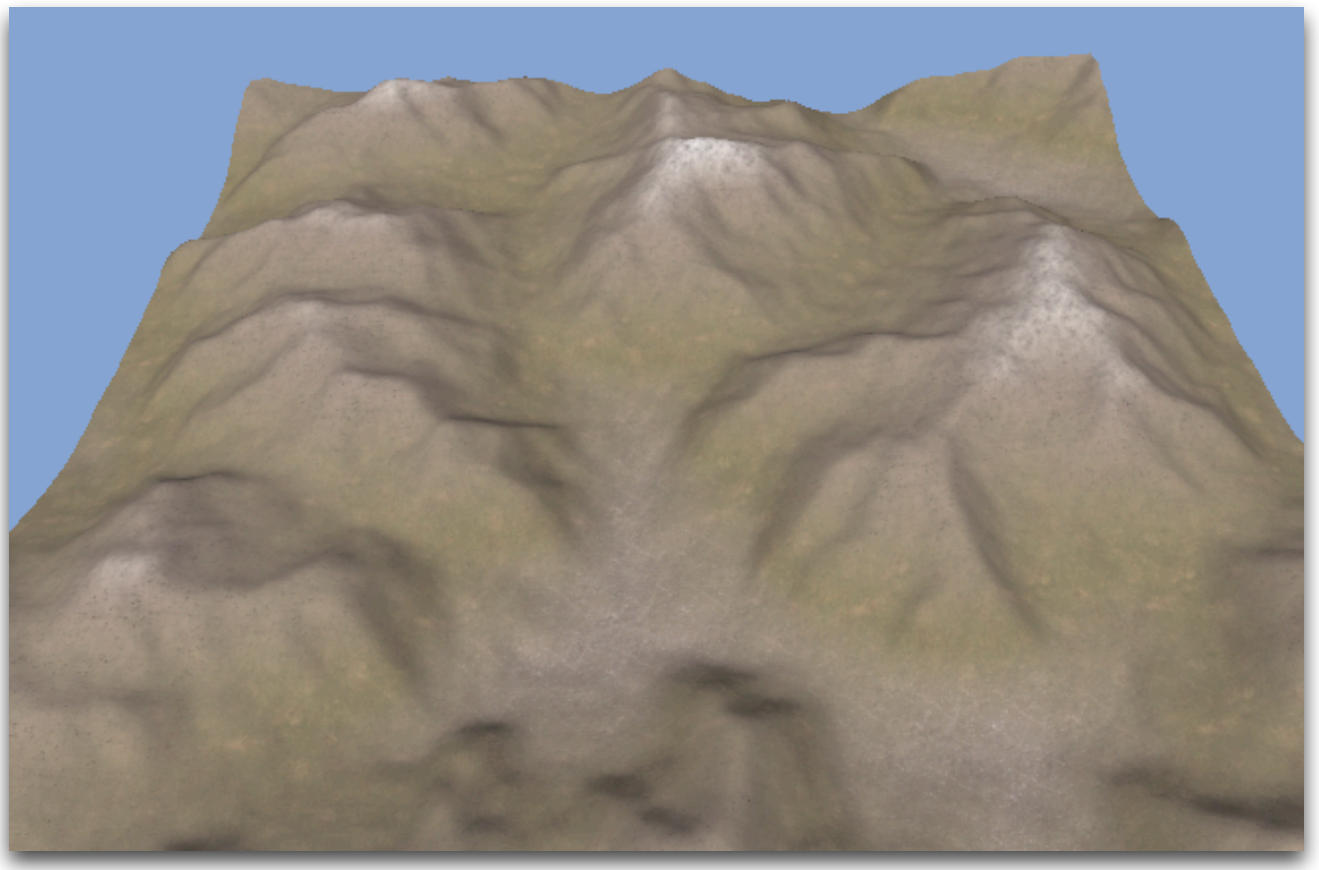
MultiStream
The Art of Writing a Next-Gen Audio Engine

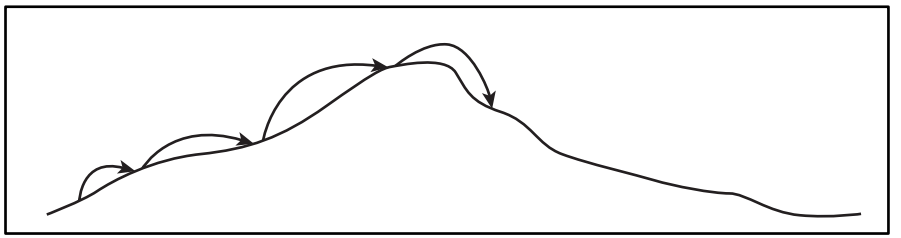


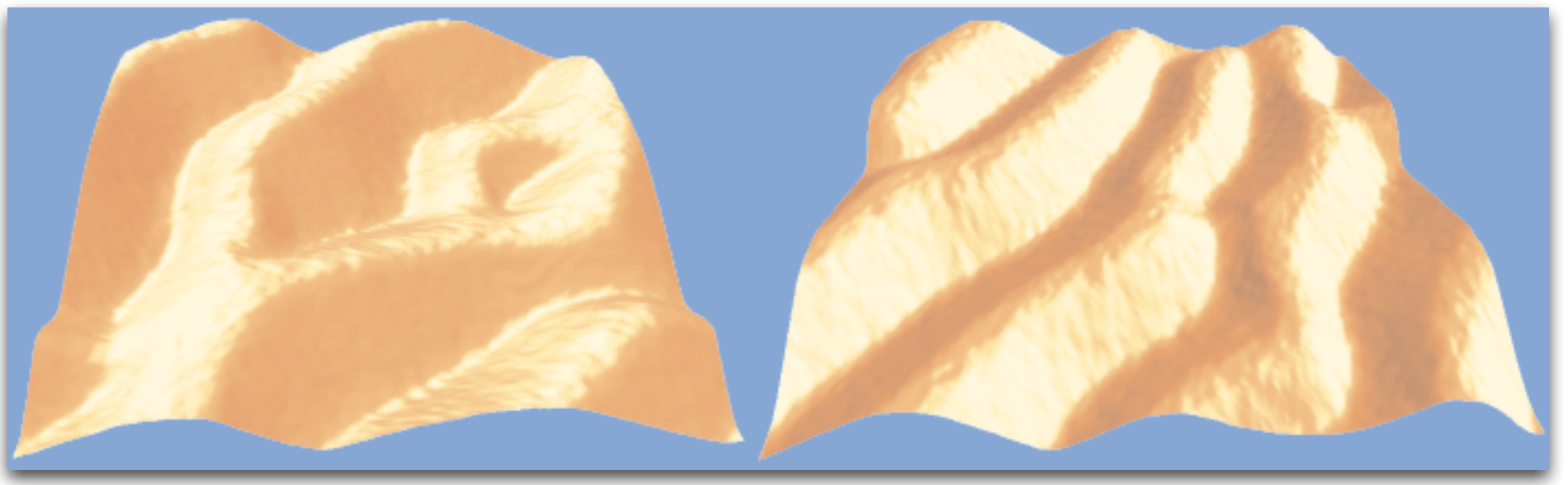


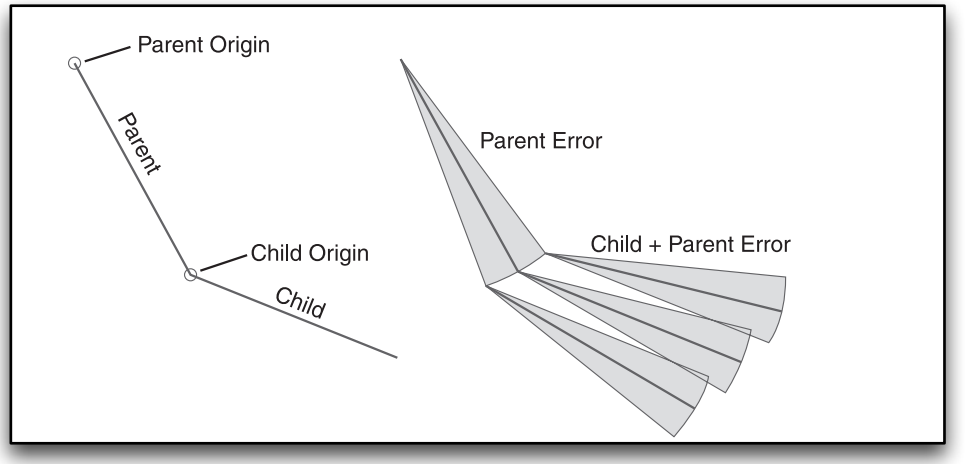


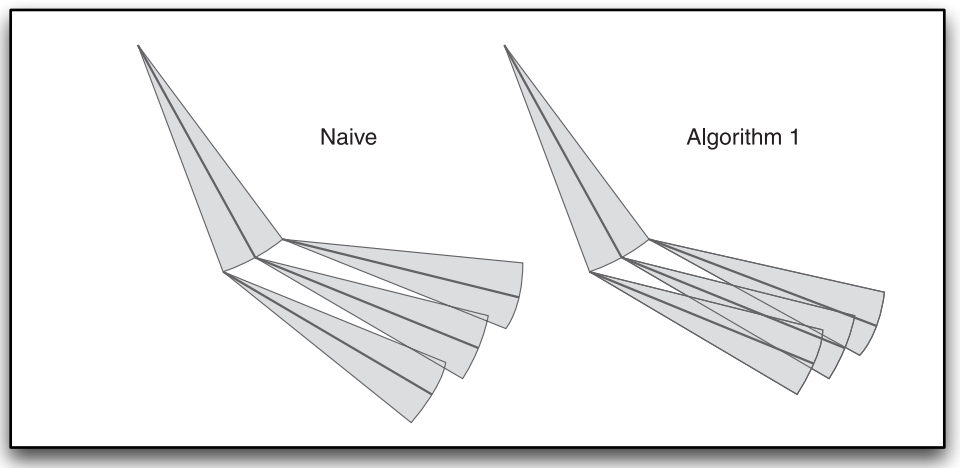


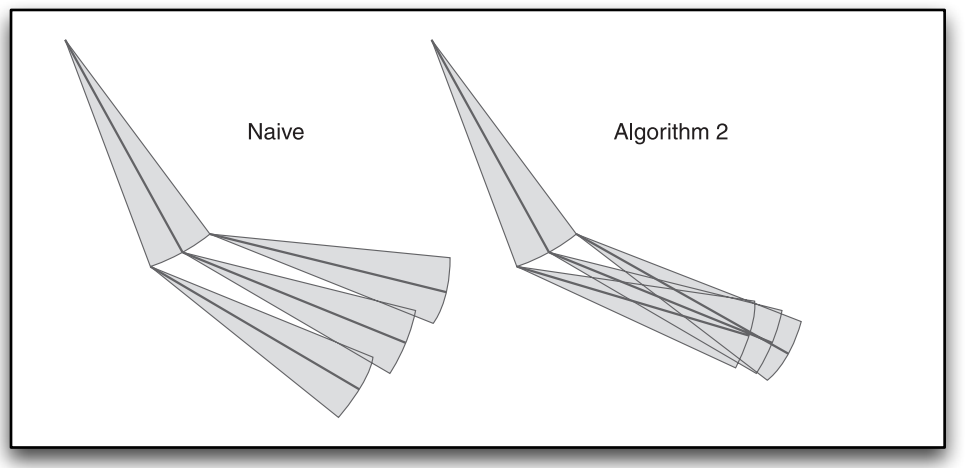


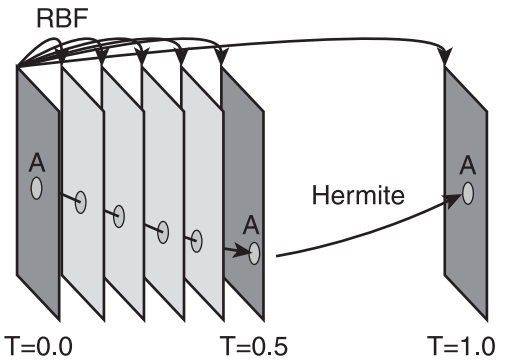


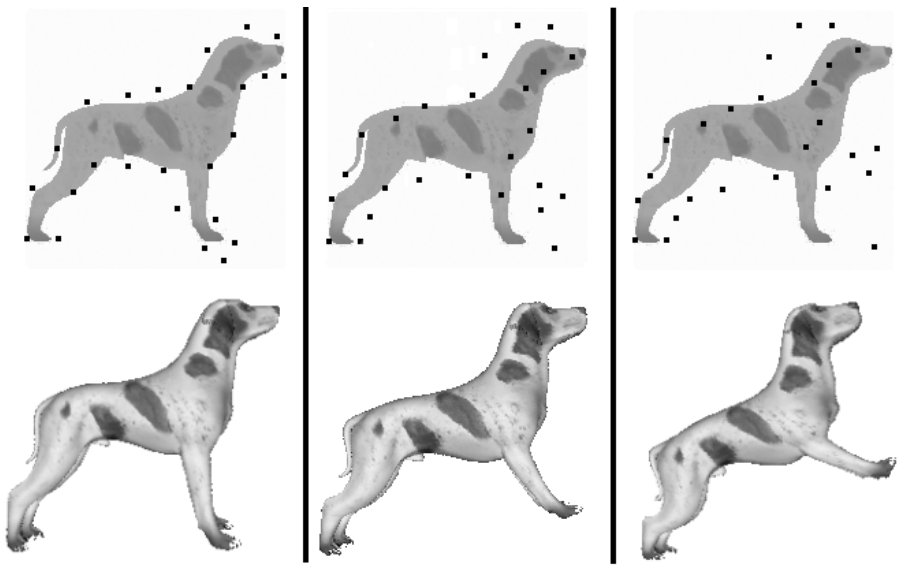



Parent Origin
Parent
Child Origin
Child
Parent Error
Child + Parent Error

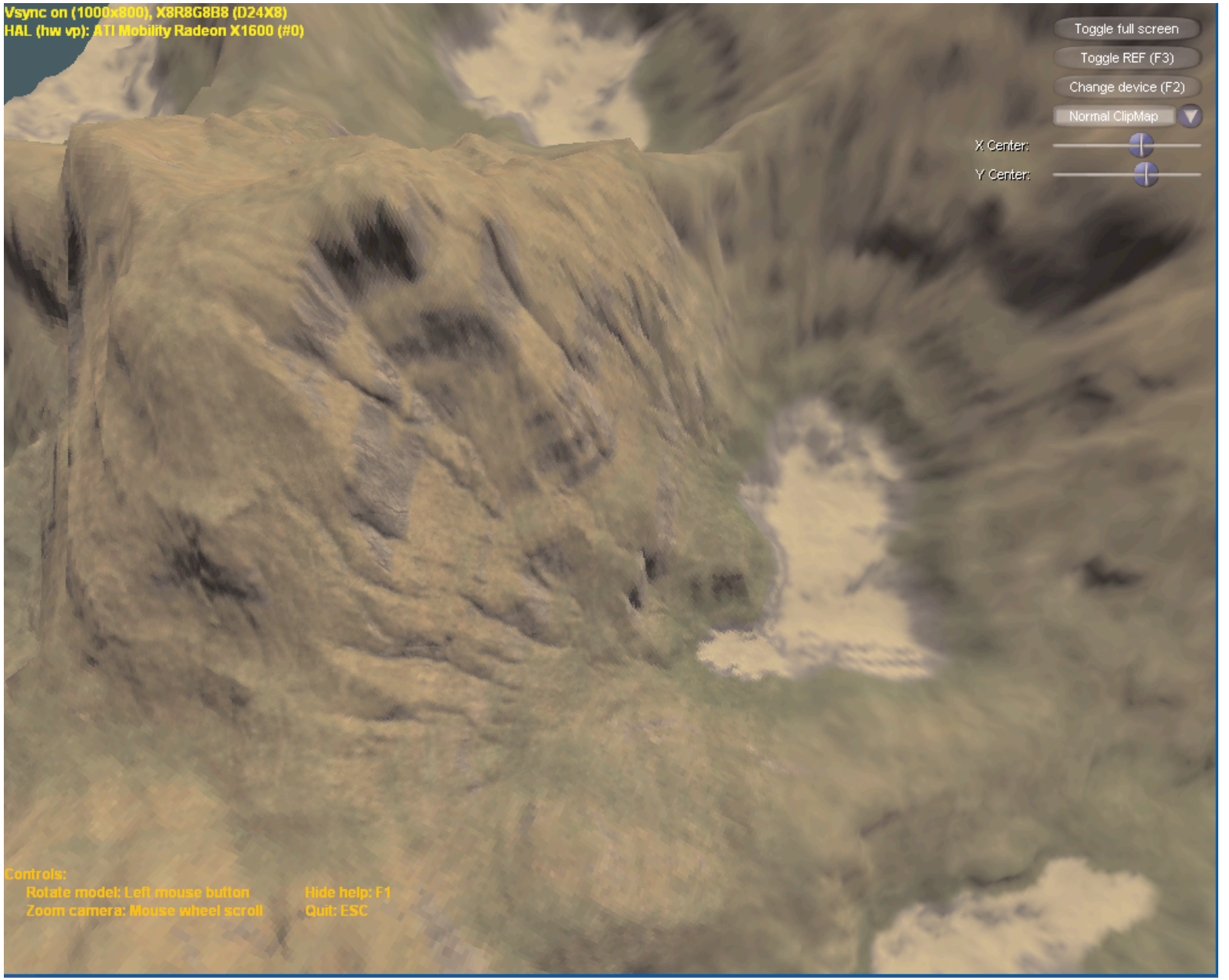

Naive
Algorithm 1

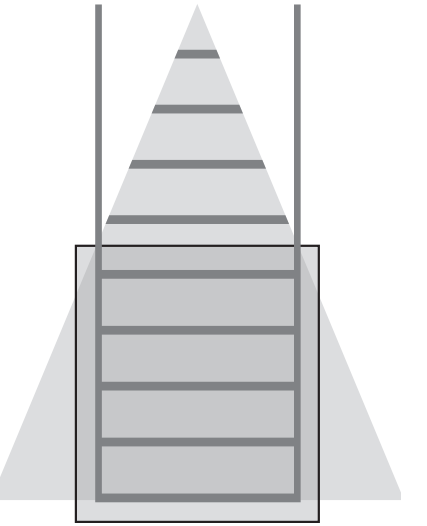

Naive
Algorithm 2


RBF
A
A
A
Hermite
T=0.0
T=0.5
T=1.0

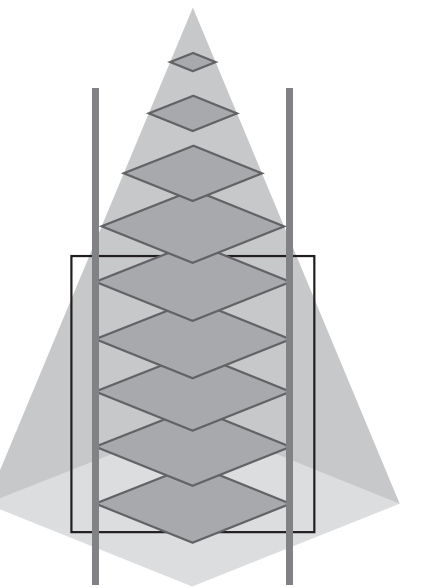


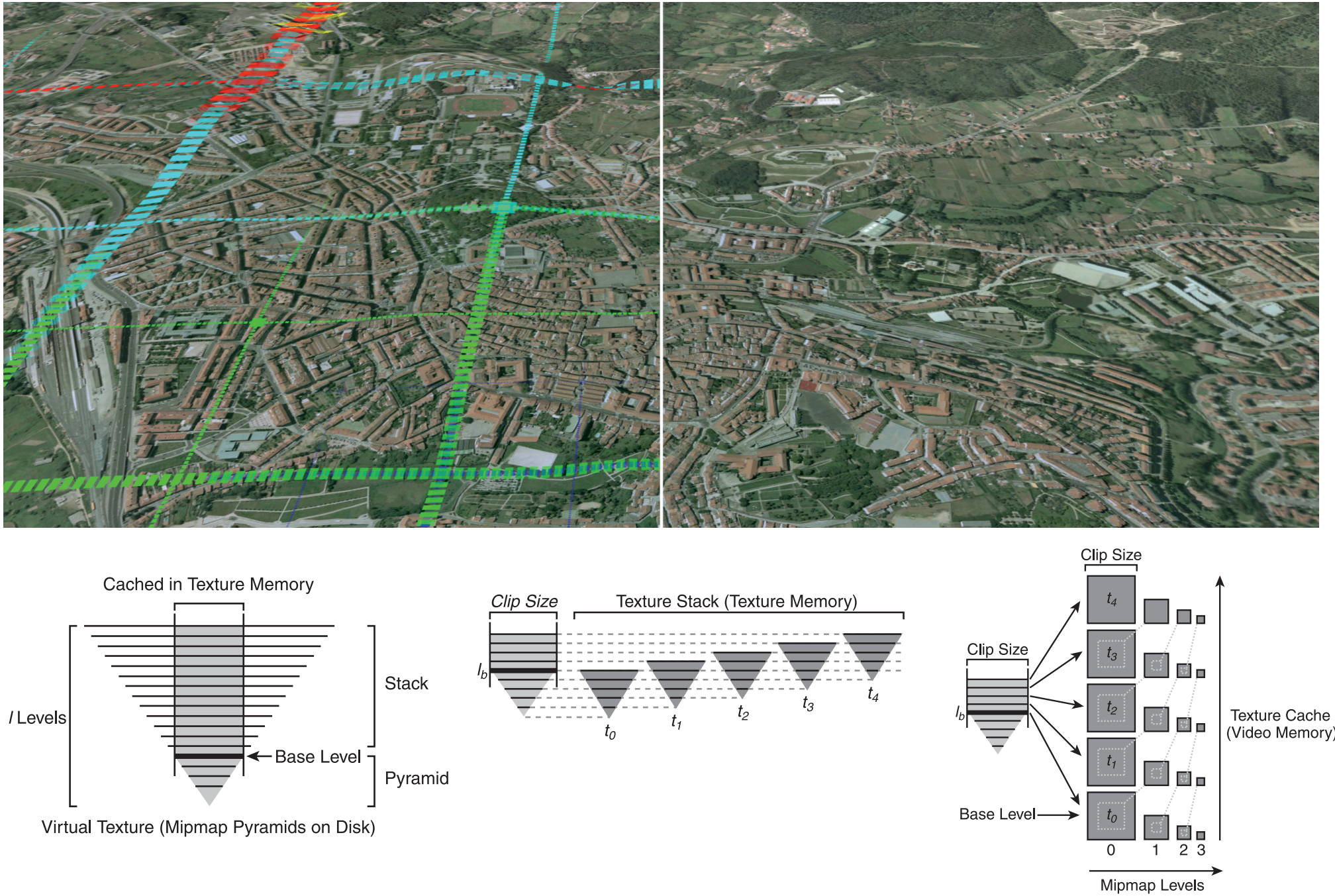

R
G
B
A

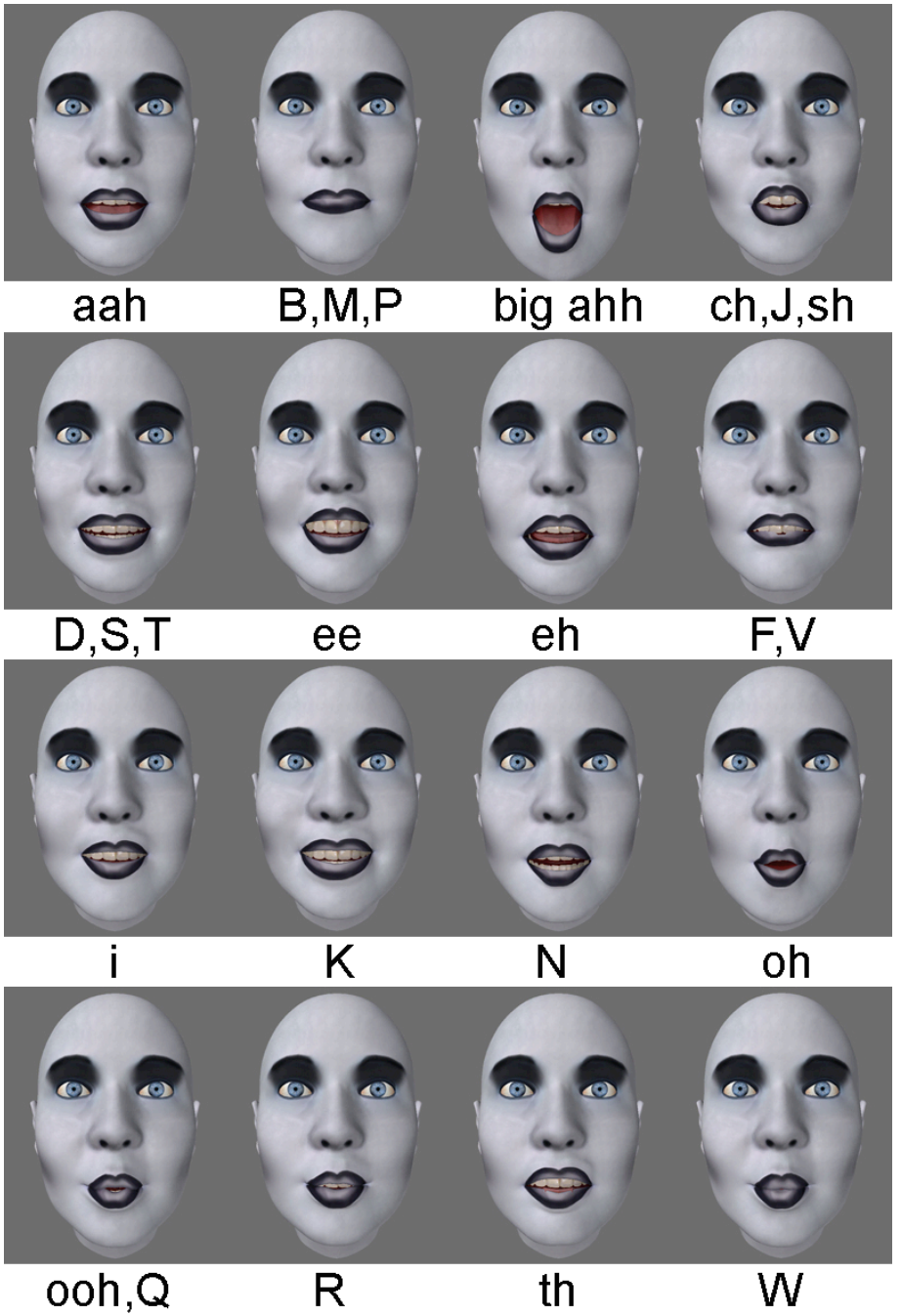




Clipstack levels




Cached in Texture Memory
Stack
l Levels
Base Level
Pyramid
Virtual Texture (Mipmap Pyramids on Disk)
Clip Size
Texture Stack (Texture Memory)
l_b
t_0
t_1
t_2
t_3
t_4
Clip Size
l_b
Base Level
Texture Cache
(Video Memory)
0
1
2
3
Mipmap Levels



Phoneme	Example	Translation
OY	Toy	T OY
P	Pee	P IY
R	Read	R IY D

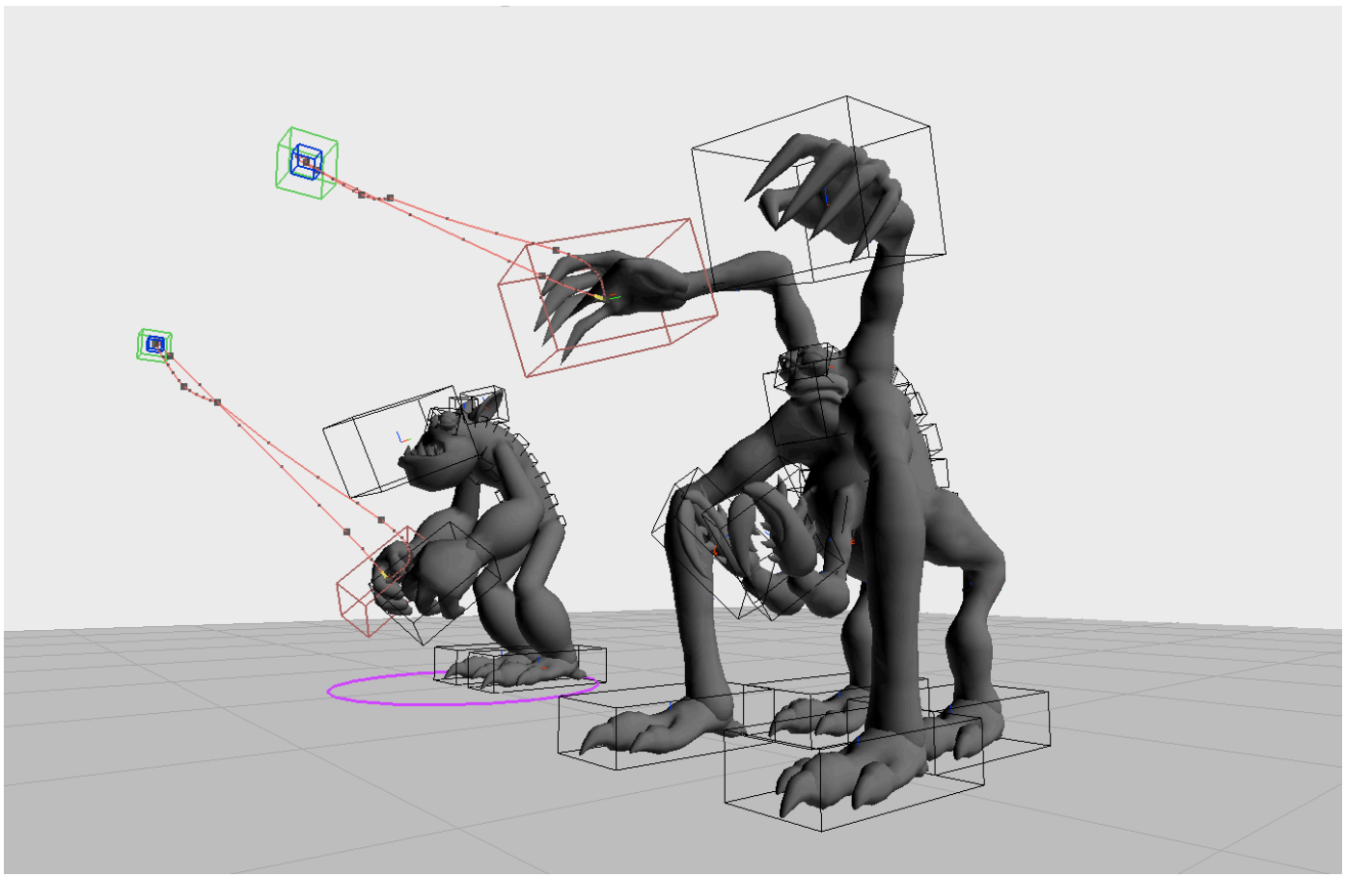
Phoneme	Viseme
AA	Big aah
CH	Ch,J,sh
ER	R

Carnegie Mellon Pronoucing Dictionary







procedural
PD
Lua

Scripting Language

Lua
Python
SQL
Flash/ActionScript

JavaScript ?

ゲーム デザイン、ターゲット ハードウェア、そして開発チーム自体も次第に大きく複雑になっているので、業界はソフトウェア開発業界の他の部分からの良いアイデアに、常に旺盛な食欲を持っていることに気が付いている。開発チームにデータベース管理者はいるだろうか？

.....

開発チームは次第に正式なプロジェクト管理と、AgileやScrumのような製作方法論を採用するようになりつつあり、我々はそこでゲーム開発外部の仲間の一般的な経験から利恩恵を得る。

.....

マルチコア マシンへのシフトにより、PCだろうと今日のゲーム機だろうと、開発者は伝統的なC/C++プログラミング言語を超えて並列性と同期の問題の解決を調べるようになり、我々が利用可能なものを知るために、HaskellやErlangのような言語を熟知した人の経験を積極的に求めている。

Exemplis discimus.