

「デジタルゲームAIと プロシージャル技術」

柔軟なデジタルゲーム・コンテンツを目指して

三宅 陽一郎

神奈川工科大学
2011.7.28
Twitter: @miyayou
#aikait

本日のコンテンツ

- (1) キャラクターAI
- (2) メタAI
- (3) プロシージャル

人工知能って何だろう？

知能ってなんだろう？

リアルワールド
(現実世界)

リアルワールド
(現実世界)



現実世界の知性

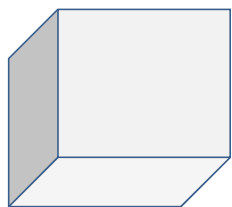


バーチャルワールド
(仮想世界)

仮想世界の知性
=人工知能

環境と知性

もし、この世界がたったひとつの白い部屋だったら、
その世界にはどんな知性が存在するだろうか？



いいや、たいてい高度な知性は存在しないだろうし、**必要ない**。

では、何が知性を
必要とさせているんだろう？

進化の歴史 = 身体と知能の歴史

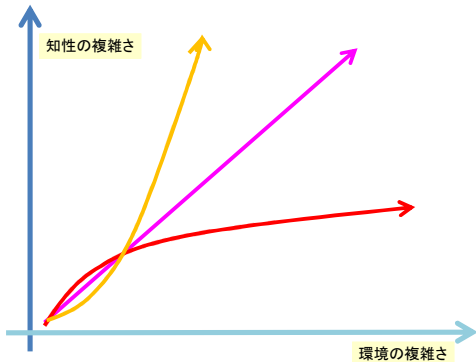


進化の歴史 = **環境**への適応の歴史

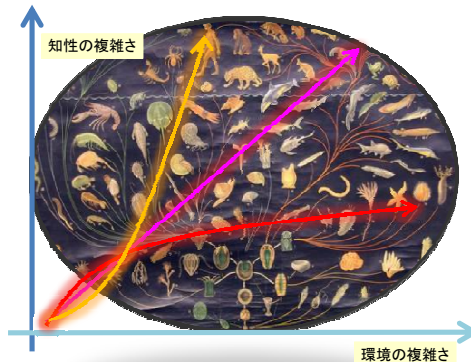
進化という長い歴史から見れば、環境が知性を育てる

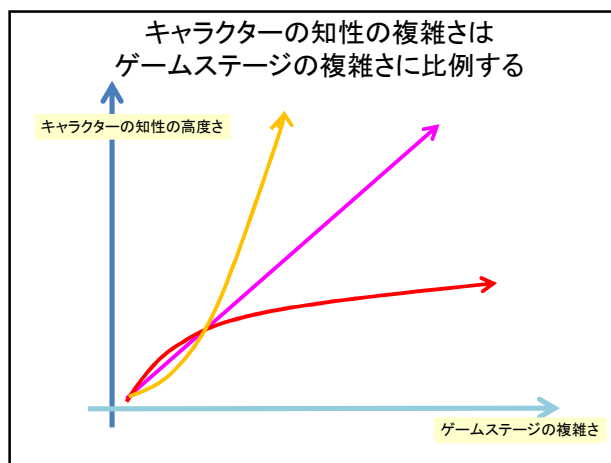
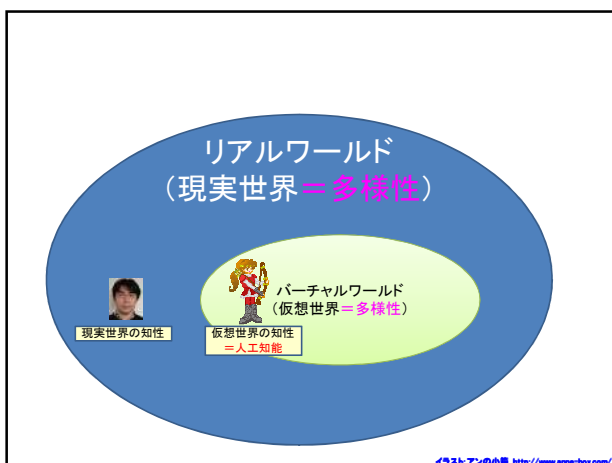
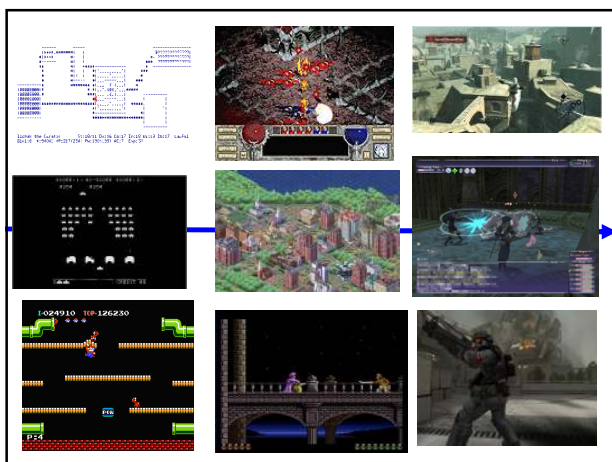
環境の複雑さ = 知性の複雑さ

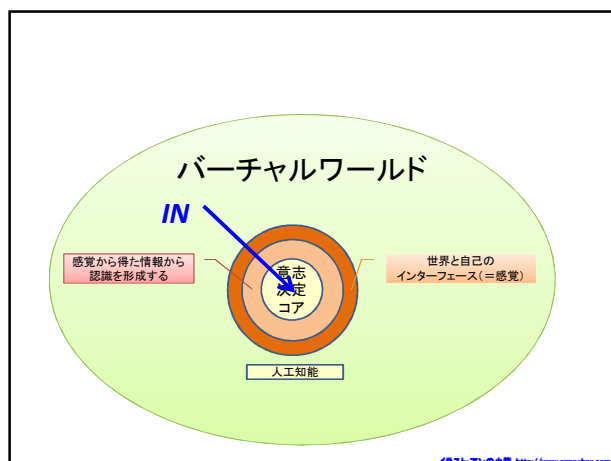
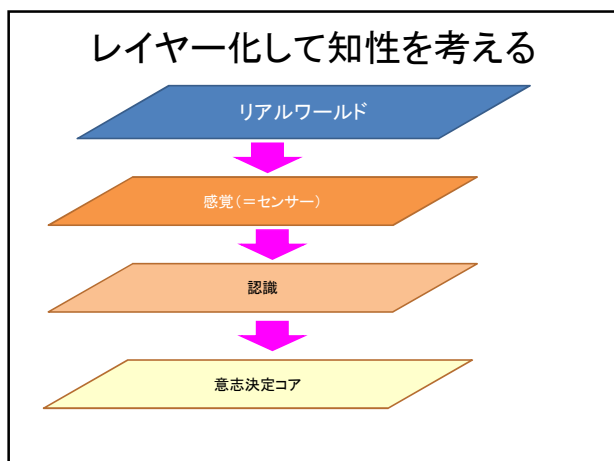
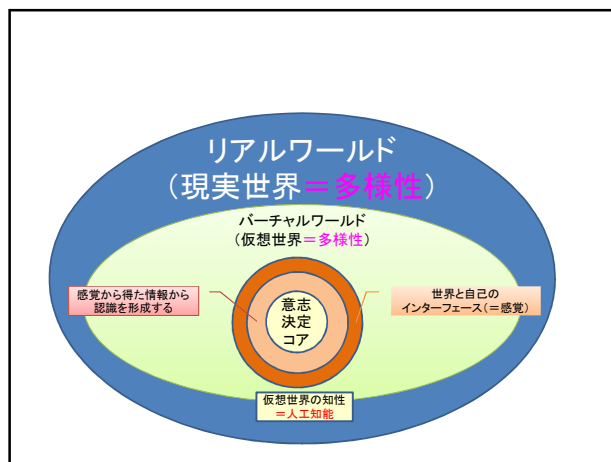
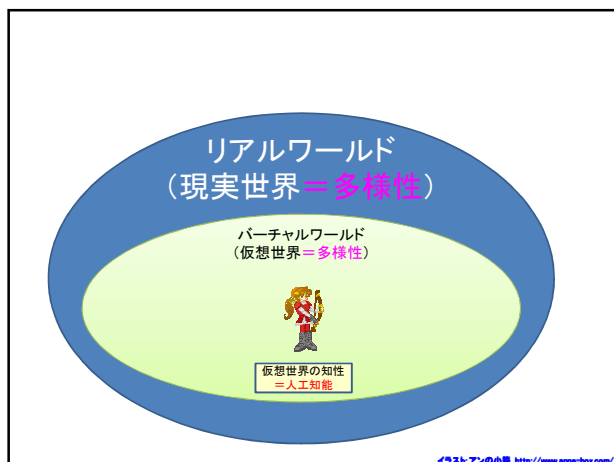
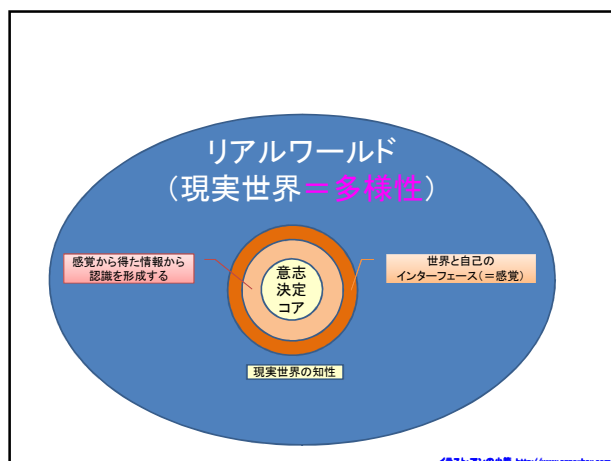
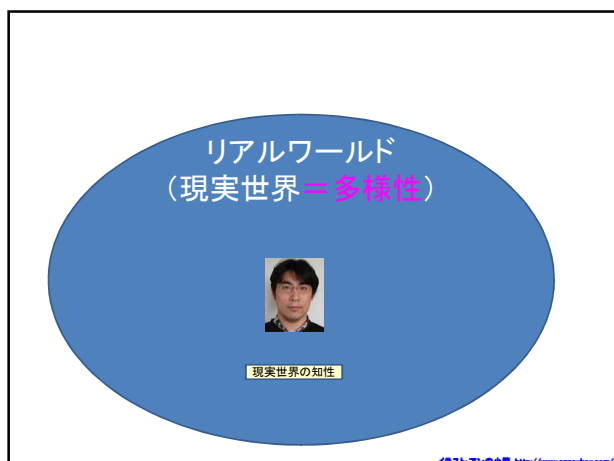
知性の複雑さは環境の複雑さに比例する

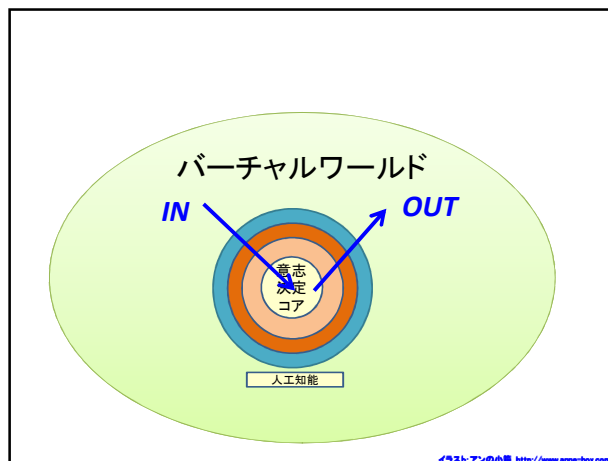
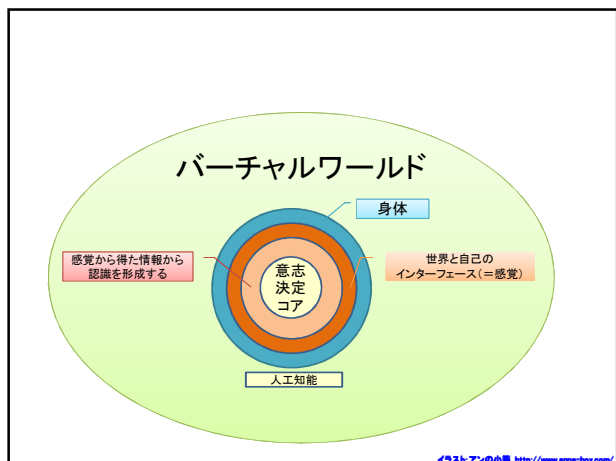
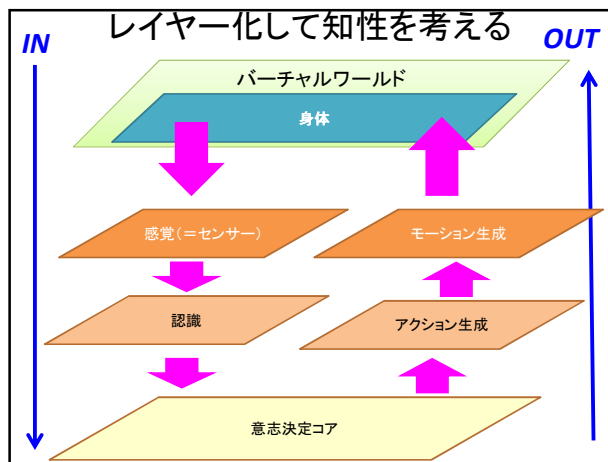
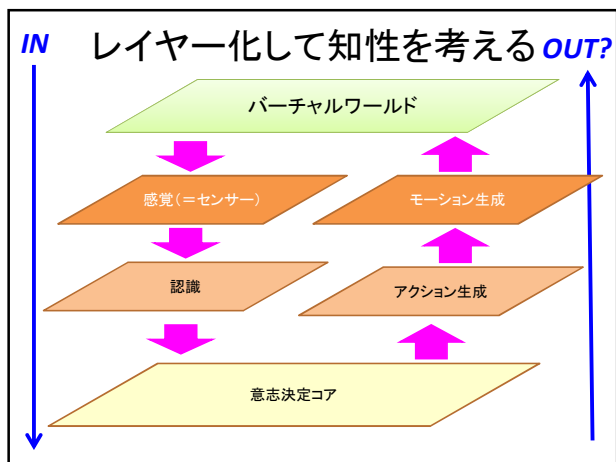
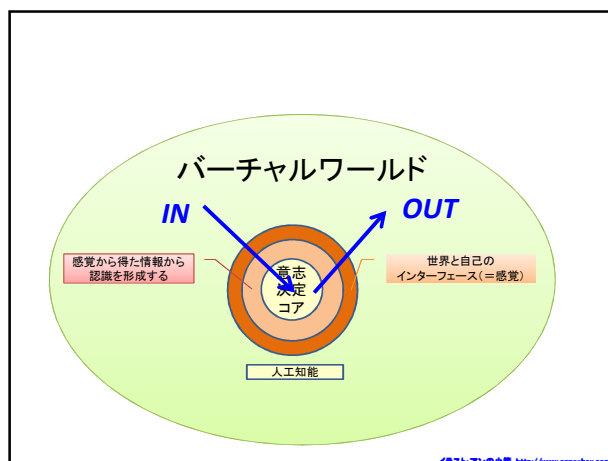
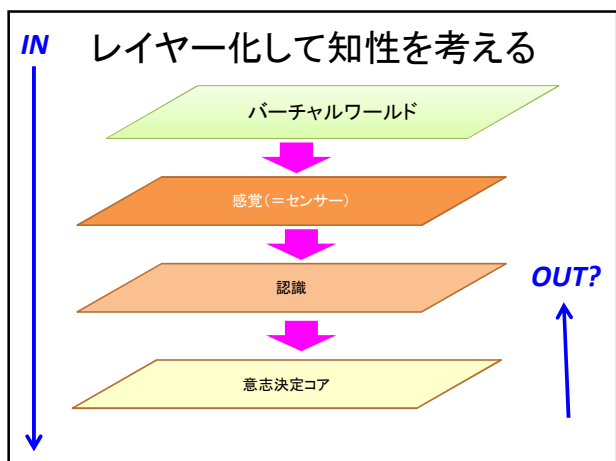


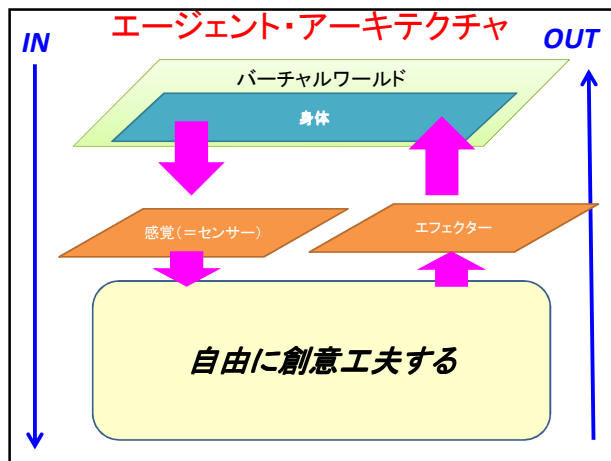
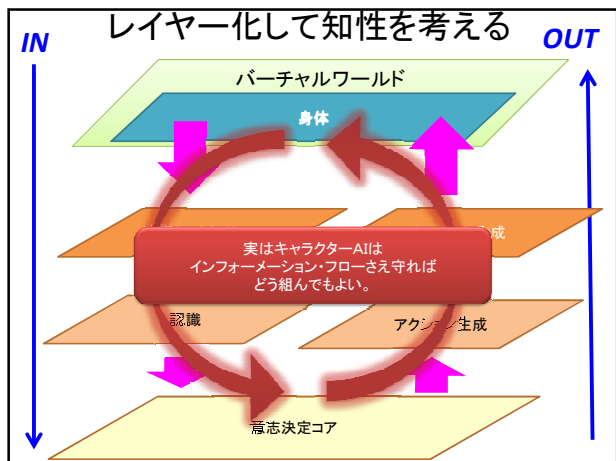
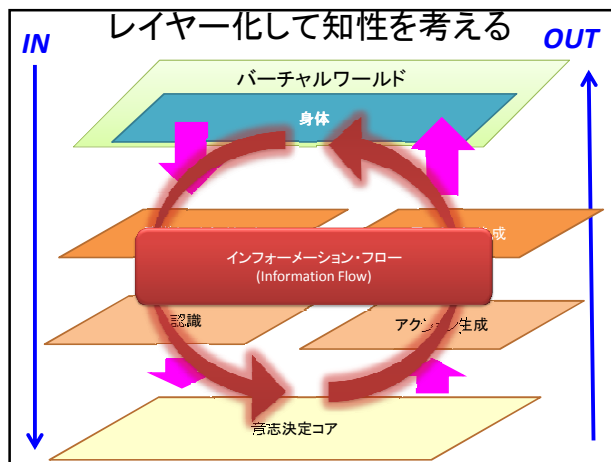
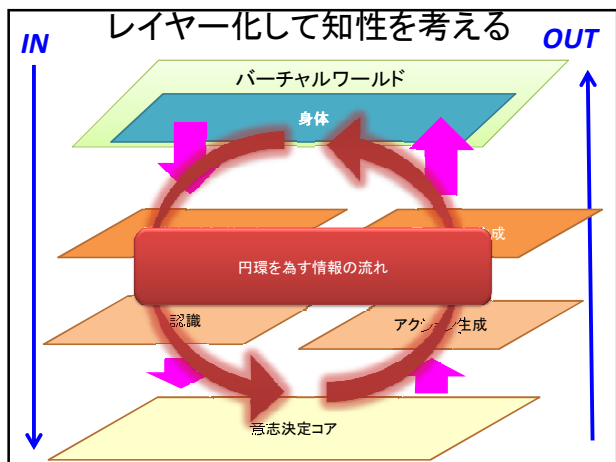
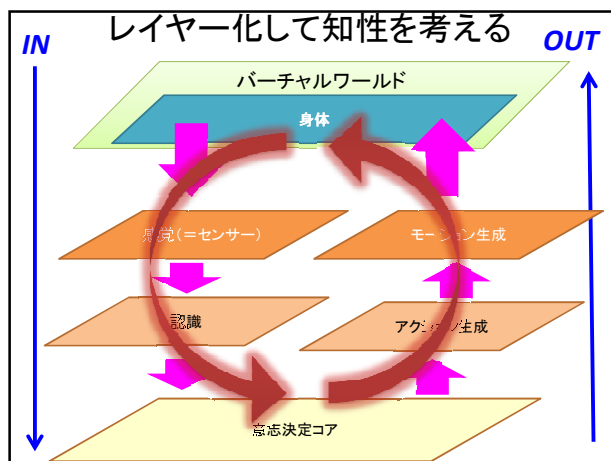
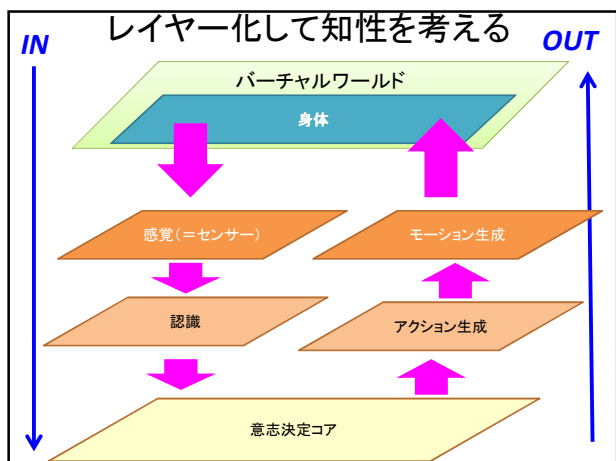
知性の複雑さは環境の複雑さに比例する

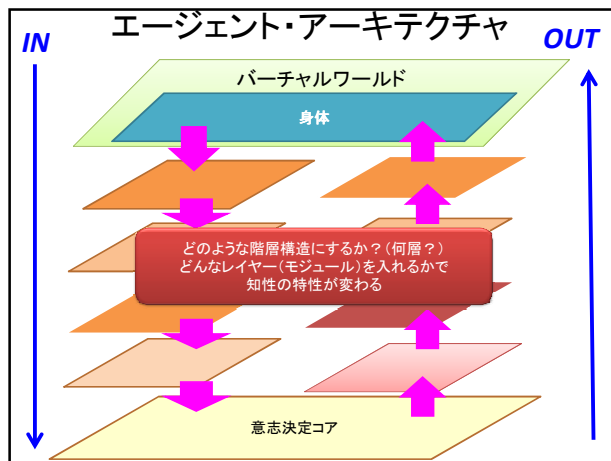
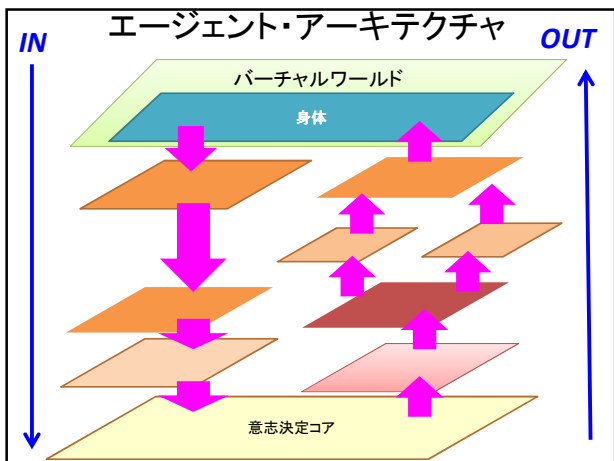












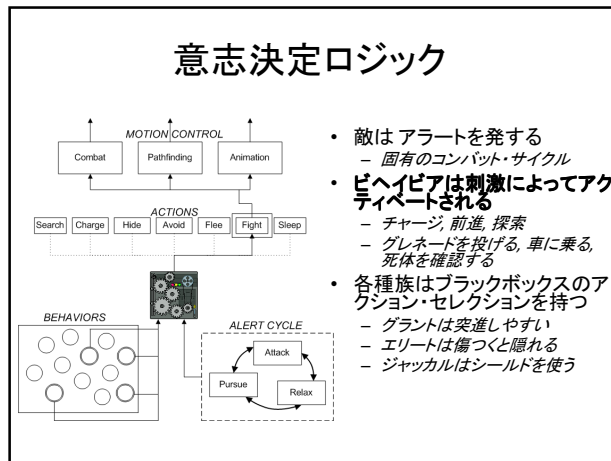
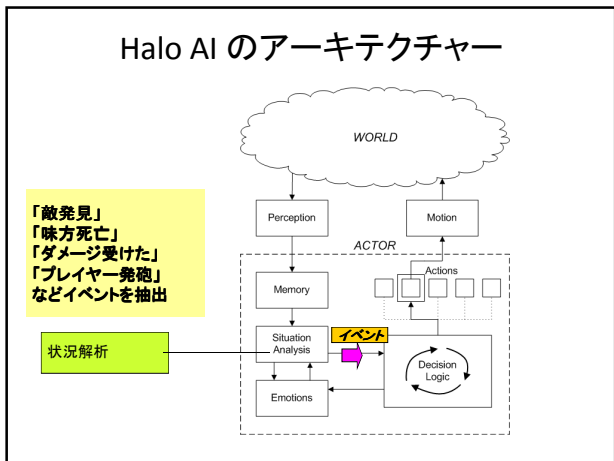
Halo

- ◆ 内容: 宇宙船や地表を舞台にしたSFのFPS
- ◆ 開発元: BUNGIE Studio
- ◆ 出版: Microsoft
- ◆ Hardware: Xbox, Windows, Mac
- ◆ 出版年: 2002年

Xbox, 全米、世界を代表するFPSの一つ(Halo 500万本 Halo2 700万 圏内10万本)
 「愛嬌のあるNPC」とその演出で、プレイヤーからの定評を得る。

Halo NPC

グラント	ジャッカル	エリート	人間
ちょこまかと動き回る。愛嬌がある。	手堅い。	大型。	普通の人間。
敵(コグナント)			味方



デモ



Reference for Halo & Halo2

- Damian Isla (2005), "Dude, where's my Warthog? From Pathfinding to General Spatial Competence", <http://www.aiide.org/aiide2005/talks/isla.ppt>
http://nikon.bungie.org/misc/aiide_2005_pathfinding/index.html
- Damian Isla (2005), Handling Complexity in the Halo 2 AI, Game Developer's Conference Proceedings., http://www.gamasutra.com/gdc2005/features/20050311/isla_01.shtml
- Jaime Griesemer(2002),The Illusion of Intelligence: The Integration of AI and Level Design in Halo, <http://halo.bungie.org/misc/gdc.2002.haloai/talk.html>
- Robert Valdes(2004), "In the Mind of the Enemy The Artificial Intelligence of Halo2", <http://www.stuffo.com/halo2-ai.htm> (現在はclosed)

Figures on following pages are from these references.

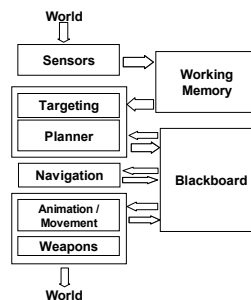
F.E.A.R



内容:閉鎖空間の中のホラーFPS
開発元: Monolith Production
出版: SIERRA
Hardware: Windows, PS3
出版年: 2004年

FPSとホラーを、映画的な演出によって結びつけたエポックメイキングな名作。長年発展させて来たAI技術の本領が発揮され、開発者、プレイヤーから高い支持を集める。

F.E.A.R. におけるエージェント・アーキテクチャ

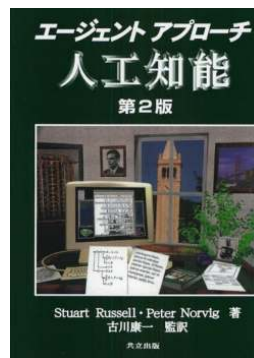


Agent Architecture Considerations for Real-Time Planning in Games (AIIDE 2005)
http://web.media.mit.edu/~jorkin/AIIDE05_Orkin_Planning.ppt

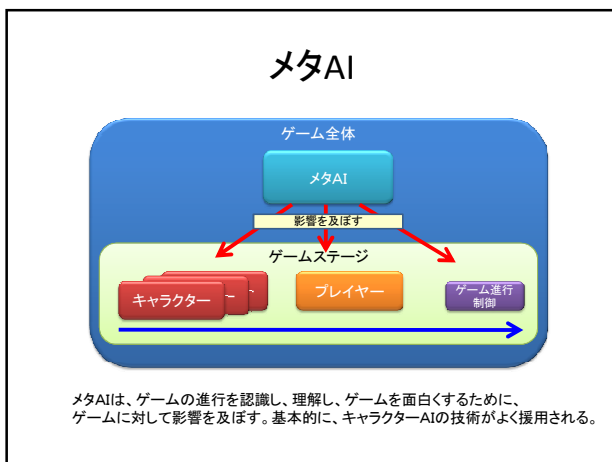
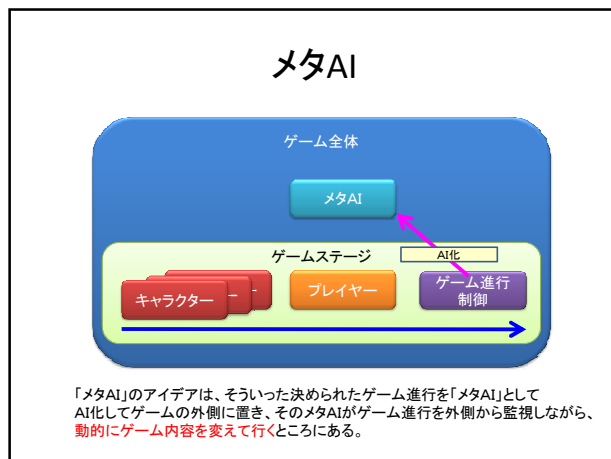
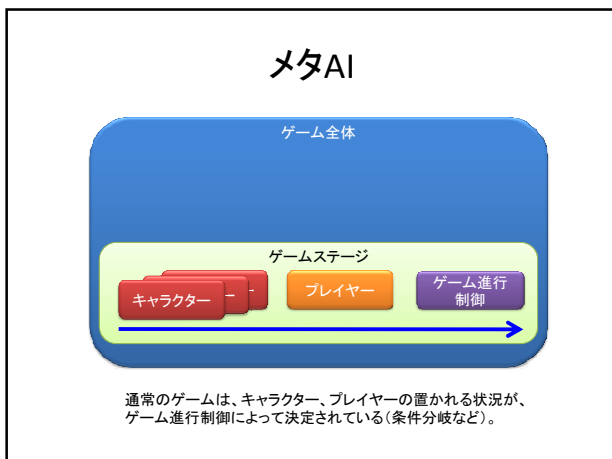
まとめ

- (1) 知性は環境を反映する。
- (2) 知性の構造は階層化レイヤーとして捉えることができる。
(⇔ サブサンプリング・アーキテクチャ)
- (3) 環境と知性を明確に分けるアーキテクチャを**エージェント・アーキテクチャ**という。
- (4) エージェント・アーキテクチャでは、環境と知性の間で**インフォメーション・フロー**が形成される。
- (5) インフォメーション・フロー上にどのようなモジュールを配置するかで、知性の質が変化する(=個性化)
- (6) どのようなモジュールがあることを知っているか？(キャラクターAIの知識、人工知能の知識)、それをどう組み合わせればどのような効果が得られるか？(ノウハウ)を習得することが、キャラクターAIを作成方法をマスターするという事。

参考文献



エージェント(=キャラクターAI)を作成する知識が編纂されている書籍。



Left 4 Dead の例

- ◆ 4人で行う協力型オンラインゲーム
- ◆ 街中でゾンビと戦う
- ◆ プラットフォーム: Windows/Xbox360
- ◆ 開発会社: Valve
- ◆ リリース: 2008.11.18

資料: <http://www.valvesoftware.com/publications.html>

Left 4 Dead の例

「Counter Strike」の世界的成功

自分たちはなぜ成功したかを徹底的に考察

それは緩急があったからだ。

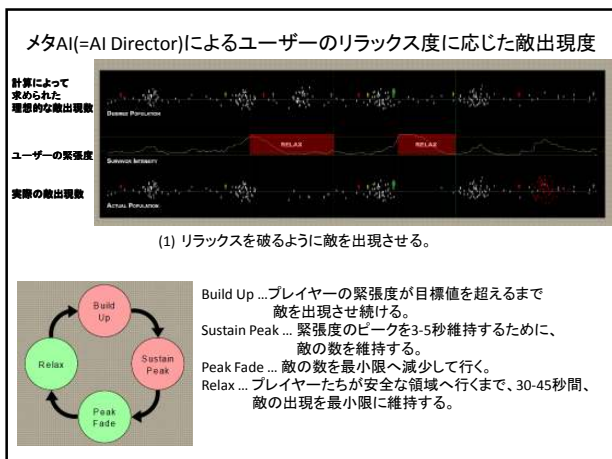
Counter Strike の「緩急＝ペース」はある程度、偶然に作られたかもしれない。では、そういったゲームが面白くなる最適なペースを、人工的に常に作り出すことはできないだろうか？

→ メタAI (=AI Director)によってペースを作る。

適応型動的ペーシング

[基本的発想]

- (1) ユーザーがリラックスしている時に、ユーザーの緊張度が一定の敷居を超えるまで敵をぶつけ続ける。
- (2) ユーザーの緊張度が一定の緊張度を超えると敵を引き上げる。
- (3) リラックスすると敵を出現し始める((1)へ)。



メタAIがゲームを認識する方法

= キャラクターAIが環境を認識する方法

キャラクター用に作成されたナビゲーションメッシュをメタAIがゲームの進行を認識するために使用する。

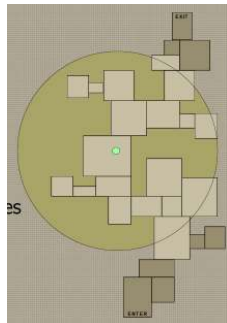


安全な領域までの道のり(Flow Distance)

プレイヤー群の経路をトレース・予測
- どこへ来るか
- どこが背面になるか

AAS に対して行うこと。

メタAIは AAS に移動に伴い、敵の群れを生成・消滅させたりする。

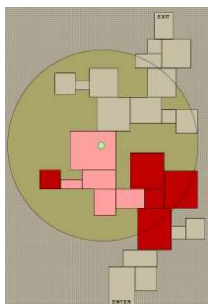


プレイヤーからの可視領域

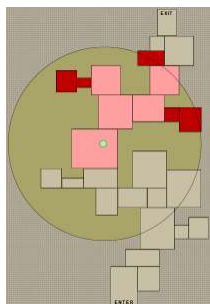
可視領域では、例えば、敵のスパウニング(発生)などはできない。



敵出現領域



背後



前方

高頻度

Goal: Promote Replayability
Procedurally Populated Environments

- Structured Unpredictability in Left 4 Dead
 - Wanderers (high frequency)
 - Common Infected that wander around in a daze until alerted by a Survivor
 - Mobs (medium frequency)
 - A large group (20-30) of enraged Common Infected that periodically rush the Survivors
 - Special Infected (medium frequency)
 - Individual Infected with special abilities that harass the Survivor team
 - Bosses (low frequency)
 - Powerful Infected that force the Survivors to change their strategy
 - Weapon Caches (low frequency)
 - Collections of more powerful weapons
 - Scavenge Items (medium frequency)
 - Pipe bombs, Molotovs, Pain Pills, Extra Pistols

低頻度

敵の種類、アイテムの種類ごとに出現頻度が違う。

ボス出現アルゴリズム

- (1) N体を予想される逃走経路上に配置
- (2) 3つのイベントパターン (何もいない, を含む) (例) Tank, Witch, 何もいない
- (3) 同じパターンのくり返しは禁止 (例) Witch, 何もいない, Witch はOK。 Witch, Witch はだめ。

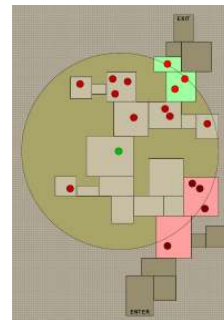


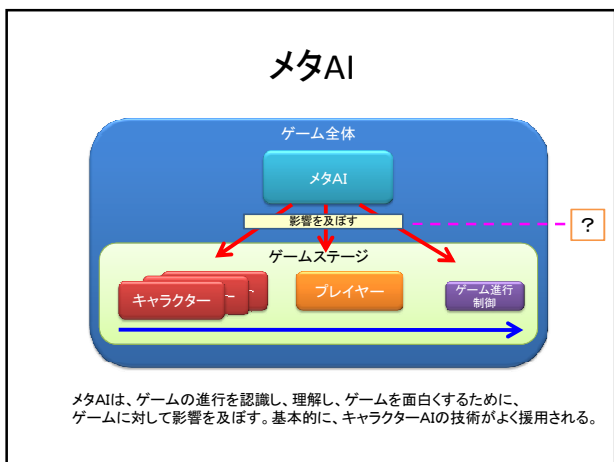
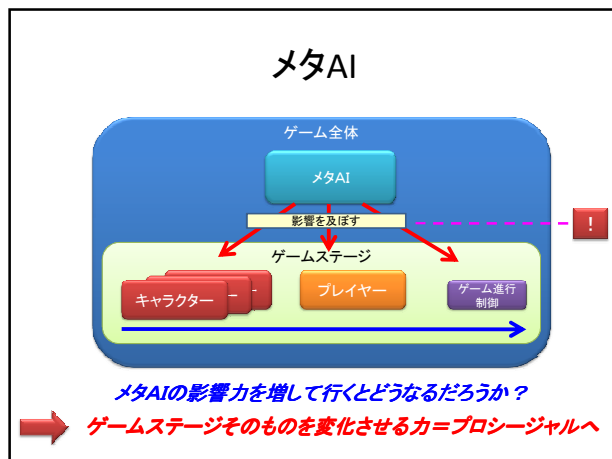
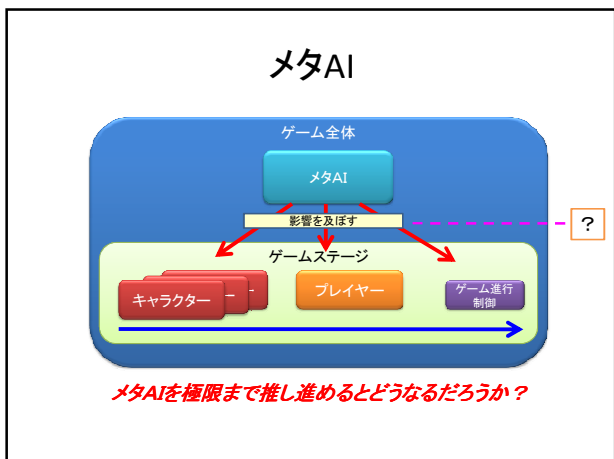
何もいない



具体的なアルゴリズム

- (1) 各エリアに、出現数 N を決定する
- (2) 出現数 N は予想される逃走経路の長さや要求される密度によって計算される。
- (3) あるエリアが AAS の中に入るとクリーチャーが N 体生成される
- (4) そのエリアが AAS の外に出ると生成が中止され、クリーチャーは消滅される。
- (5) N はそのエリアがプレイヤーから見えている場合、或いは、プレイヤーがリラックスモードの場合には、強制的に 0 になる。





プロシージャル技術って何ですか？

くぐってみよう！

- コンテンツを計算によって生成する技術。
- 或いは、ある程度のデータから、新しいコンテンツを生成する技術。



(例) Battlefield 2



- ◆ オフライン/オンラインゲーム
- ◆ 広大な大自然の中でFPS
- ◆ プラットフォーム: Xbox360/PS3/PC
- ◆ 開発会社: EA/Dice リリース: 2010.12.2

SIGGRAPH 2007



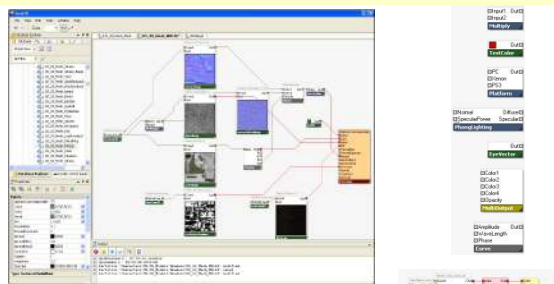
- ① オブジェクト生成
(We can generate objects with procedural techniques
-Then use rules to deform / destroy / modify / move them
-Better interactivity)
- ② Semi-procedural surface shader
- ③ Procedural shader
- ④ Procedurally distributed on the fly



GDC 2007 Frostbite "Rendering Architecture and Real-time Procedural Shading & Texturing Techniques"
http://developer.amd.com/assets/Anderson_Tatarchuk_FrostbiteRenderingArchitectureGDC07_AMD_Session1.pdf
 GDC 2007 "The Importance of Being Noisy: Fast, High Quality Noise", N. Tatarchuk
http://developer.amd.com/Assets/Tatarchuk_NoiseGDC07_D3D_Day1.pdf
 SIGGRAPH 2007 Johan Andersson "Terrain Rendering in Frostbite using Procedural Shader Splatting"
[http://ati.amd.com/developer/gdc/2007/Andersson-TerrainRendering\(Siggraph07\).pdf](http://ati.amd.com/developer/gdc/2007/Andersson-TerrainRendering(Siggraph07).pdf)

Semi-procedural Surface Shader

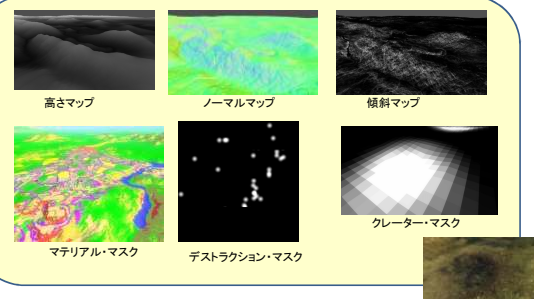
表面状態などに応じてシェーダーが形成される(グラフベースでアーティストも作成可能)



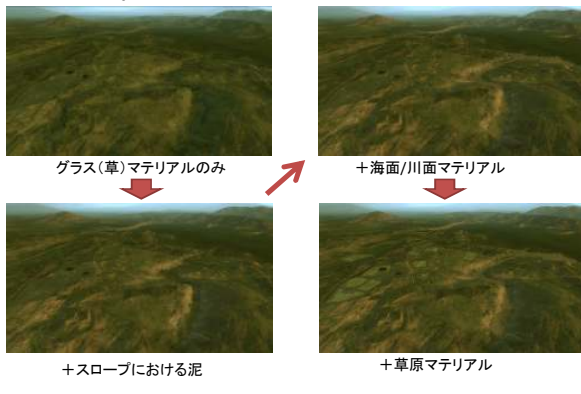
- 組み込み可能ノード
- 基本演算 (mul, add, div)
 - 地形条件 (fresnel, refraction)
 - ロジカル (プラットフォームなど)
 - パラメータ (スカラー, ベクトル, ブール)

- パラメータ (Z, Normal, eye-vector)
- ライトニング (phong, sub-surface)
- ルート (general, offset, multi-output)
- その他 (script, curve)

プロシージャルに生成するシェーダー用のマップ・マスク



Semi-procedural Surface Shader 実例



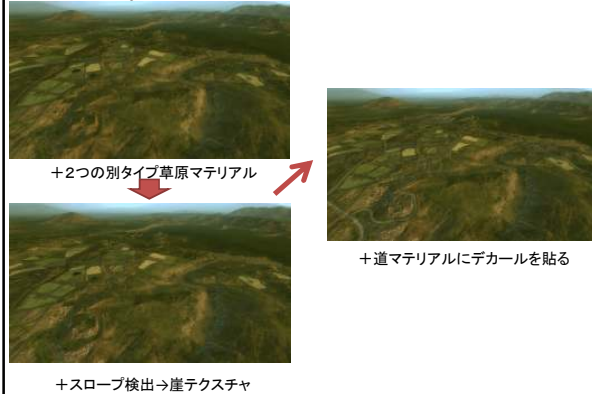
グラス(草)マテリアルのみ

+海面/川面マテリアル

+スロープにおける泥

+草原マテリアル

Semi-procedural Surface Shader 実例




+2つの別タイプ草原マテリアル

+道マテリアルにデカールを貼る

+スロープ検出→崖テクスチャ

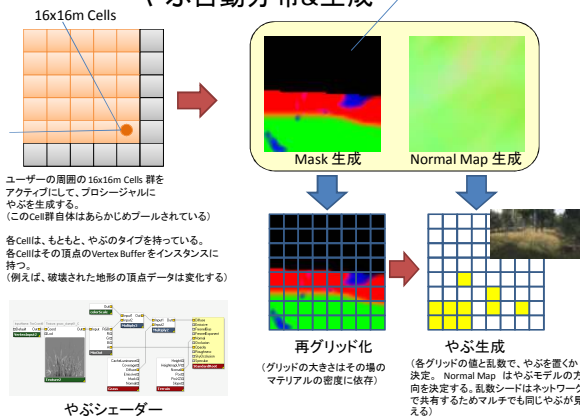
やぶ自動分布&生成



Terrain Rendering in Frostbite Using Procedural Shader Splatting, Johan Andersson
<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1.1.161.89798&rep=rep1&type=pdf>

やぶ自動分布&生成

黒 = やぶ生成なし



ユーザーの周囲の 16x16m Cells 群をアクティブにして、プロシージャルにやぶを生成する。(このCell群自体はあらかじめプールされている)

各Cellは、もともと、やぶのタイプを持っている。各Cellはその頂点のVertex Bufferをインスタンスに持つ。(例えば、破壊された地形の頂点データは変化する)

再グリッド化 (グリッドの大きさはその場のマテリアルの密度に依存)

やぶ生成 (各グリッドの値と乱数で、やぶを置くか決定。Normal Map はやぶモデルの方向を決定する。乱数シードはネットワークで共有するためマルチでも同じやぶが見える)

やぶシェーダー

地形自動生成 & 自動雪シェーダー



Snow Coverage

- Snow coverage determined procedurally
- Artists paint slope's control points as vertex colors
- Slope used for smoothstep to define snow masks
 - Mask 1: smoothstep based on `geometric.normal.x` component
 - Mask 2: smoothstep based on `normal.map.z` component
 - Final snow coverage mask = Mask1 * Mask2

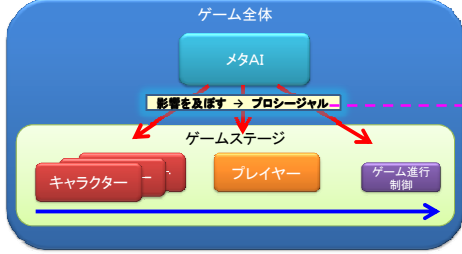
- シェーダーでマルチ分解能の Displacement map で山脈生成
 - 乱数は fBM (fractal Brownian Motion)



- 地形マップのノーマルマップ
 - パンプマップのノーマルマップから Snow Coverage を計算

The Importance of Being Noisy-Fast, High Quality Noise, Natasya Tatarchuk
http://developer.amd.com/articles/Tatarchuk-Noise/0007_0010_0001.pdf

メタAI × プロシージャル技術



ゲーム全体

メタAI

影響を及ぼす → プロシージャル

ゲームステージ

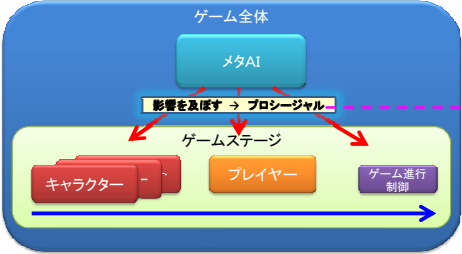
キャラクター プレイヤー ゲーム進行制御

メタAIの影響を増して行くとどうなるだろうか?

→ ゲームステージそのものを変化させる力 = プロシージャルへ

→ 世界そのものをリアルタイムで書き換えて行く力を手に入れる

メタAI × プロシージャル技術



ゲーム全体

メタAI

影響を及ぼす → プロシージャル

ゲームステージ

キャラクター プレイヤー ゲーム進行制御

メタAIの影響を増して行くとどうなるだろうか?

→ ゲームステージそのものを変化させる力 = プロシージャルへ

→ 世界そのものをリアルタイムで書き換えて行く力を手に入れる

デジタルゲームデザイン、未踏の領域

ゲームの行方

3者に共通するもの

- キャラクターAI → その場の状況に応じて
- メタAI → 進行に合わせてゲームを変化
- プロシージャル → その場でコンテンツを生成してゲームを展開

「人力による巨大固定コンテンツ」から
 「高い技術による柔軟なコンテンツ」へ

[私的コメント]
 海外は残虐な戦闘型ゲームが多いが、これらの技術を活かした日本型の優しいゲームもあるはずだ。

ゲームAIラウンドテーブル・オン・ツイッター

- 毎月1回
- テーマを一つ決めて議論
- ツイッターアカウントがあれば誰でも参加できる。
- 司会: @miyayou or @hudepen
- 議論は togetter にまとまっている
<http://togetter.com/li/154135>



書籍紹介



第22章「プロシージャル技術」
第23章「デジタルゲームAI」



第5章「これからデジタルゲームのAIの
進む道を知るために知っておきたいこと」
第5章付録「デジタルゲームAI入門」(教科書紹介)

ご清聴ありがとうございました！



ご質問: y.m.4160@gmail.com Twitter: miyayou まで

<http://www.hana.com/board/post.php?sid=0912301>