キャラクターAIの基礎理論
「環境と思考」

－ 20の実例を踏まえて －

筑波大学コンテンツ工学セミナー

三宅　陽一郎

(株式会社フロム・ソフトウェア)

y.m.4160@gmail.com

筑波大学

2008.10.30 (木)

Contact Information

Youichiro Miyake

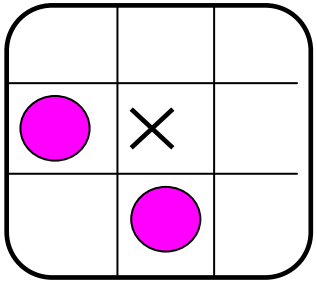
- Mail: y.m.4160@gmail.com
- Twitter: @miyayou
- Blog: http://blogai.igda.jp
- LinkedIn: http://www.linkedin.com/in/miyayou
- Facebook: http://www.facebook.com/youichiro.miyake

ゲーム実例リスト

- Halo 1,2,3
- Counter Strike
- Chromehounds
- Age of Empire1,2
- Killzone
- F.E.A.R.
- The Sims
- Assassin’s Creed

商業ゲームの実装例を集まるのはなかなかたいへんだが、
どんな理論よりも一番説得力があるのは実装されたゲームである。

AIが対象とする状態空間の数

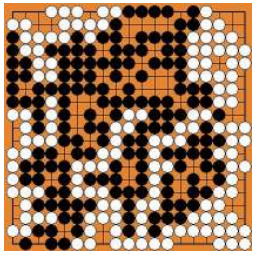


高々、9! 個

数

オフタイム
ディスクリート(離散)

将棋　10の220乗



囲碁　10の360乗

スケール

オフタイム
コンティニュアス(連続)

バーチャル

リアル

エンターテインメント
（ディジタルゲーム）

スポーツ（F1）

エンターテインメントか
スポーツか？

リアルか
バーチャルか？

ゲームAIとは何か？

(1) 人工知能技術をゲームに適用したもの。

(2) ゲームにおける知能的な機能を代用するもの。

（例）キャラクター、アバター、囲碁などの対戦AI

誰もきちんと定義しようとしない。

① きちんと定義しよう。

② 議論をするときには、定義を明らかにする。

今回は（2）の特に「キャラクターAI」の意味で使います！

ゲームAIの目標

①最初の目標　　時間と空間を支配できるAI

（技術的目標）

②最後の目標　　ユーザーの心理をコントロールできるAI

（エンターテインメントAIとしての目標）

今回の講演の主旨

キャラクターAIにおいては、

AIとそれを取り巻く環境の関係こそが、

高度なAIを実現するために

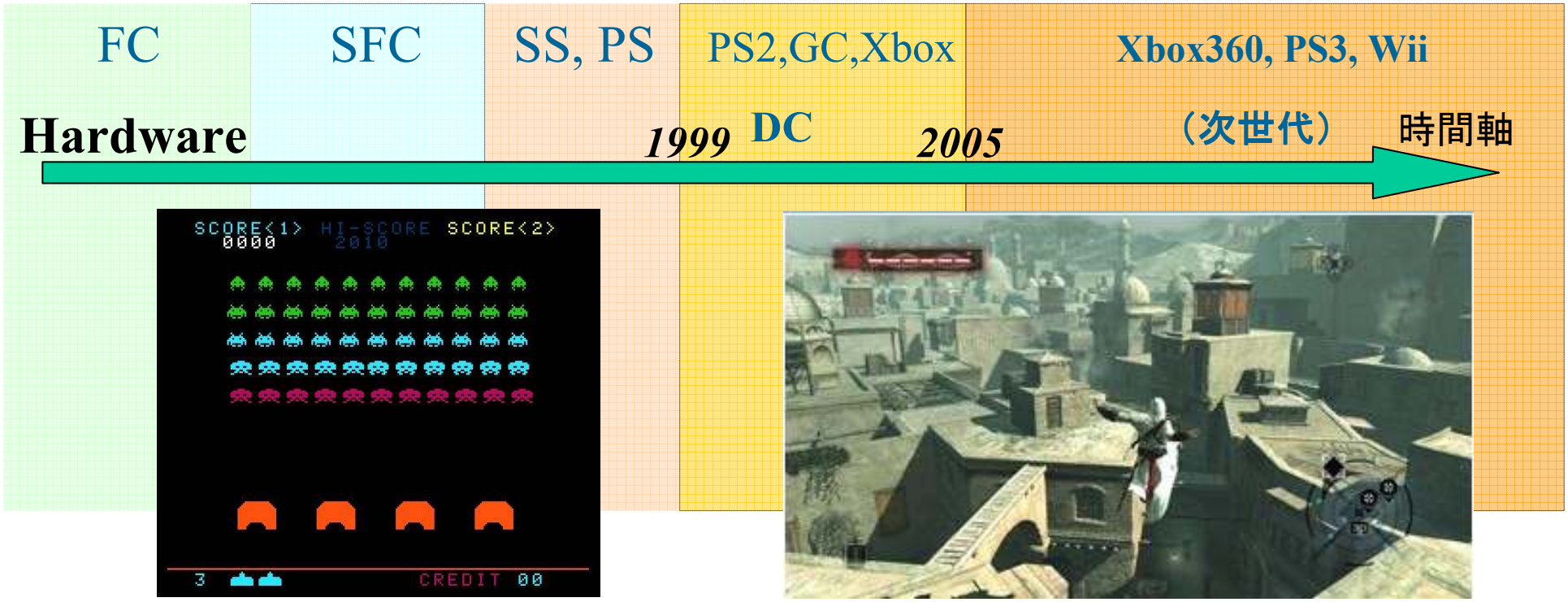
最も重要であることを理解する。

コンテンツ

第1章

人の知性からキャラクターの知性へ

ゲームの進化と人工知能



単純な世界の
シンプルなAI

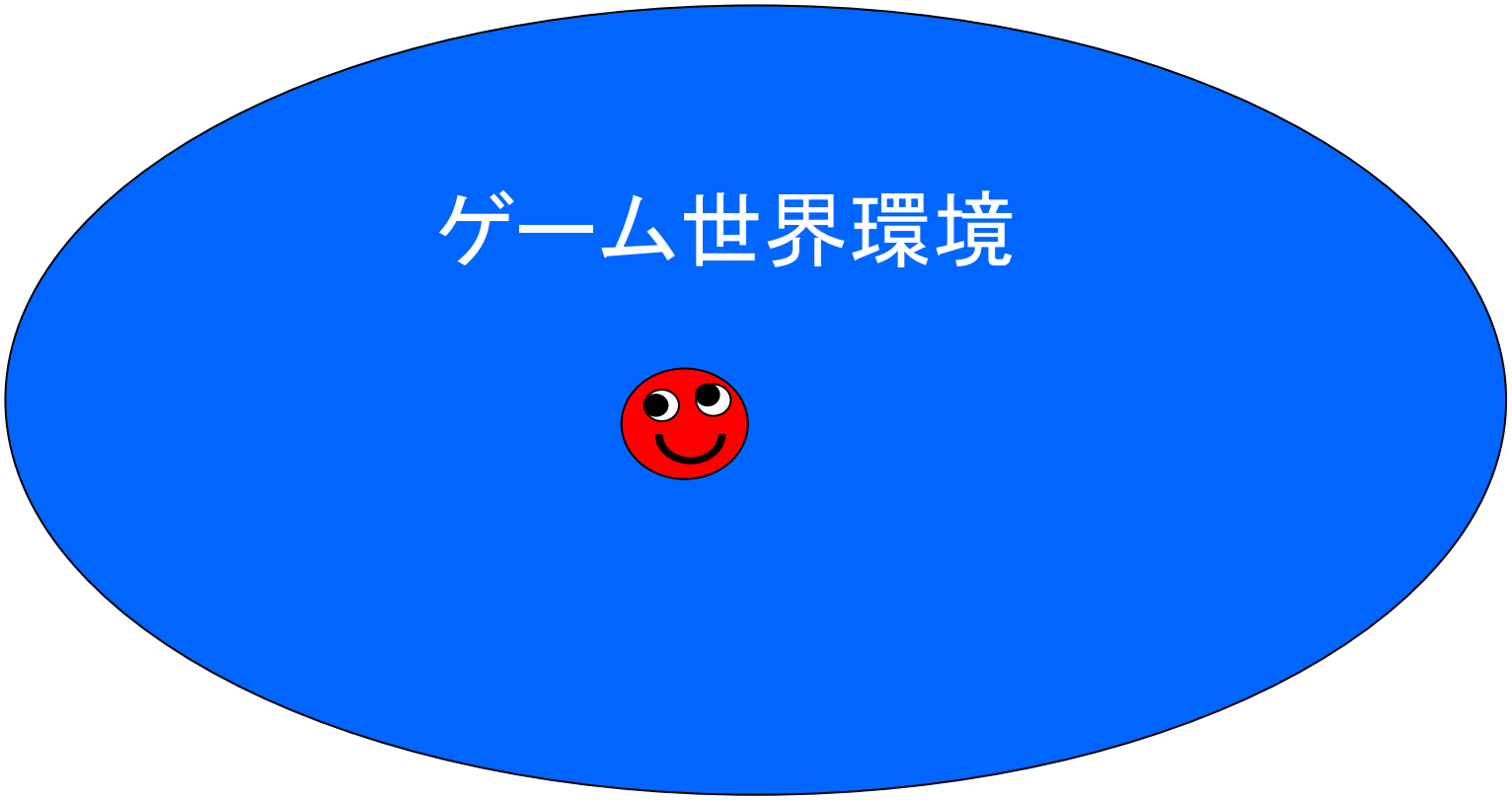
複雑な世界の
複雑なAI

ゲームも世界も、AIの身体と内面もますます複雑になる。

知性は常に環境と結び付けて考えなければならない

自然全体は、長い時間をかけて適切な場所に適切な生命を進化させ配置し、全体を連携させて生存して来た。

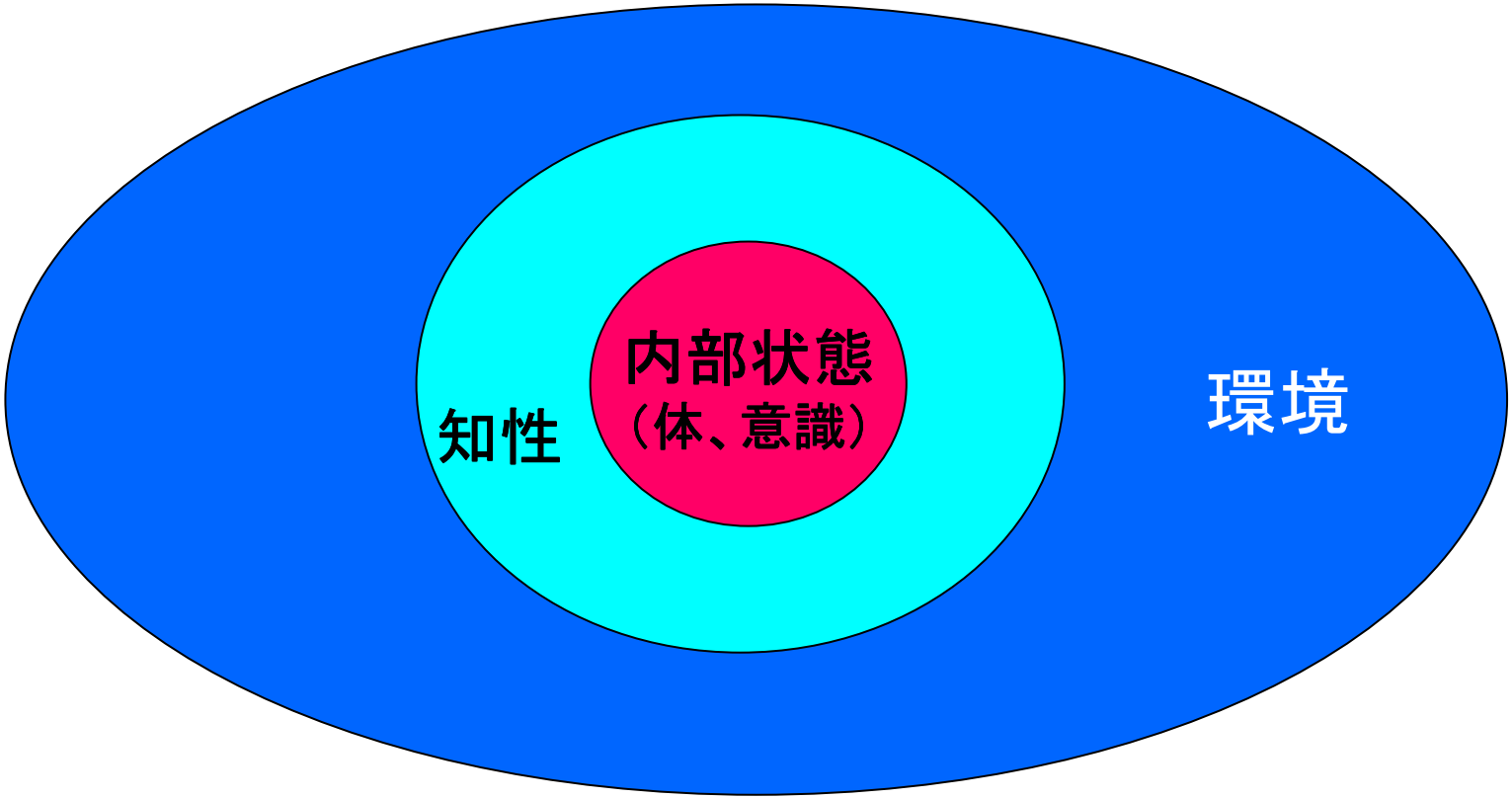
ゲーム世界とキャラクターAI



キャラクターAIを考えるときも、必ず周囲の環境と共に考えなければならない。

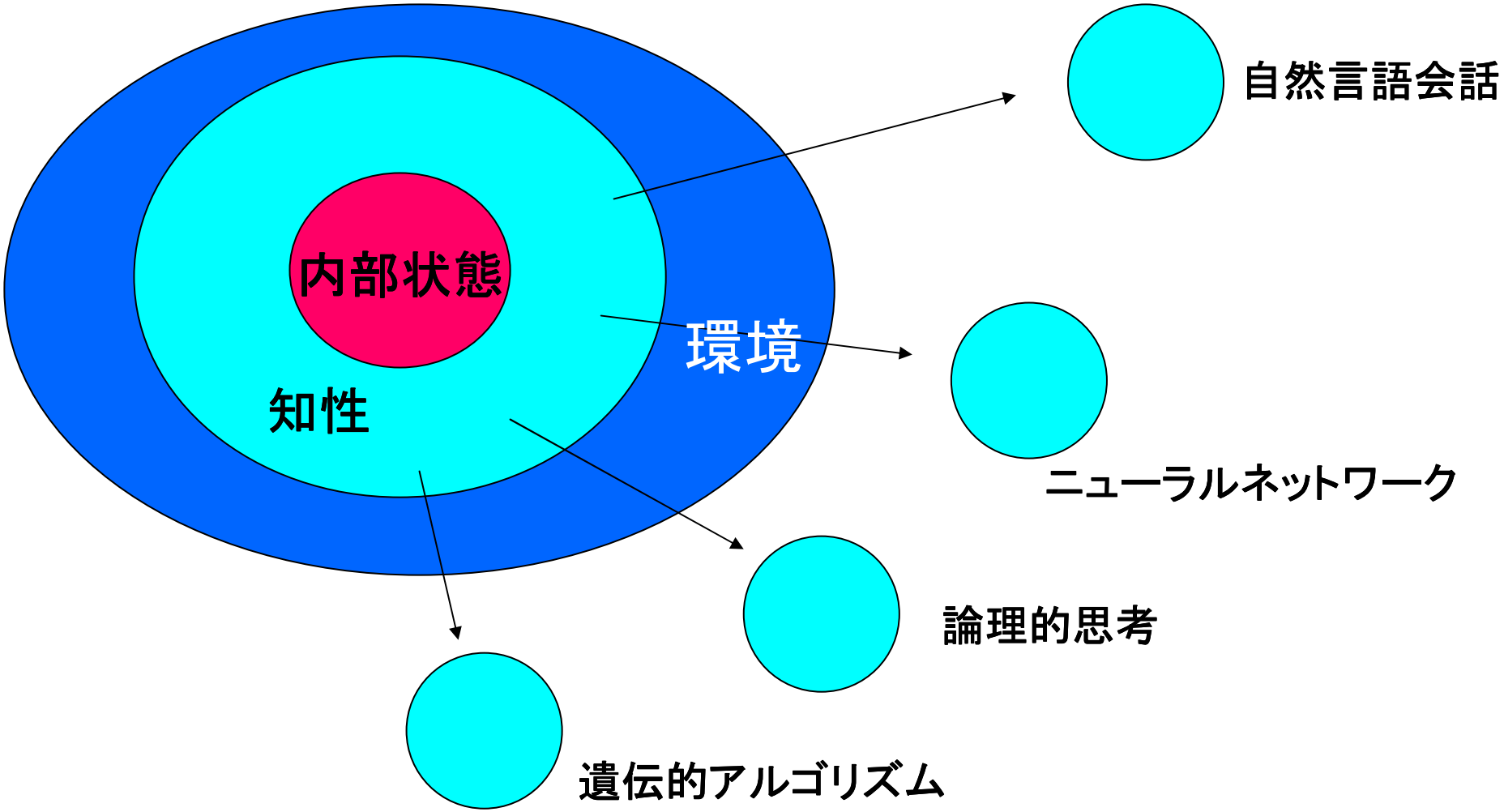
そもそも知性とは？

- 生き物が、外部の環境に対して、適応するために持ったシステム



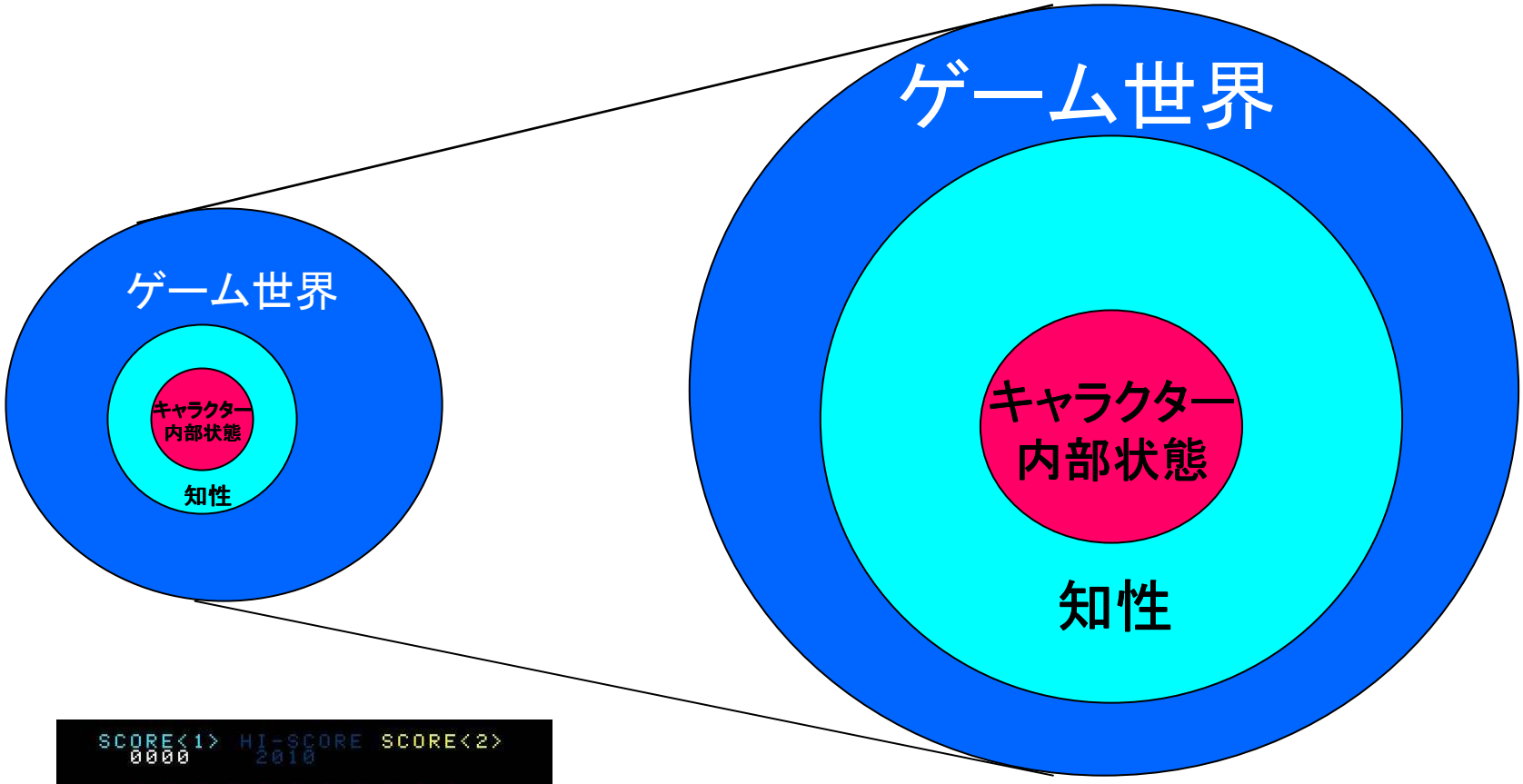
知性は境界に宿る

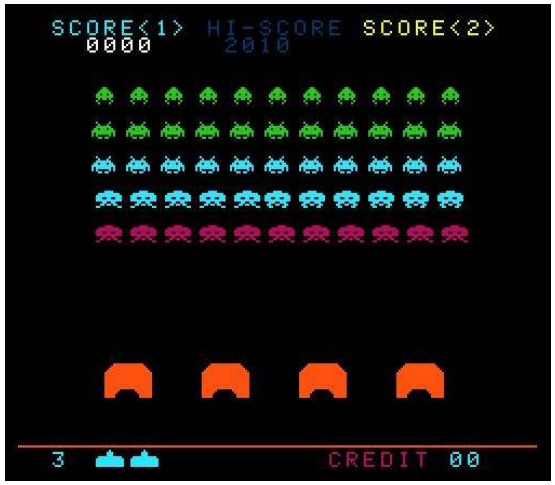
人工知能とは？



生命の知能を機能ごとに、取り出して発展させたもの。

キャラクターAIとは？


ゲーム世界
キャラクター
内部状態
知性
ゲーム世界
キャラクター
内部状態
知性


SCORE<1> HI-SCORE SCORE<2>
0000 2010
3
CREDIT 00

今日の技術テーマ

キャラクターをゲーム世界で
動かすにはどうすればいいんだろう？

（ここで学生に質問）

その前に…

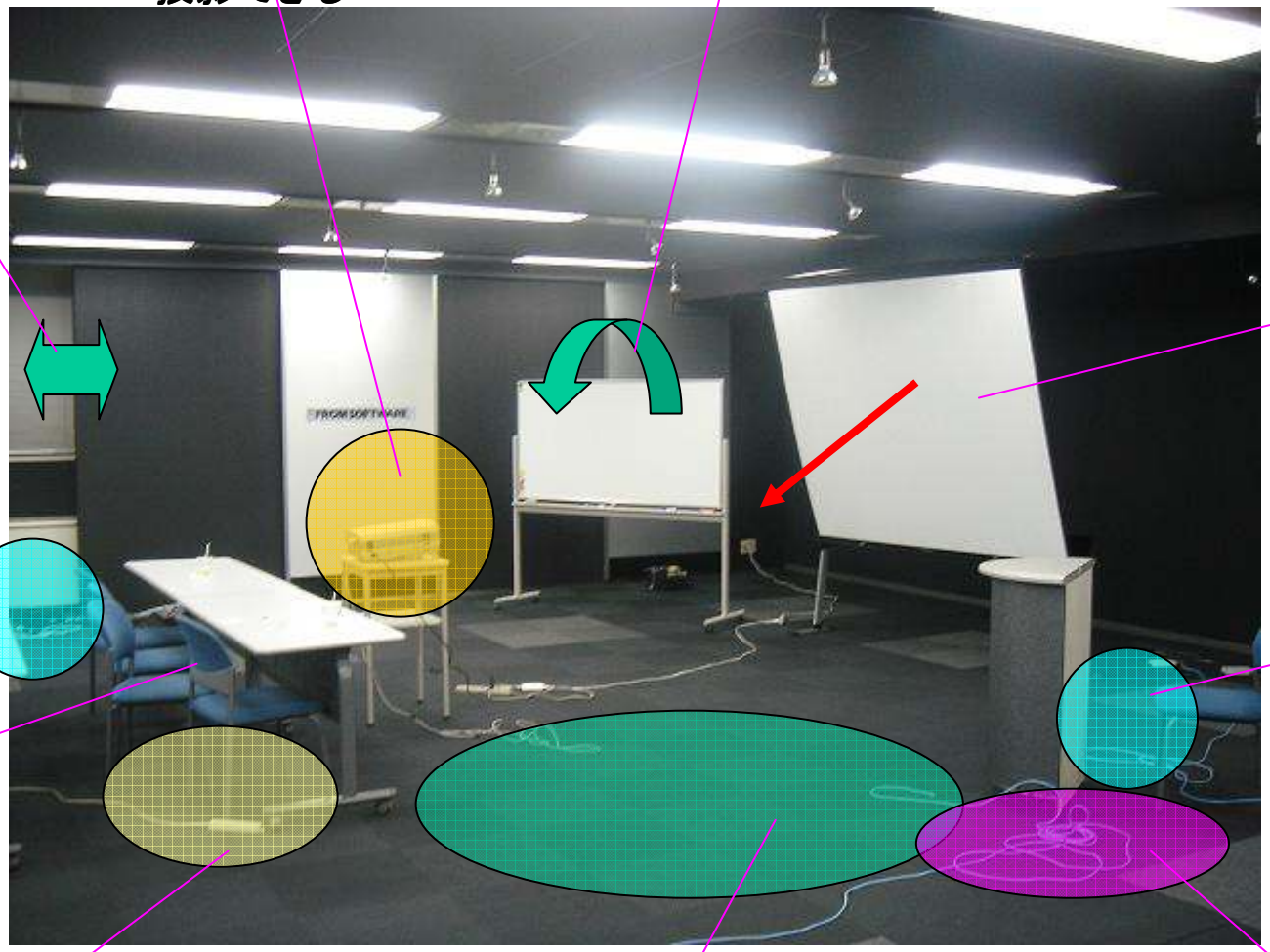
人間はどのようにして現実世界で
活動しているんだろう？

環境と知性はどのように
結び付いているだろうか？

この風景を見て何を思いますか？

（ここで学生に質問）

環境に対していろいろな情報が埋め込まれている



左右に動かせる

スイッチを入れれば
投影できる

折り曲げることができる

倒れるかも

通れない

通れるかな？

引けば座れる

プロジェクターの
コンセント届くかもしれない

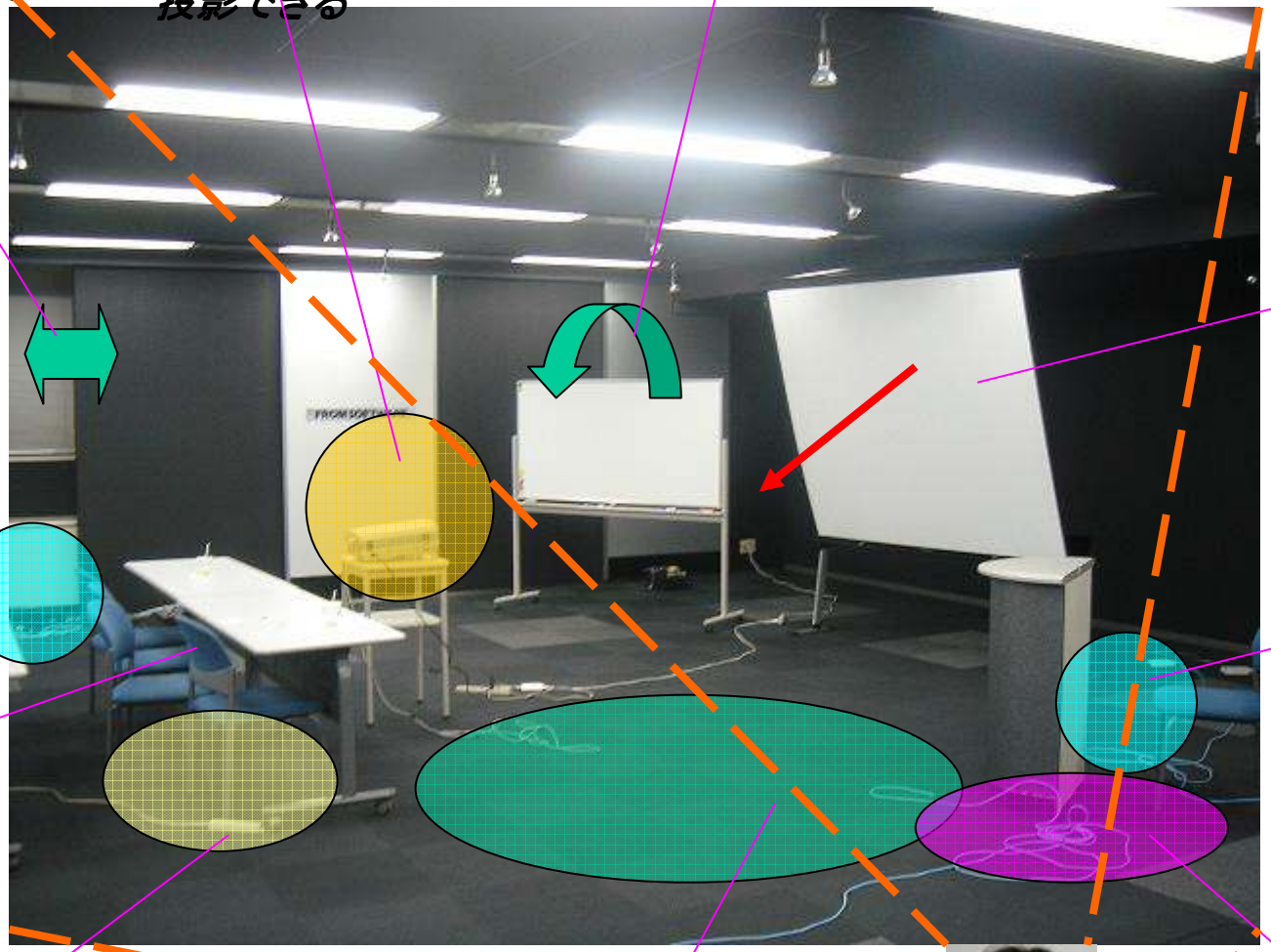
歩くことができる

こけるかも

…だから行動できる

誰が情報を埋め込んだの？

環境に対していろいろな情報が埋め込まれている



左右に動かせる

スイッチを入れれば
投影できる

折り曲げることができる

倒れるかも

通れない

通れるかな？

引けば座れる

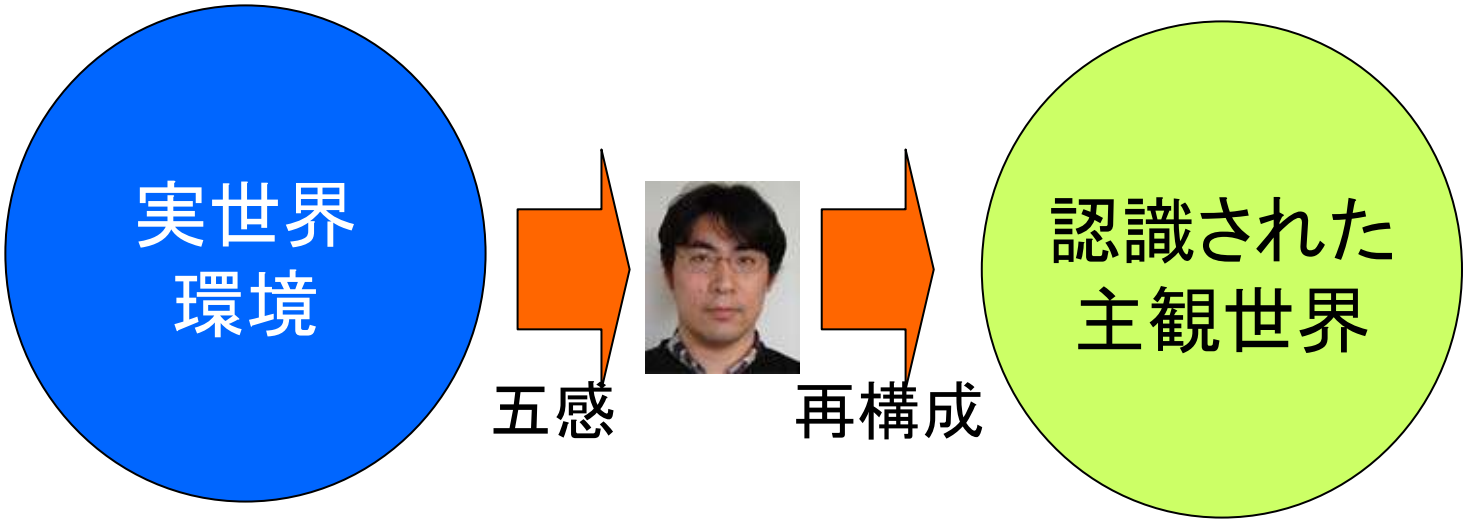
プロジェクターの
コンセント届くかもしれない

歩くことができる

こけるかも

自分！

人は五感で世界を再構成する過程で
自分の身体を基準に環境に情報を埋め込んで認識している



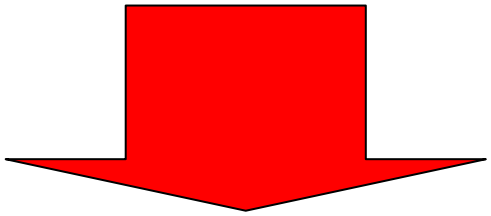
認識された世界には、
無意識の中で、様々な情報が
既に埋め込まれている

① 知識（常識）
② 可能な行動　＝　アフォーダンス
③ 質感（クオリア）
・・・

自分の知識や経験、身体情報から構成される

結論①

(1)　動物は自分の身体を基準にして、環境に対して
いろいろな情報が埋め込んでいる。
(2)　リアルタイムに判断することもある。
(3)　その情報をもとにして行動することができる。



実はそれはキャラクターAIにとっても同様である

キャラクターAIも環境情報がなければ何も行動できない

http://www.bungie.net/Inside/publications.aspx

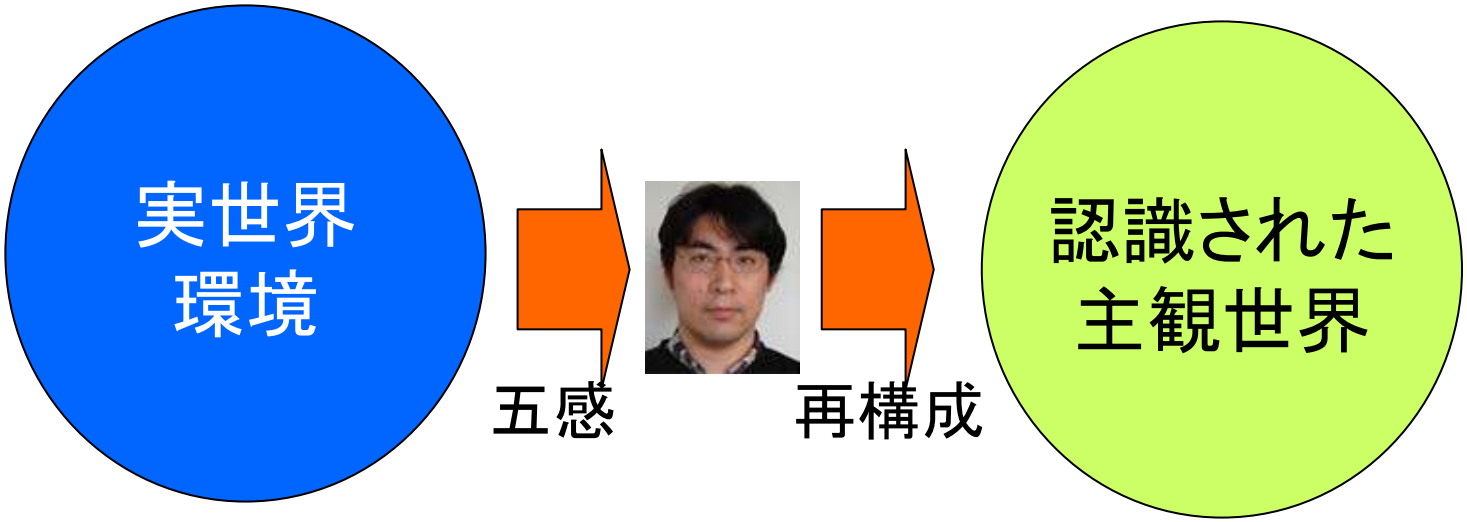
かつその情報はキャラクターの身体とその能力に応じたものでなけれならない

キャラクターAIも環境情報がなければ何も行動できない

http://www.bungie.net/Inside/publications.aspx

かつその情報はキャラクターの身体とその能力に応じたものでなけれならない

人は五感で世界を再構成する過程で
自分の身体を基準に環境に情報を埋め込んで認識している



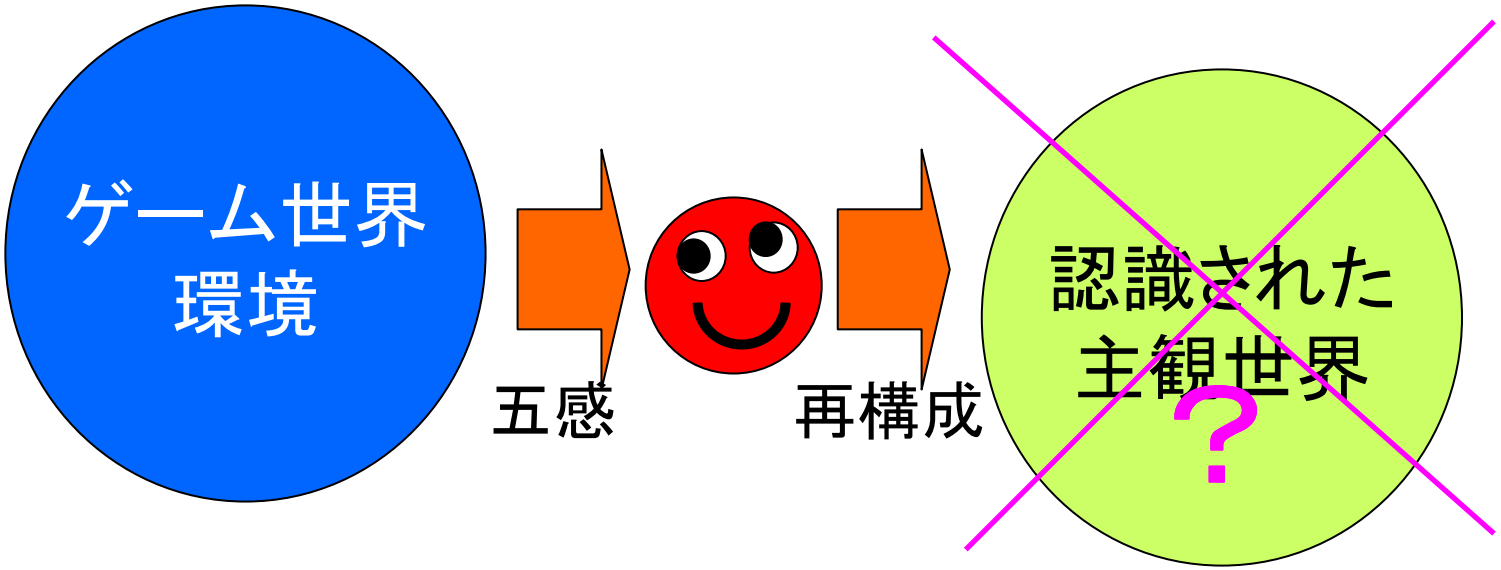
人は無意識の中で環境と知性を結び付ける能力（内的認識）を持っている。

可能な行動　＝　アフォーダンス
質感（クオリア）
知識表現

認識された世界には、
無意識の中で、様々な情報が
既に埋め込まれている

そういった情報は、自分の知識や
経験、身体情報から構成される

AIは五感で世界を再構成する過程で
自分の身体を基準に環境に情報を埋め込んで認識している？



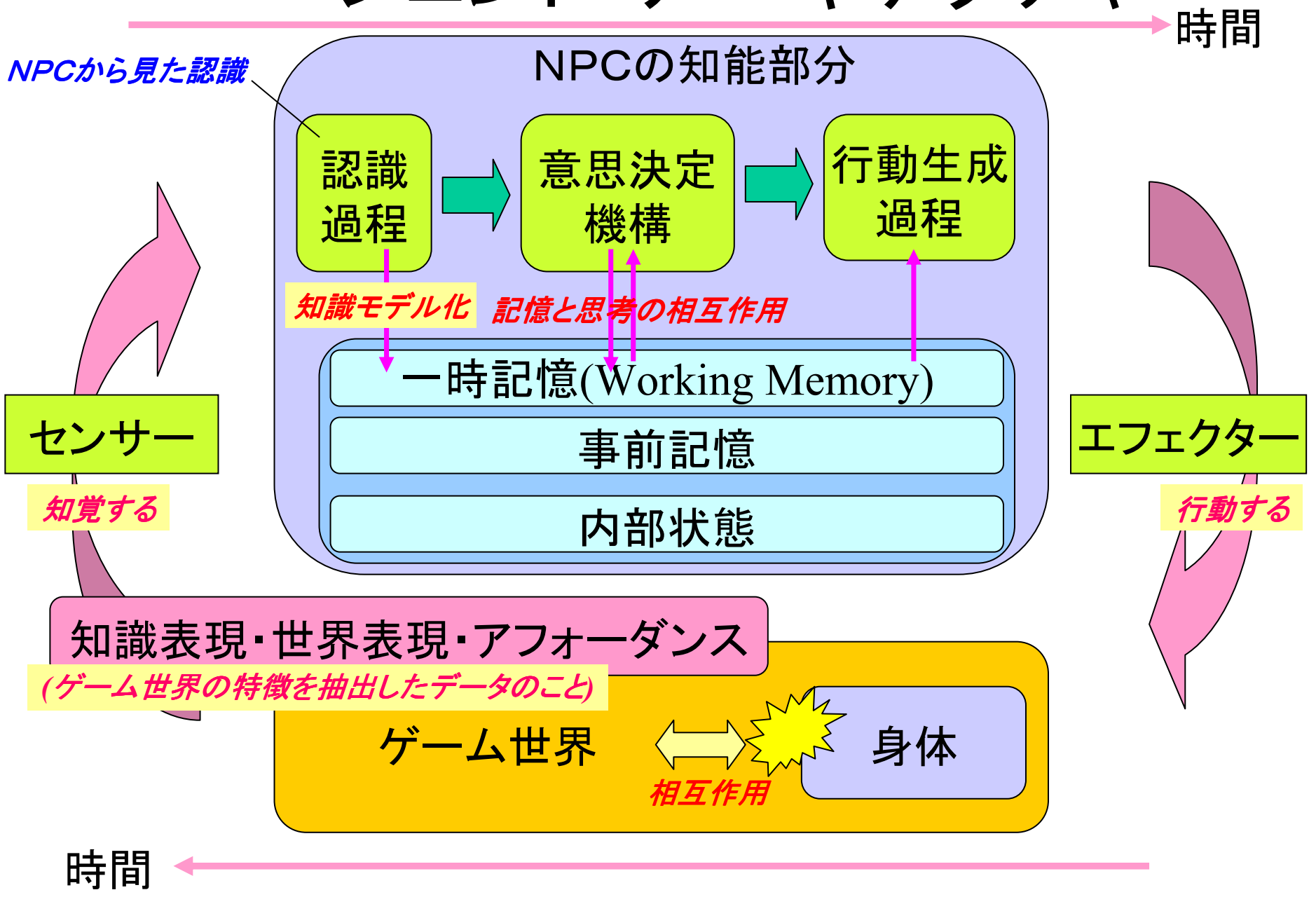
人工知能には無意識の中で環境と知性を結び付ける能力（内的認識）が欠如している。

人工的に環境と知性を結び付ける仕組みが必要

人工知能のための認識（動作）モデル＝エージェント・アーキテクチャ

知識モデル　＝　知識表現（世界表現）＋アフォーダンス

エージェント・アーキテクチャ



第1章まとめ

(1) 人は五感から得た情報によって世界を再構成している。

(2) 人が再構成した主観世界には無意識のうちに多くの情報が埋め込まれている。

(3) 知識、アフォーダンス、判断、クオリア。

(4) 人工知能にはそういった能力が無いために、人の認識に変わる機能が必要である。

参考文献

(1) Damian Isla, “Halo AI Retrospective: 8 Years of Work on 30 Seconds of Fun”, http://www.bungie.net/Inside/publications.aspx

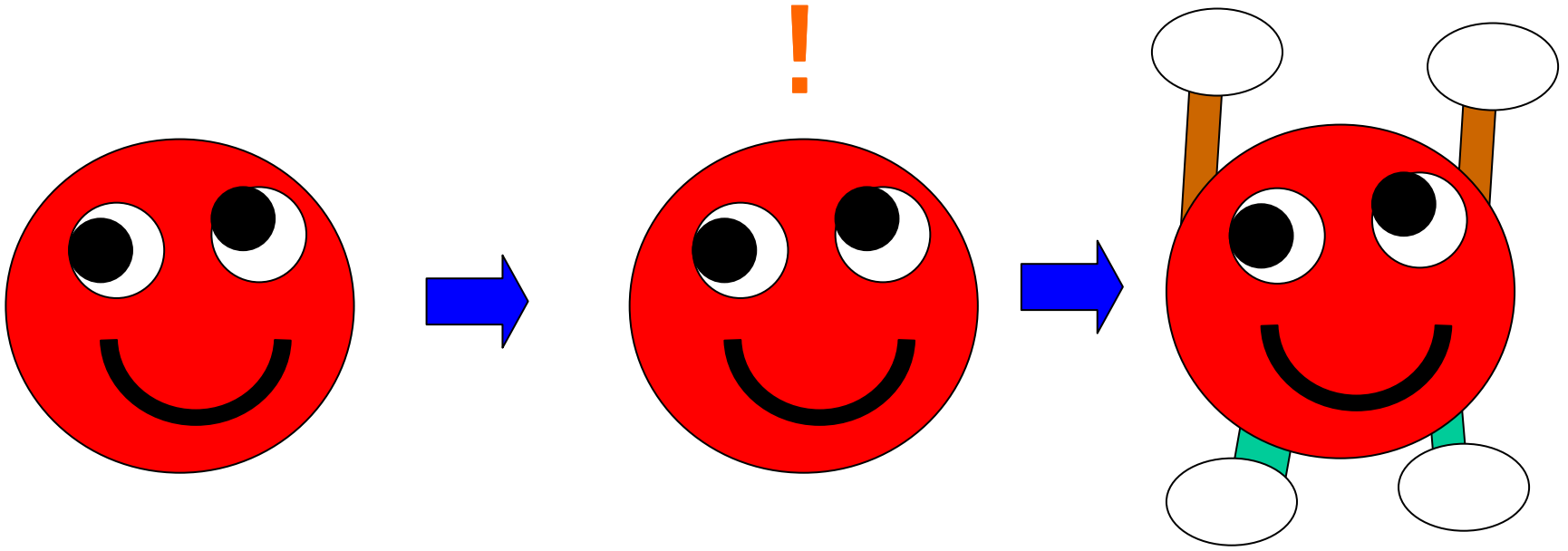
(2) IGDA日本ゲームAI連続セミナー第３回資料

第2章

エージェント・アーキテクチャ

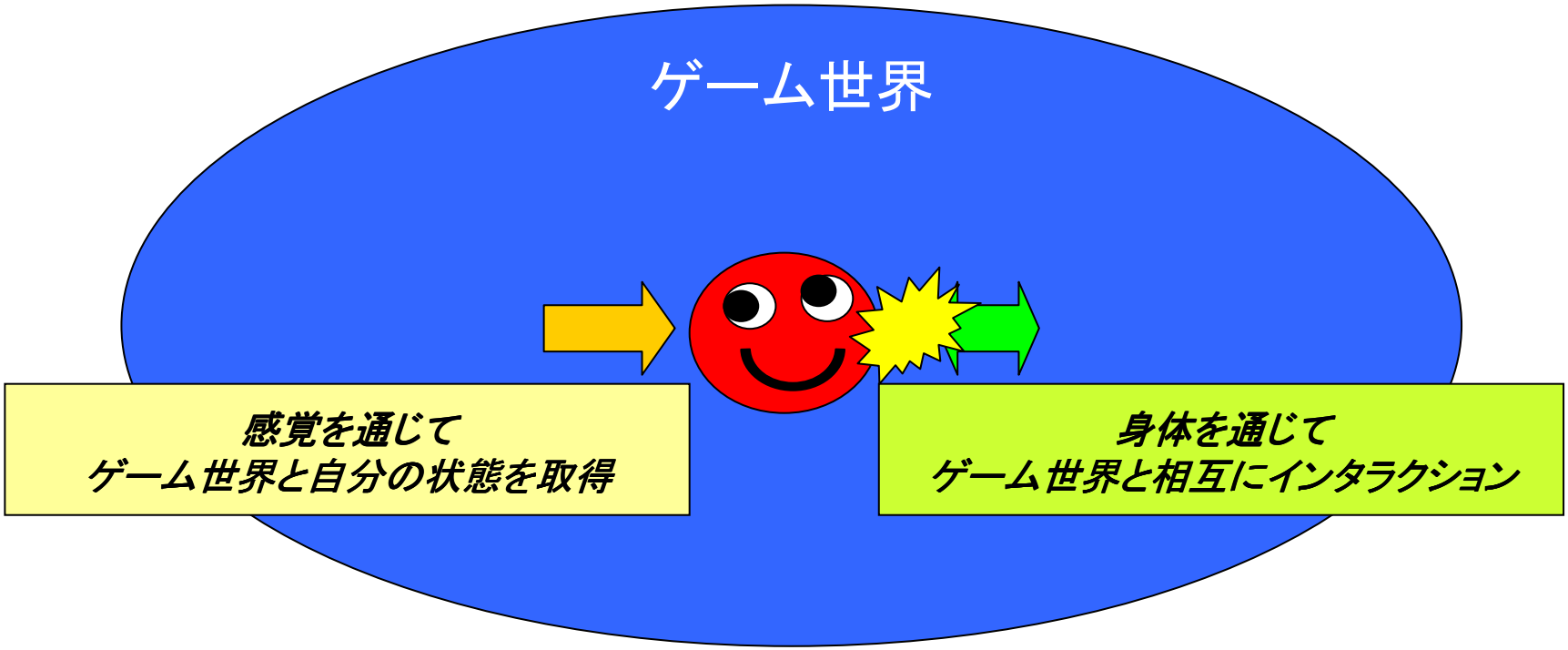
エージェントとは？

① 環境に対して情報を集める感覚（センサー）を持つ
② 自ら判断する能力を持つ。
③ 環境に対して働きかけることができる能力を持つ。

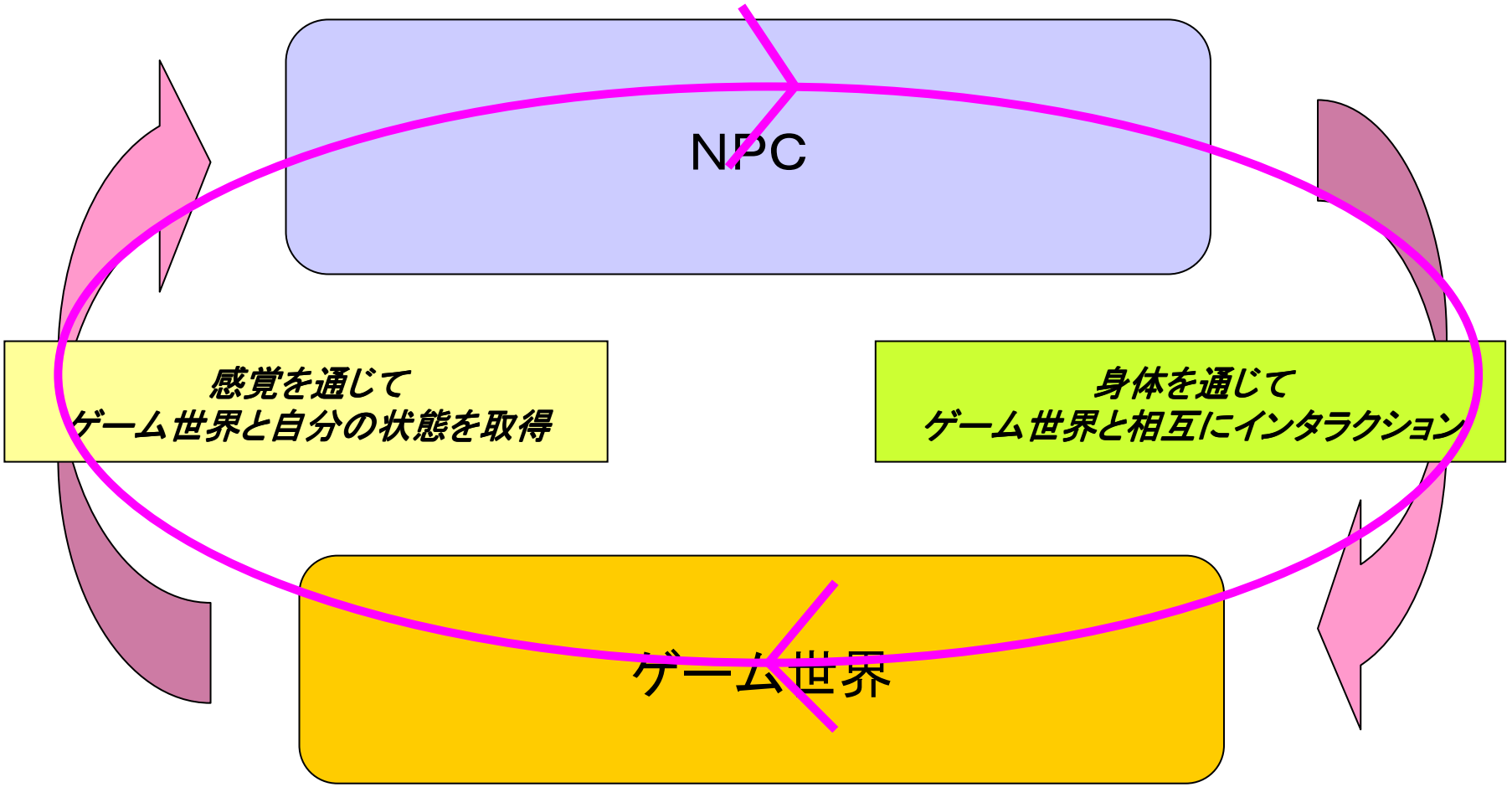


感覚を持ち　　自ら判断して　　世界に働きかける能力を持つ

ゲームにおけるエージェント



エージェント・アーキテクチャーにおける情報の流れ

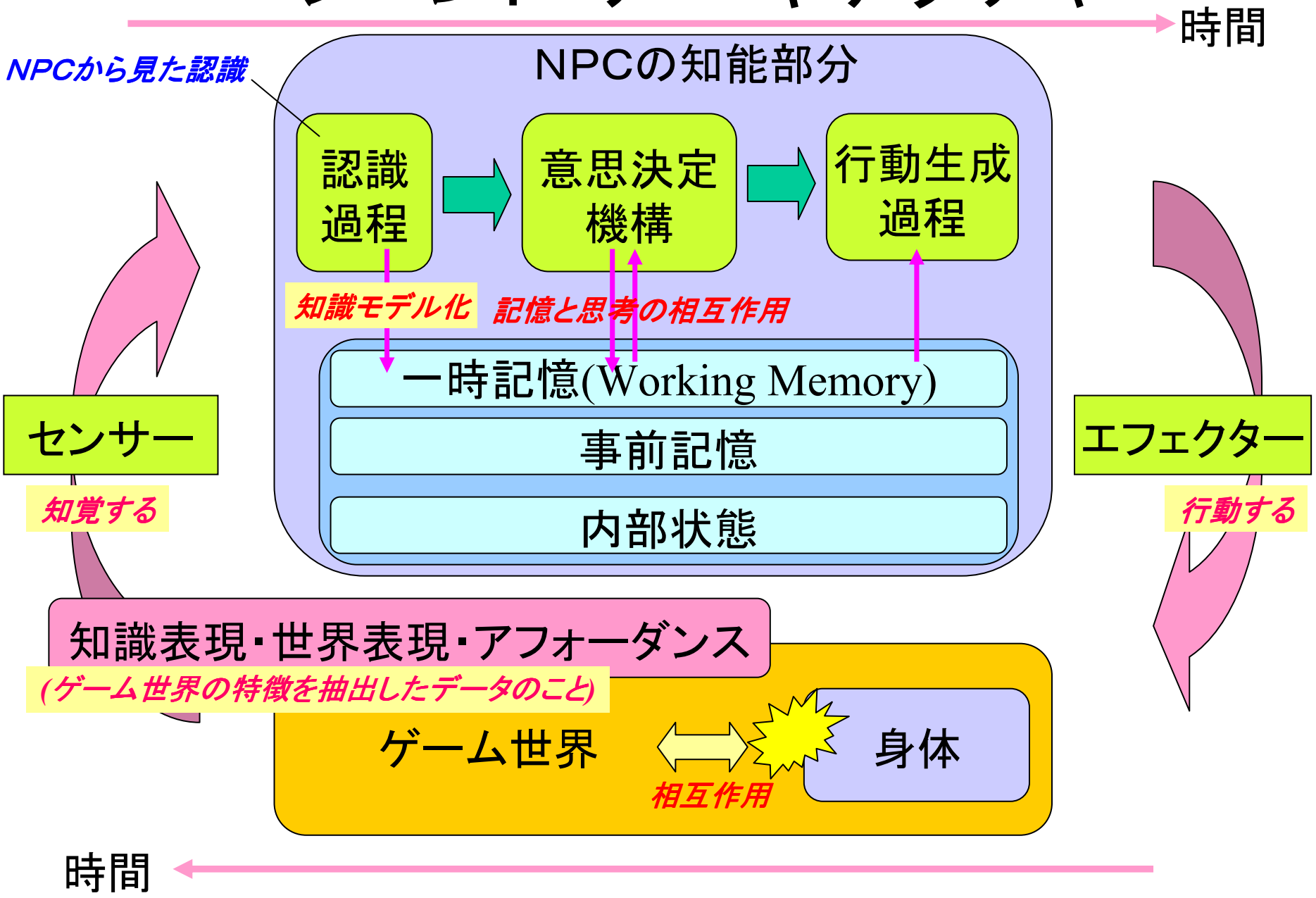


この情報の流れに仕掛けをしてみよう！

「人工知能＝からくり」

作り方を勉強しよう！

エージェント・アーキテクチャー



知識表現・世界表現・アフォーダンスを理解しよう

① アフォーダンス

② 知識表現・世界表現

アフォーダンスとは？

アフォーダンス
＝身体が世界で可能な行動可能性

この吊橋を渡れますか？

渡れるというアフォーダンス

渡れないというアフォーダンス

どちらも自分の身体情報を基本にしている

アフォーダンス
＝身体が世界で可能な行動可能性

体の大きさが2分の一になったとしてこの吊橋を渡れますか？

渡れるというアフォーダンス

渡れないというアフォーダンス

どちらも自分の身体情報を基本にしている

アフォーダンス
＝身体が世界で可能な行動可能性

体重が3倍になったとしてこの吊橋を渡れますか？

渡れるというアフォーダンス

渡れないというアフォーダンス

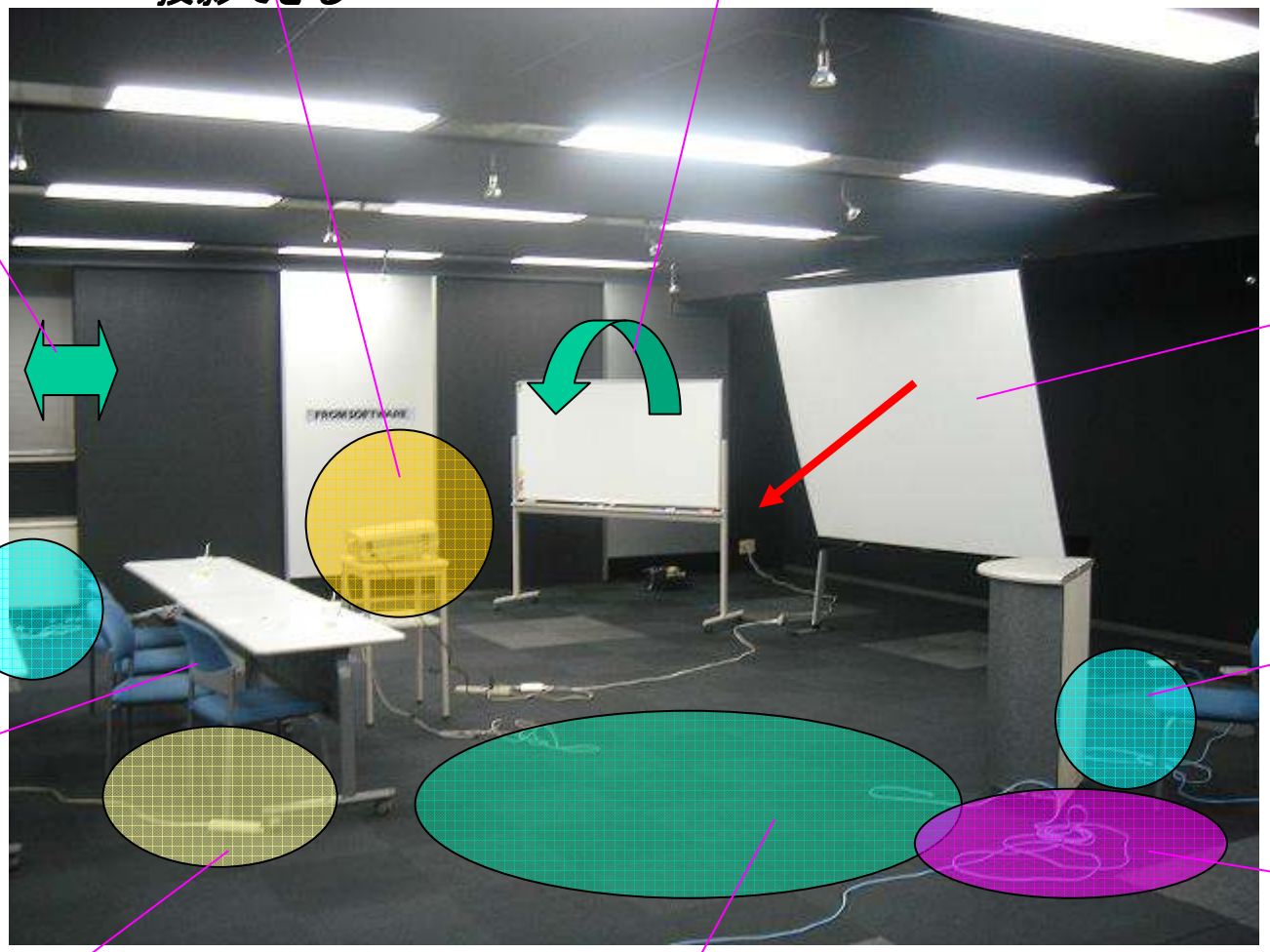
どちらも自分の身体情報を基本にしている

この風景を見て何を思いますか？

体の幅が3倍になったとしてこの部屋がどう見えるか？

人の頭脳が環境に埋め込む情報を更新するのに、しばらく経験と時間が必要！

人は自分の身体を基準にして環境に対して いろいろな情報が埋め込んでいる



…だから行動できる

Halo2におけるオブジェクトに埋め込まれている アフォーダンス情報

BUNGIE

Spatial Object Features

Much like “Affordances” (Gibson, Norman, Wright)

- An object advertises the things that can be done with / to it
- But they must do so in a geometrically precise way in order to be useful

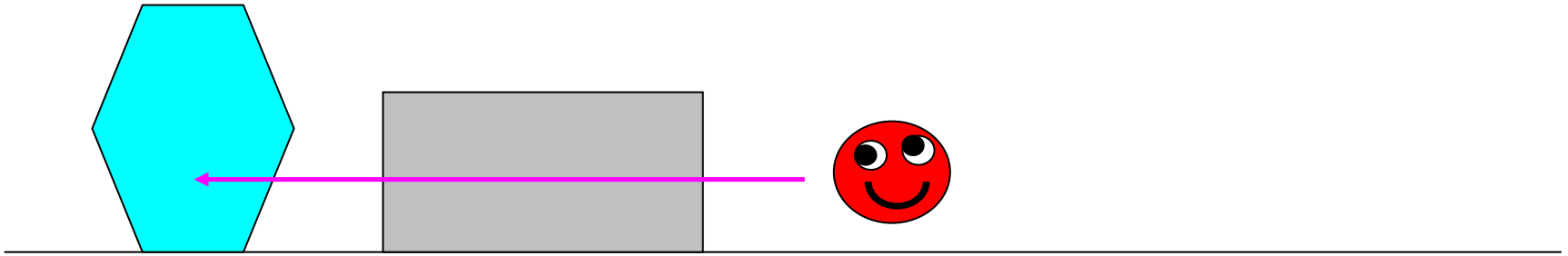
Implementation: “object markers”

- Rails or points
- Orientation vector indicates when the affordance is active
- An object has different properties at different orientations

Damian Isla (2005), “Dude, where’s my Warthog? From Pathfinding to General Spatial Competence”, http://www.aiide.org/aiide2005/talks/isla.ppt

一連のアフォーダンス情報が
キャラクターの行動を賢明に見せる

Can I go through ?
Yes or No



通れるか？
通れないか？

X

壊せるか？　壊せないか？
動かせるか？　動かせないか？

＝

突破できる？
突破できないか？

Tips: アクティブ・アフォーダンス

- ゲームでは壊せそうなものは壊せそうでなければならない
- 食べれるものは食べれそうに見えなければならない

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)

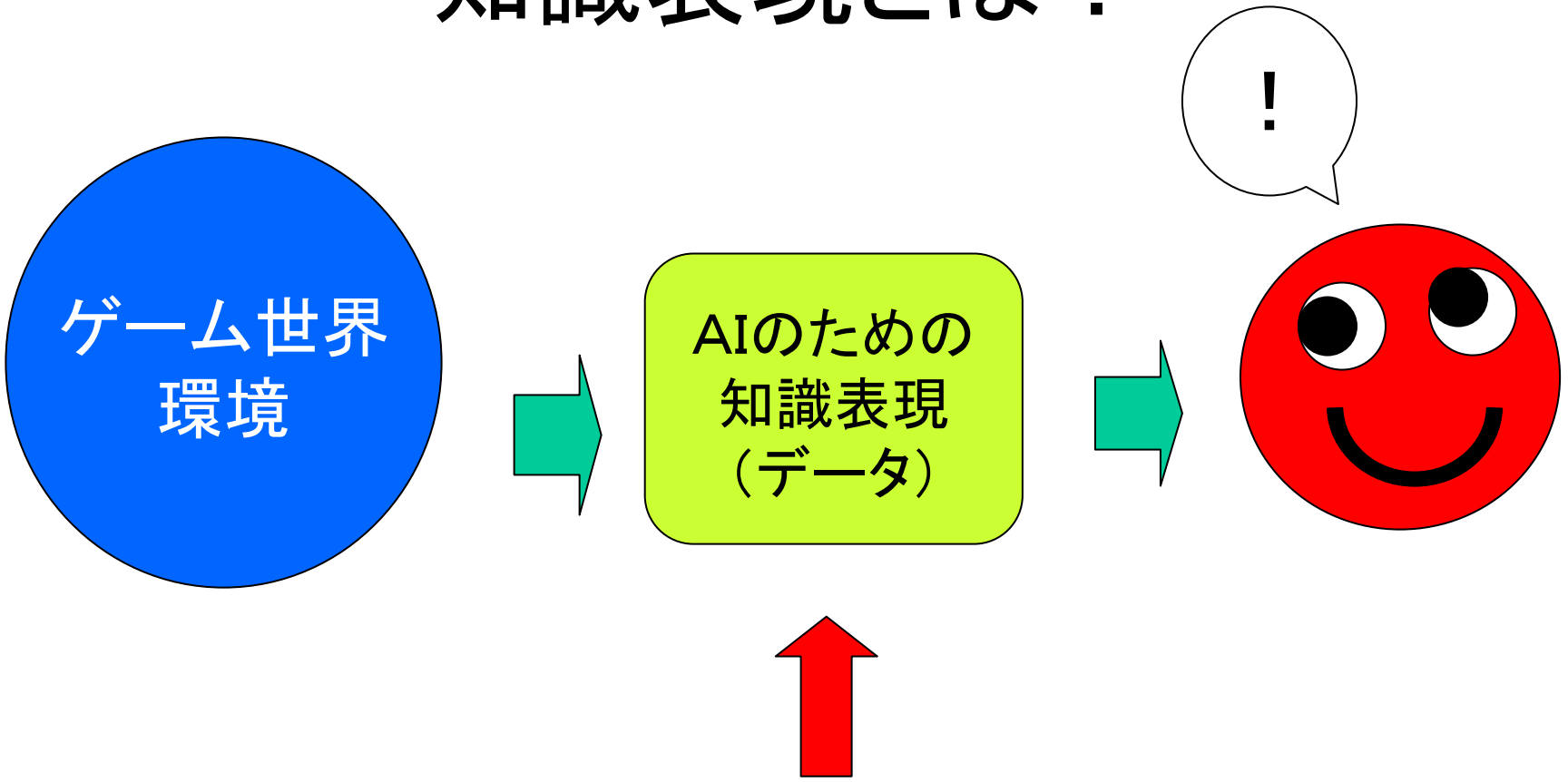
アフォーダンス説明終わり

知識表現・世界表現・アフォーダンスを理解しよう

① アフォーダンス

② 知識表現・世界表現

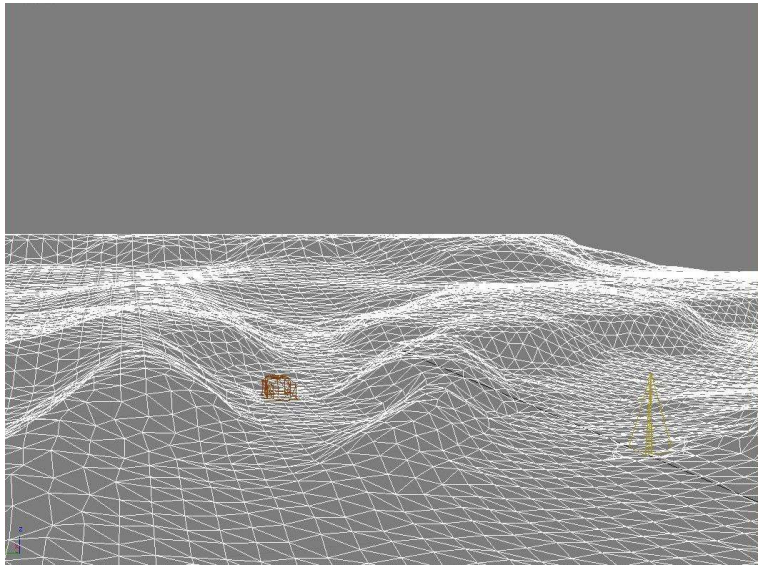
知識表現とは？

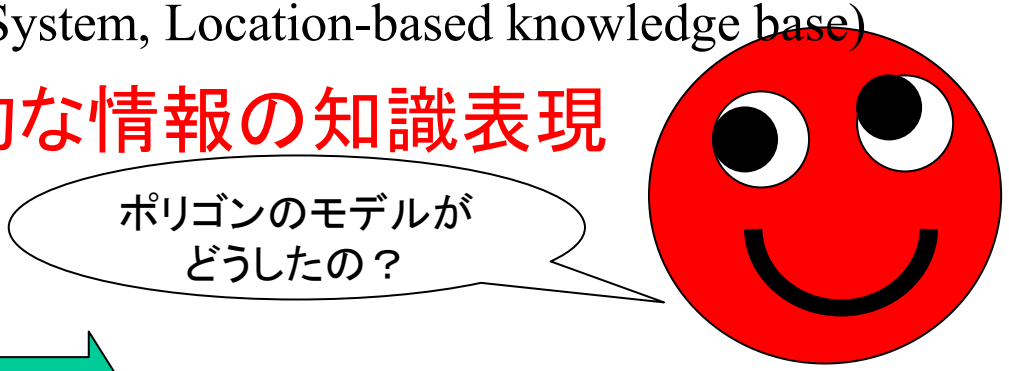


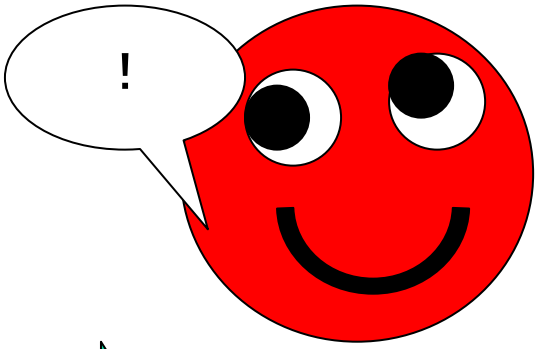
世界表現とは？

(World Representation,
Location-based Information System, Location-based knowledge base)

AIの属する世界の大局的な情報の知識表現

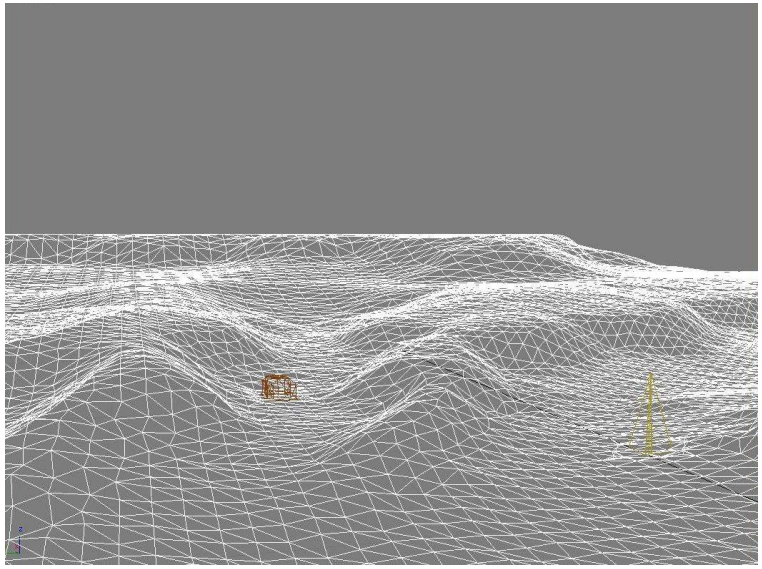




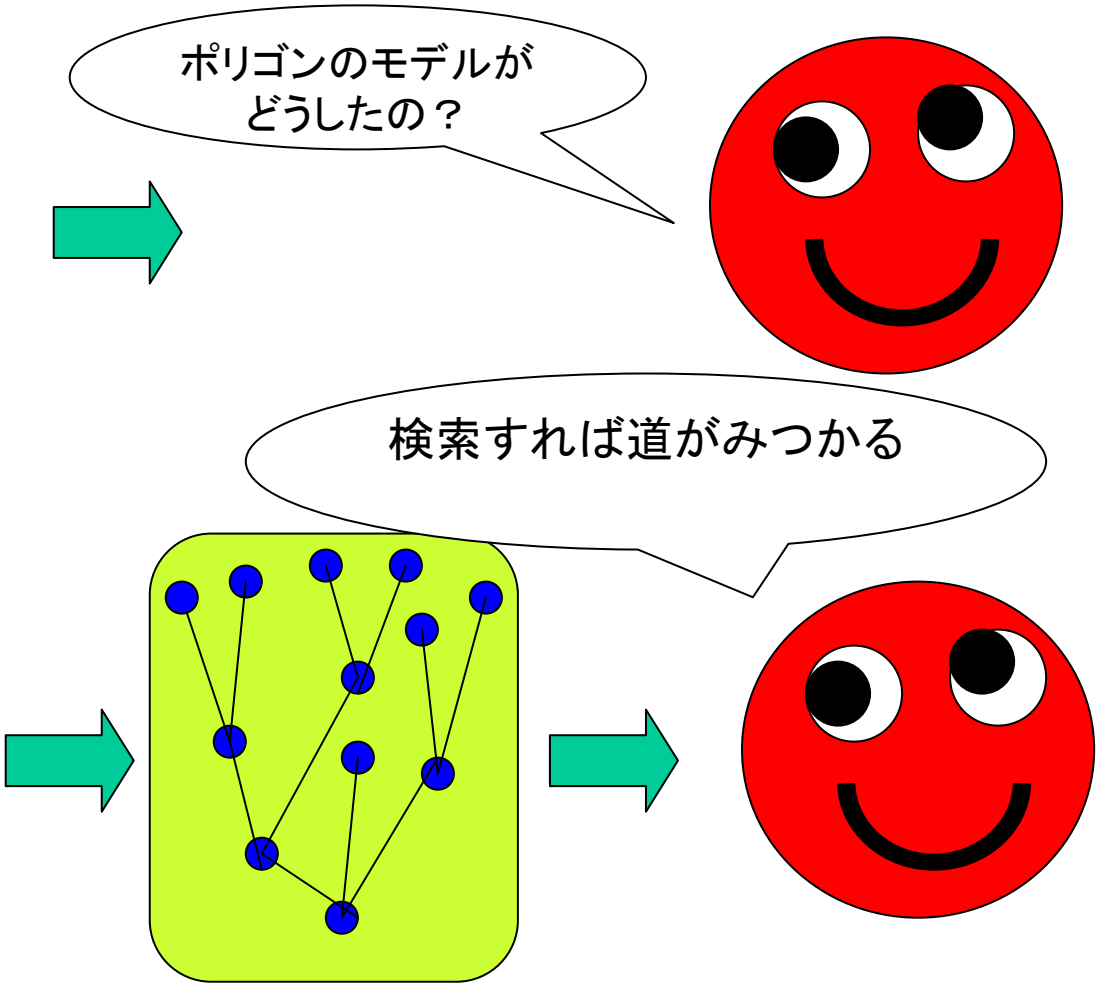


AIのための
データ

世界表現の例　パス検索データ

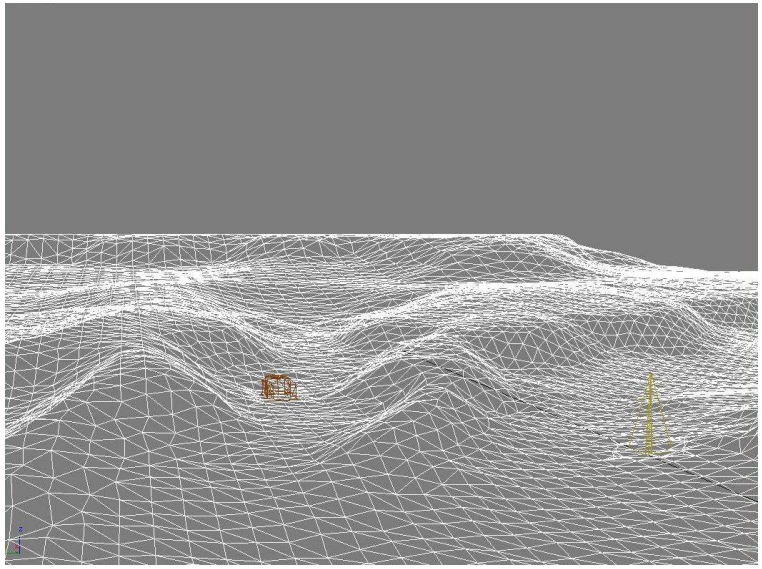


地形データ

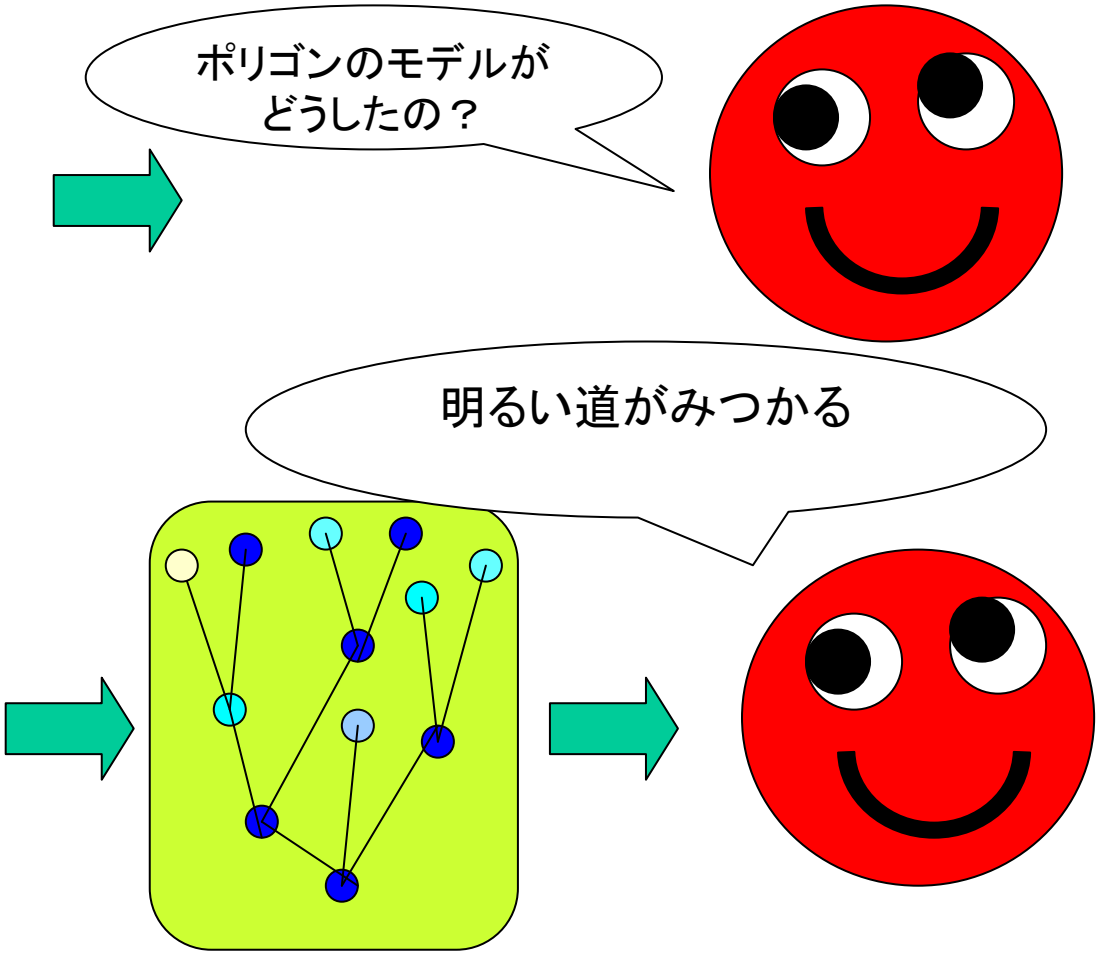


ウエイポイントデータ

世界表現の例
パス検索データ＋ウエイポイントの明るさ

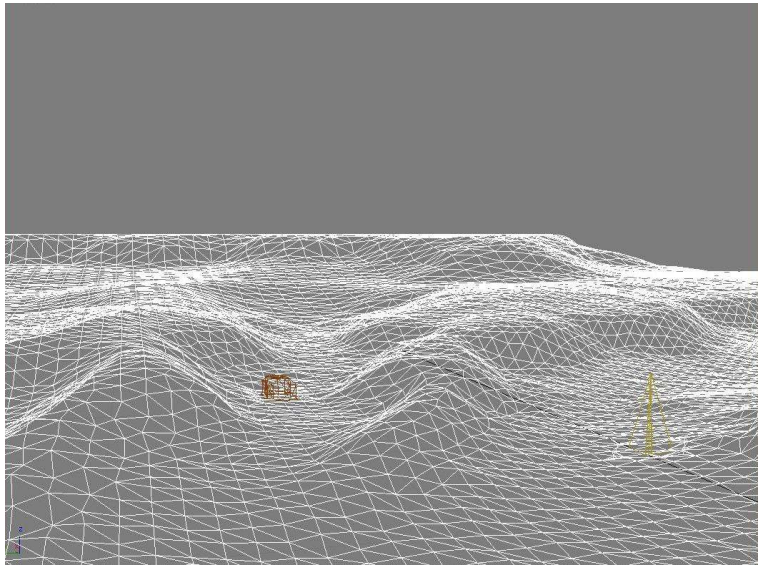


地形データ

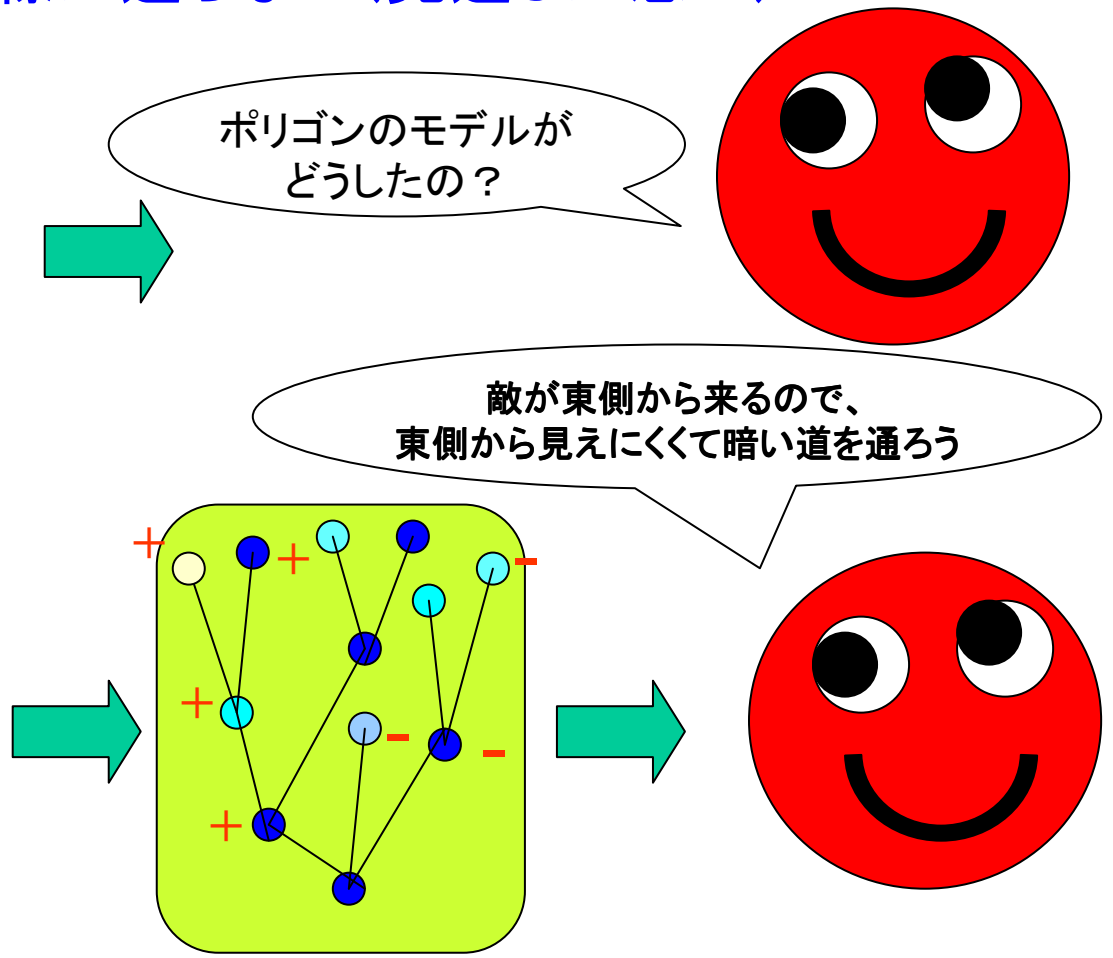


ウエイポイントデータ
ウエイポイントの明るさ

世界表現の例　パス検索データ＋ウエイポイントの明るさ
＋ 東側に対して視線が通らない（見通しが悪い）

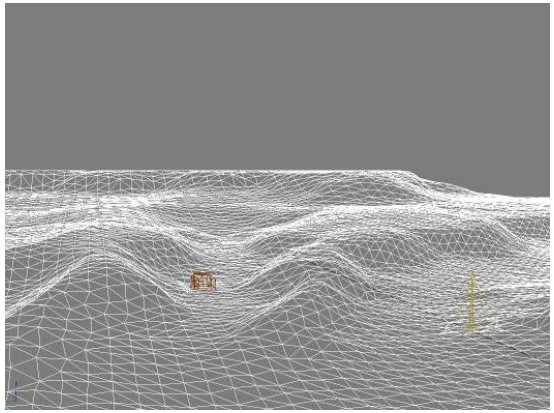


地形データ



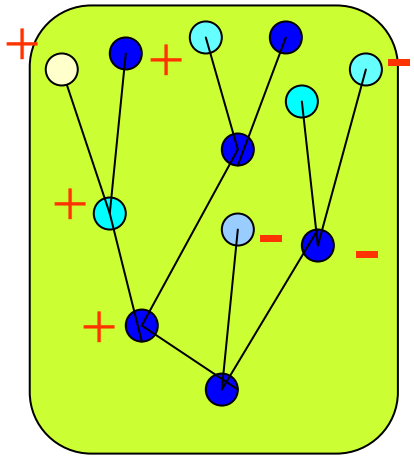
ウエイポイントデータ
ウエイポイントの明るさ
東側から見えやすいかどうか（＋、－）

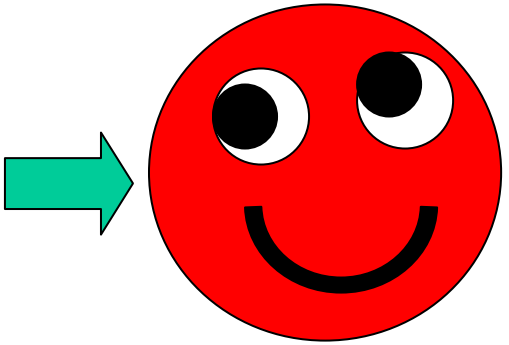
世界表現と地形解析



地形データ

地形解析





AIのための世界表現

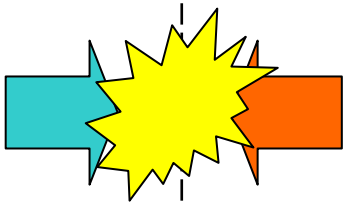
Killzone NPCの課題

広大なマップで戦術的に移動し攻撃するNPC

バトルフィールドでは
最大7vs7(一人はプレイヤー)
で戦う

味方

人間



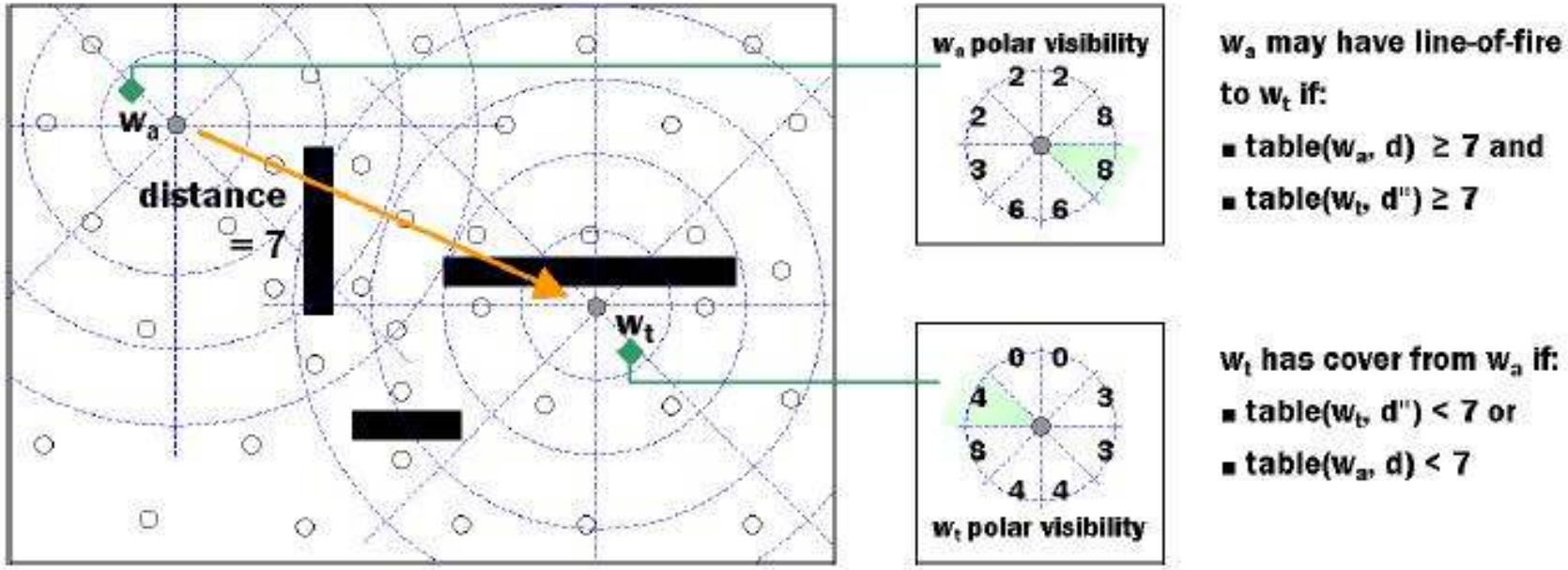
(強化)人間

敵

世界表現の使い方①

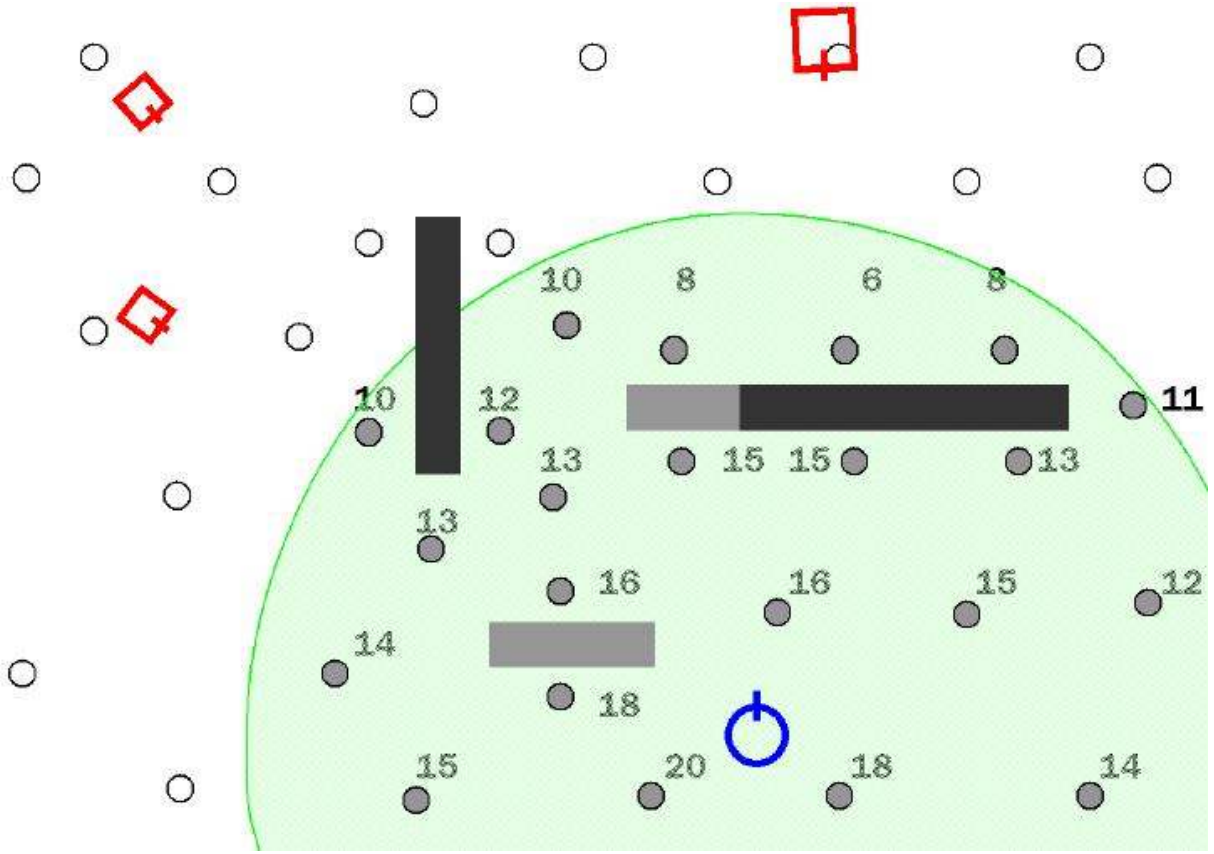
軽い射線判定を用いて戦術的な移動ポイントを
リアルタイム計算しよう！

8方向に対して、可視距離（キャラクターの体が見える限界距離）を計算する（参照テーブルとして持つ）



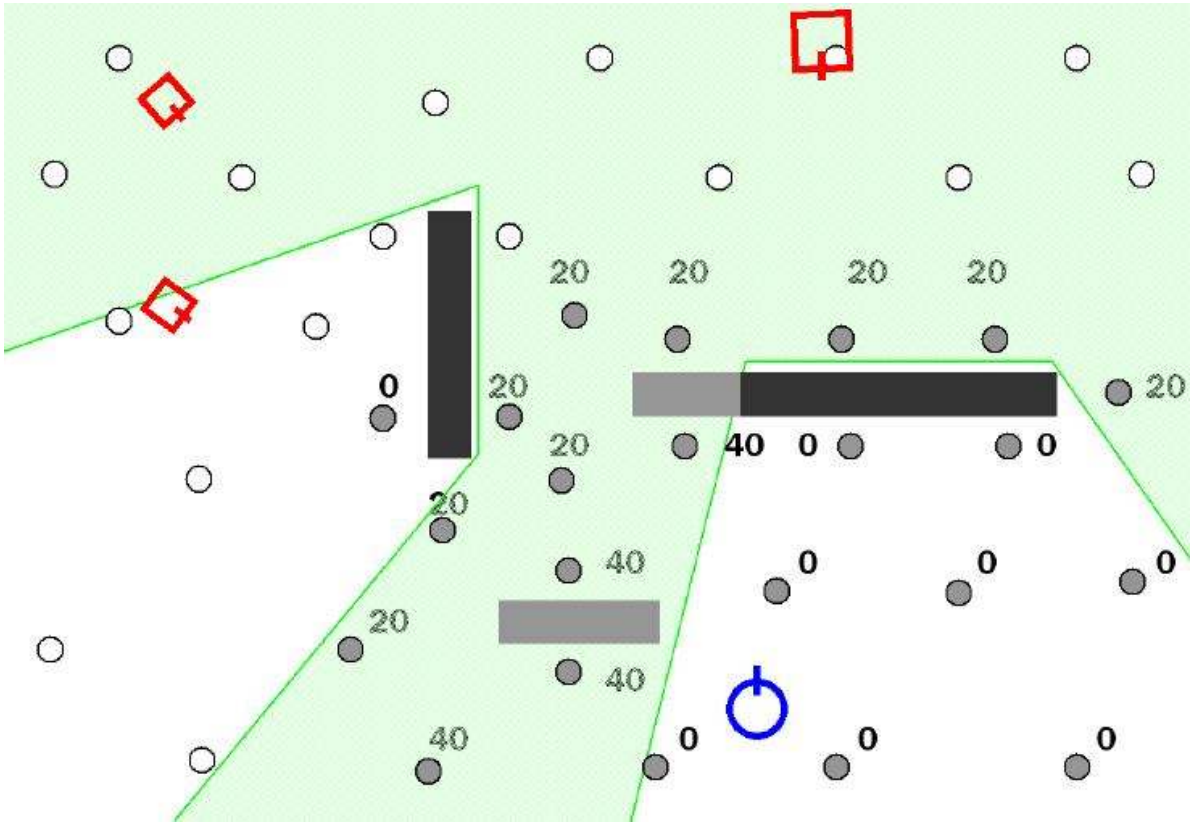
完全な精度ではないが、射線判定が軽い演算で出来てしまう！

Step1　　一定半径内のポイントを候補に上げる



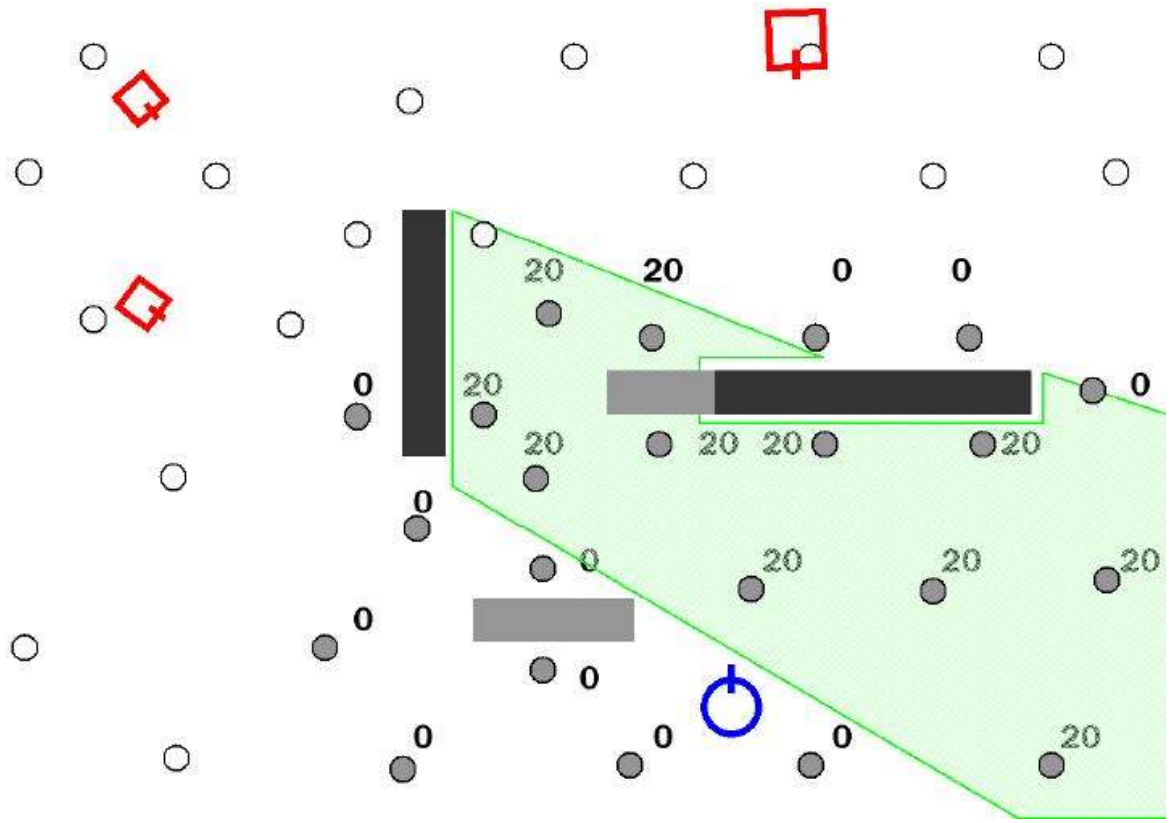
この時点で他のポイントはこれ以降、評価しない　⇒　計算量の軽減

Step2　第一の敵から射線の通るポイントを評価する

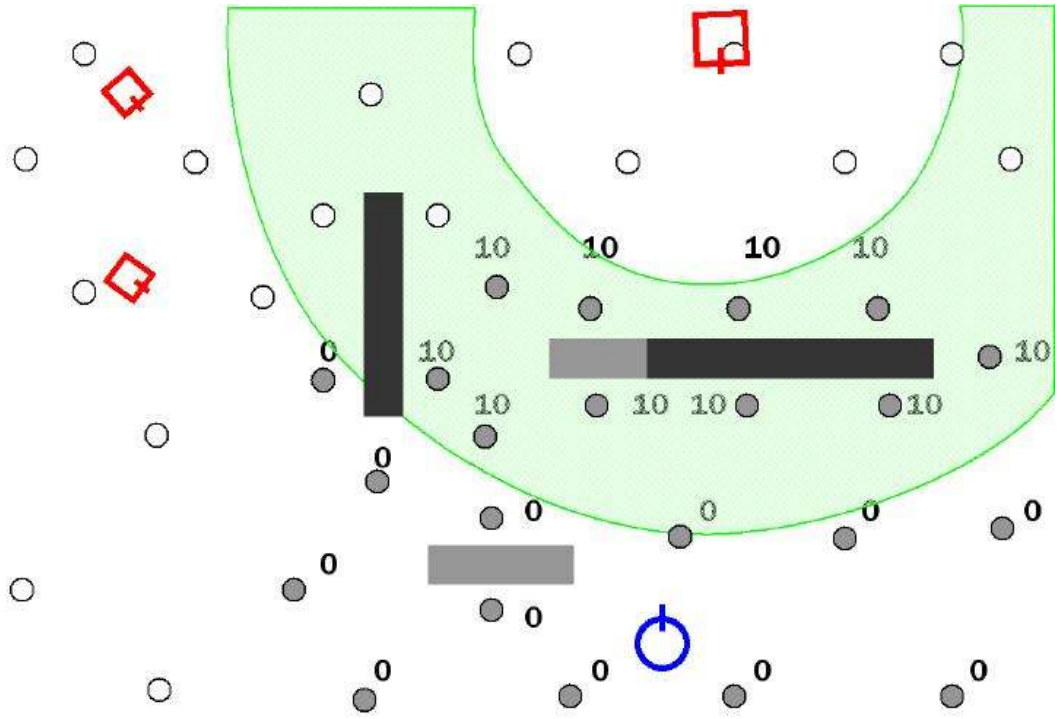


４０点　…　立った状態のみ射線が通る
２０点　…　座っても立っても射線が通る

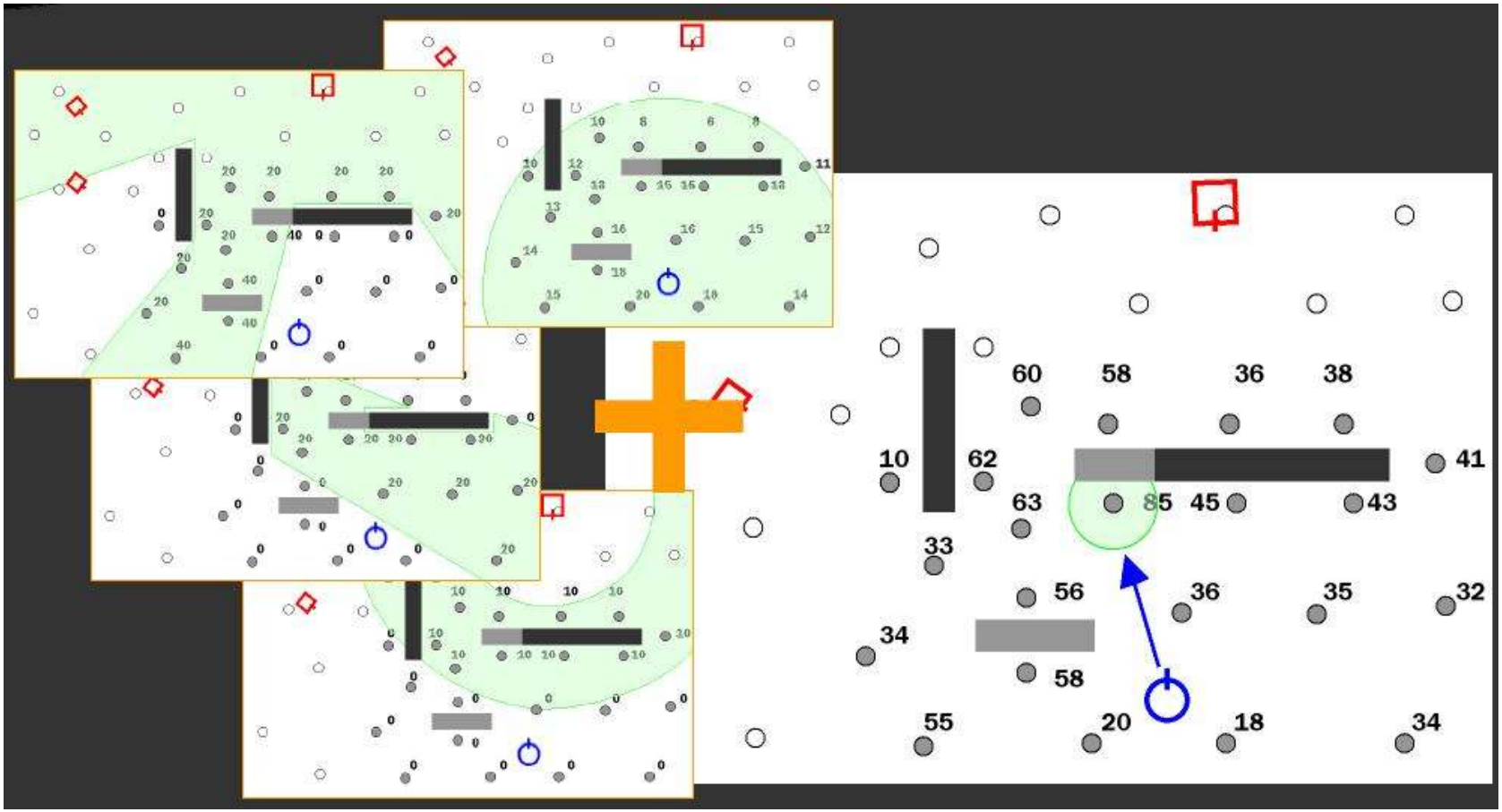
Step3　第二の敵たちから射線の通らないポイントを評価する



Step4 攻撃最適領域内のポイントを評価する



Step5　Step1~4の評価値を重みをかけて足し合わせる

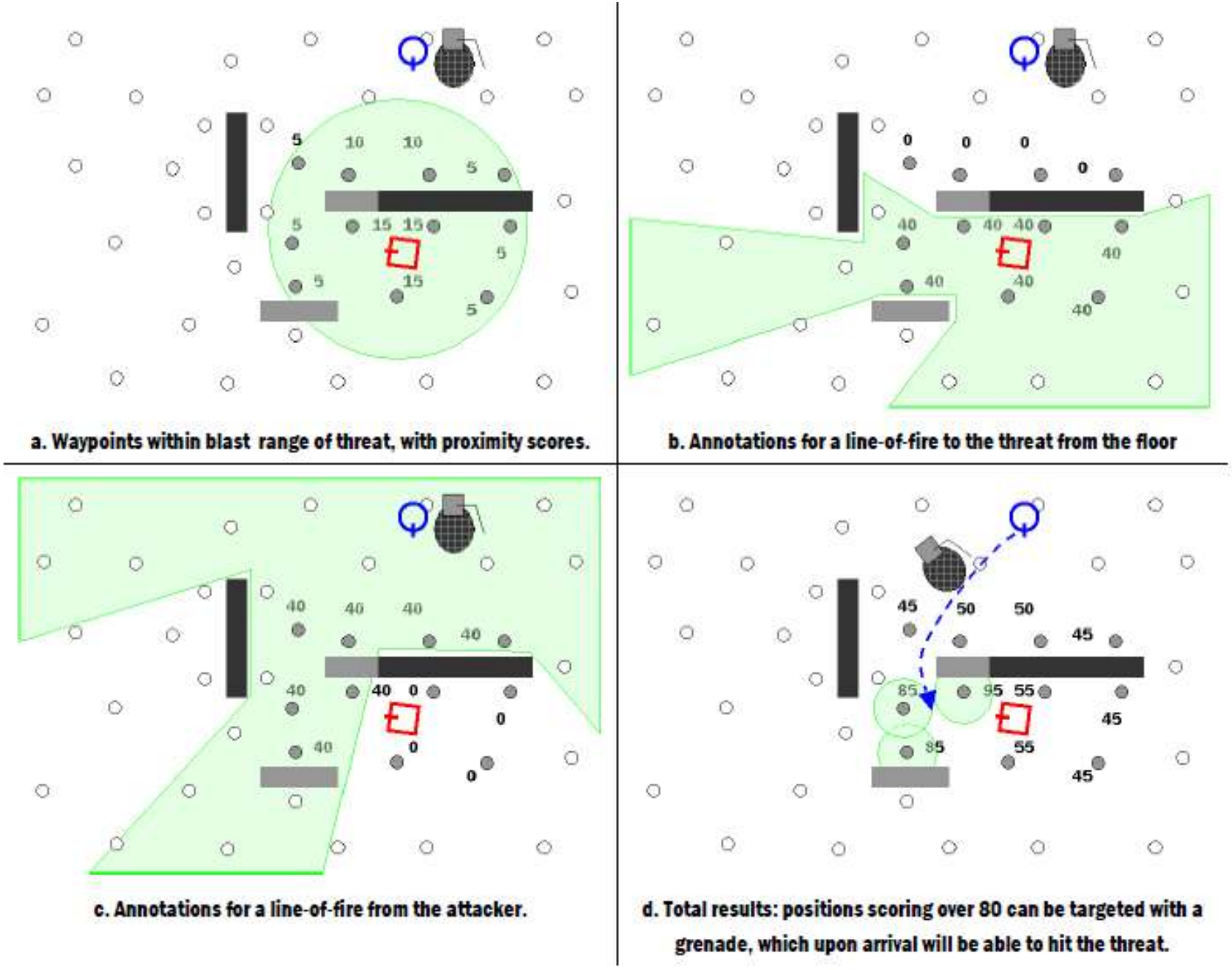


動的な戦術位置検出

動作例２

敵が隠れている場所から移動しそうな場所に手榴弾を投げる

(2) 敵が隠れている場所から移動しそうな場所に手榴弾を投げる



a. Waypoints within blast range of threat, with proximity scores.

b. Annotations for a line-of-fire to the threat from the floor

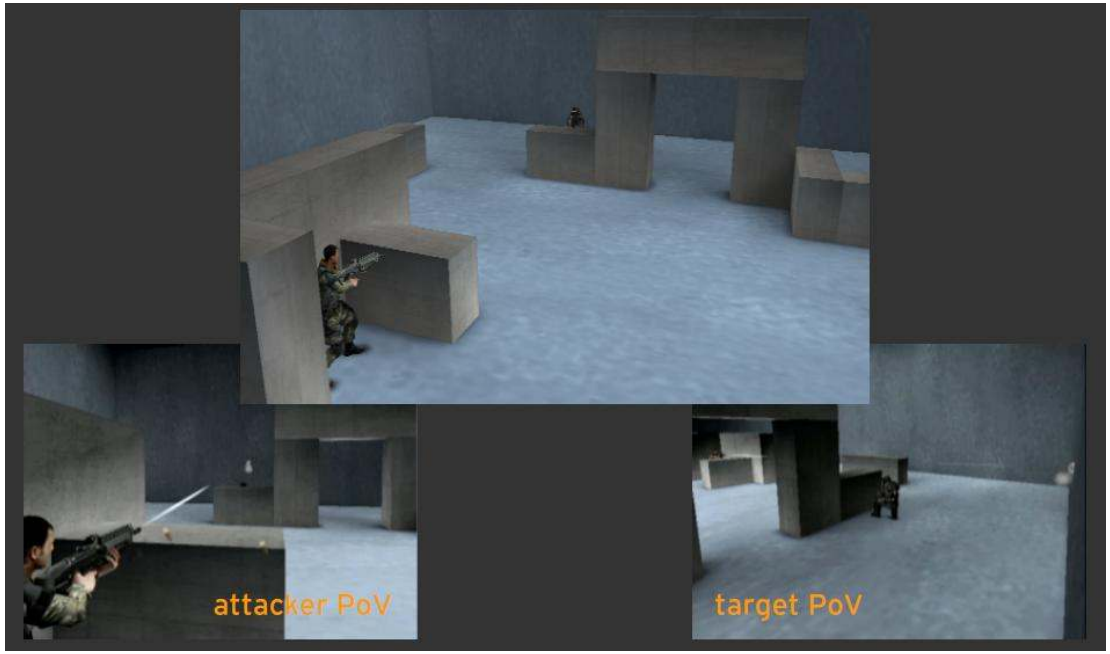
c. Annotations for a line-of-fire from the attacker.

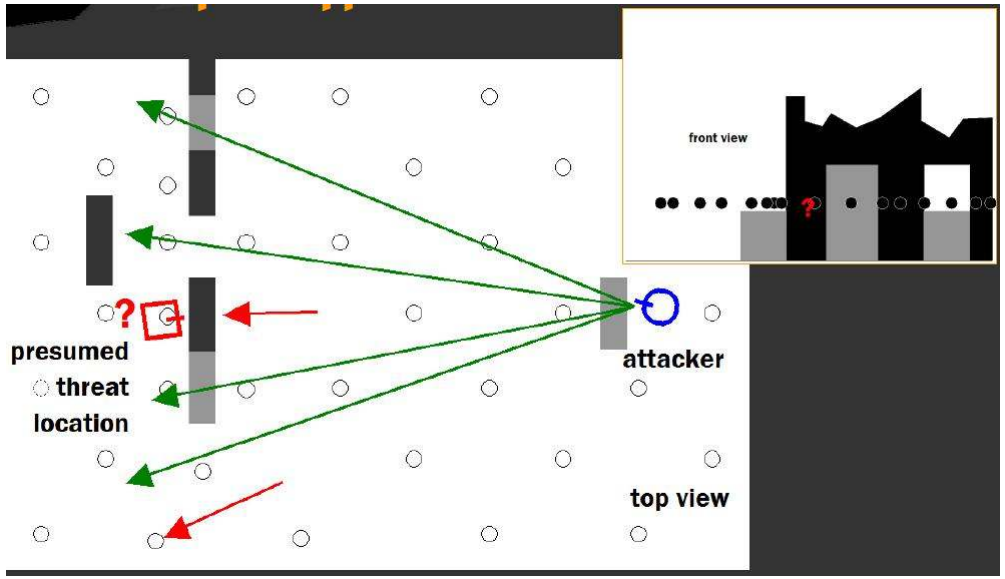
d. Total results: positions scoring over 80 can be targeted with a grenade, which upon arrival will be able to hit the threat.

動作例３

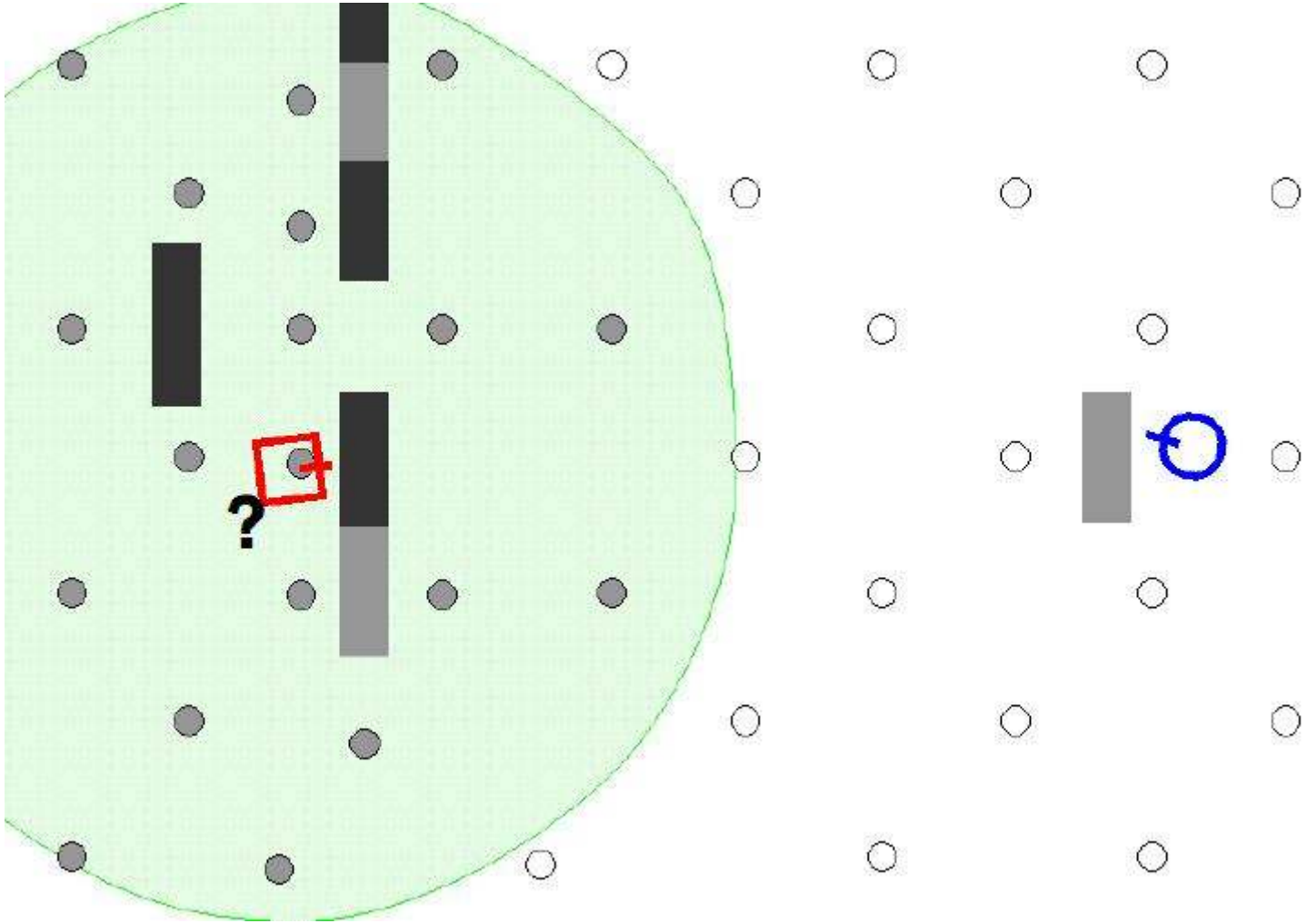
隠れている敵に対して威嚇射撃を行う

（3） 隠れている敵に対して威嚇射撃を行う

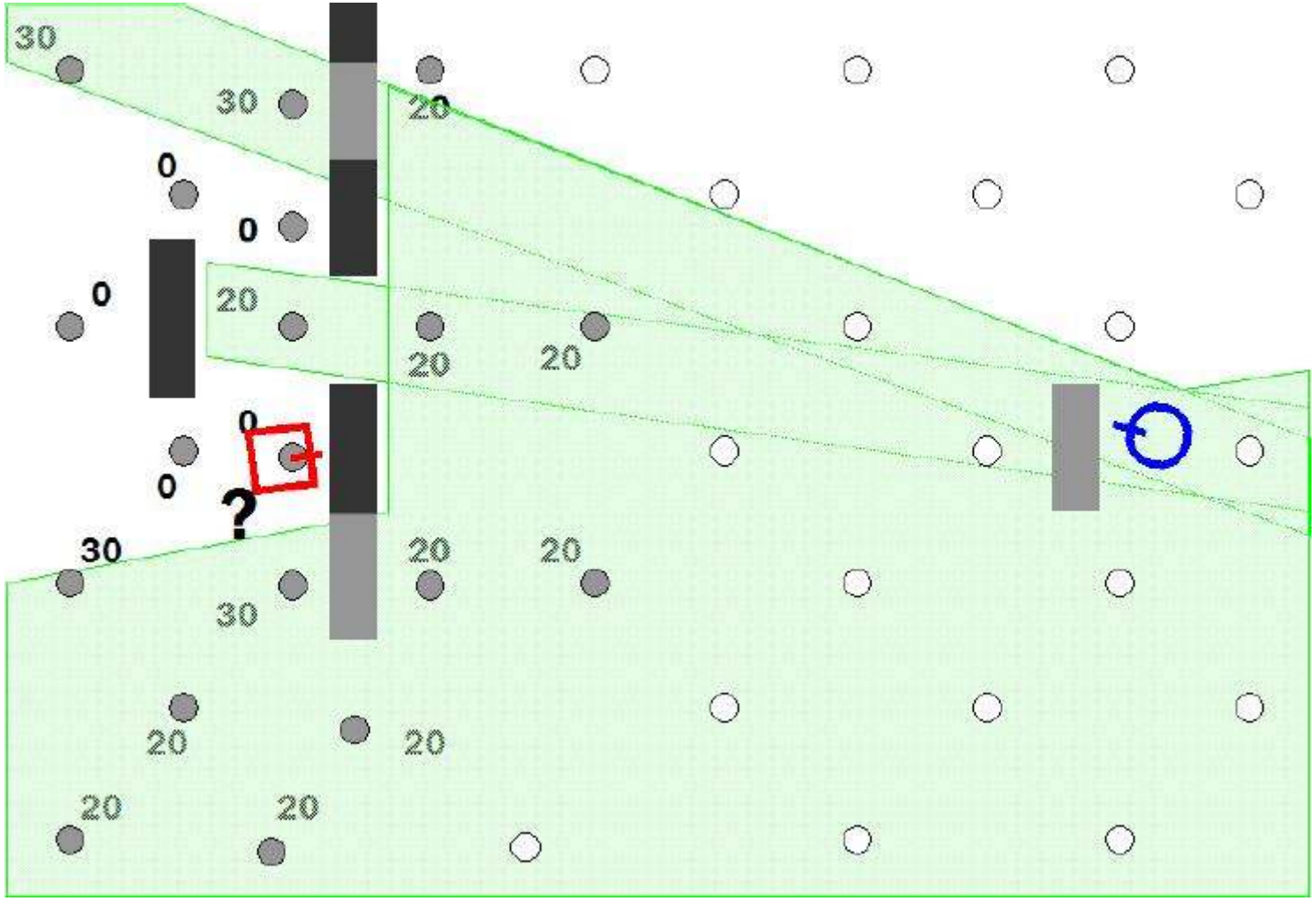




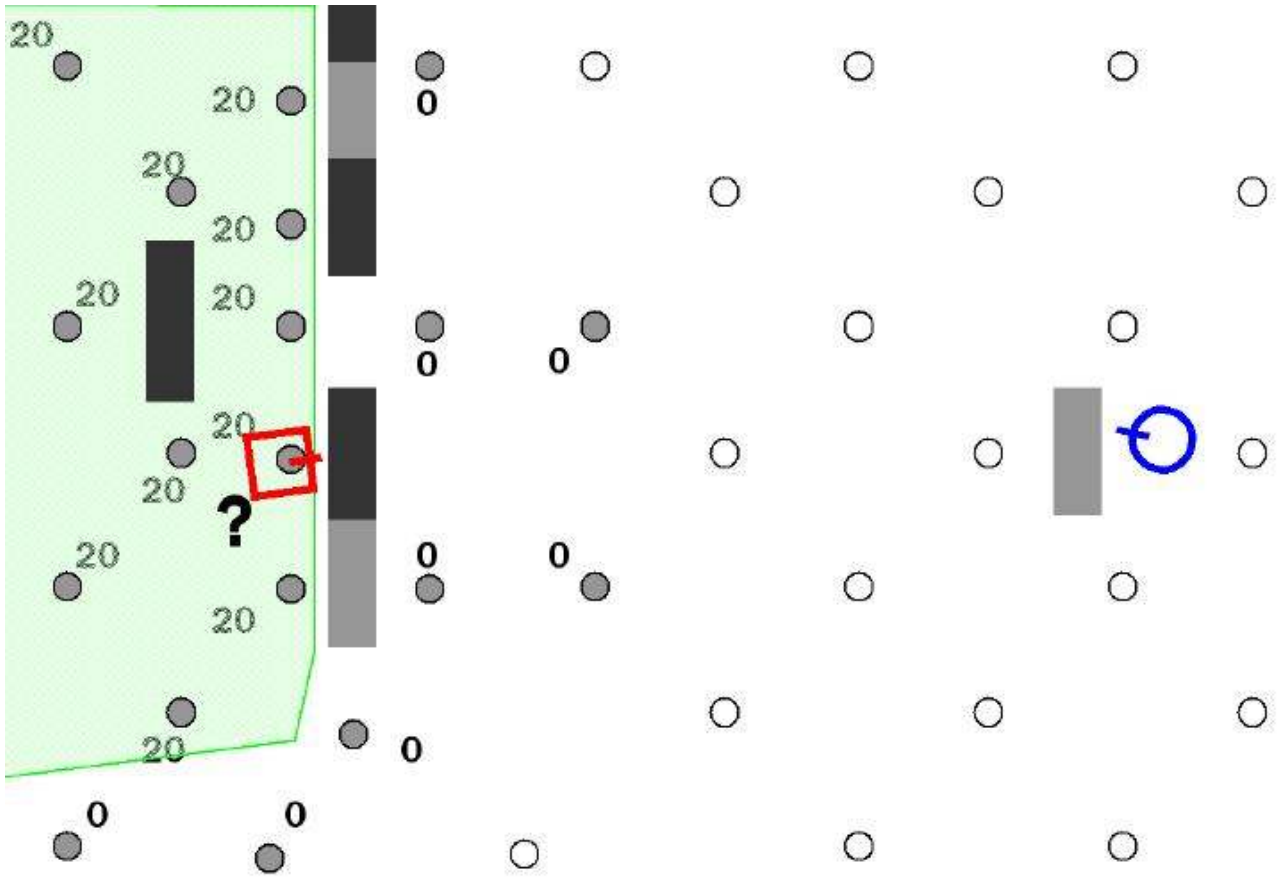
Step1　威嚇する敵一定半径内のポイントを候補に上げる



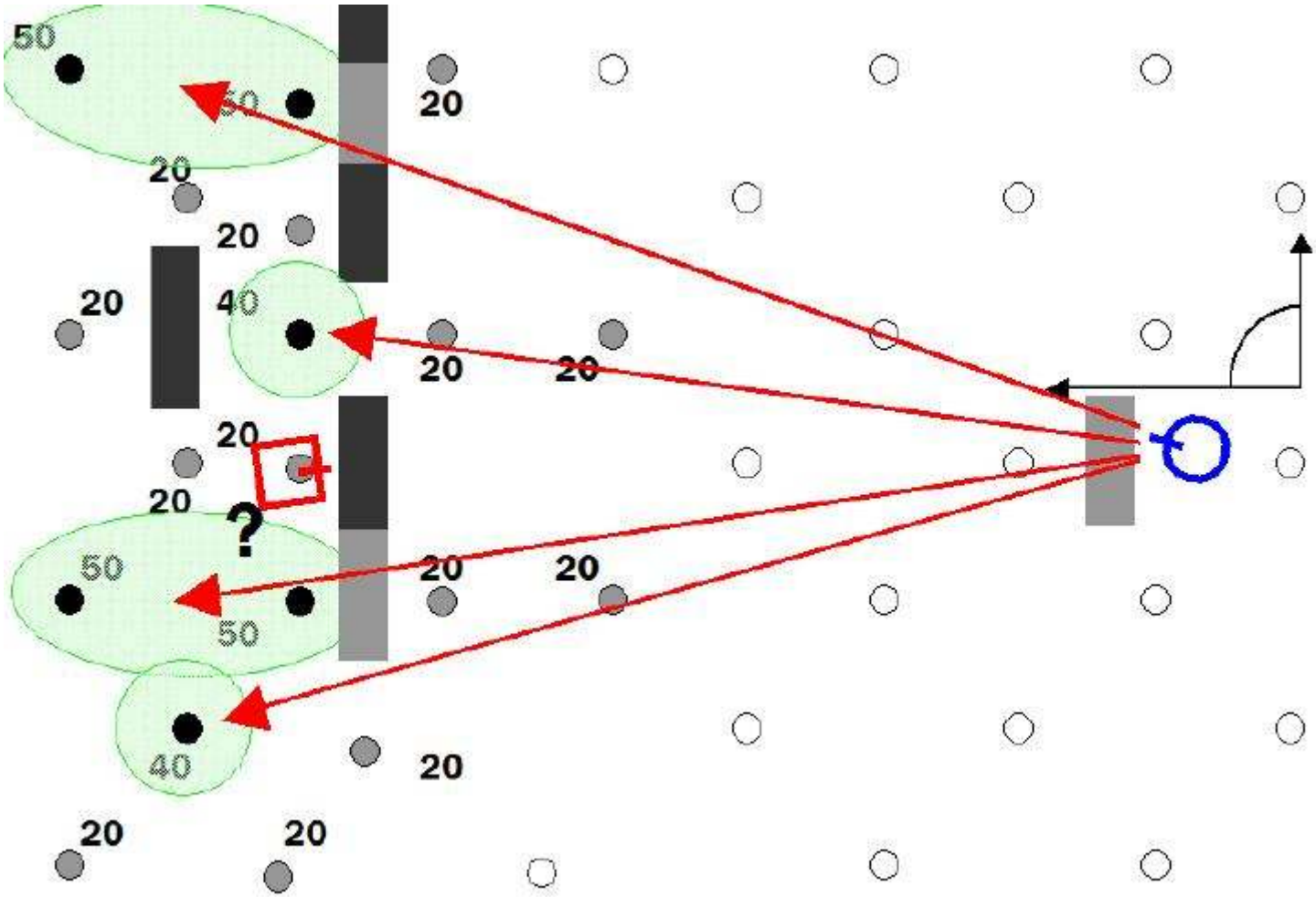
Step2　自分（COM）の射線が通るポイントを評価する



Step3 COMの方向と逆に敵が遮蔽物が側にあるポイントを評価する



Step4　Step1~Step3 を総合して、威嚇ポイントの候補を上げ、
COMから見て同じ立体角にあるものはまとめる



References for Killzone

[1] **William van der Sterren** (2001),"Terrain Reasoning for 3D Action Games", *http://www.cgf-ai.com/docs/gdc2001_paper.pdf*

[2] **William van der Sterren** (2001) ,"Terrain Reasoning for 3D Action Games(GDC2001 PPT)", *http://www.cgf-ai.com/docs/gdc2001_slides.pdf*

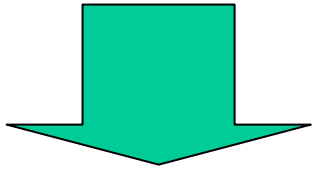
[3] **Remco Straatman, Arjen Beij, William van der Sterren** (2005) ,"Killzone's AI : Dynamic Procedural Combat Tactics", *http://www.cgf-ai.com/docs/straatman_remco_killzone_ai.pdf*

[4] **Arjen Beij, William van der Sterren** (2005),"Killzone's AI : Dynamic Procedural Combat Tactics (GDC2005)", *http://www.cgf-ai.com/docs/killzone_ai_gdc2005_slides.pdf*

[5] **Damian Isla** (2005), "Dude, where's my Warthog? From Pathfinding to General Spatial Competence", *http://www.aiide.org/aiide2005/talks/isla.ppt*

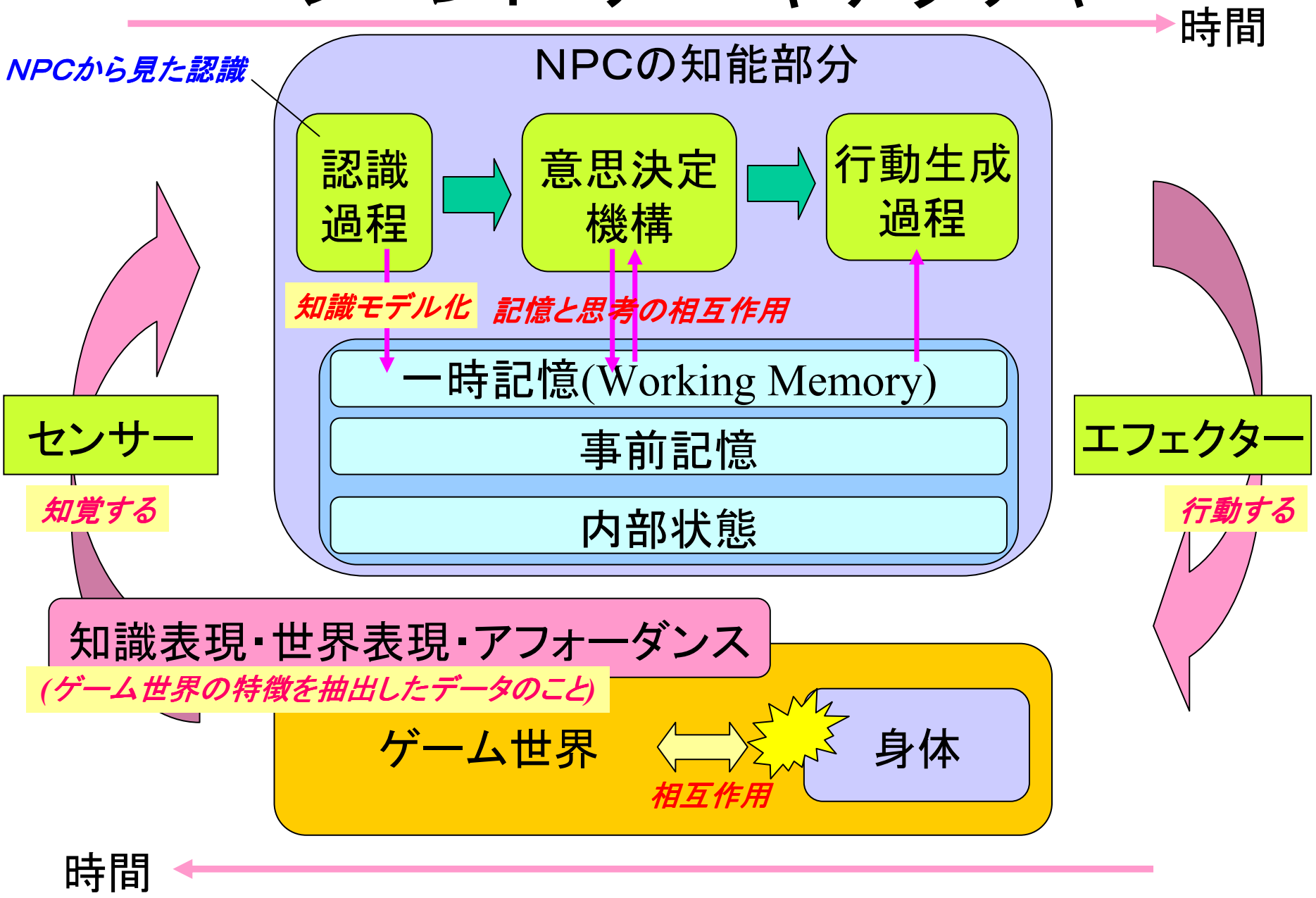
知識表現・世界表現・アフォーダンスを理解しよう

① アフォーダンス
② 知識表現・世界表現

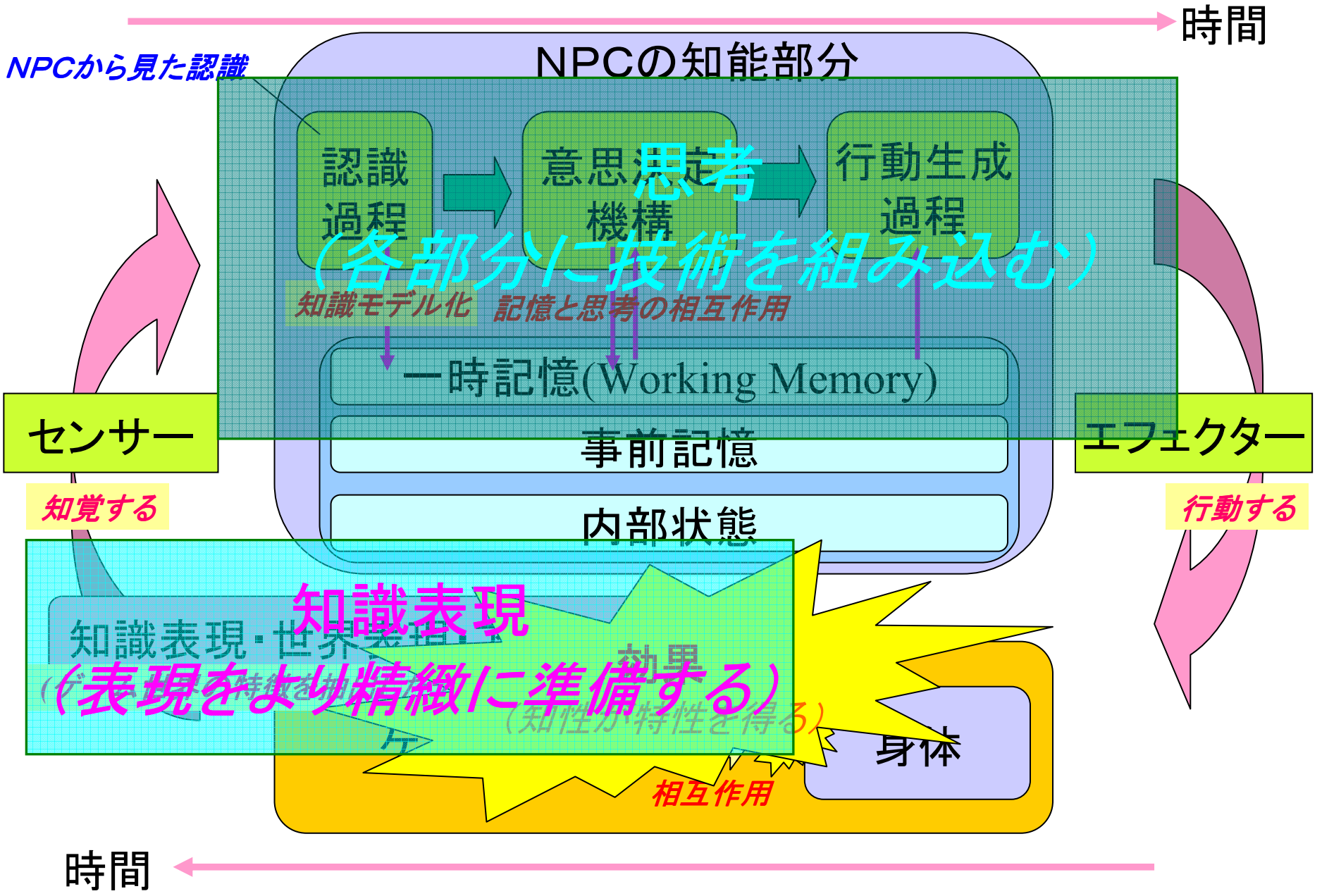


世界と知能を結び付ける情報

エージェント・アーキテクチャー



『技術を仕掛けて効果を得る』



第2章まとめ

これで「エージェント・アーキテクチャ」の概要の説明は
終わりました。

このエージェント・アーキテクチャのフレームの上に
たくさんの技術を整理・蓄積して行くことができます。

そして、その技術がゲームにおいて、
一体どういった効果が得られるか、を研究することが、

「エージェント・アーキテクチャによるキャラクターAIの研究」

です。

第３章

エージェント・アーキテクチャ

空間編

この章では、前章で説明した
「エージェント・アーキテクチャ」上の
さまざまな技術を見て行くことにしましょう。

特にゲームAIでは、
ゲーム世界の空間表現（世界表現）を
充実させることでAIを賢くさせる、
という性質があります。

この章では、そういった実例を解説します。

ゲームAIの目標

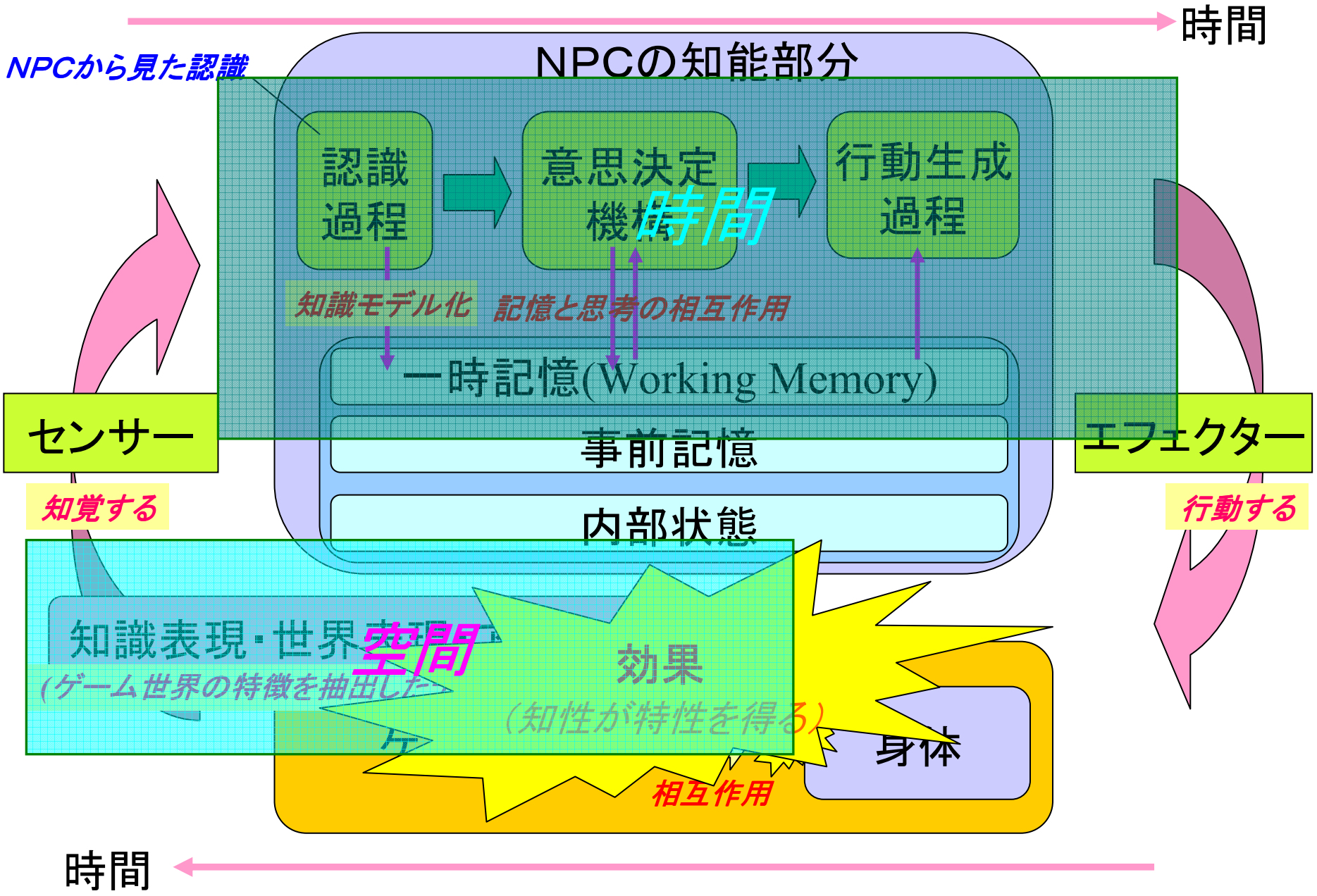
①最初の目標　　時間と空間を支配できるAI

（技術的目標）

②最後の目標　　ユーザーの心理をコントロールできるAI

（エンターテインメントAIとしての目標）

『技術を仕掛けて効果を得る』



世界表現上の4つの技術

(1) Influence Map

細かく区切ったマップの単位に情報を載せる手法

（例）Killzone, Chromehounds

(2) Smart Terrain(Object)

オブジェクト側にキャラクターを制御する情報を載せる手法

（例）The Sims

(3) Terrain Analysis　地形解析

地形から特徴的な情報を抽出する手法

（例）Age of Empire (AOE)

(4) パス検索

任意の２点間の経路を探索する

（例）カウンターストライクなど多数

第3章　エージェント・アーキテクチャ　空間編

(1) Influence Map

細かく区切ったマップの単位に情報を載せる手法

（例）Killzone

Age of Empire

Chromehounds

(2) Smart Terrain(Object)

オブジェクト側にキャラクターを制御する情報を載せる手法

（例）The Sims

(3) Terrain Analysis　地形解析

地形から特徴的な情報を抽出する手法

（例）Age of Empire (AOE)

(4) パス検索

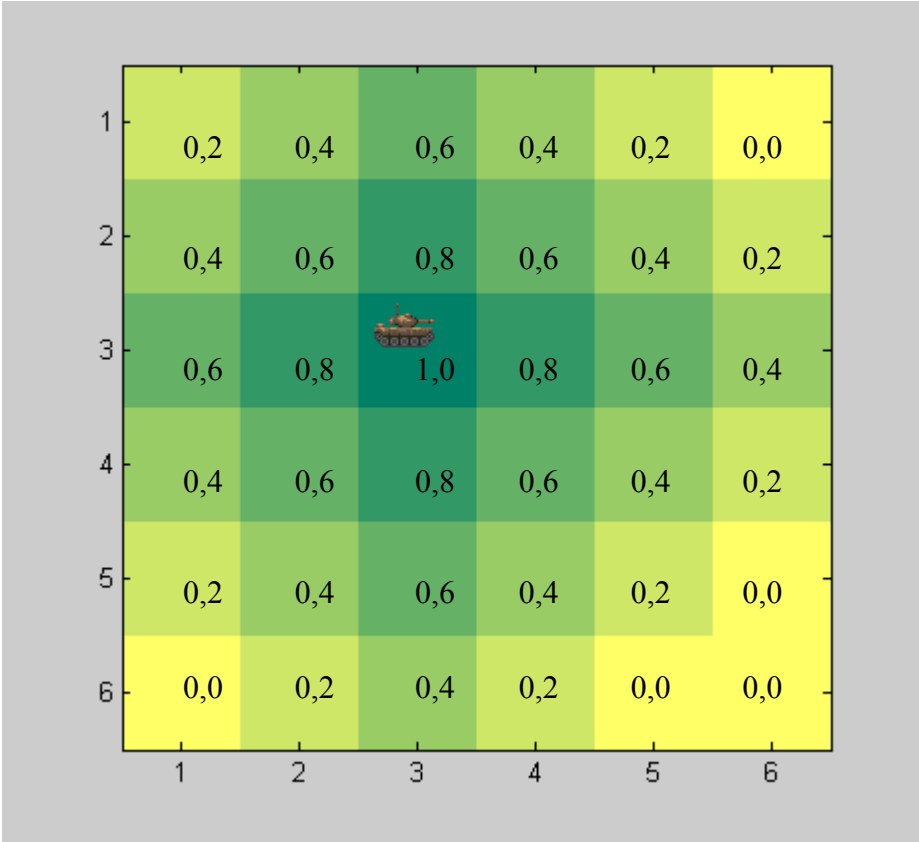
任意の２点間の経路を探索する

（例）カウンターストライクなど多数

Influence Map(影響マップ)

セル分割されたマップに、問題とする性質の評価値を記録して行く方法

（例）①占有度

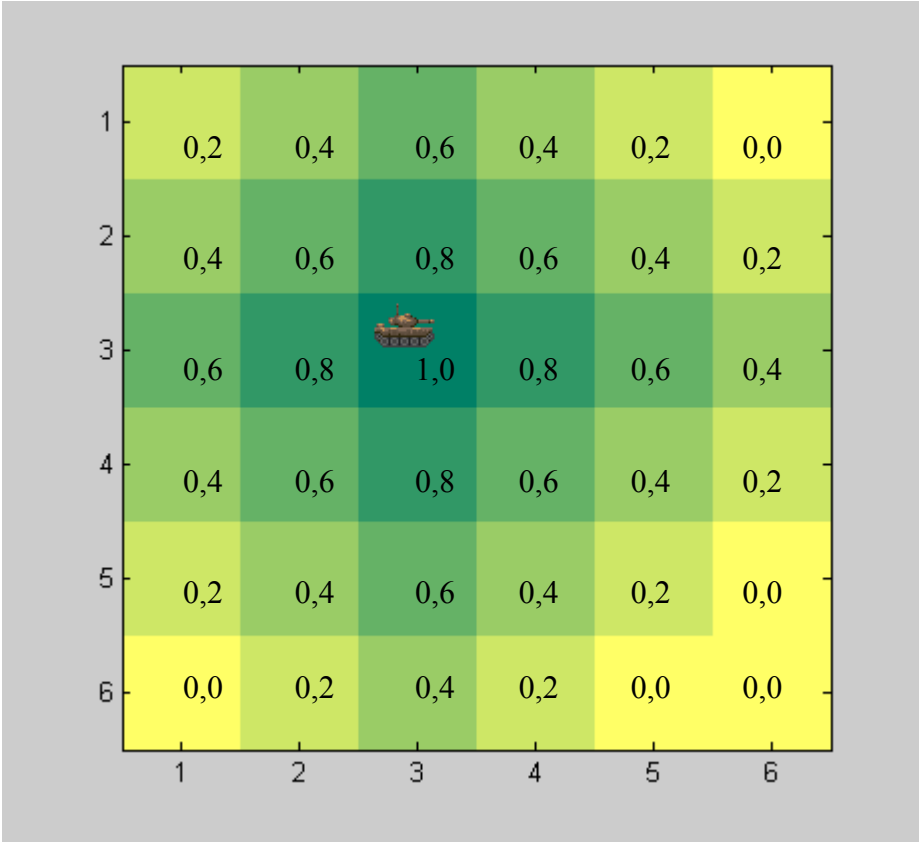


http://www.fdaw.unimaas.nl/education/4.5GAI/slides/Influence%20Maps.ppt

Influence Map(影響マップ)

セル分割されたマップに、問題とする性質の評価値を記録して行く方法

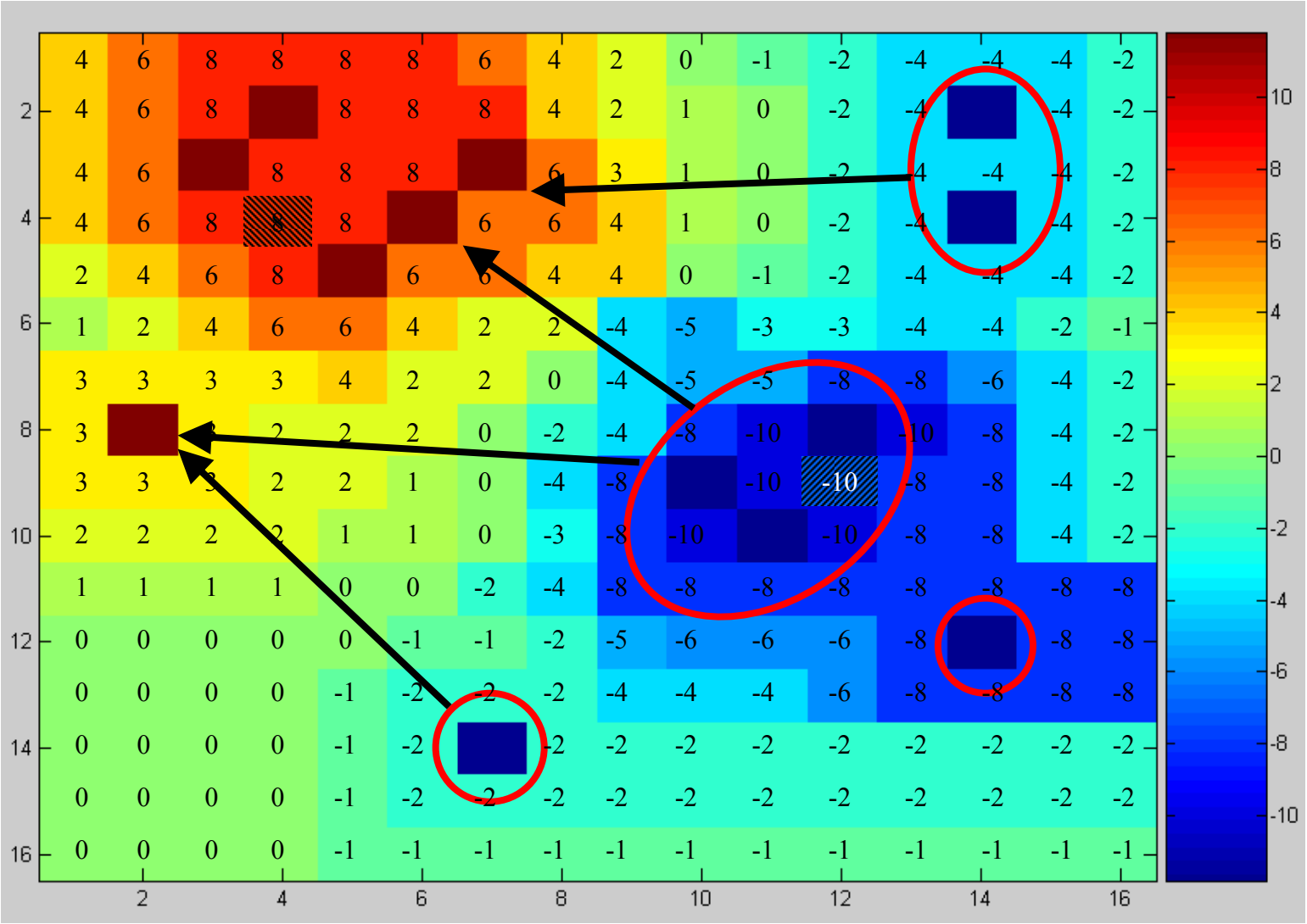
（例）①占有度
②通過可能確率



http://www.fdaw.unimaas.nl/education/4.5GAI/slides/Influence%20Maps.ppt

Influence Map(影響マップ)

（例）様々なIMから計算して戦略的な位置取りを計算する



http://www.fdaw.unimaas.nl/education/4.5GAI/slides/Influence%20Maps.ppt

第3章　エージェント・アーキテクチャ　空間編

(1) Influence Map

細かく区切ったマップの単位に情報を載せる手法

（例）Killzone

Age of Empire

Chromehounds

(2) Smart Terrain(Object)

オブジェクト側にキャラクターを制御する情報を載せる手法

（例）The Sims

(3) Terrain Analysis　地形解析

地形から特徴的な情報を抽出する手法

（例）Age of Empire (AOE)

(4) パス検索

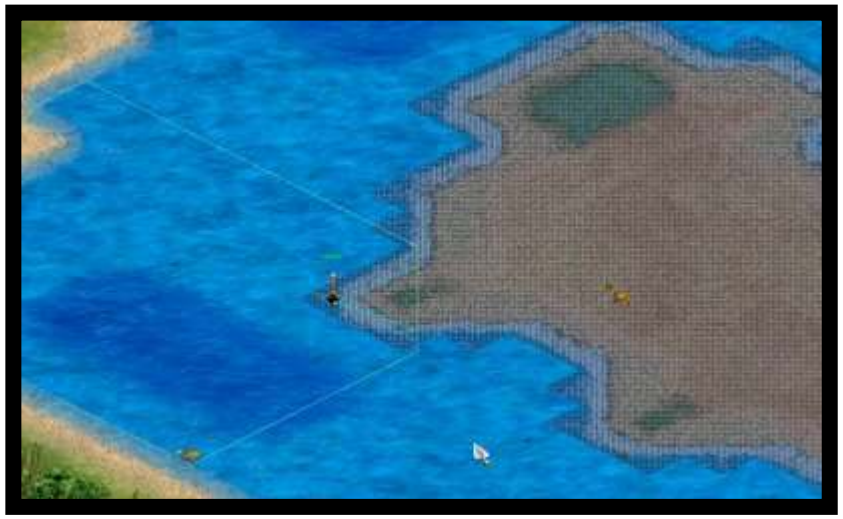
任意の２点間の経路を探索する

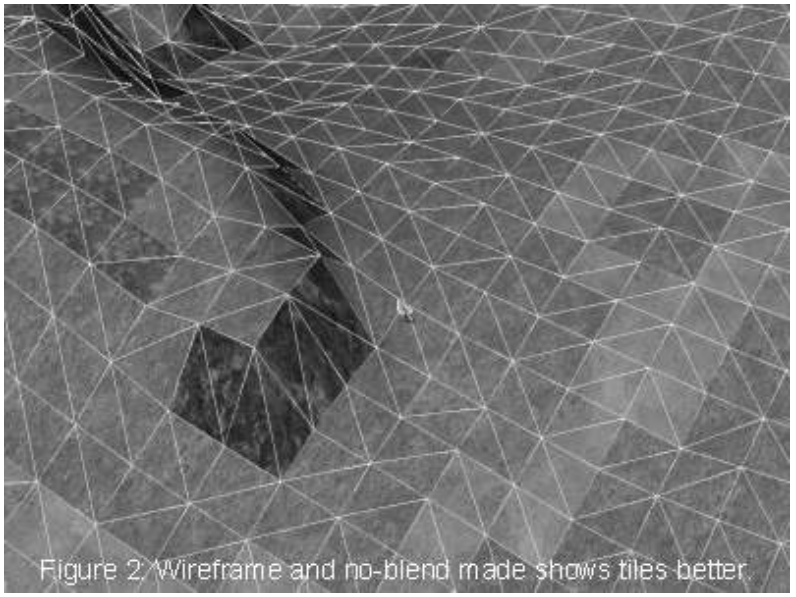
（例）カウンターストライクなど多数

Age of Empire(AOE)における利用例

AOEはタイルベースの
RTS（リアルタイムストラテジー）

マップは毎回、自動生成

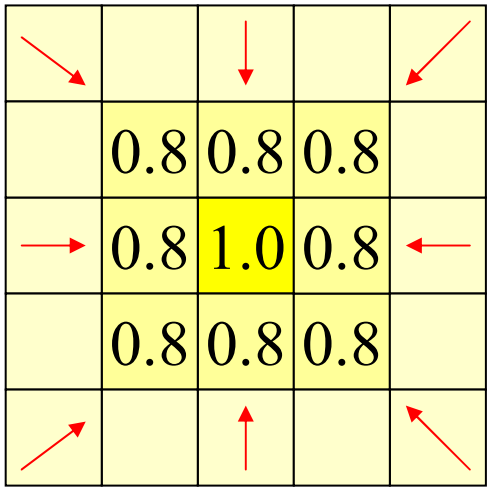




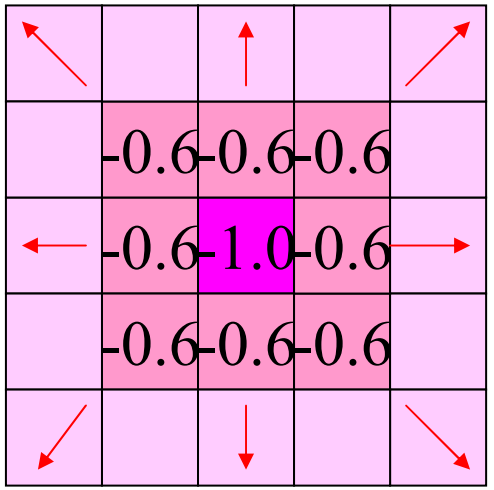
Age of Empire(AOE)における利用例①

金鉱発掘小屋の最適位置自動検出

Figure 3: Influence map around gold mines.



金鉱＝アトラクター
(吸引源)



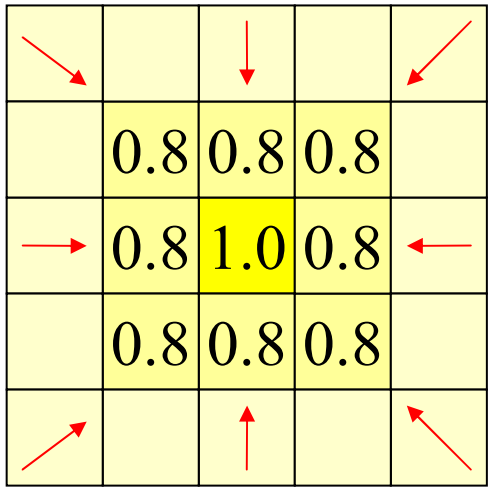
防壁＝デトラクター
（排斥源）

AIに金鉱の発掘小屋を適切な位置に置かせるには？
＝　金鉱に近いが、近すぎてはいけないところに置く

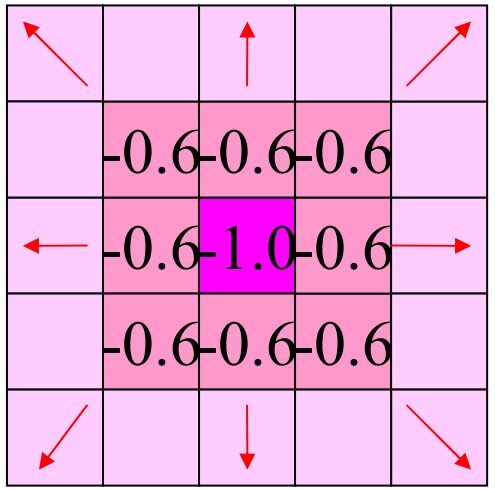
金鉱の回りのタイル（仮想的）防壁を置いて、
近すぎも遠すぎもしない点に最高点のタイルが来るようにする
＝　自動位置検出

Age of Empire(AOE)における利用例②

兵隊の配置の最適位置自動検出



高い場所＝アトラクター
(吸引源)

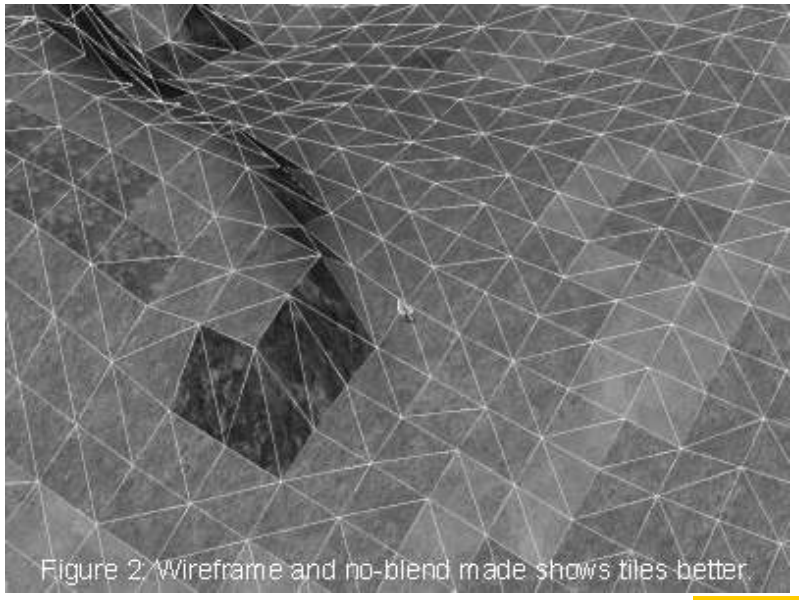


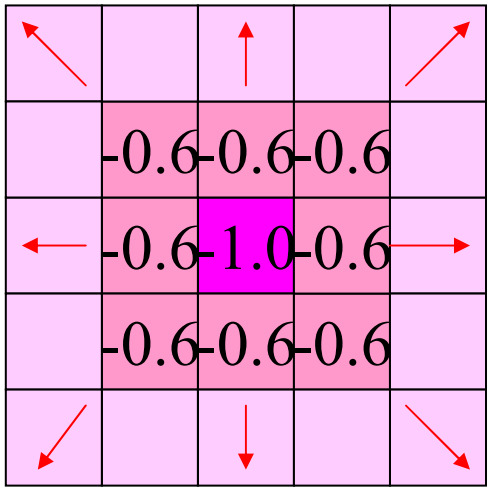
家がある場所＝デトラクター
（排斥源）

兵士を街からなるべく遠く、高台の場所に置かせたい
＝　高い場所(アトラクター)、家がある（デトラクター）とする

Age of Empire(AOE)における利用例③

兵隊の配置の最適位置自動検出

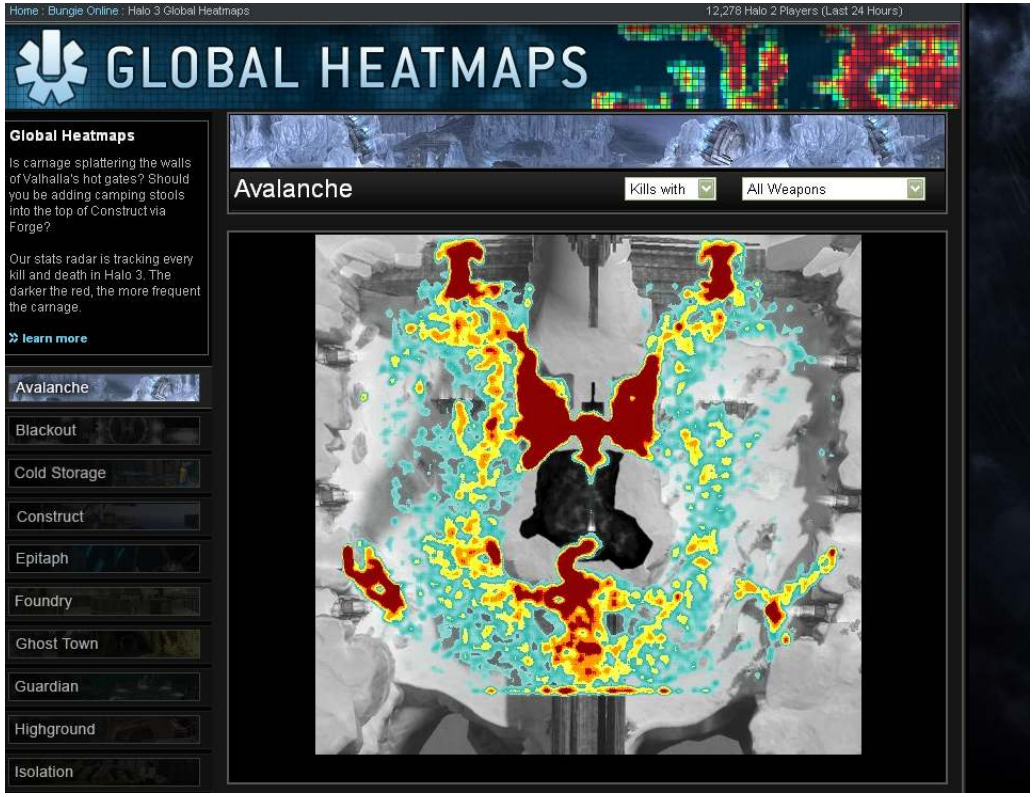


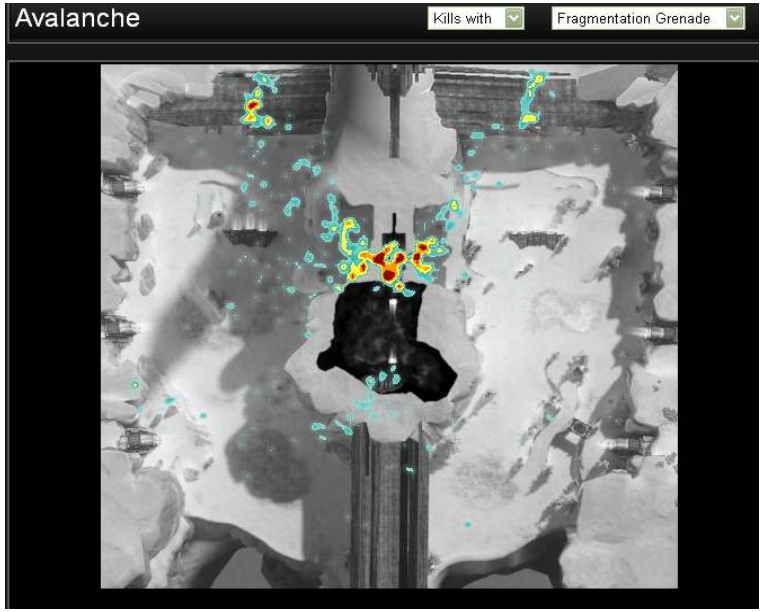


既知の敵のビル、以前通ったルート ＝ デトラクタ（排斥源）

この排斥源が与えるコストをパス検索のコストとすると、
「なるべく遠ったことにない敵の基地からみつかりにくいパス」
を見つけることが出来る。

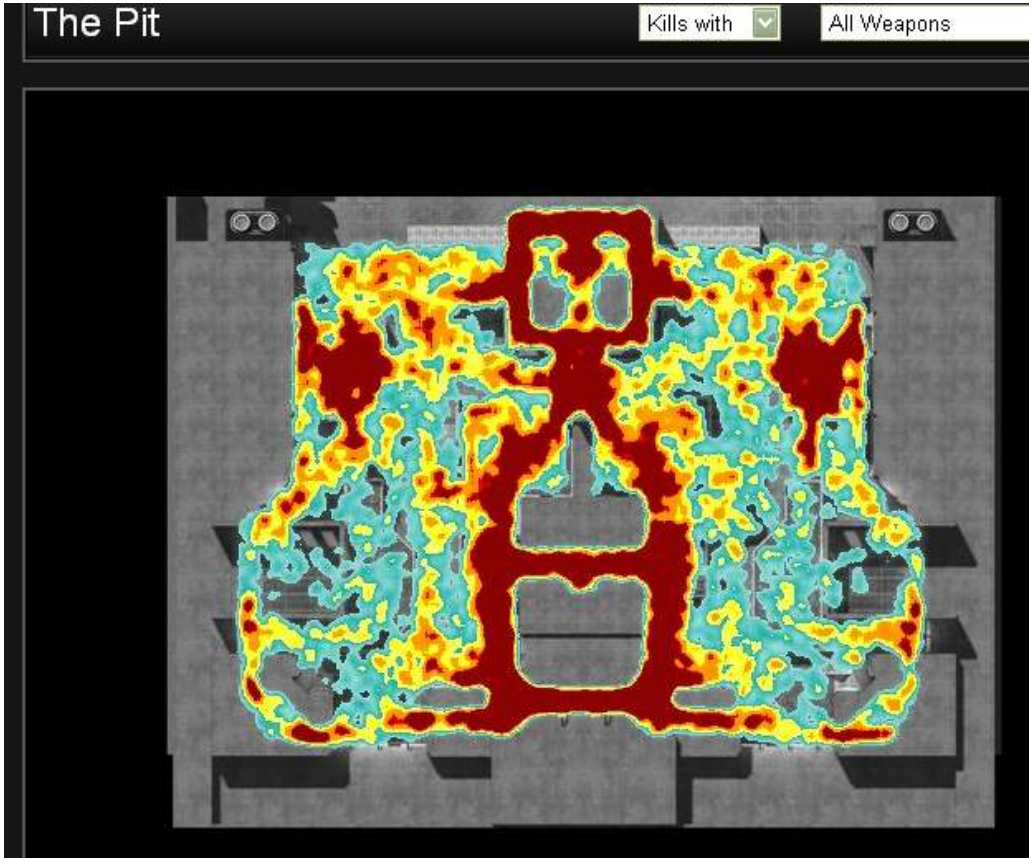
Halo3 Global Heatmaps





http://www.bungie.net/online/heatmaps.aspx

Halo3 Global Heatmaps



第3章　エージェント・アーキテクチャ　空間編

(1) Influence Map

細かく区切ったマップの単位に情報を載せる手法

（例）Killzone

Age of Empire

Chromehounds

(2) Smart Terrain(Object)

オブジェクト側にキャラクターを制御する情報を載せる手法

（例）The Sims

(3) Terrain Analysis　地形解析

地形から特徴的な情報を抽出する手法

（例）Age of Empire (AOE)

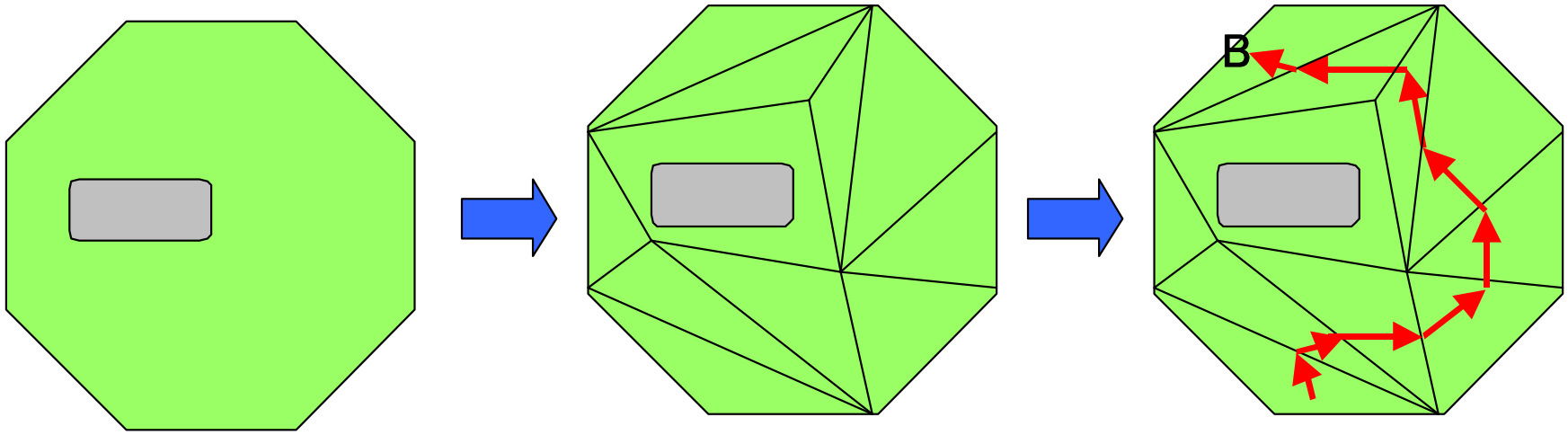
(4) パス検索

任意の２点間の経路を探索する

（例）カウンターストライクなど多数

Navigation Mesh 法とは？

マップを凸多角形で埋めてキャラクターを移動させる方法



手順

（1） マップを障害物を含まない三角形に分割する（データを用意する）

（2） その三角形の情報から、パスを検索する（ゲーム内リアルタイム）。

（3） 検索したパスに従って移動する（ゲーム内リアルタイム）。

ナビゲーションメッシュの利点

適用できる場合

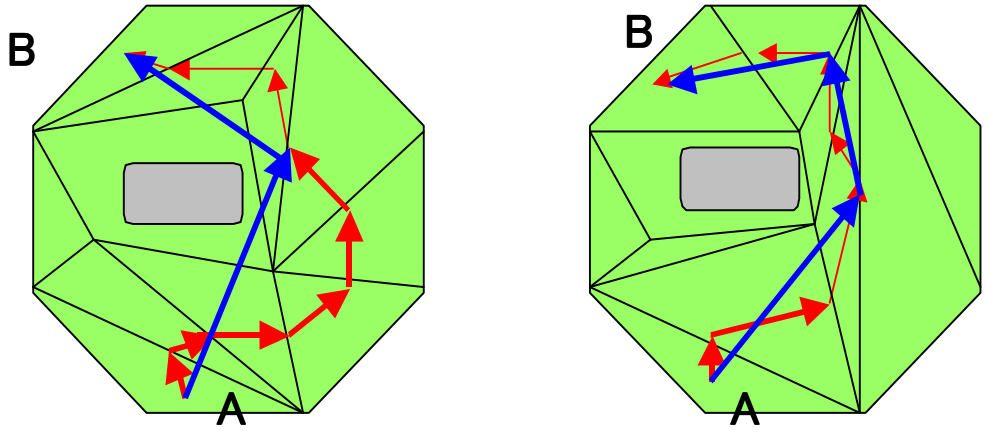
キャラクターが床から離れない場合にのみ有効な方法

利点

（1）マップ上のどの点からどの点でも、
ゲーム中にリアルタイムに経路を探索し移動できる。

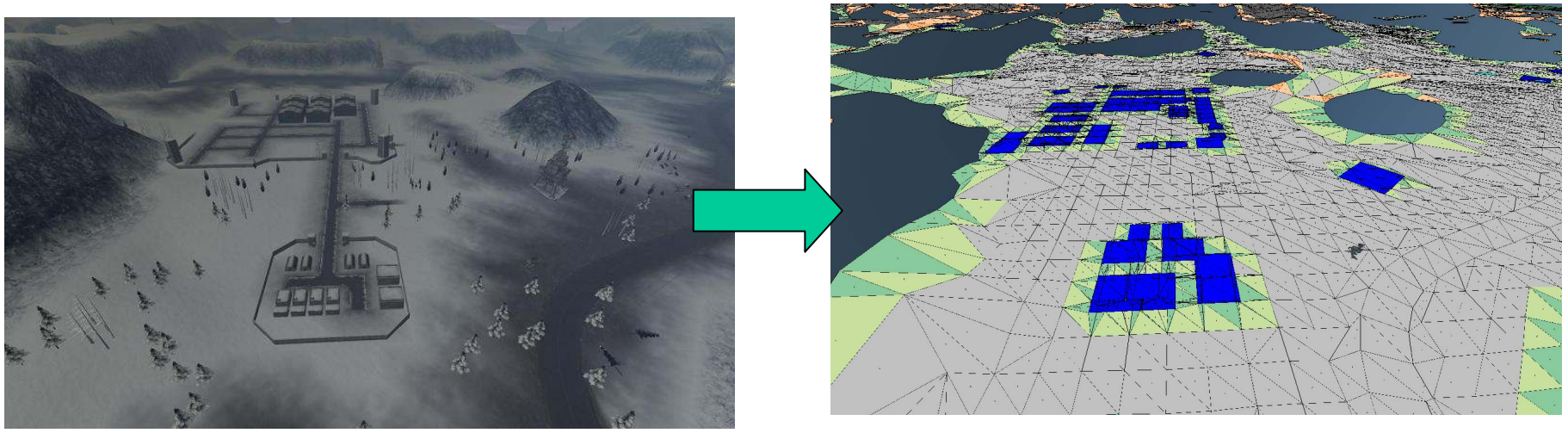
（2） 平面的につながっていれば、どんな地形にも対応できる。

（3） マップの情報をデータに埋め込みやすい。



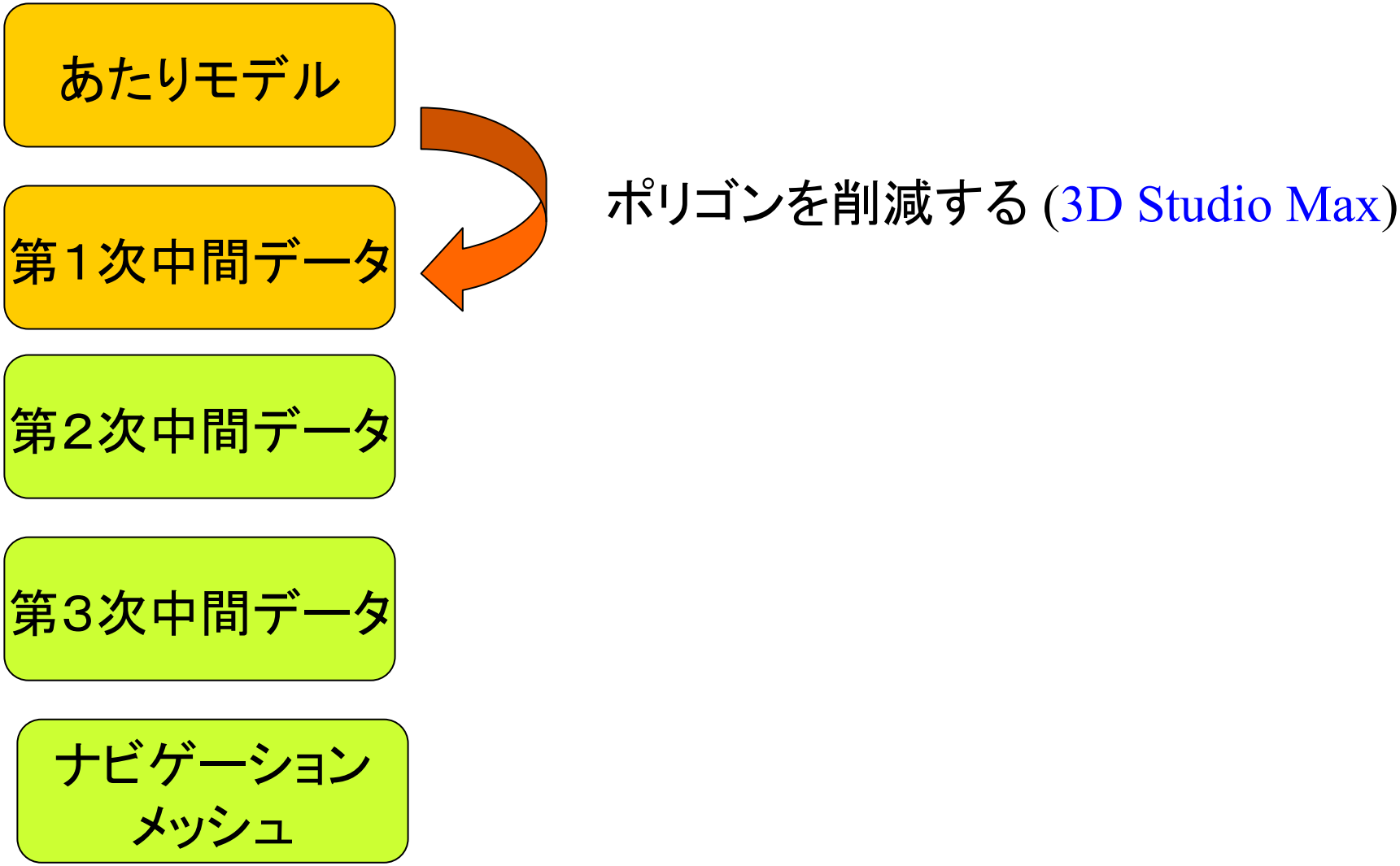
クロムハウンズにおけるナビゲーションメッシュ

(1) 30000 – 80000 メッシュ
(2) 複雑な地形にも対応
(3) メッシュを当りモデルから自動生成
(4) マルチ分解能
(5) 地形表面の性質（雪、砂など）の情報が埋め込まれたメッシュ



山岳、街、湖など、80に及ぶバリエーションに富んだマップに対し、
単一のデータ形式、アルゴリズムで対応することが出来た（汎用性）。

ナビゲーションメッシュ工程



ナビゲーションメッシュ工程

あたりモデル

第１次中間データ

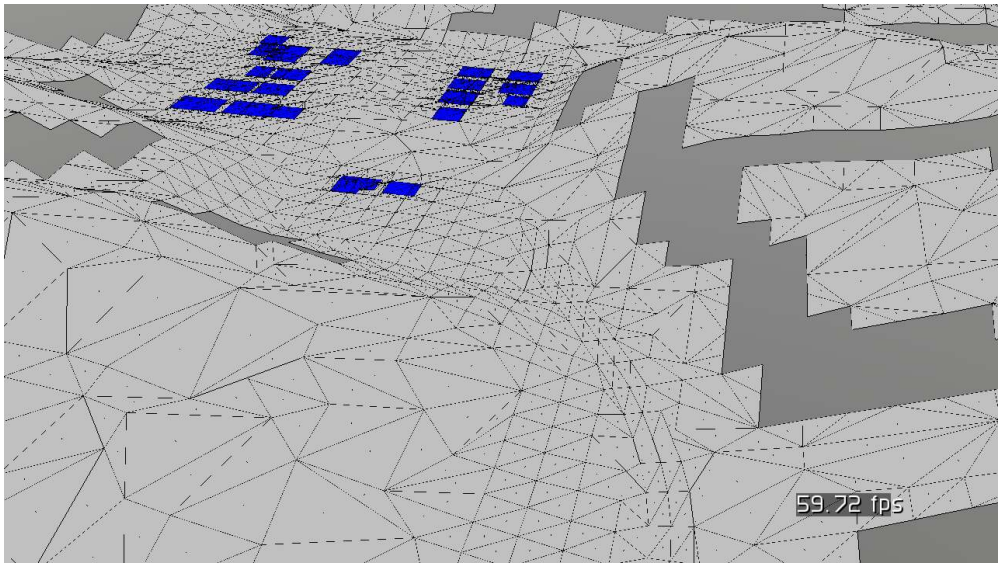
第２次中間データ

第３次中間データ

ナビゲーション
メッシュ

デバッグ

配置オブジェクトとのあたりを取る(3D Studio Max plugin)



ナビゲーションメッシュ工程

あたりモデル

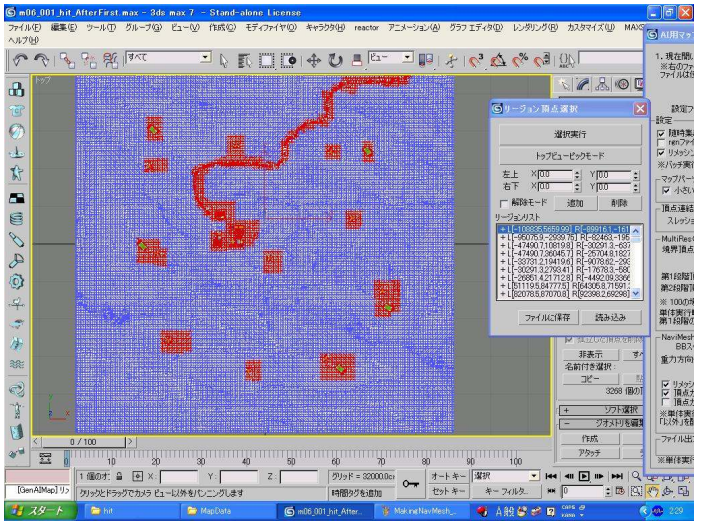
第１次中間データ

第２次中間データ

第３次中間データ

ナビゲーション
メッシュ

デバッグ



ポリゴンを削減する(3D Studio Max)

ナビゲーションメッシュ工程

あたりモデル

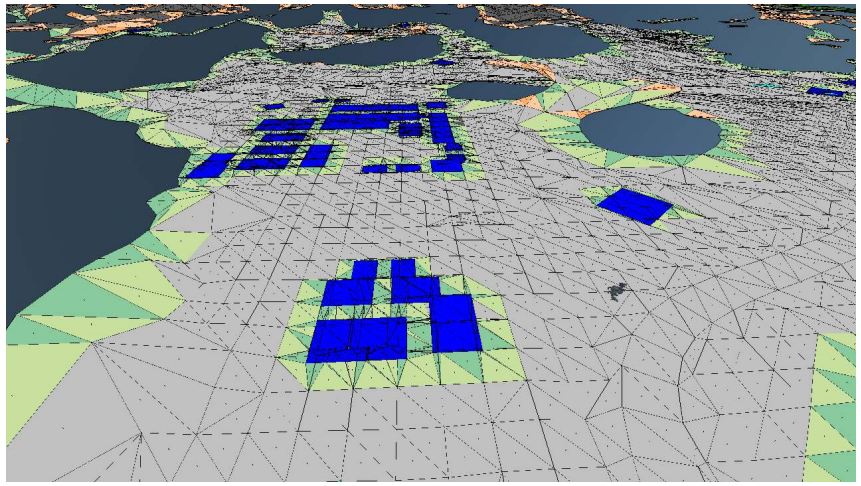
第１次中間データ

第２次中間データ

第３次中間データ

ナビゲーション
メッシュ

デバッグ



プログラムで地形と表面の情報を埋め込む

ナビゲーションメッシュ工程

あたりモデル

第１次中間データ

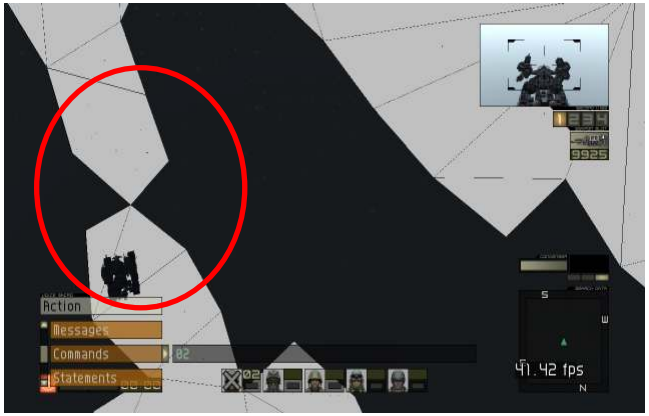
第２次中間データ

第３次中間データ

ナビゲーション
メッシュ

デバッグ

（1） あたりモデルと比較する。



ナビゲーションメッシュ工程

あたりモデル

第１次中間データ

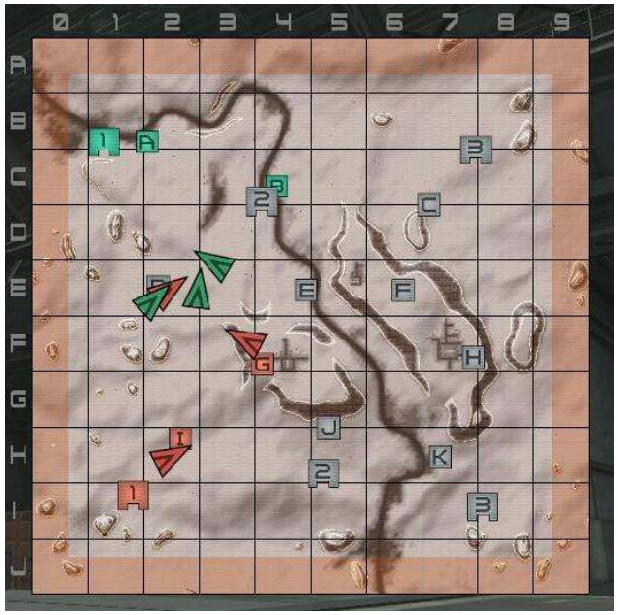
第２次中間データ

第３次中間データ

ナビゲーション
メッシュ

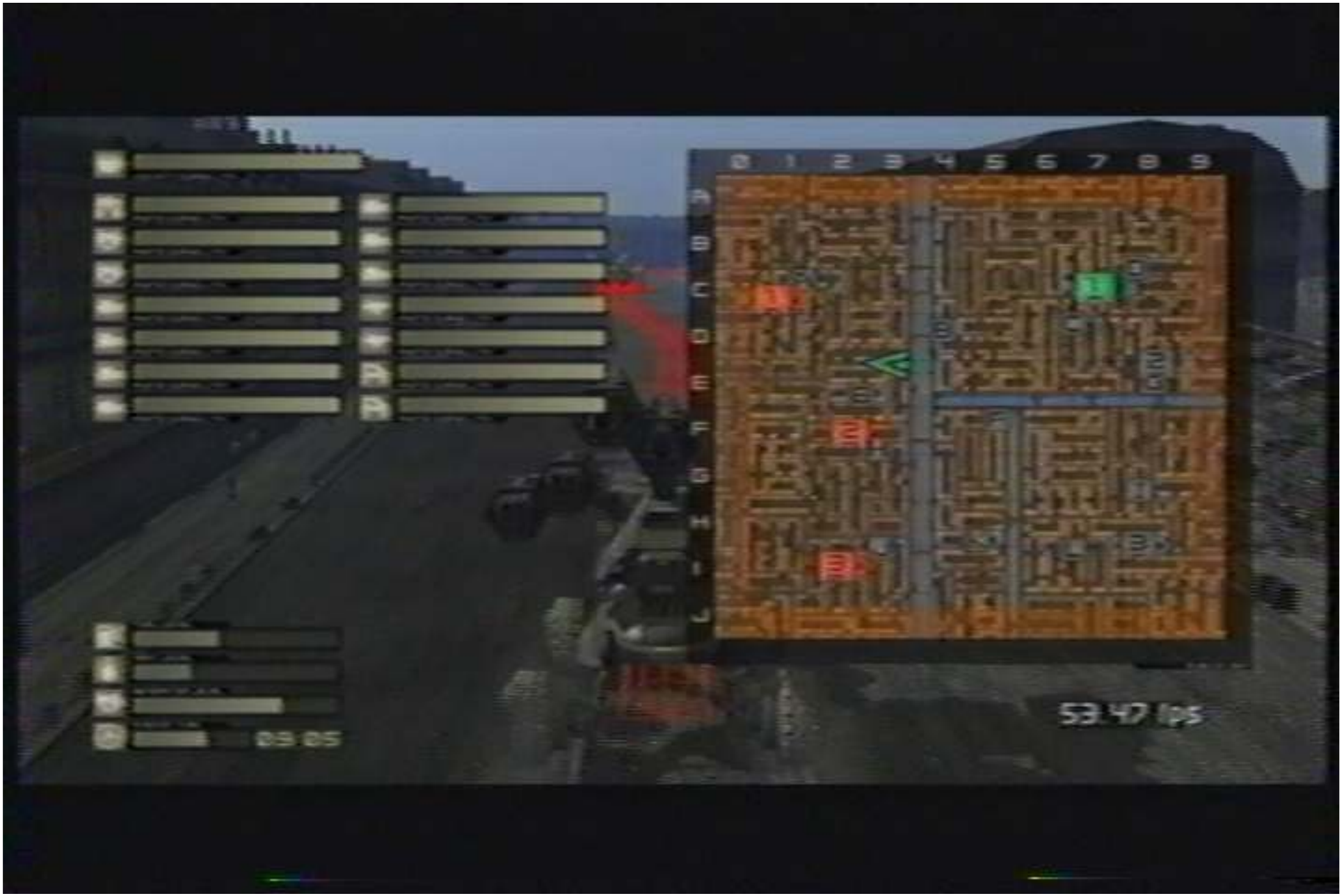
デバッグ

(2) 実機でテストを行う 。

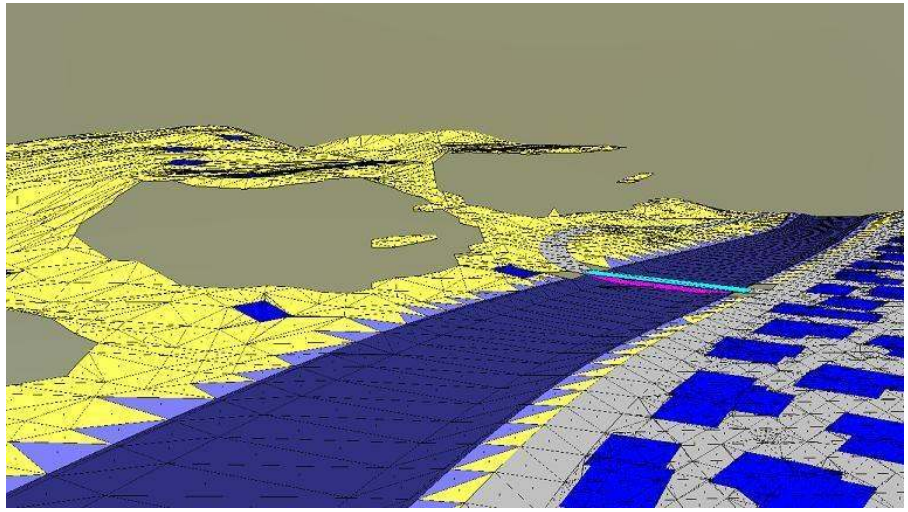


各、本拠地同士を結ぶパスと、
コンバスを巡回するパスをチェック

デバッガーさん
80 x 10 x 2 = 1600 以上のチェック


0 1 2 3 4 5 6 7 8 9
A
B
C
D
E
F
G
H
I
J
53.47 fps

情報が埋め込まれたナビゲーションメッシュ



（1） 水や砂地は、ハウンズのスピードを減速させるので、
メッシュに表面の性質を埋め込んでおく

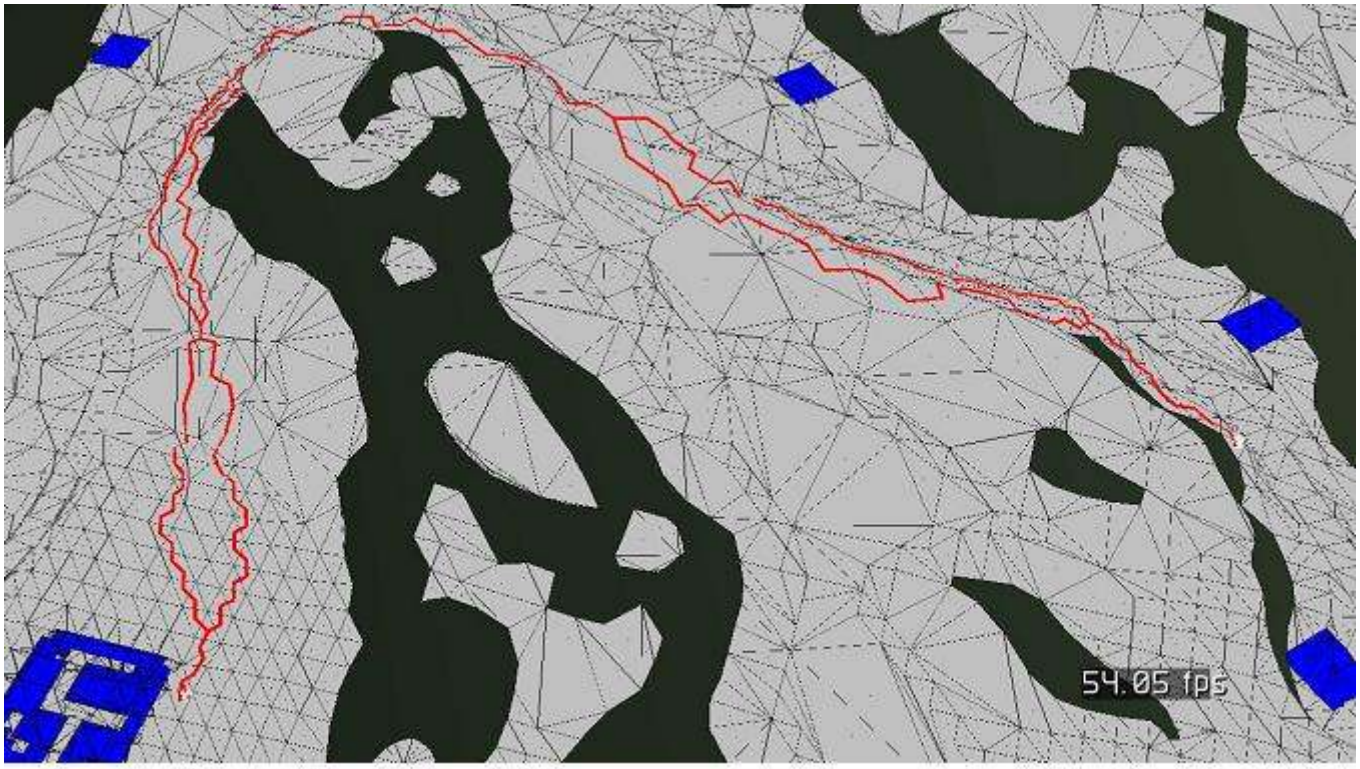
➡ 最短時間の経路を導く

ハウンズが地表効果を考慮して移動する

（2）障害物が破壊されたら、メッシュのデータを更新する

➡ **ハウンズが状況の変化に対応して移動する**

ナビゲーションメッシュ上を
A*アルゴリズムによって
ハウンズが移動するデモをご覧ください。



第3章　エージェント・アーキテクチャ　空間編

(1) Influence Map

細かく区切ったマップの単位に情報を載せる手法

（例）Killzone

Age of Empire

Chromehounds

(2) Smart Terrain(Object)

オブジェクト側にキャラクターを制御する情報を載せる手法

（例）The Sims

(3) Terrain Analysis　地形解析

地形から特徴的な情報を抽出する手法

（例）Age of Empire (AOE)

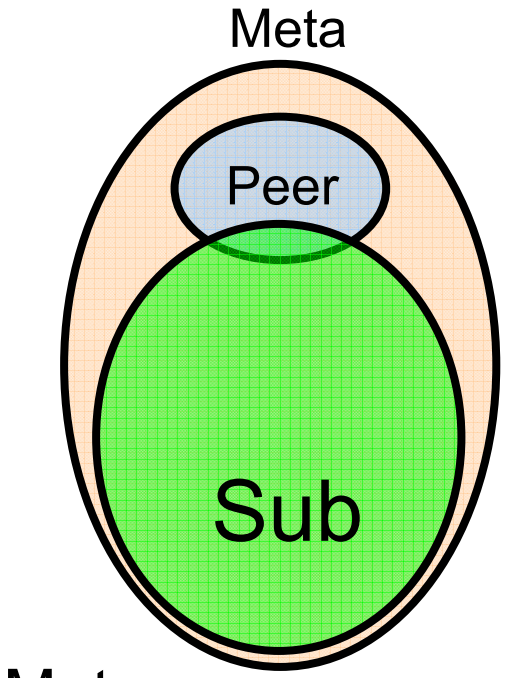
(4) パス検索

任意の２点間の経路を探索する

（例）カウンターストライクなど多数

The SimCity Sub AI

SimCity シリーズのAIの作り方



Meta
Peer
Sub

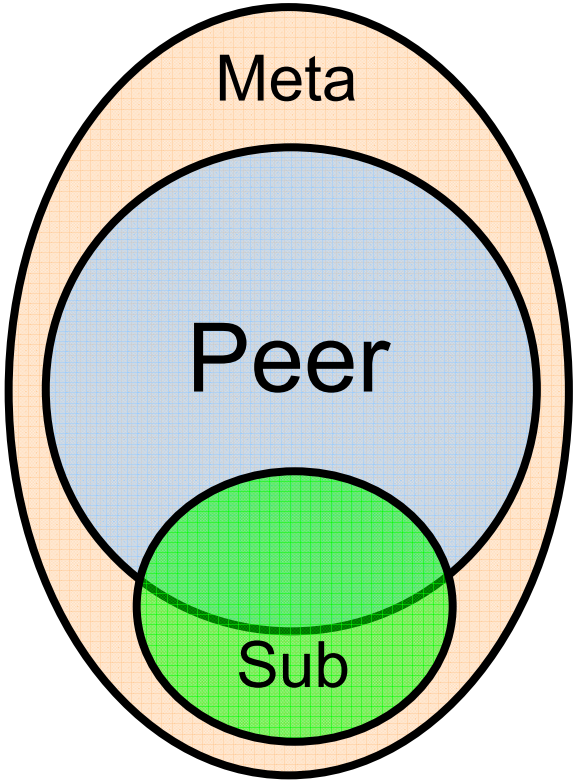
Crime = Pop. Density^2 - Land Value - Police Effect

Land Value = Distance[Zonetype] + Terrain + Transport

世界をダイナミクス（力学系、動的な数値の仕組み ）として動かす。
世界を動かす SubAI(=シミュレーション)　を構築。

The Sims Peer AI

The Sims シリーズのAIの作り方



Meta
Peer
Sub

[原則] 周囲の対象に対する、あらゆる可能な行動から、Happiness 係数を最大化する行動を選択する。

Sims (not under direct player control) choose what to do by selecting, from all of the possible behaviors in all of the objects, the behavior that maximizes their current happiness.

人をダイナミクス（力学系、動的な数値の仕組み ）として動かす。
世界を動かす PeerAI(=キャラクターAI)　を構築。

オブジェクトに仕込むデータ構造

Data (Class, Sate)

Graphics (sprites, z-
Animations (skeletal)

Sound Effects

Code (Edith)
-Main (obiect thread)
-External 1
-External 2
-External 3

パラメーター

グラフィックス
アニメーション

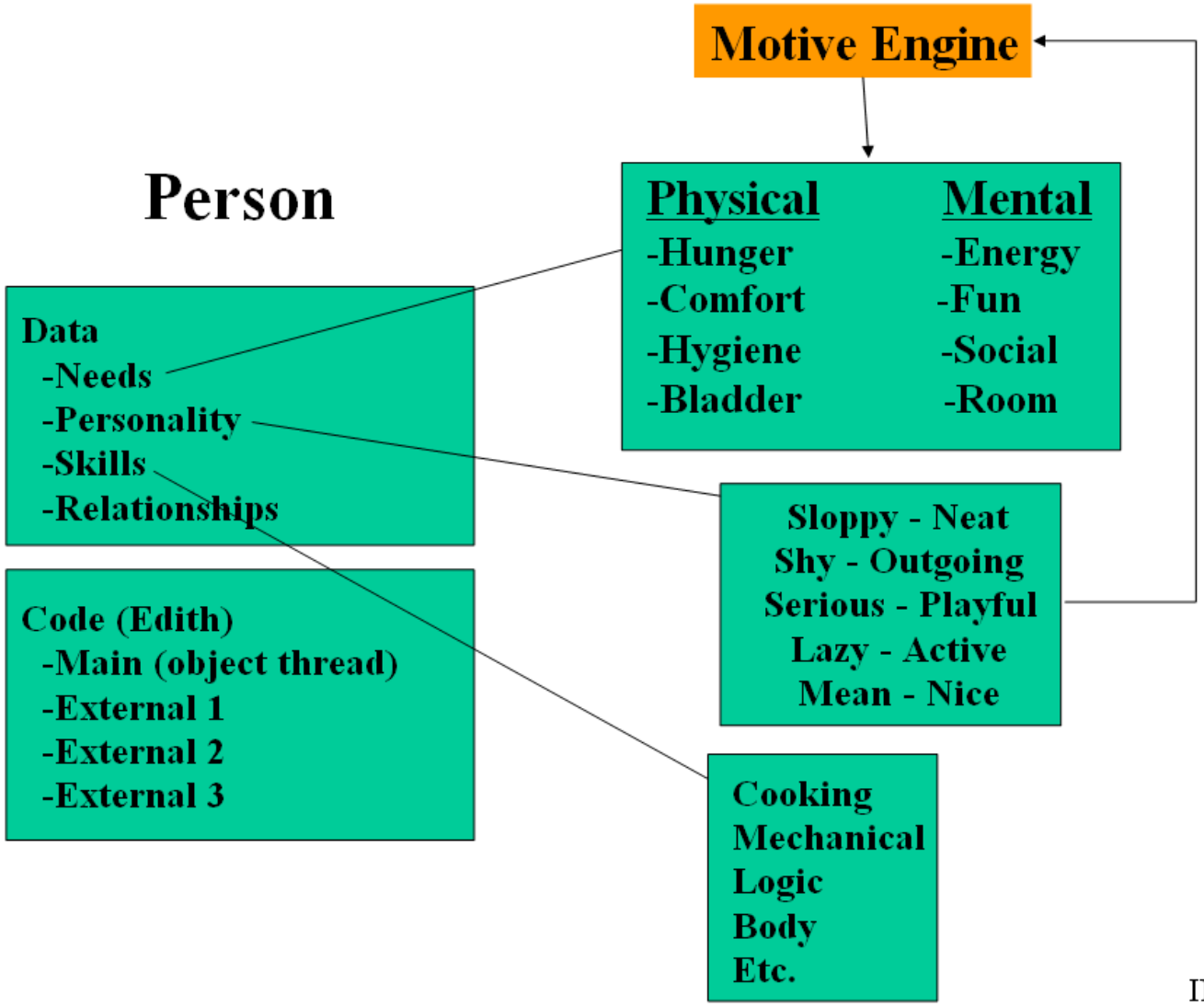
サウンド

メインスレッド

いろいろなインタラクションの仕方

Ken Forbus, "Simulation and Modeling: Under the hood of The Sims" (NorthWerstern大学、講義資料)
http://www.cs.northwestern.edu/%7Eforbus/c95-gd/lectures/The_Sims_Under_the_Hood_files/frame.htm

NPCに仕込むデータ構造



INTF

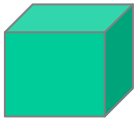
Ken Forbus, "Simulation and Modeling: Under the hood of The Sims" (NorthWerstern大学、講義資料)
http://www.cs.northwestern.edu/%7Eforbus/c95-gd/lectures/The_Sims_Under_the_Hood_files/frame.htm

最適な行動を選択する

Hunger +20
Comfort -12
Hygiene -30
Bladder -75
Energy +80
Fun +40
Social +10
Room -60

Mood +18

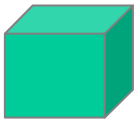
Toilet



Mood +26

-Urinate (+40 Bladder)
-Clean (+30 Room)
-Unclog (+40 Room)

Bathtub

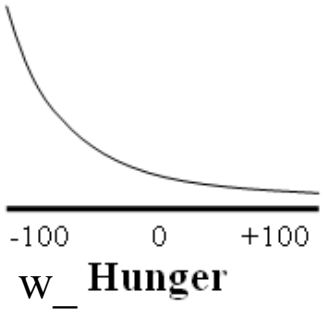


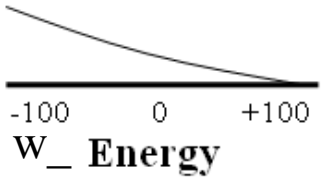
Mood +20

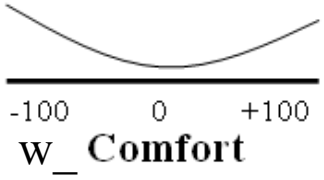
-Take Bath (+40 Hygiene)
(+30 Comfort)
-Clean (+20 Room)

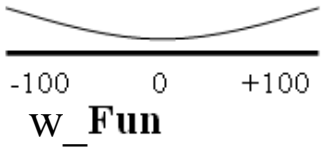
[原則] 周囲の対象に対する、あらゆる可能な行動から、Happiness（ここではMood）係数を最大化する行動を選択する。

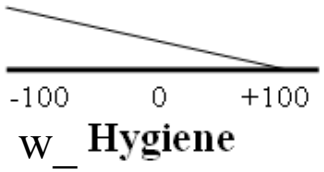
Happiness 係数の計算のためのウエイト

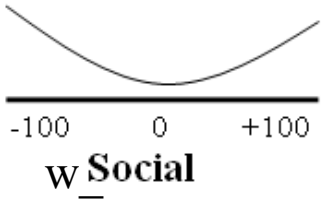


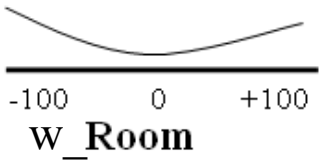








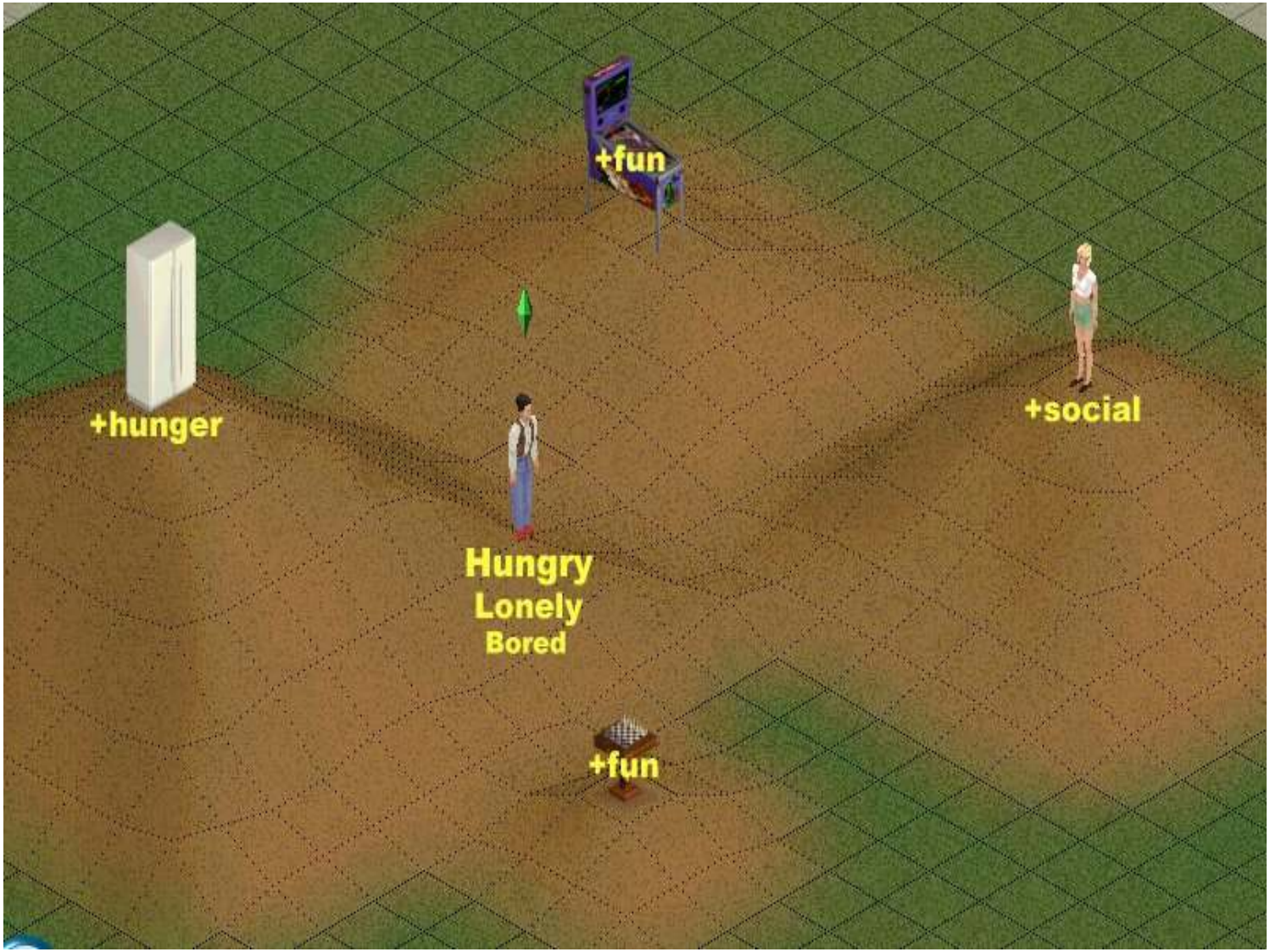




= Mood (-100..+100)

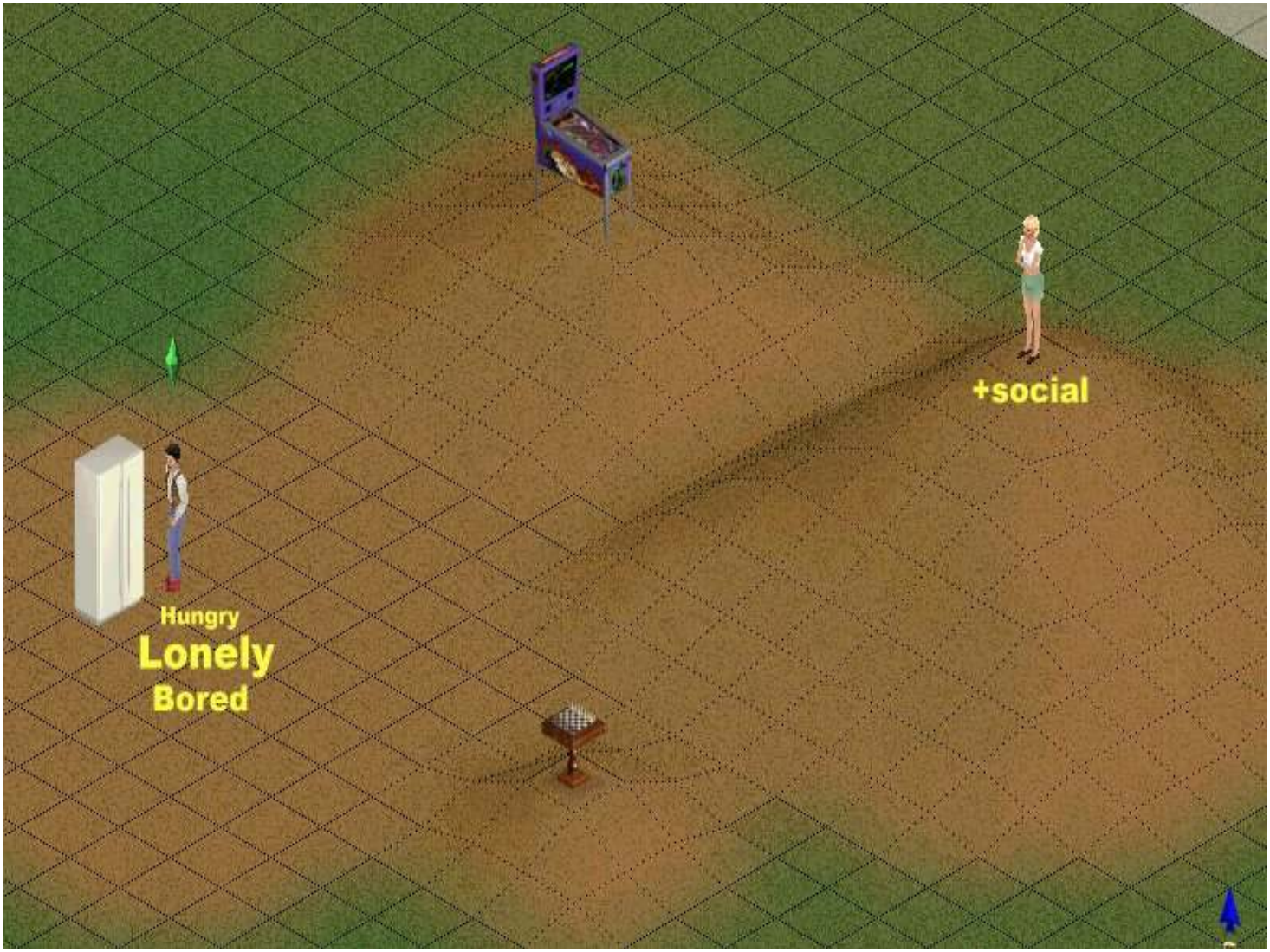
Hapiness = W_Hunger * Hunger + W_Engergy * Energy + …

Happiness を最大化



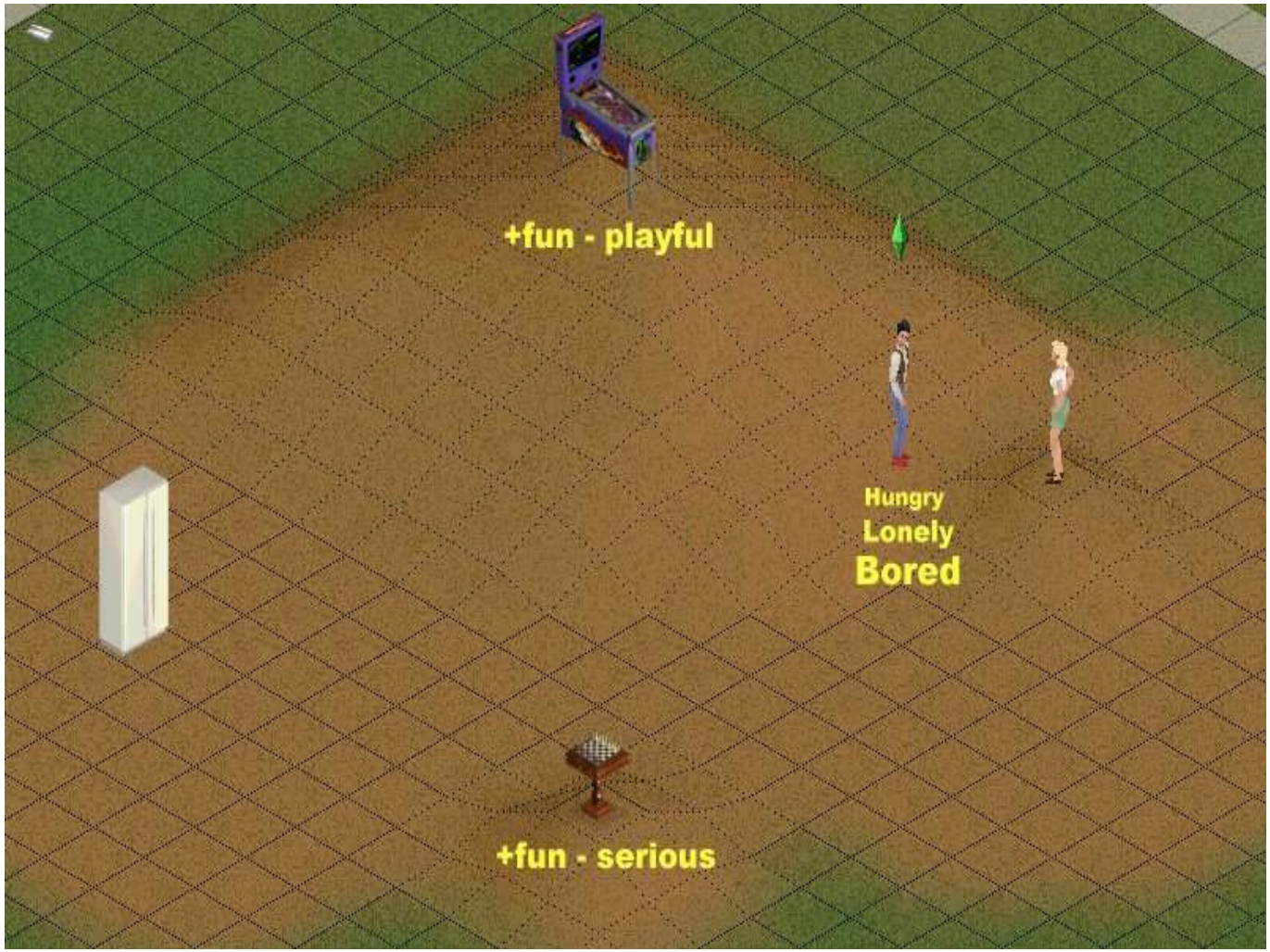
冷蔵庫が最も総合的にHapinessを上昇させるから

Happiness を最大化



冷蔵庫へ行きます。

Happiness を最大化



お腹が膨れたので、ちょっと退屈だから、女の子と話します。

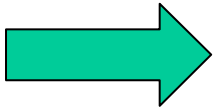
[余談] Halo3

上記の手法は、

Declarative Method としてHalo3 へ

Halo3　9講演のPPT資料　http://www.bungie.net/Inside/publications.aspx

Sims の話へ戻って…



Edith

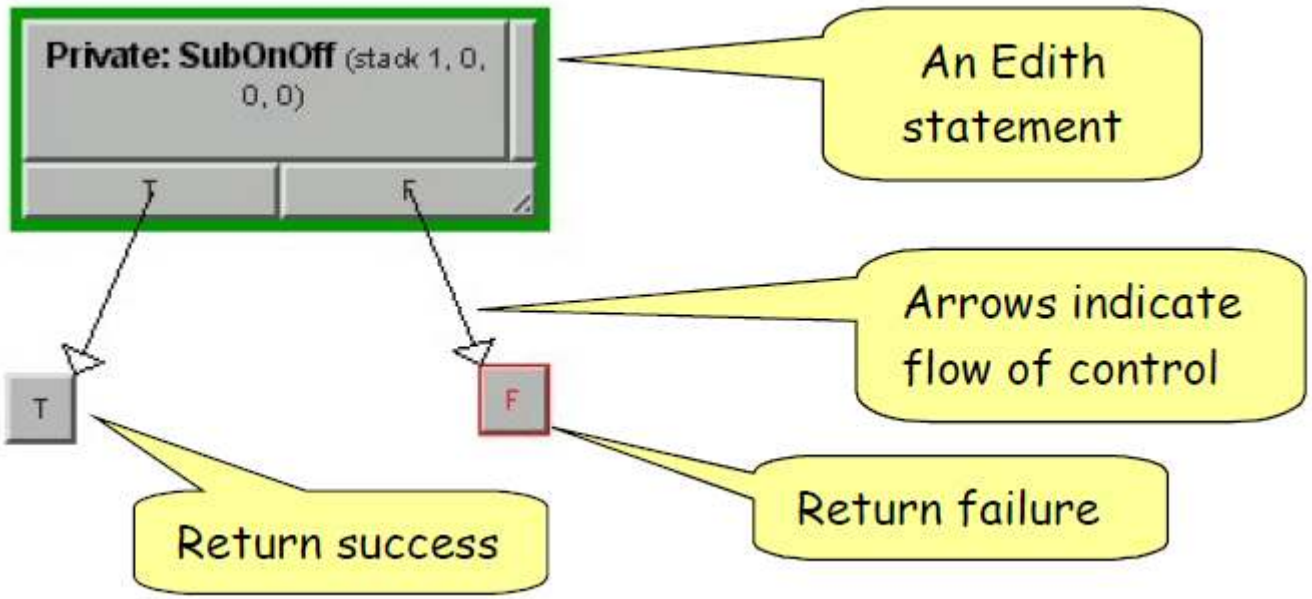
これだけだと、
原始的で単発の行動しかできないけれど...

実際は、
一連の行動のシークエンスをAIに行わせたい。

そのためのビジュアル・プログラミング環境が、
Edith

Edith

プログラミング・オブジェクトを繋げて行く
ビジュアル・プログラミング環境

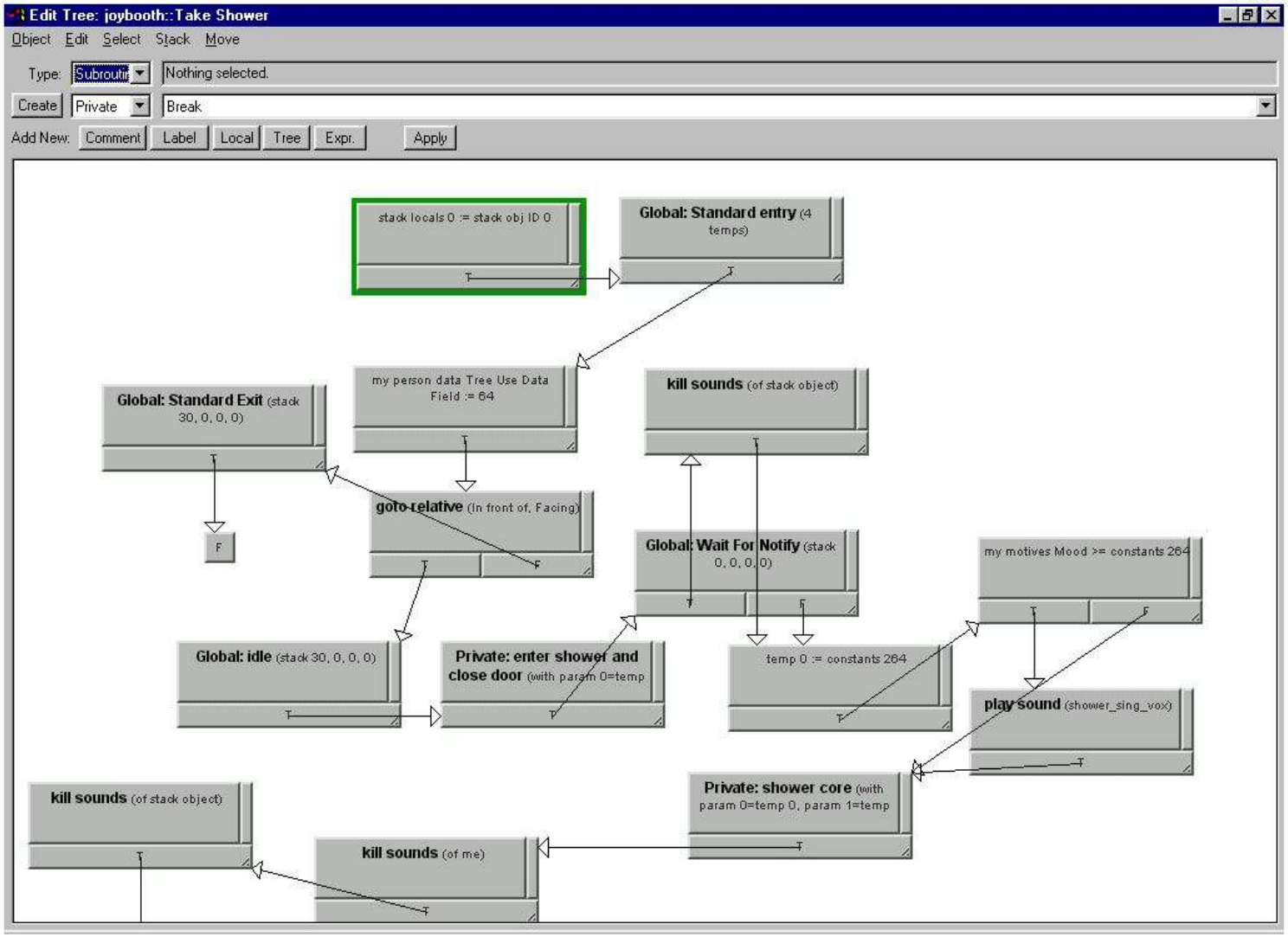


Kenneth D. Forbus “Some notes on programming objects in. The Sims”

http://www.qrg.northwestern.edu/papers/Files/Programming_Objects_in_The_Sims.pdf

Edith

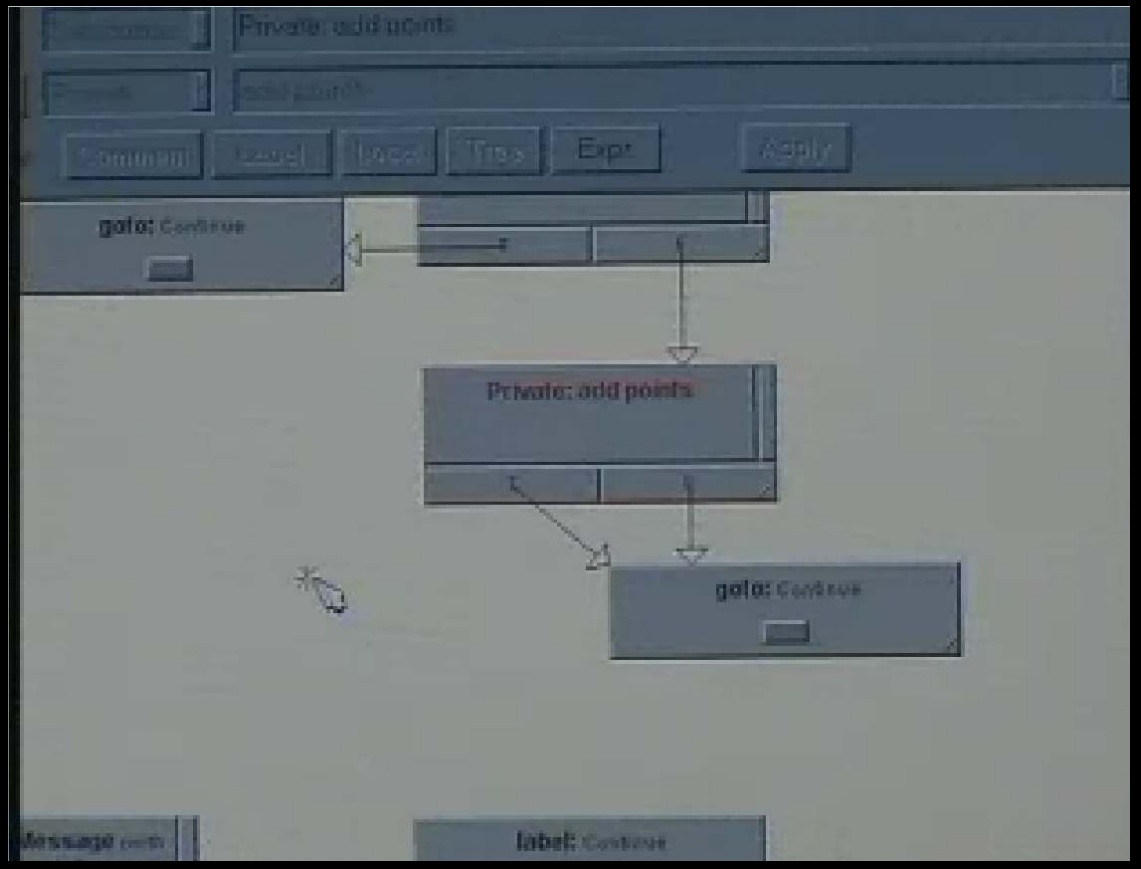
プログラミング・オブジェクトを繋げて行くビジュアル・プログラミング環境



Kenneth D. Forbus “Some notes on programming objects in. The Sims”

http://www.qrg.northwestern.edu/papers/Files/Programming_Objects_in_The_Sims.pdf

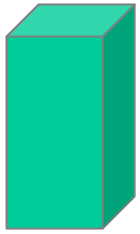
Edith Demo



http://www.DonHopkins.com/home/movies/TheSimsPieMenus.mov

実例① 「調理して食べる」

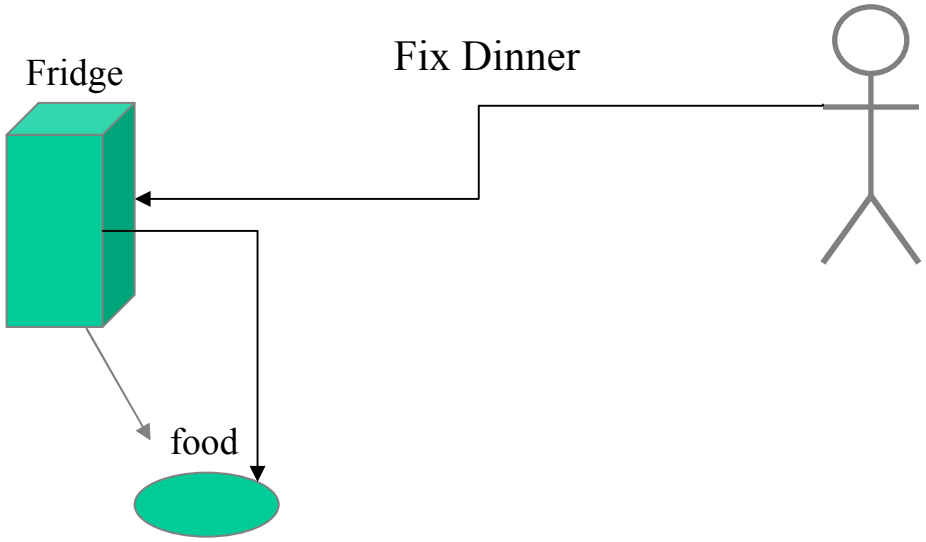
Fridge



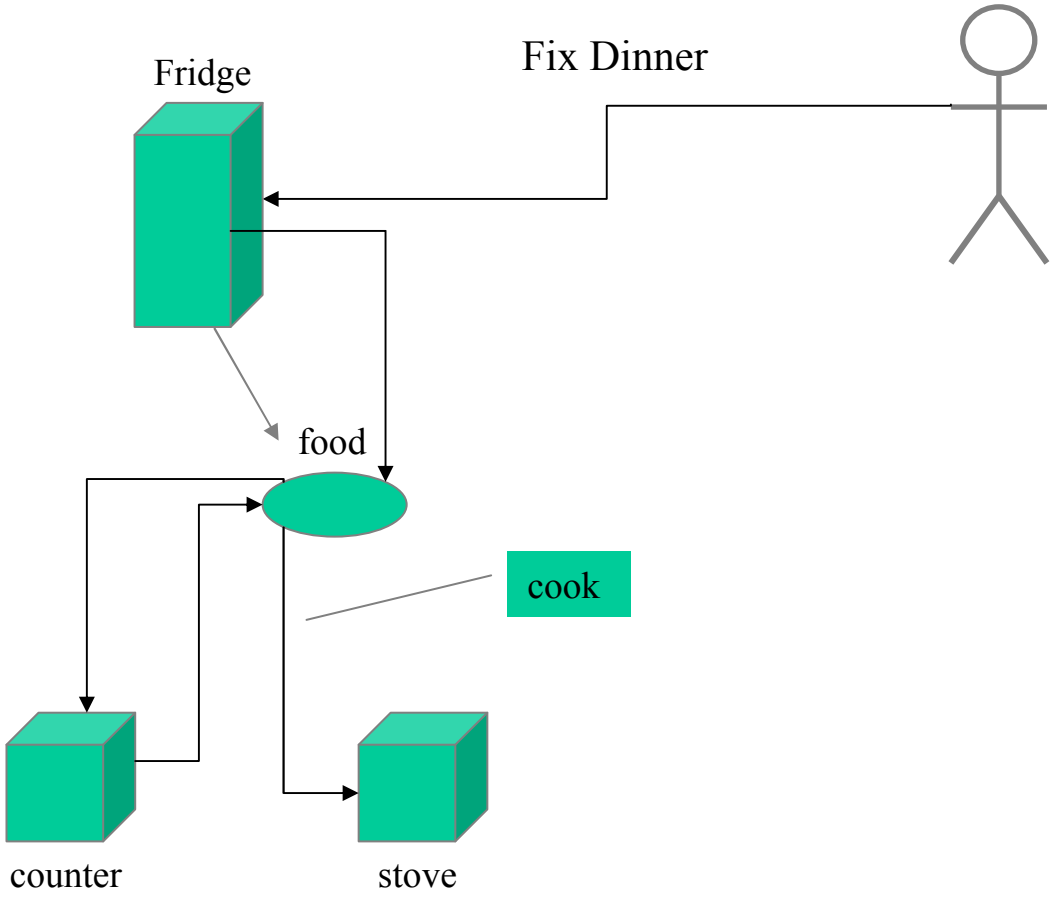
Hunger +30

Hungry

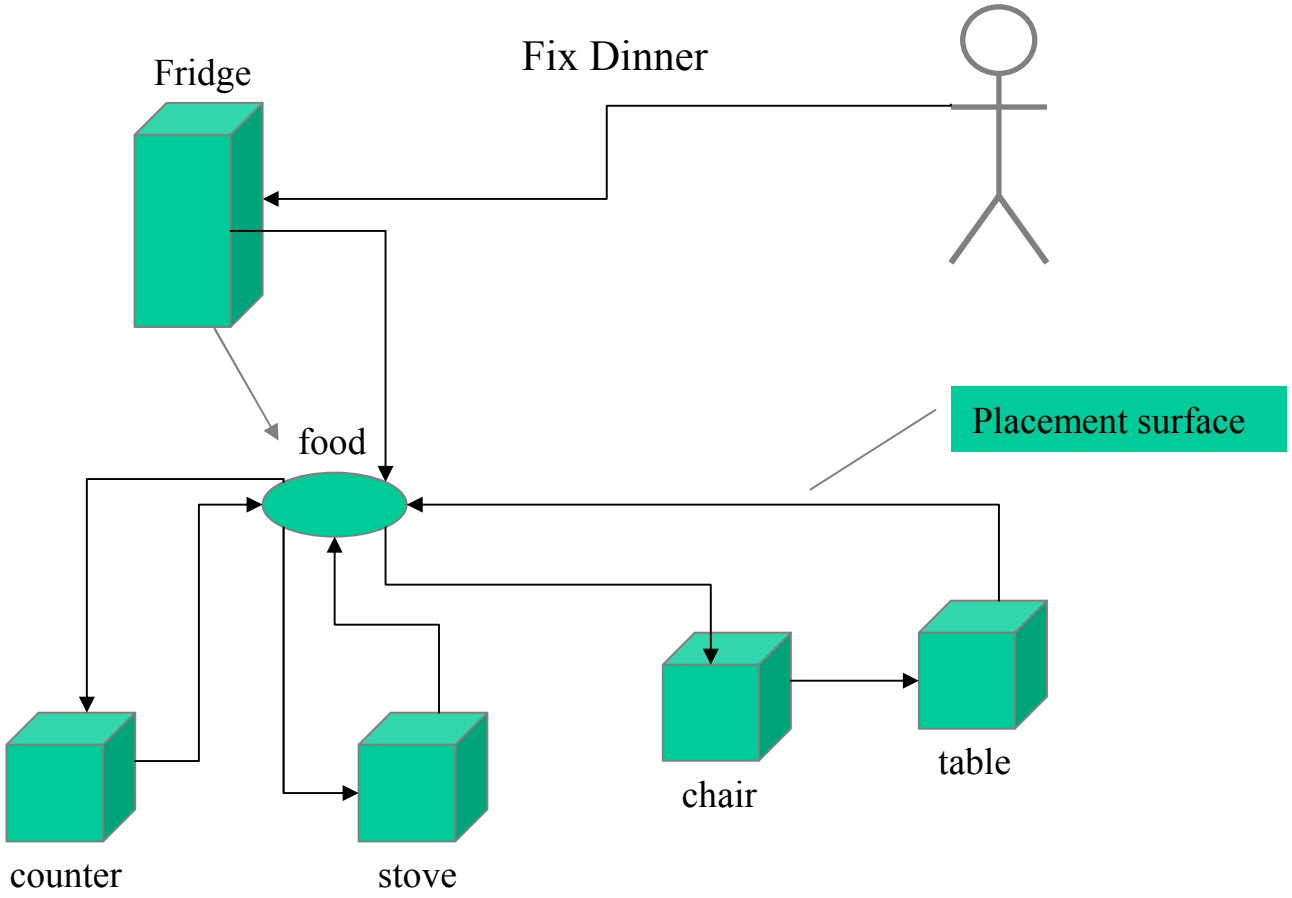
実例① 「調理して食べる」



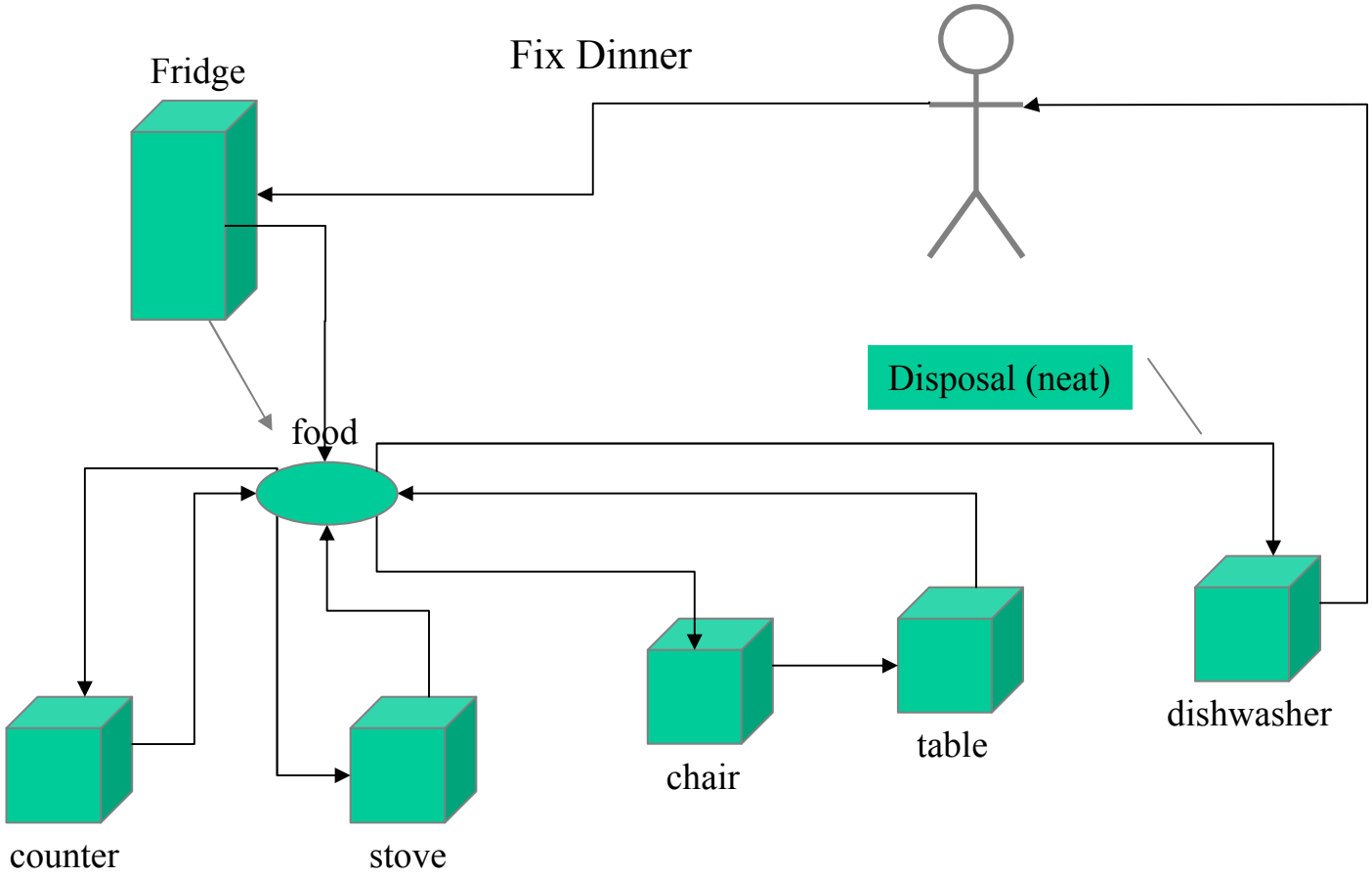
実例① 「調理して食べる」



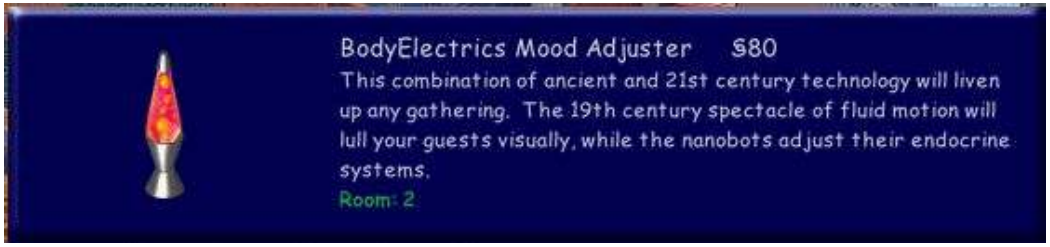
実例① 「調理して食べる」

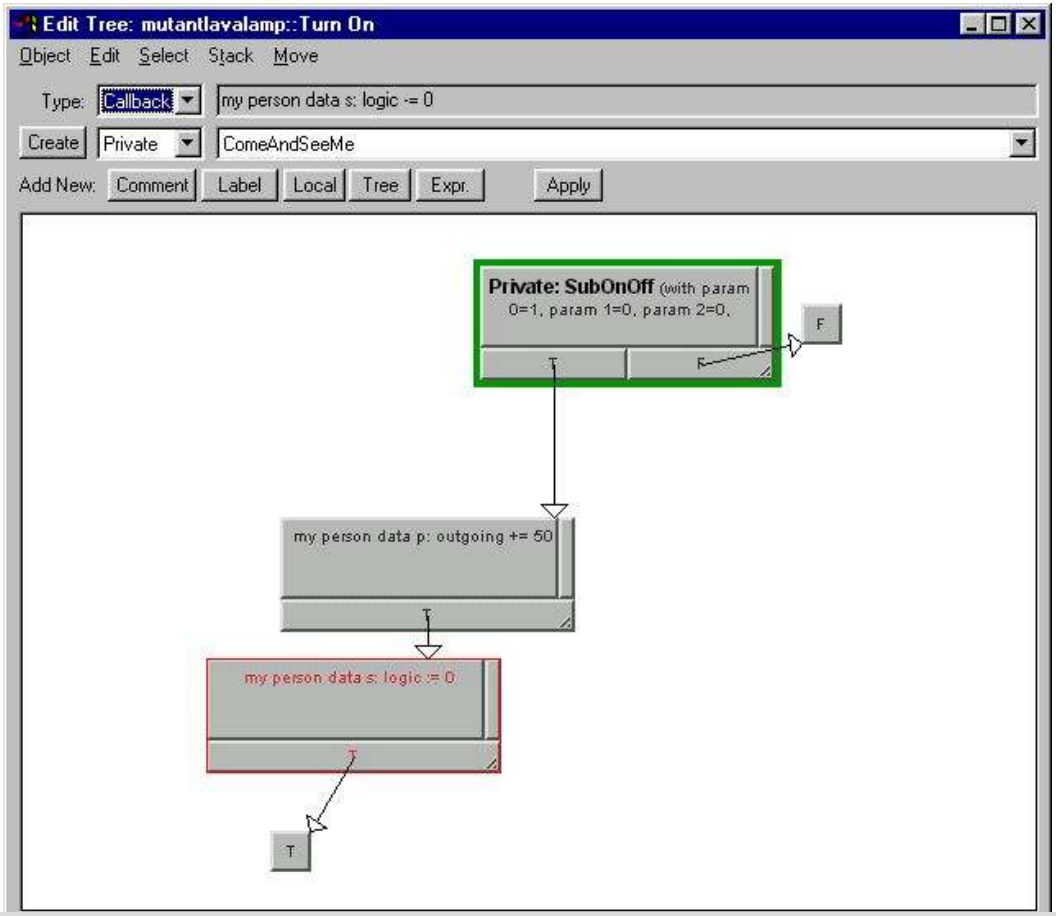

Fix Dinner
Fridge
food
Placement surface
counter
stove
chair
table

実例① 「調理して食べる」


Fix Dinner
Fridge
Disposal (neat)
food
counter
stove
chair
table
dishwasher

Edith for Mood Adjuster (パーティ盛り上げグッズ)





Kenneth D. Forbus “Some notes on programming objects in. The Sims”

http://www.qrg.northwestern.edu/papers/Files/Programming_Objects_in_The_Sims.pdf

第3章　エージェント・アーキテクチャ　空間編

(1) Influence Map

細かく区切ったマップの単位に情報を載せる手法

（例）Killzone

Age of Empire

Chromehounds

(2) Smart Terrain(Object)

オブジェクト側にキャラクターを制御する情報を載せる手法

（例）The Sims

(3) Terrain Analysis　地形解析

地形から特徴的な情報を抽出する手法

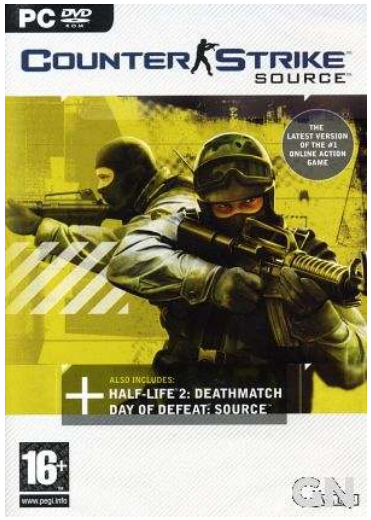
（例）Age of Empire (AOE)

~~(4) パス検索~~

任意の２点間の経路を探索する

（例）CounterStrike

Assassin’s Creed



CounterStrikeにおける ナビゲーション・メッシュ

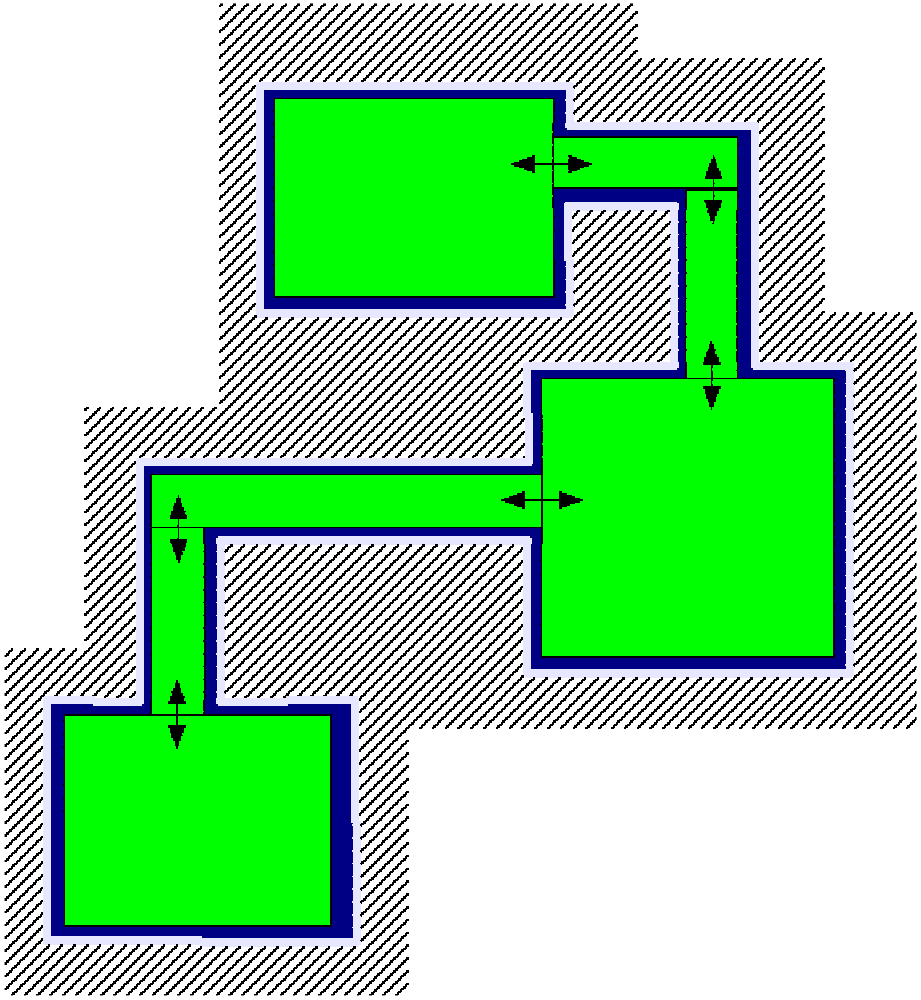
(1) マップからナビゲーション・メッシュを自動作成

(2) メッシュに対してビヘイビア（動作）を埋め込むことができる。

スマートなメッシュ

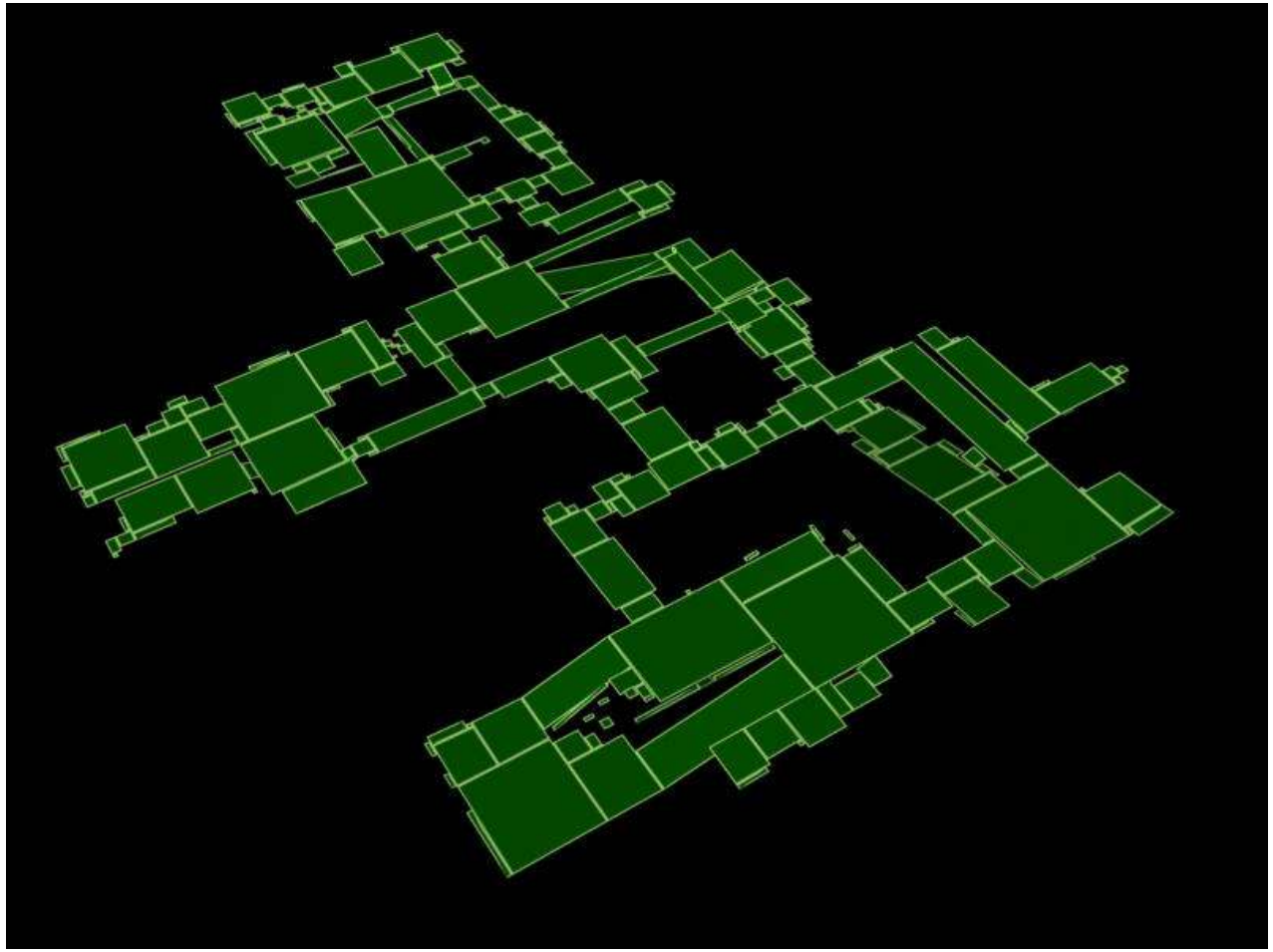
GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

Simple Navigation Area Example



GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

Navigation Areas: Dust

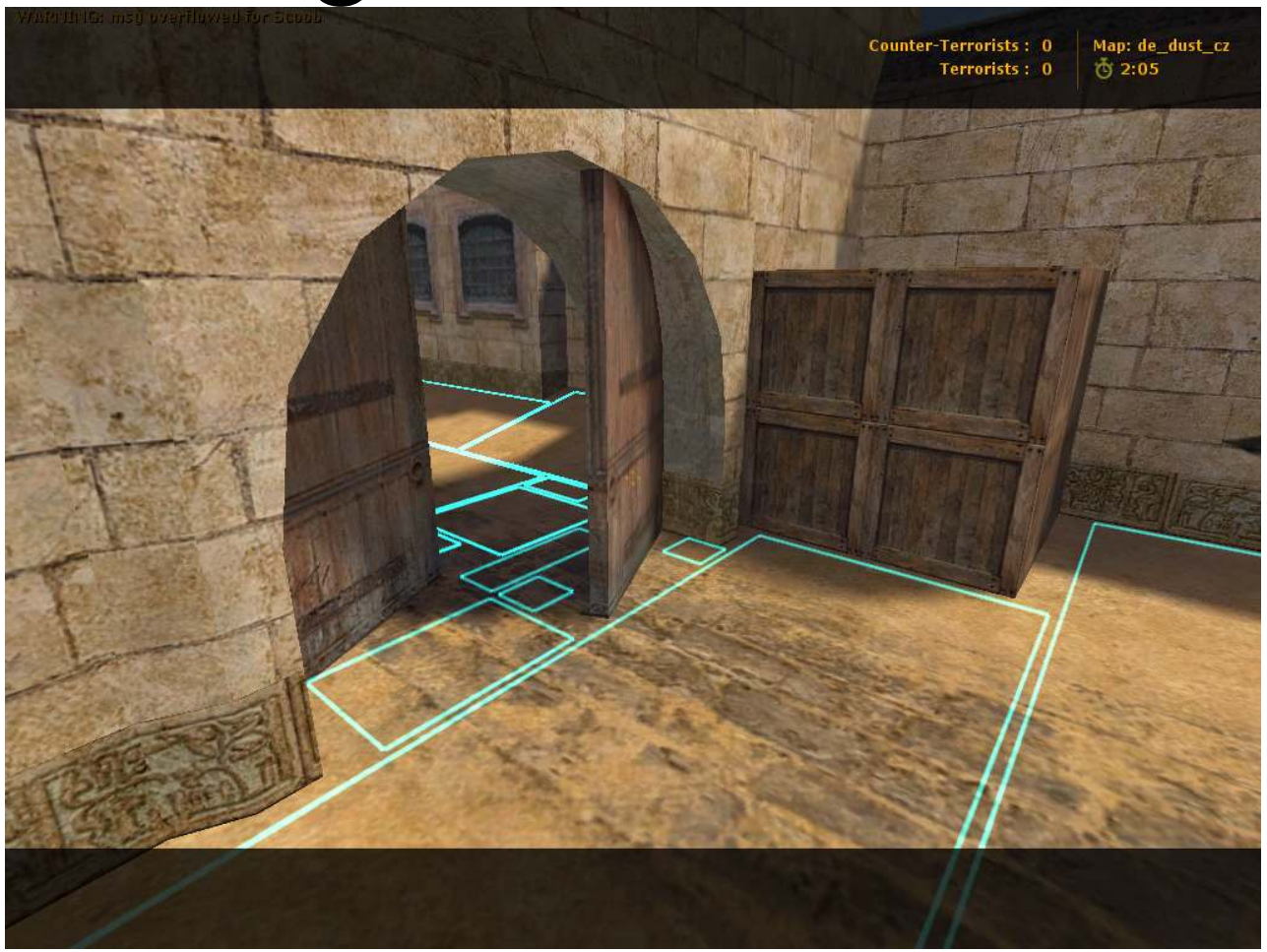


GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

Navigation Areas: Dust

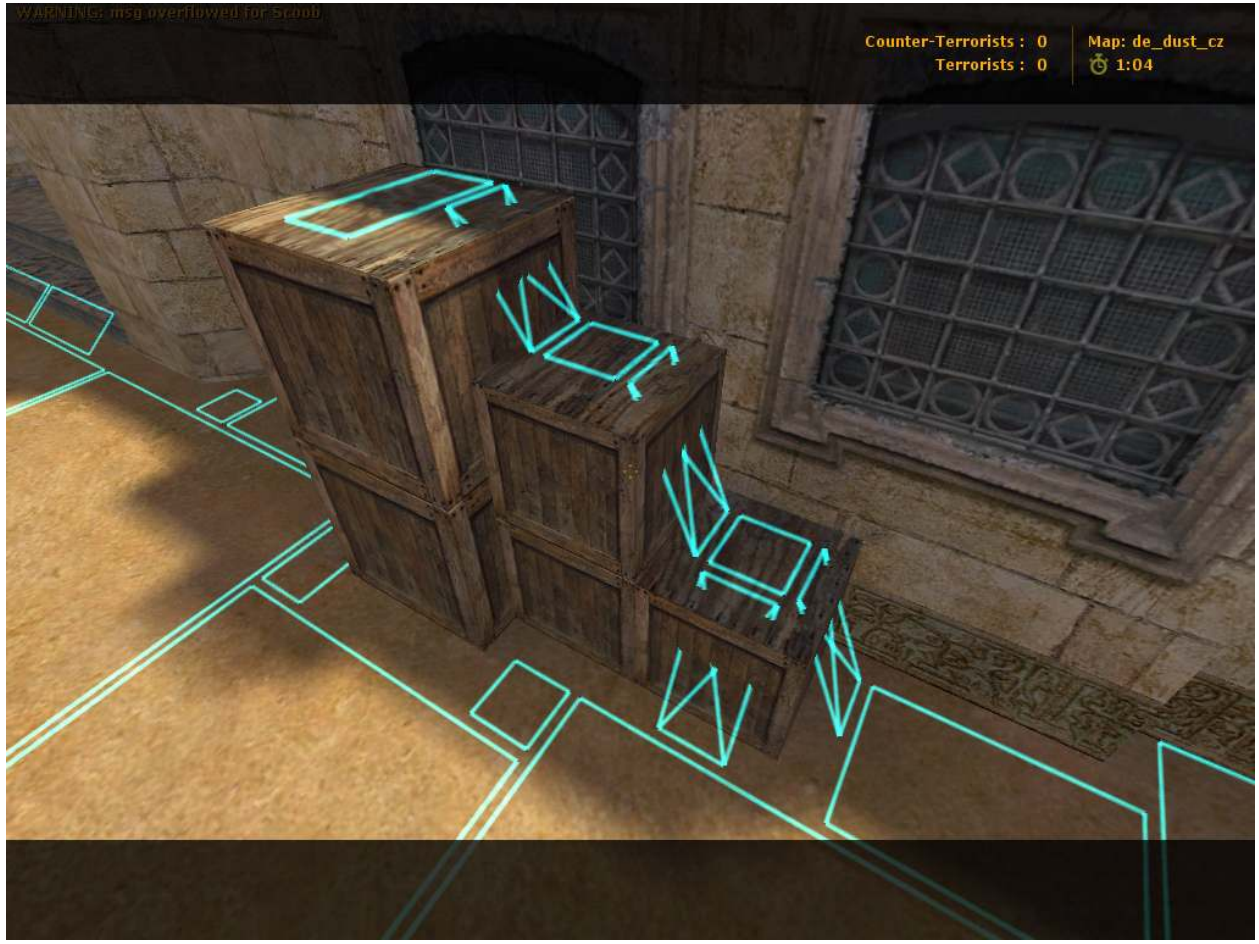
GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

Navigation Areas: Dust



GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

Navigation Areas: Dust

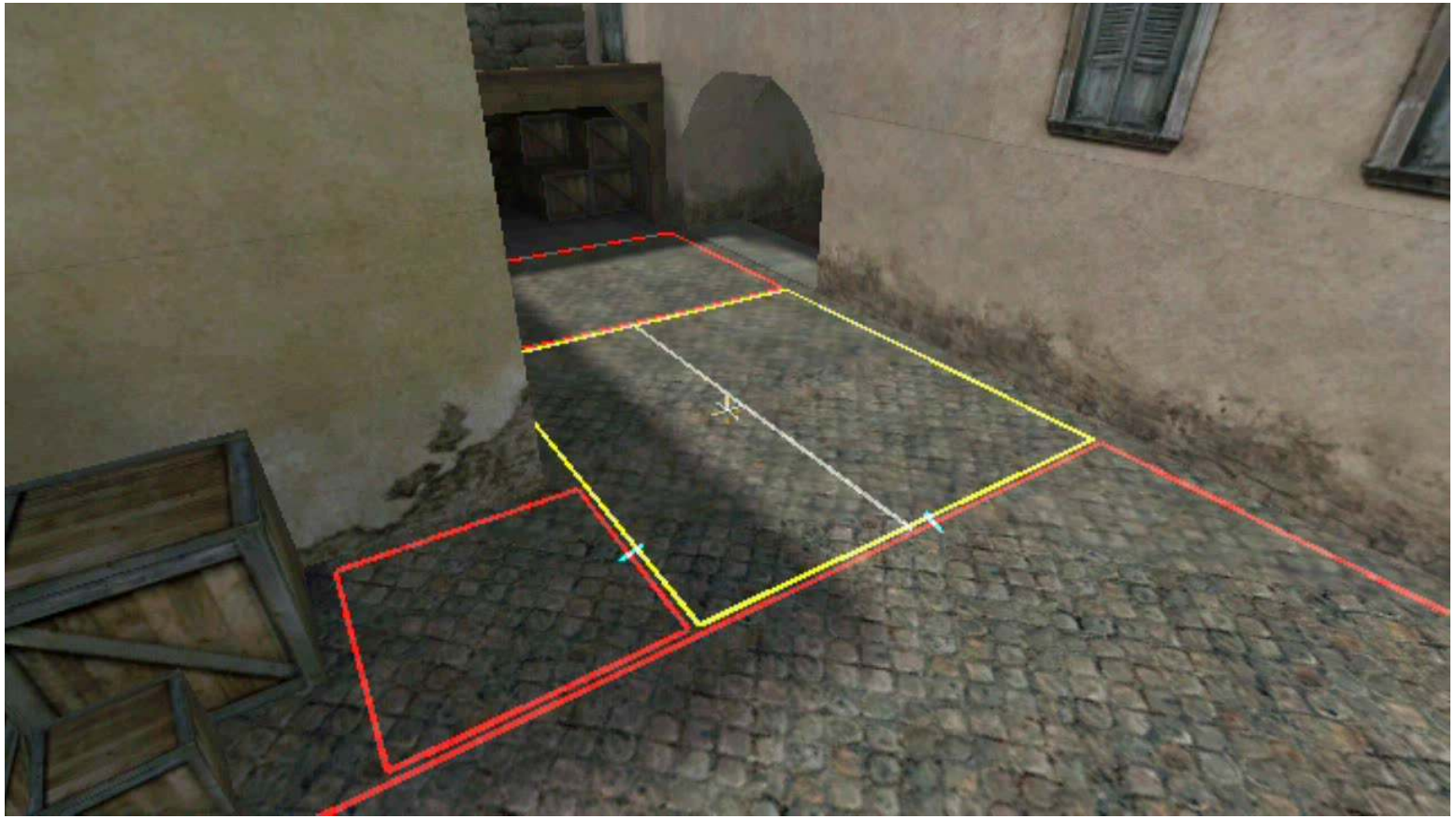


GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

埋め込まれている情報

- 角、突起　小さなオブジェクト
- 飛び上がる場所
- 飛び超える溝
- 一方的に降りる（降りたら上がれない）
- ドア
- はしご
- しゃがんで通る場所
- 破壊可能物（窓など）

Ledge and Gap Jumping Navigation Mesh

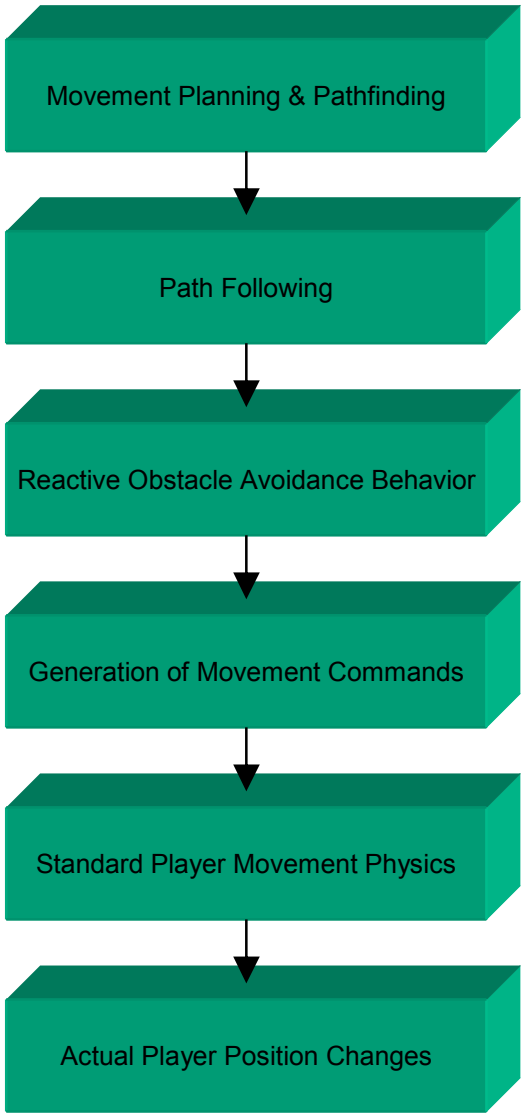


GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

Ledge and Gap Jumping Example

GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

Movement Hierarchy



GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

Path Following

GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

Reactive Obstacle Avoidance

GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

Navigation Example: Italy

GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

Navigation Example: Office

GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

環境における動作

- パス検索
  - A* コスト関数
    - ジャンプ、かがむ、梯子といったアクションの経路はコストを高める
    - 最も安全か、最も早い経路によって、危険度のコストを加算する

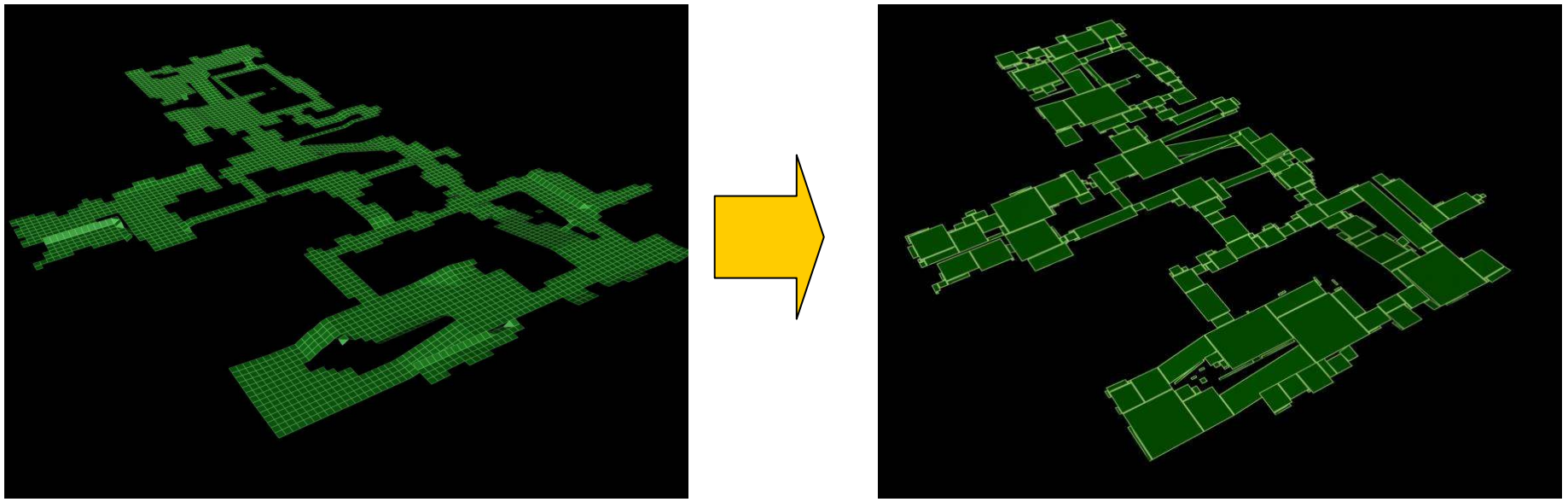
環境における動作

- パスを辿ることの失敗
  - 原因
    - 小さな、予測外のオブジェクト
    - 梯子
    - パスから落ちる
    - キャラクターとの衝突
  - 検出方法
    - 一定時間内の平均速度をモニター
  - 失敗しないために…
    - ランダムにくねって走る
    - ランダムにジャンプ

使用例
マップ自動学習

- ユーザーコミュニティーによって作成されたマップ
- マップが作成されるち、数分でそれを学習して新しいナビゲーションメッシュを自動生成する
- Learning samples the map, and aggregates the samples into Nav Areas via a greedy algorithm
- 隠れる場所とアプローチポイントを自動検出

Simple to Use:
Automated Map Learning



GDC 2004 : The Making of the Official *Counter-Strike* Bot
http://www.gdcvault.com

第3章　エージェント・アーキテクチャ　空間編

(1) Influence Map

細かく区切ったマップの単位に情報を載せる手法

（例）Killzone

Age of Empire

Chromehounds

(2) Smart Terrain(Object)

オブジェクト側にキャラクターを制御する情報を載せる手法

（例）The Sims

(3) Terrain Analysis　地形解析

地形から特徴的な情報を抽出する手法

（例）Age of Empire (AOE)

~~(4) パス検索~~

任意の２点間の経路を探索する

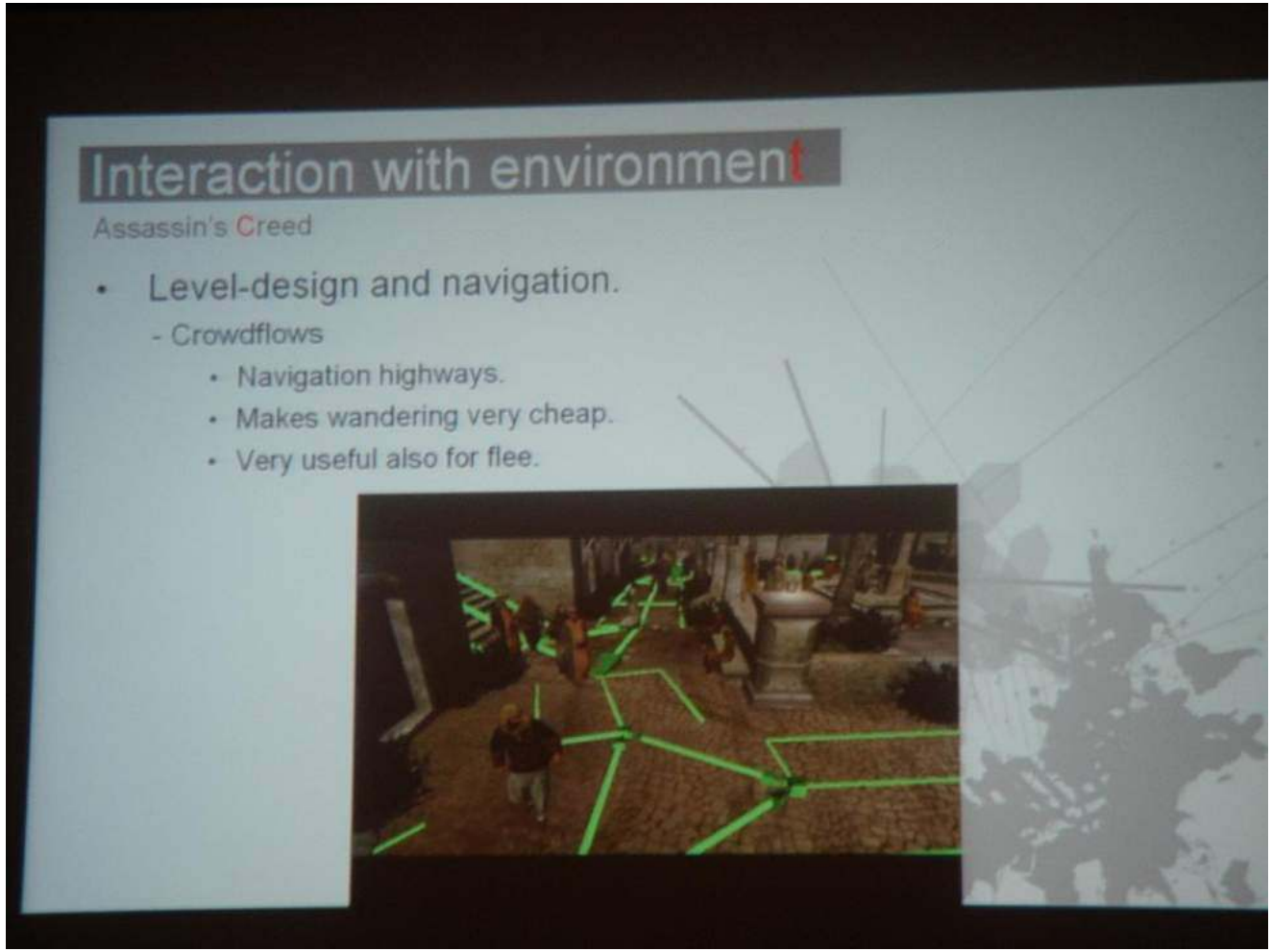
（例）CounterStrike

Assassin’s Creed

階層型パス検索システム

(1)パス検索の必要のない群集

- ナビゲーション・ハイウェイ
（デフォルトの進行ルートを巡行するだけ）



階層型パス検索システムの作り方

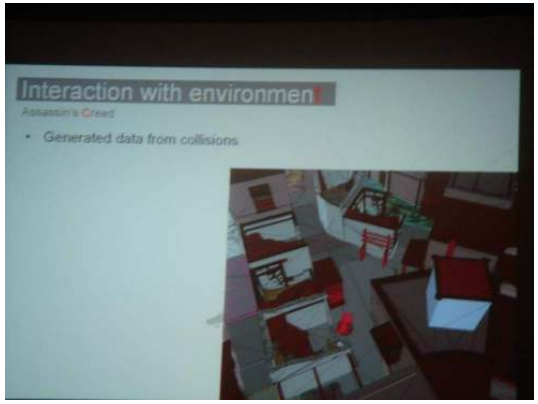
(2)　パス検索

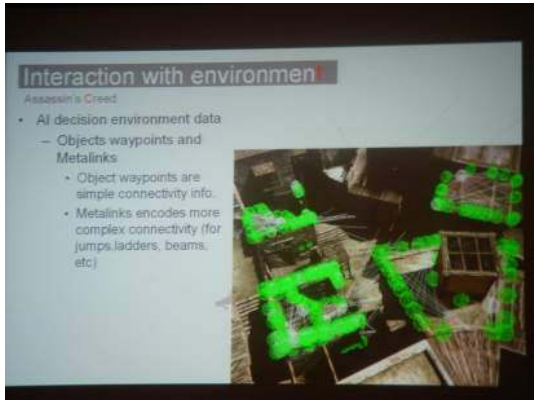
Step1: 衝突モデルからナビゲーション・メッシュとウエイポイントを自動生成

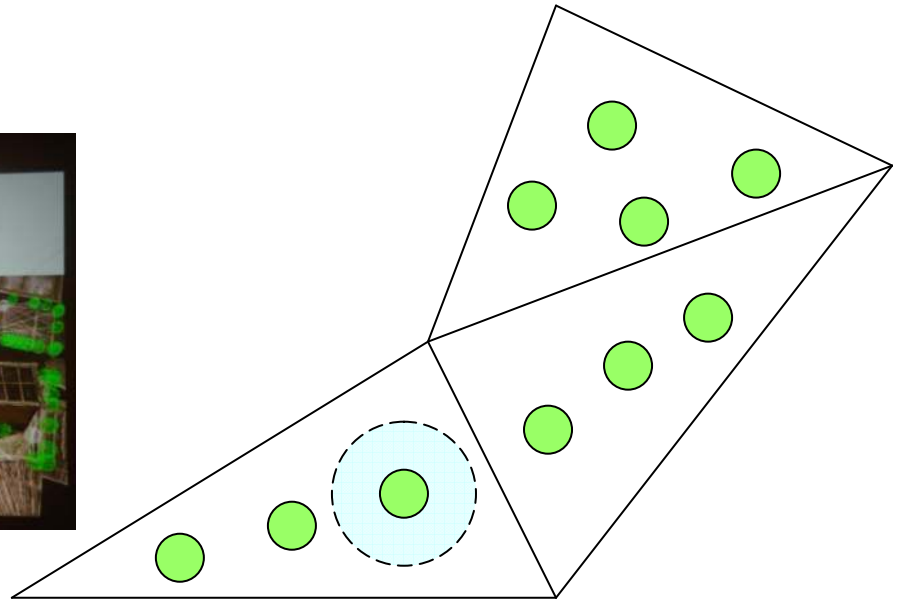
（ウエイポイントを置くときは周囲の障害物とコリジョンを
しないチェックもしているはずだ（三宅））

Step2: ナビゲーション・メッシュとウエイポイントを対応づける

Step3: 複雑なトポロジー（地形情報）上のウエイポイントには
メタリンク情報を埋め込む

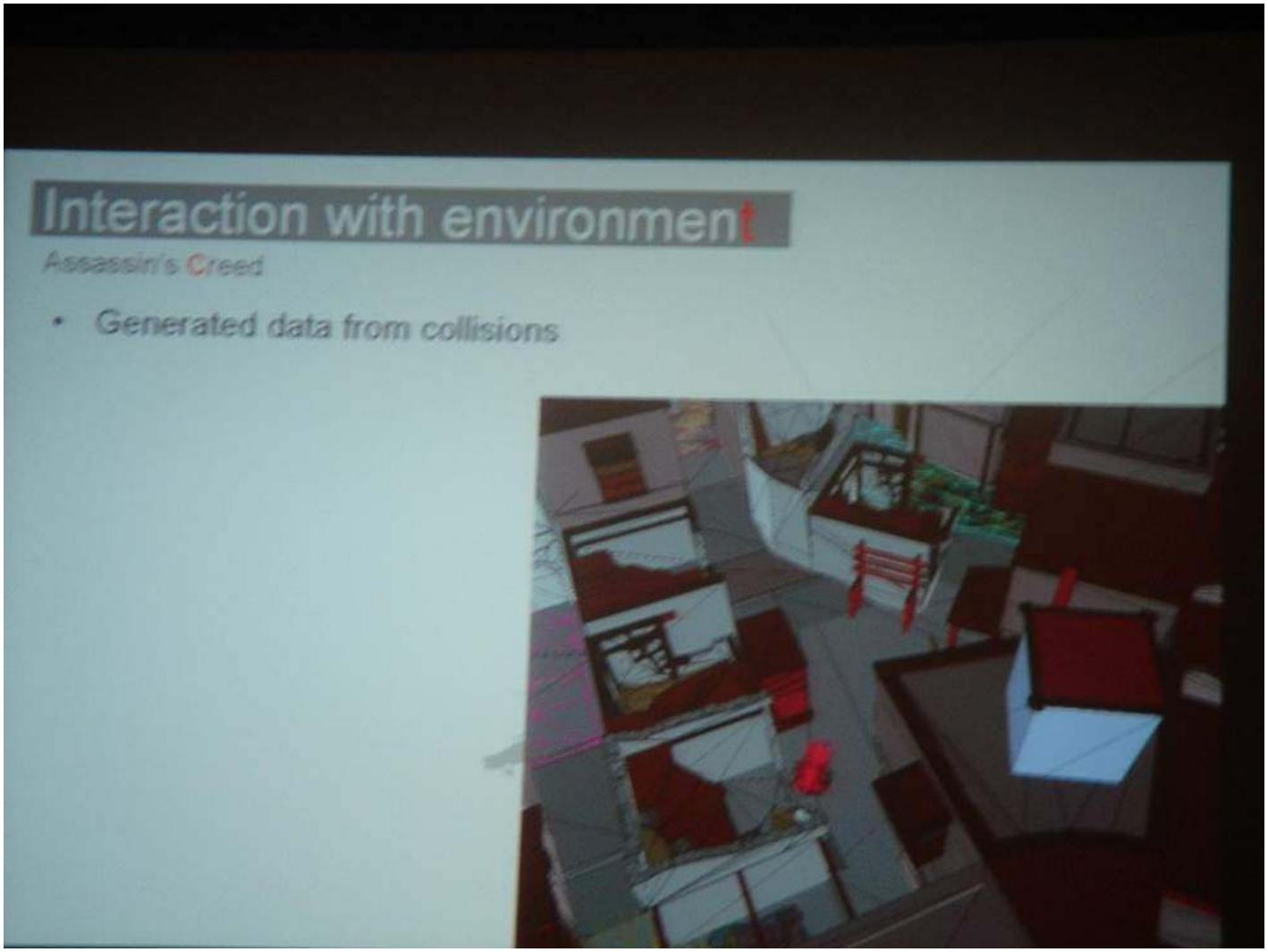




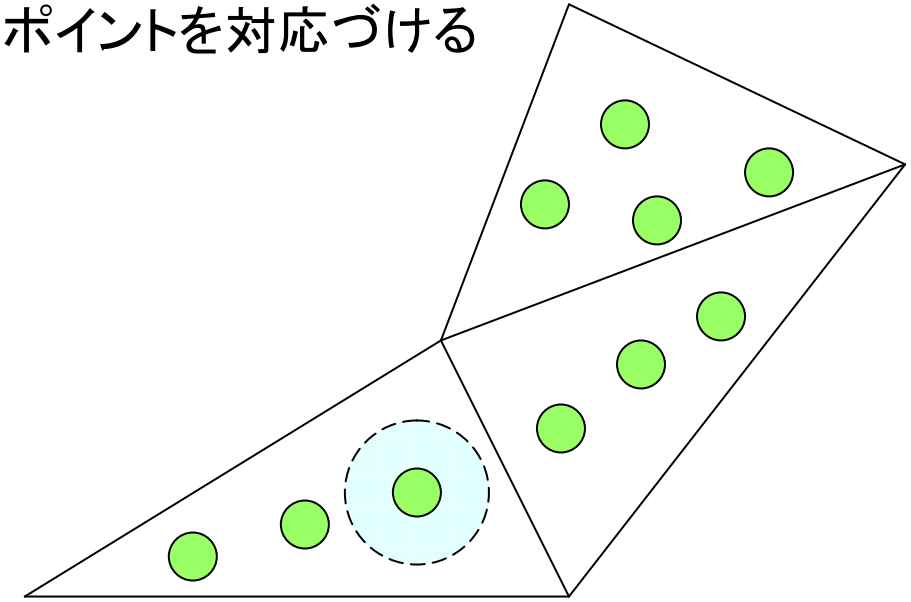


データ構造：　各メッシュはその上にあるウエイポイントをリストとして持つ

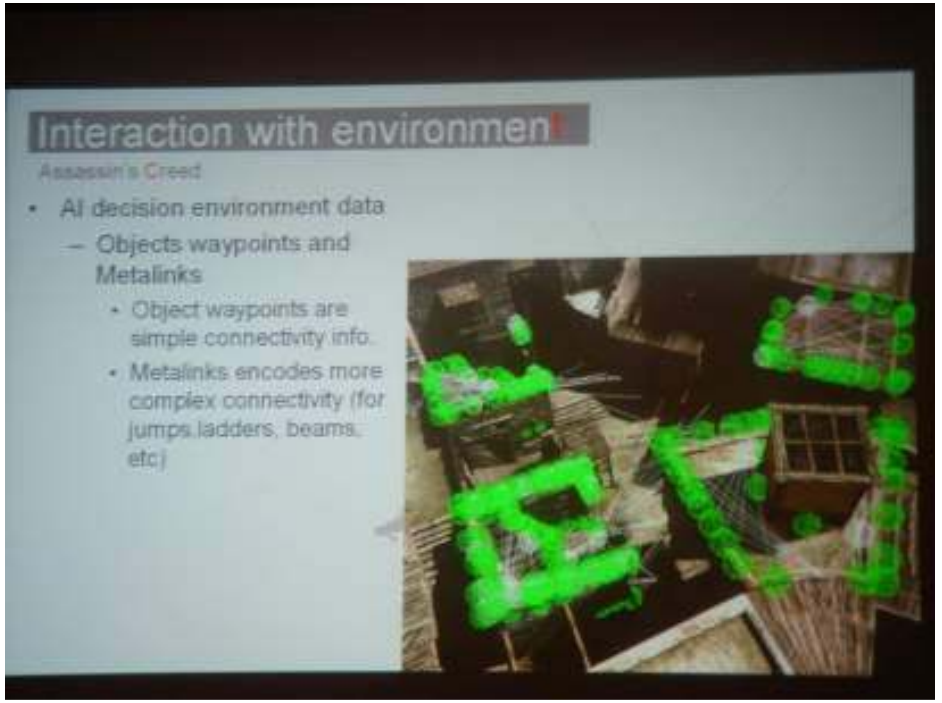
Step1: 衝突モデルからナビゲーション・メッシュとウエイポイントを自動生成



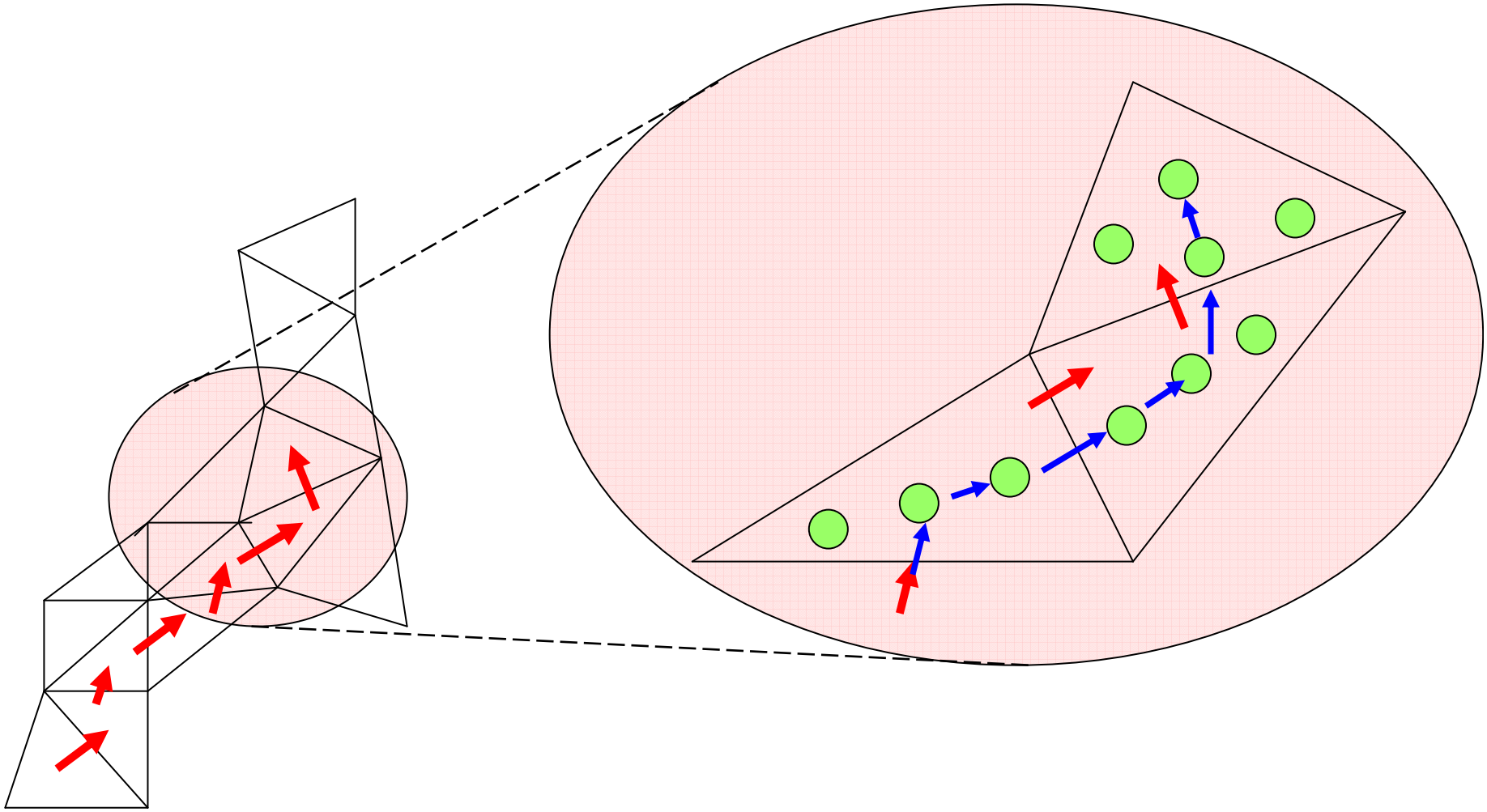
Step2: ナビゲーション・メッシュとウエイポイントを対応づける



Step3: 複雑なトポロジー（地形情報）上のウエイポイントには
メタリンク情報を埋め込む



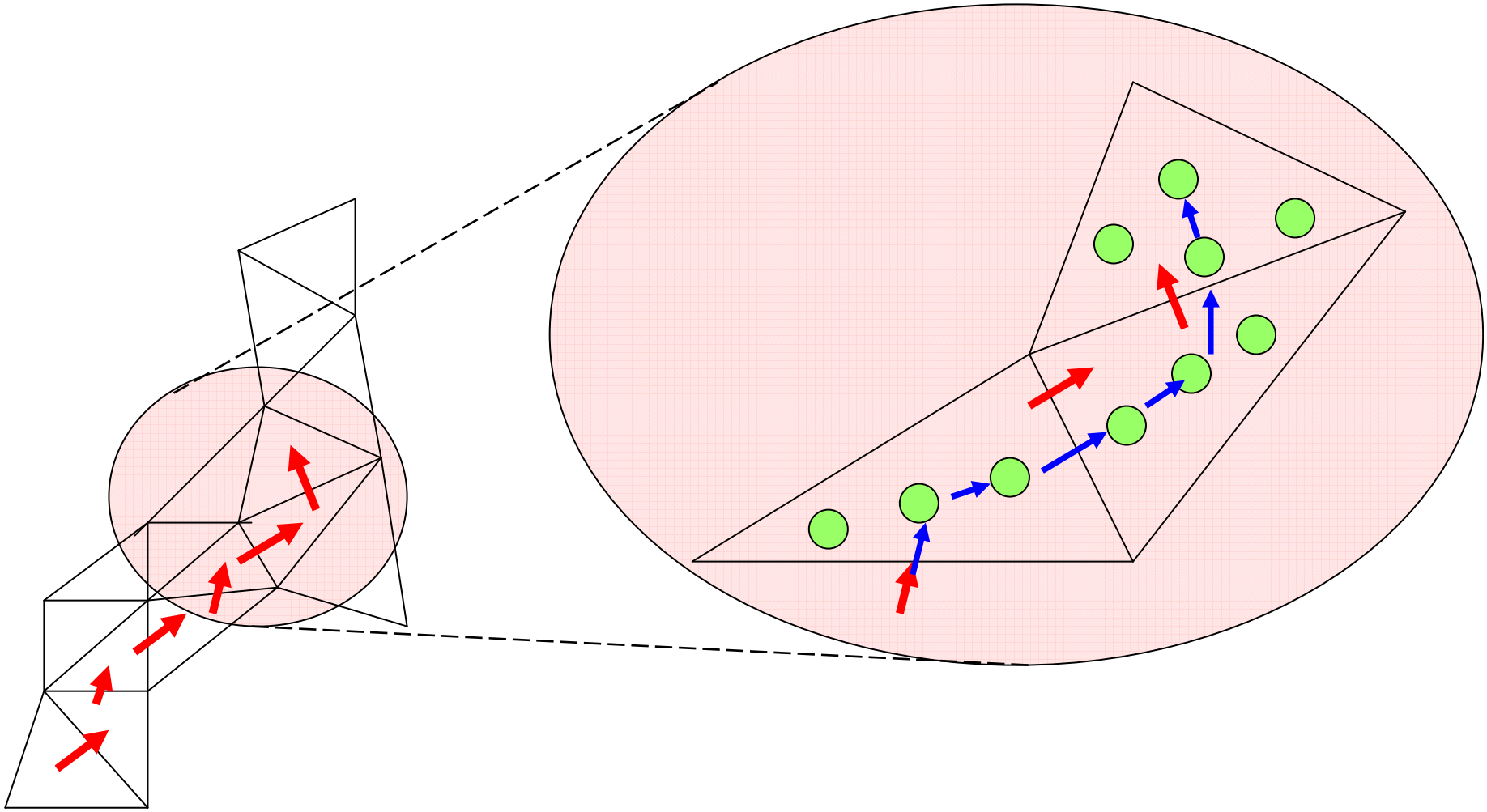
階層型パス検索システムの使い方



まずナビゲーション・メッシュでグローバルに検索する →
1つのメッシュの中ではウエイポイントで検索する →

（A＊検索の計算量を軽減することができる。特にACのようなモブ制御には必須）

パス検索システムの使い方



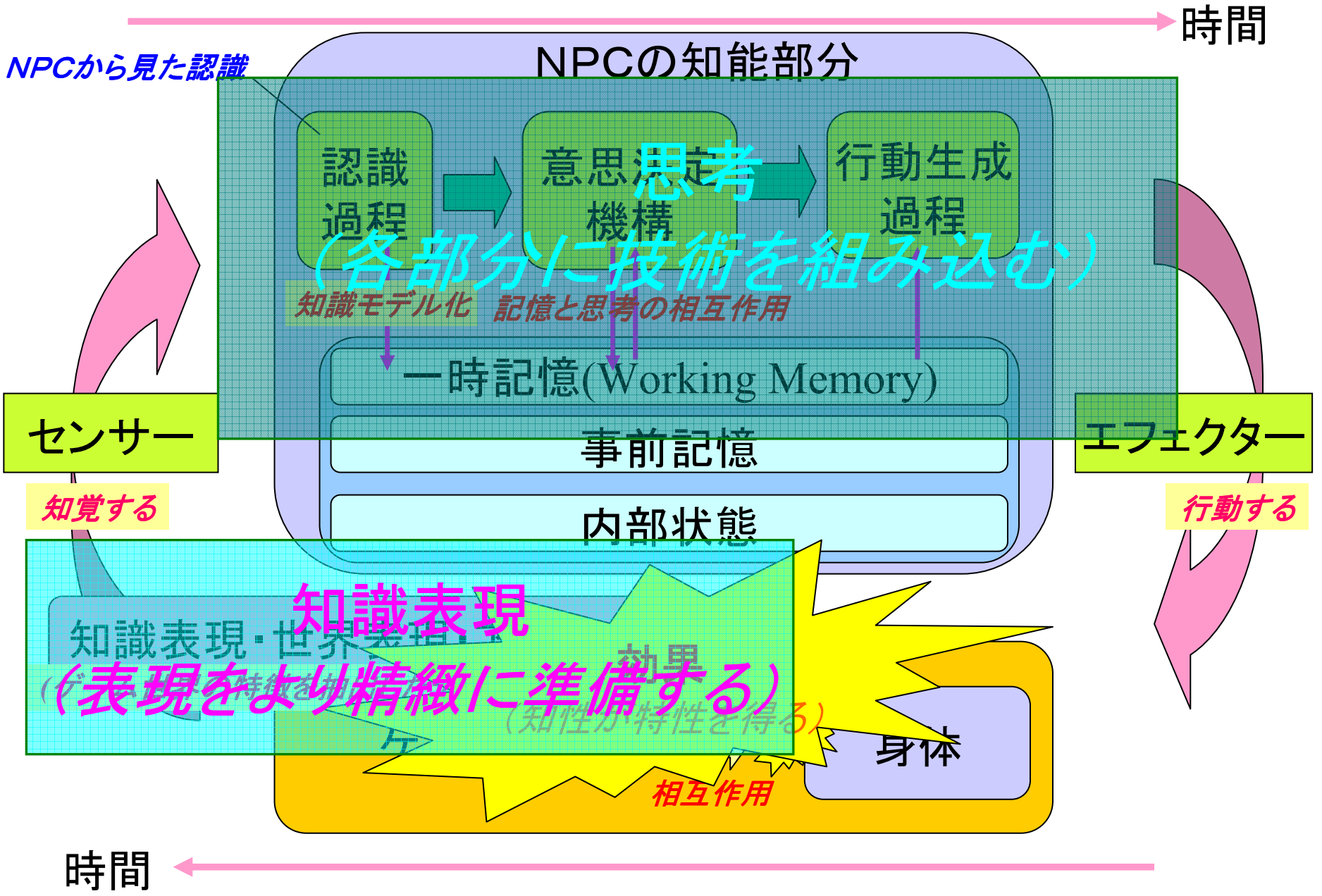
パス検索データはスケールによって
階層化するといろいろ便利！

（Ａ＊検索の計算量を軽減することができる。特にACのようなモブ制御には必須）

参考文献

「ASSASSIN‘S CREED」における群衆プログラム(GameWatch)
http://www.watch.impress.co.jp/game/docs/20080223/assasin.htm

『技術を仕掛けて効果を得る』



第3章　まとめ

この章では、AIが仮想空間（ゲーム空間）を
如何に使用できるかを見て来ました。

移動、動作、パス検索など、
世界表現上の多様な情報を使って
ゲーム空間を利用しています。
AIはもう十分にこの空間を利用できそうです。

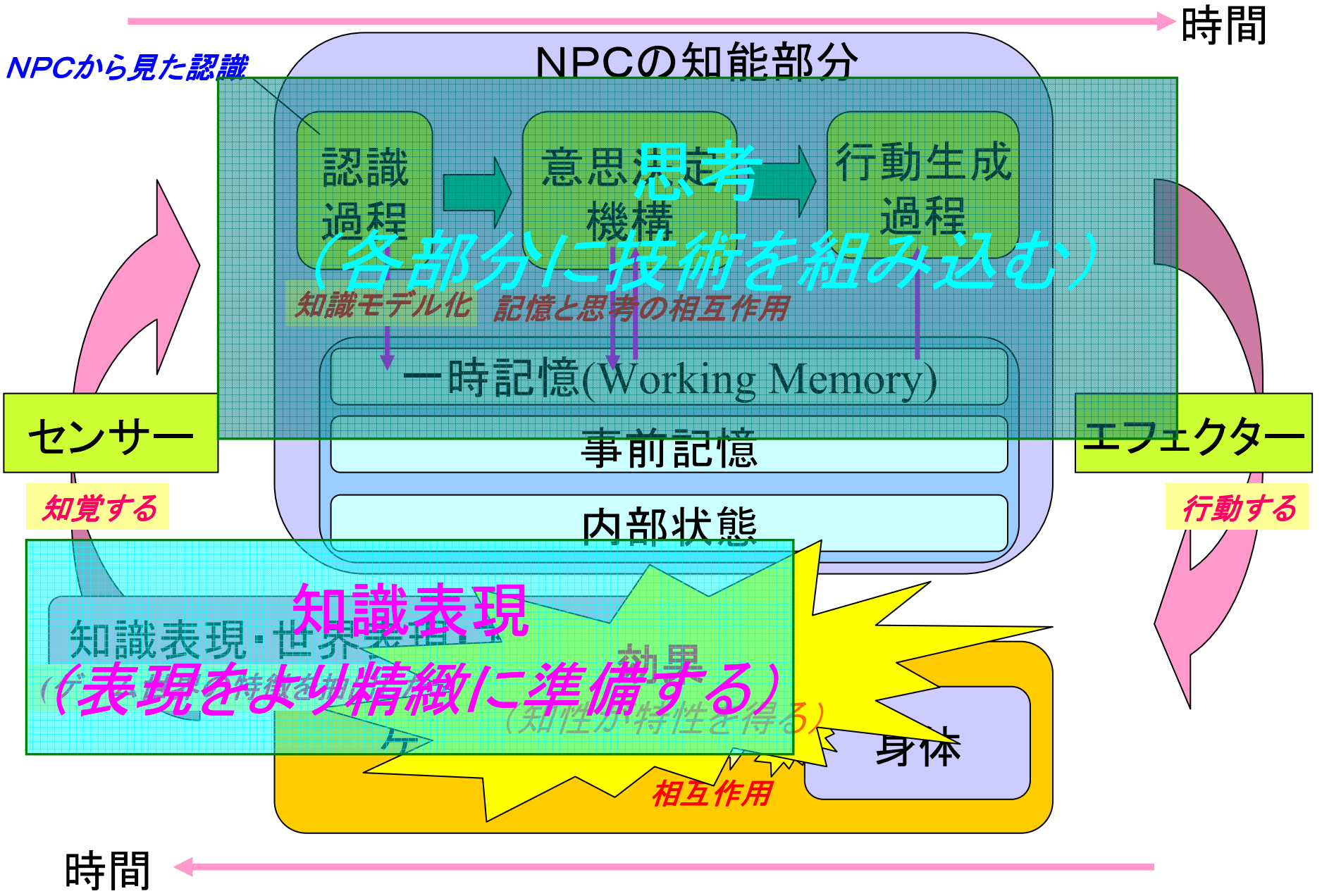
でも、そもそもの目的は何だったでしょうか？
AIの思考は、この能力とどう関係するのか、
次の章で見てみましょう。

第４章

エージェント・アーキテクチャ

時間編

『技術を仕掛けて効果を得る』



プランニング

「思考部分」には、
本当にいろいろな組み立て方がありますが、

ここでは特に時間と関係の深い
「プランニングの手法」を解説します。

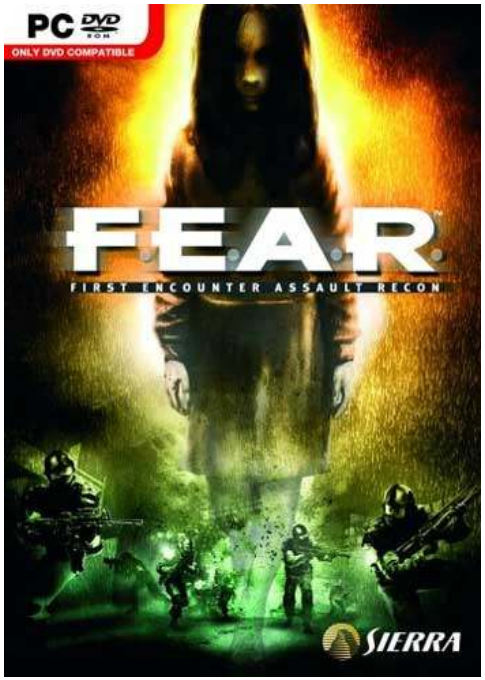
第４章　エージェント・アーキテクチャ
時間編

プランニング

(1) 連鎖プランニング（F.E.A.R.）

(2) 階層型プランニング（Chromehounds）

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ーキングな名作。
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 NPCの課題

何をするべきか、を自分で見つける。

見つけた目的を、如何にするべきか、を自分で考える。

（C4アーキテクチャー＋ゴール指向プランニング）

ラット

動き回るが攻撃しない。

アサシン

壁や天井を這う

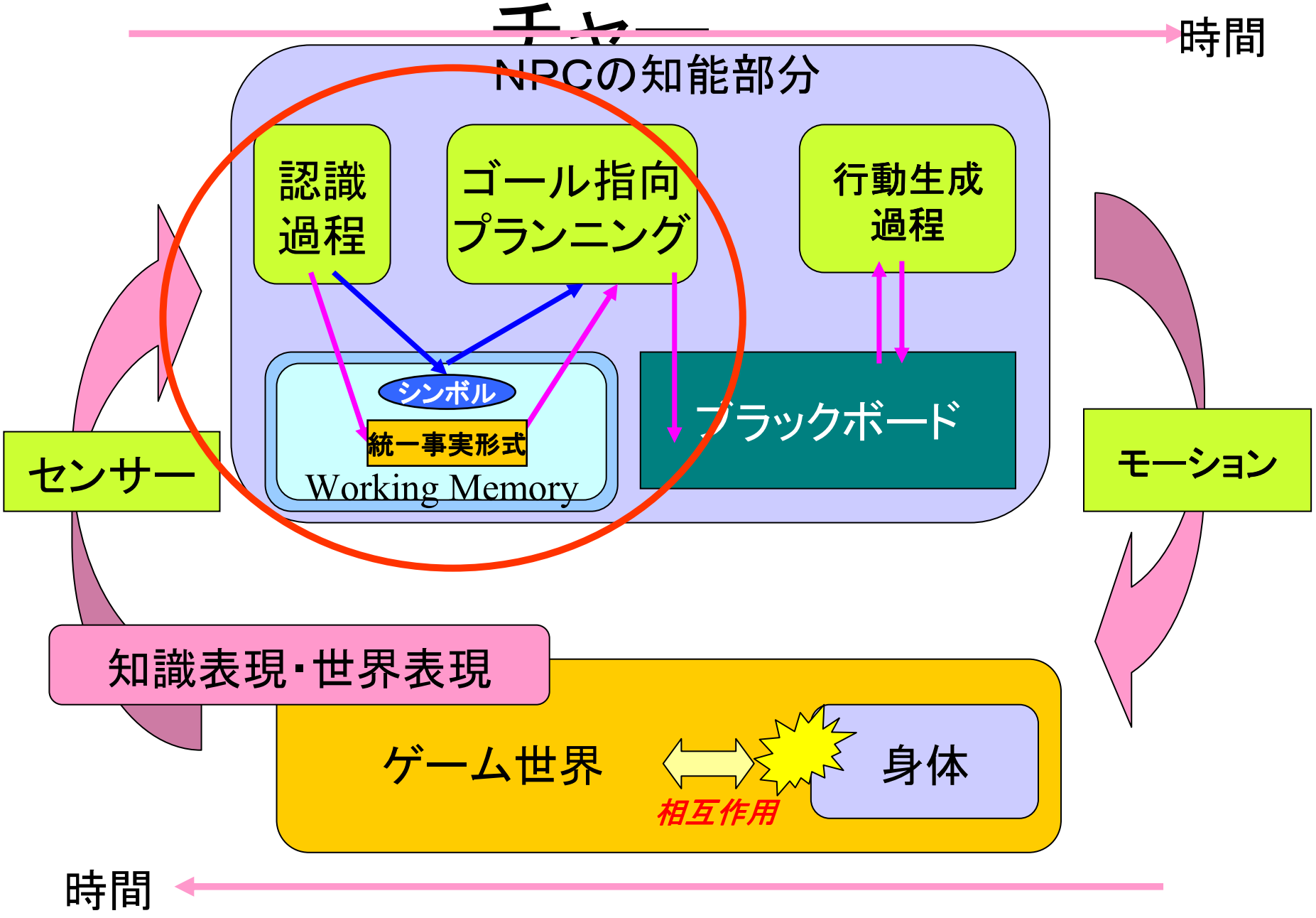
人間

普通の人間。

敵

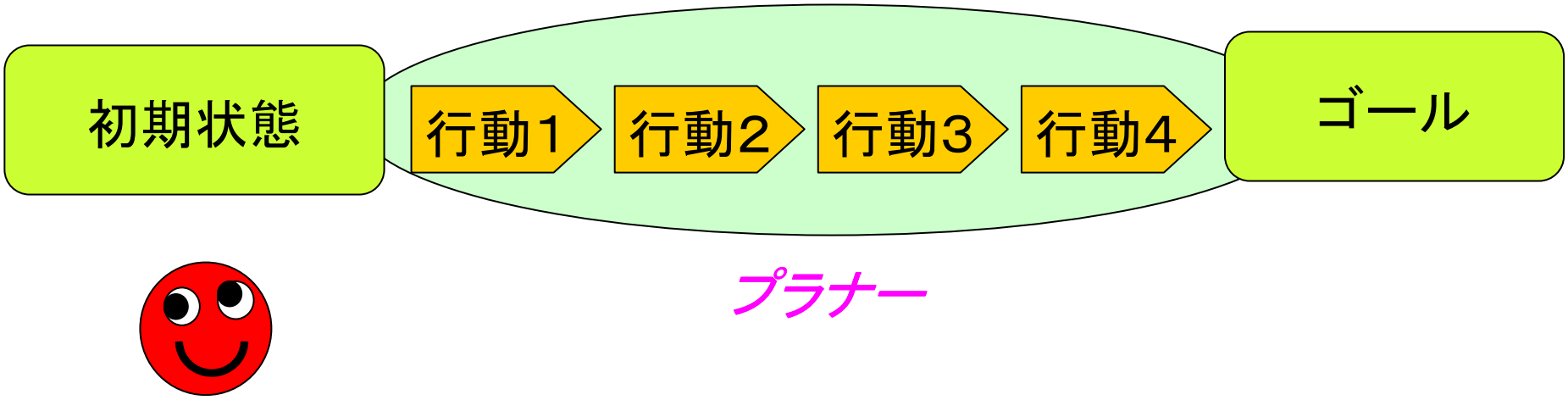
学術的な成果（C4アーキテクチャー）を、
どう実際のゲームに応用しているか見てみよう！

F.E.A.R NPCのAIのアーキテクチャ



プランニングとは？

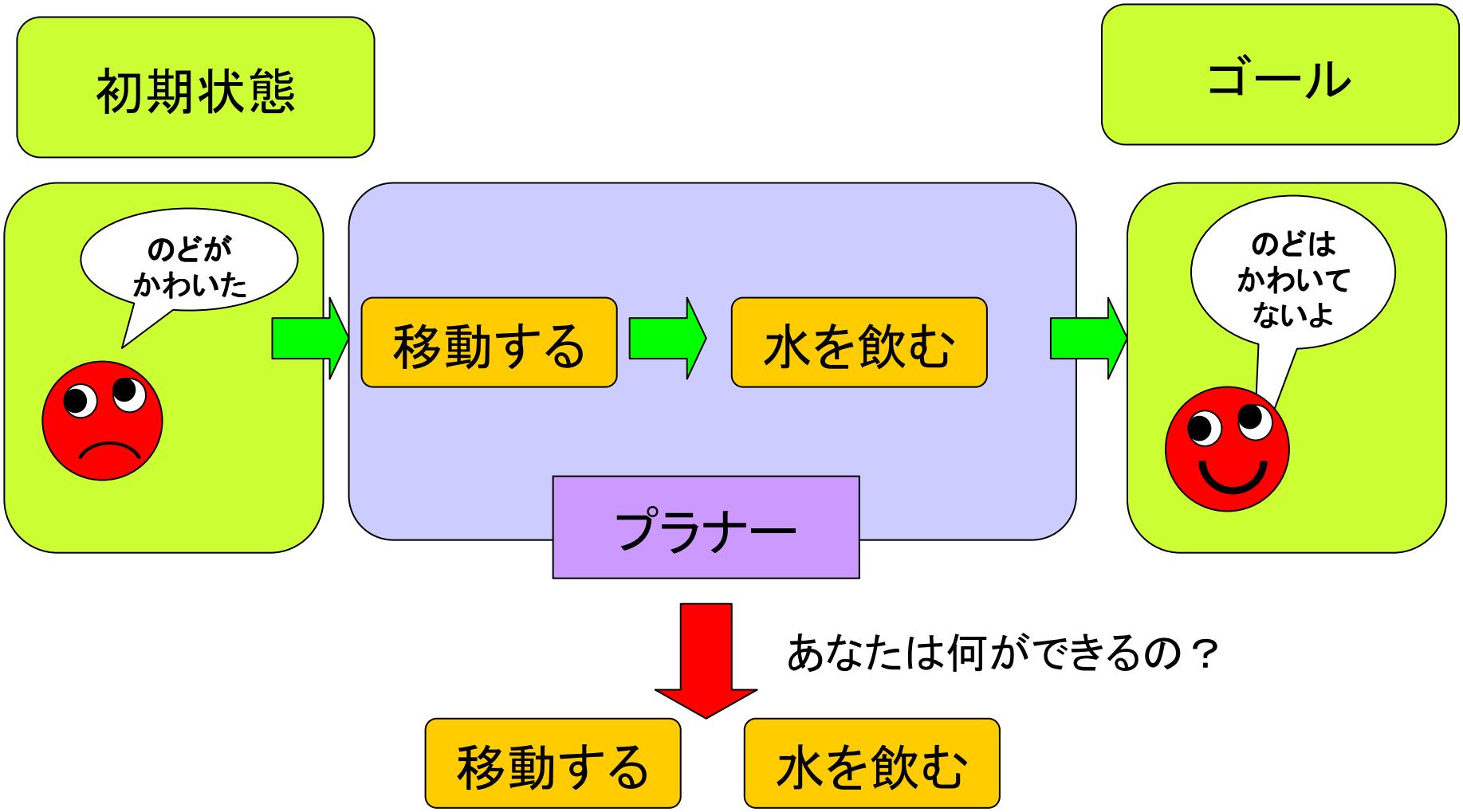
プランニングとは？



基本概念：　初期状態　　ゴール　　プラナー

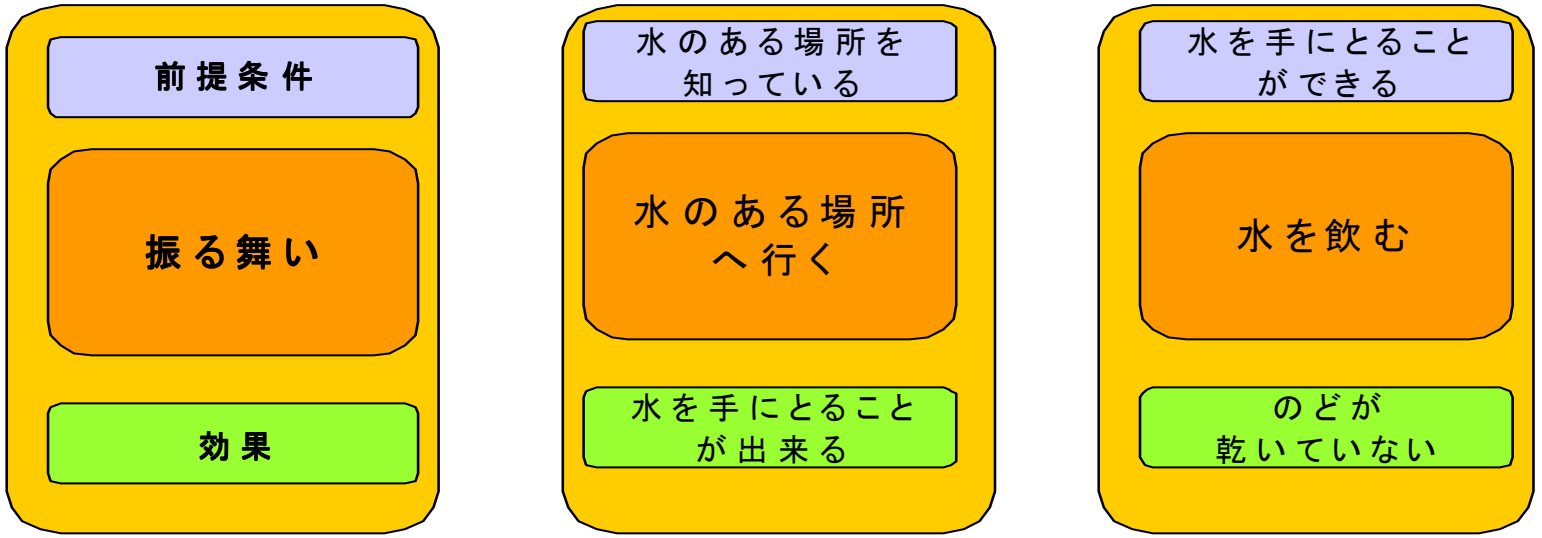
アクションプランニングの例

行動(アクション)によるプランニング＝　アクションプランニング



連鎖による方法

プランニングにおける行動の表現

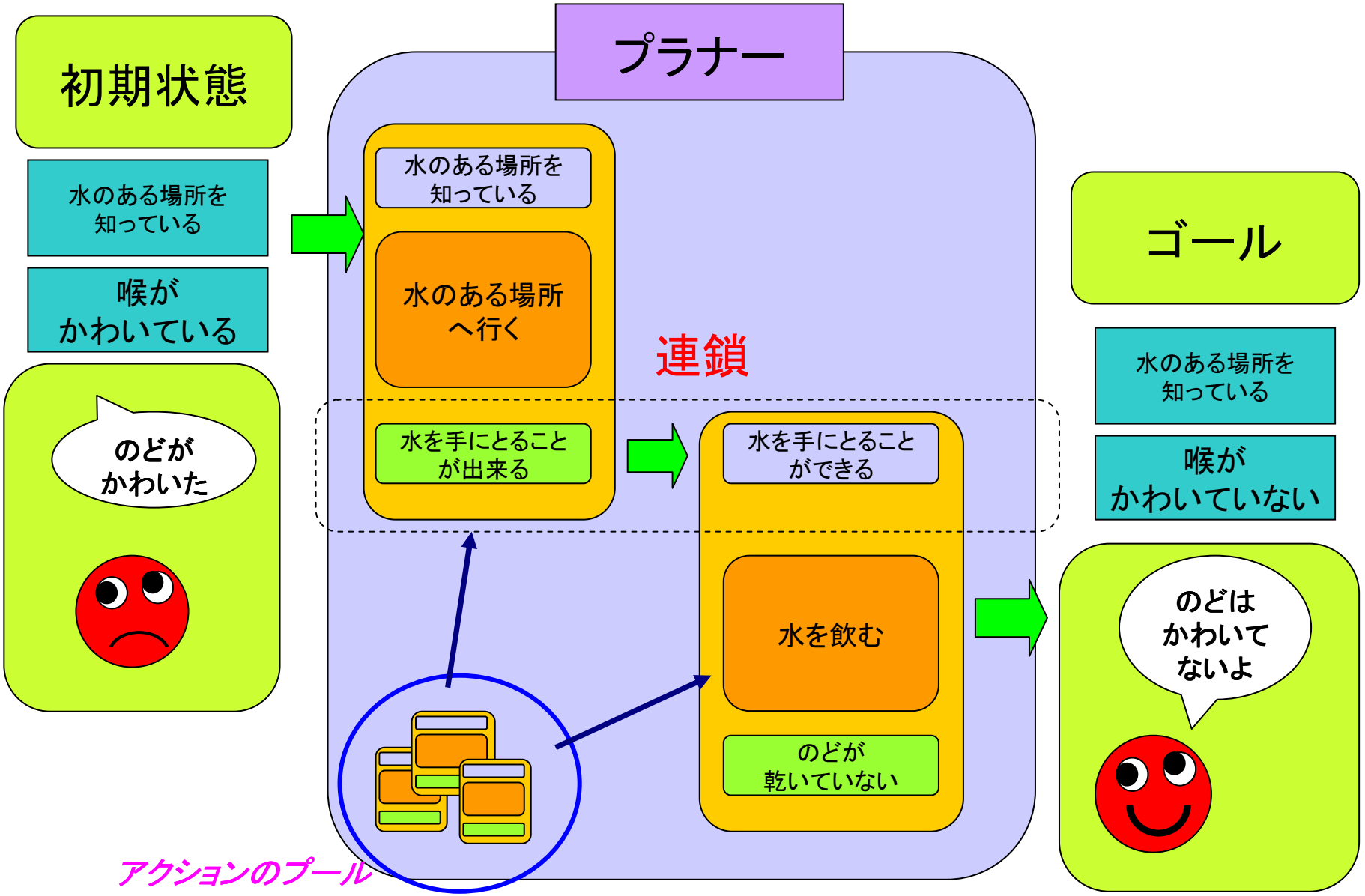


前提条件　＝　その行動を実行するために必要な条件
効果　　＝　その行動を起こしたことによる効果

F.E.A.Rでは、前提条件、効果をシンボルで記述する。

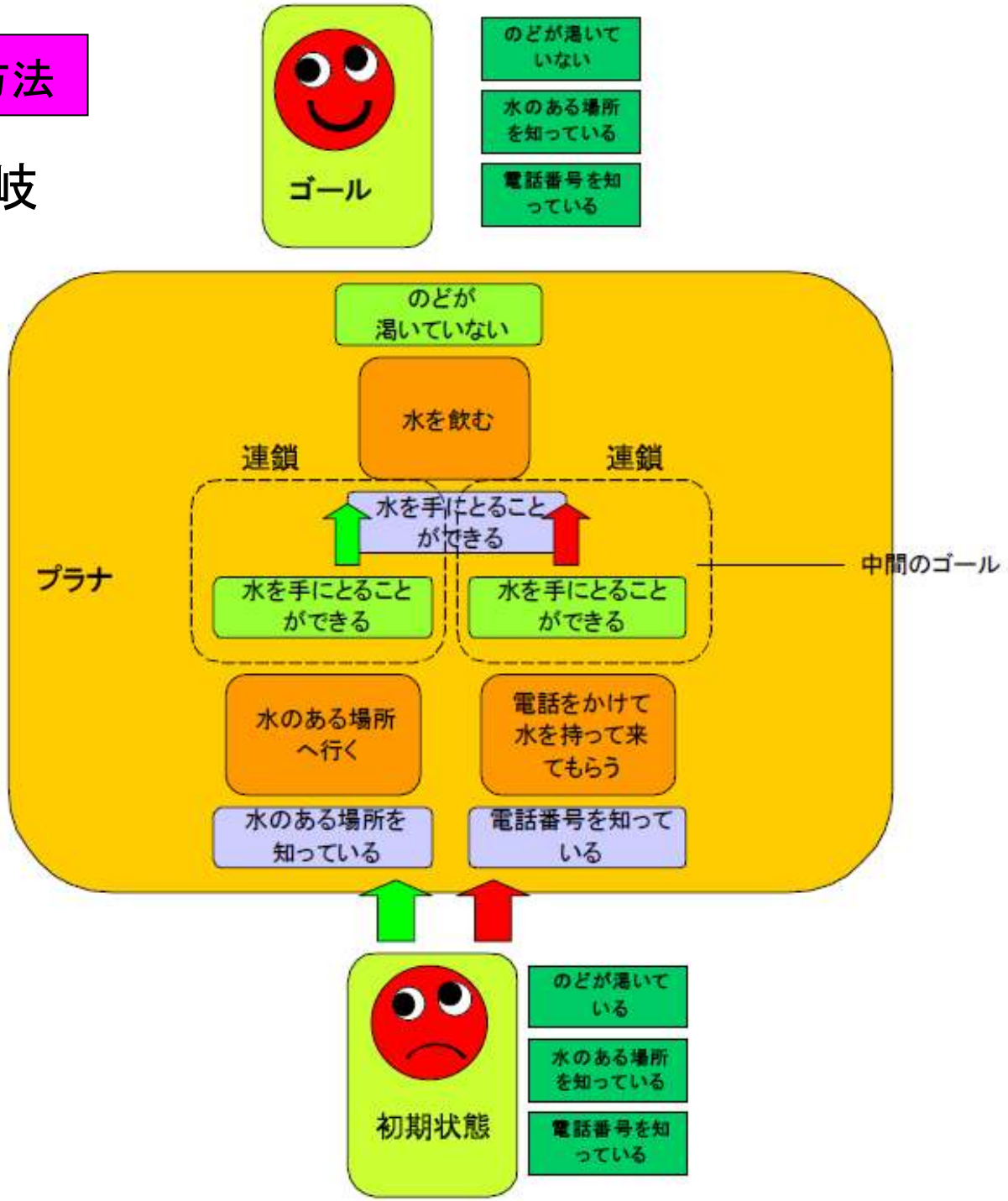
連鎖による方法

連鎖によるプランニング



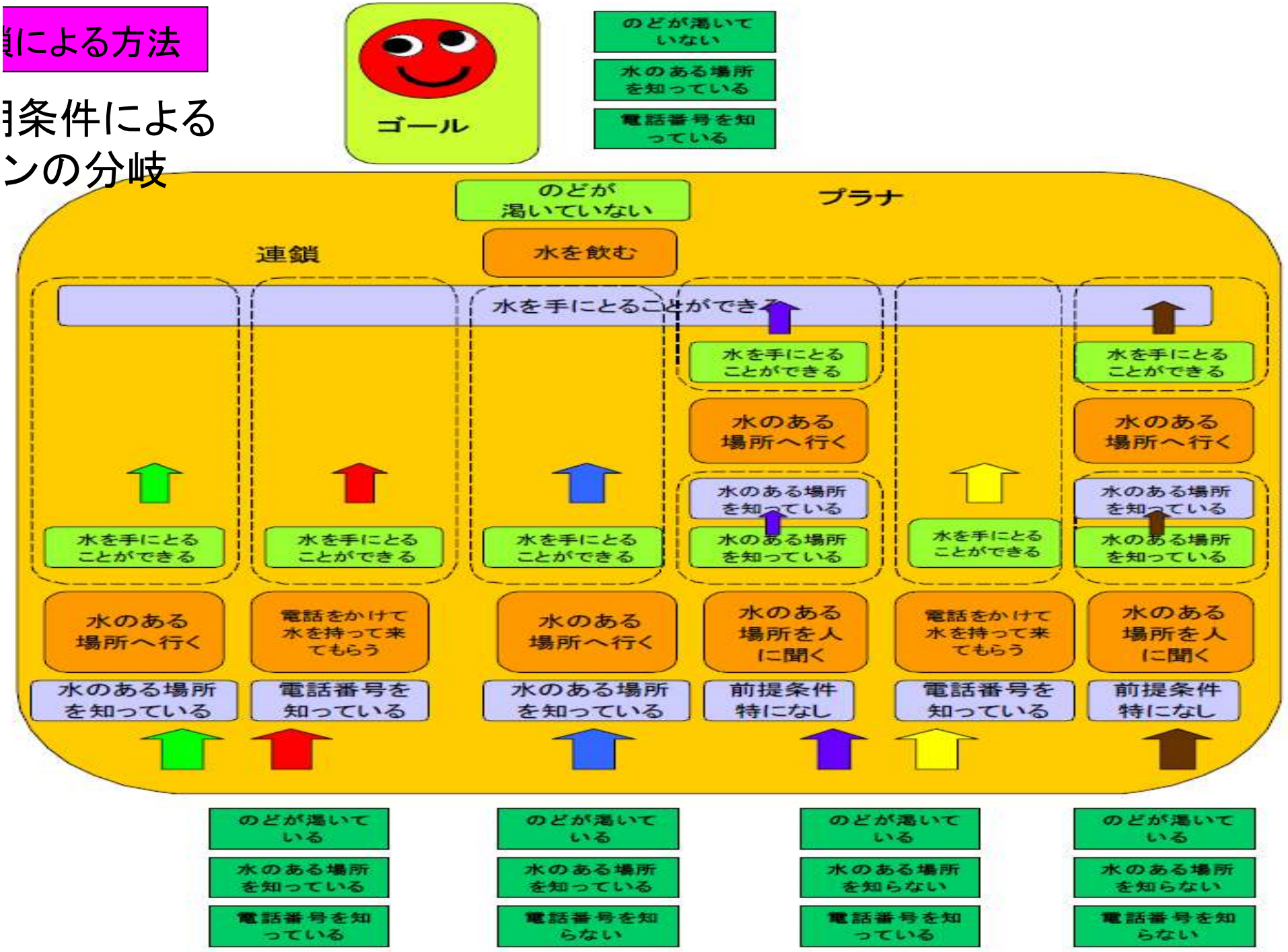
連鎖による方法

プランの分岐



識による方法

前条件による
ンの分岐



プランニング説明終了

F.E.A.Rのプランニング①　シンボル

kTargetIsDead = ture
この兵士Aは死んだ

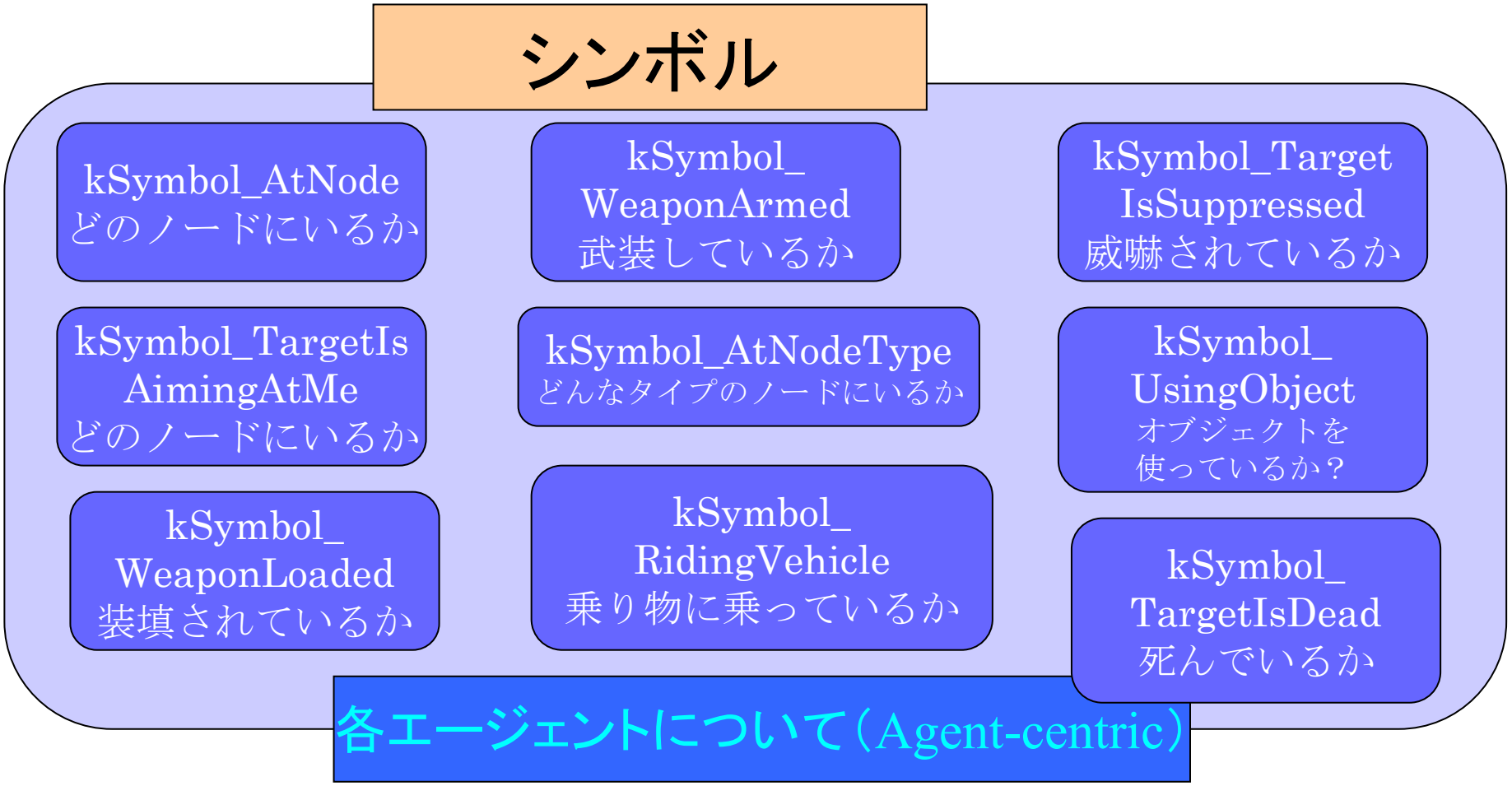
kTargetAtme = ture
この兵士Bは自分を狙っている

kWeaponIsLoaded = false
私Cの武器は装填済みでない

F.E.A.Rのプランニング①　シンボル

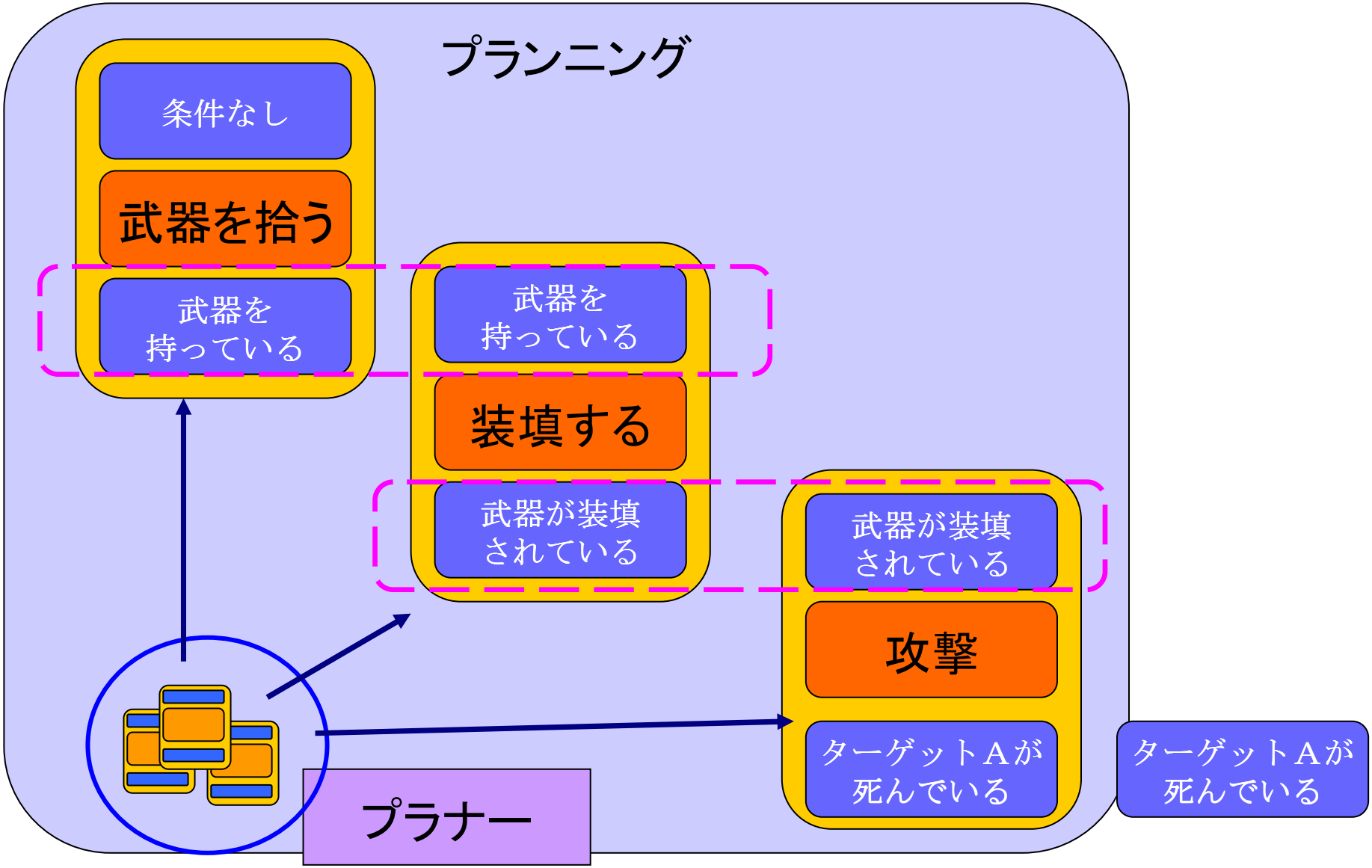
エージェントの認識する世界をもっとシンプルに表現したい

20個のシンボルで世界を集約して表現する



このシンボルをプランニングへ

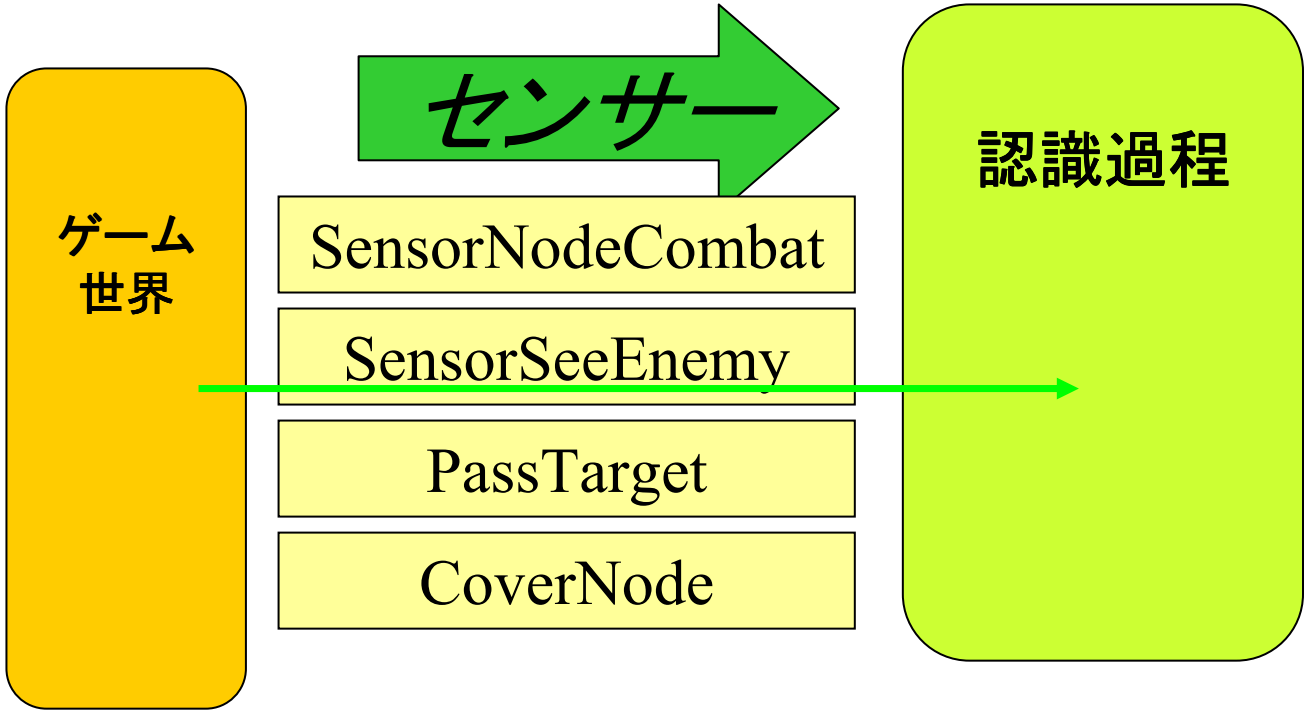
F.E.A.R.のプランニング②
シンボルによる連鎖プランニング



プランニング

F.E.A.R のNPCのセンサー

センサー ＝ 五感からの情報、及び、五感から得られるはずの情報を模擬する機能

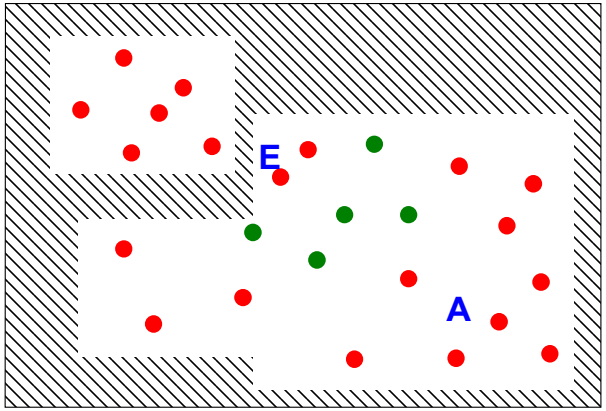


レイキャスト（視線チェック）、パス検索など重たい処理からなる

SensorNodeCombat　隠れる、或いは、隠れながら攻撃できる場所を探す。
SensorSeeEnemy　敵が見えるかチェック
PassTarget　戦術ポイントまでのパスを見つけ、かつそのパスが敵から安全であることをチェックする。
CoverNode　隠れることができるノード

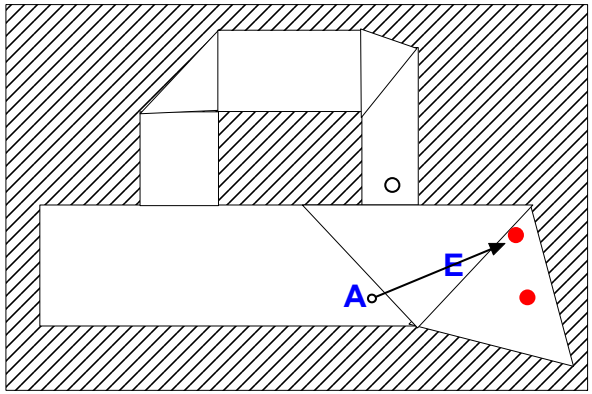
F.E.A.R のCOMの感覚（センサー）

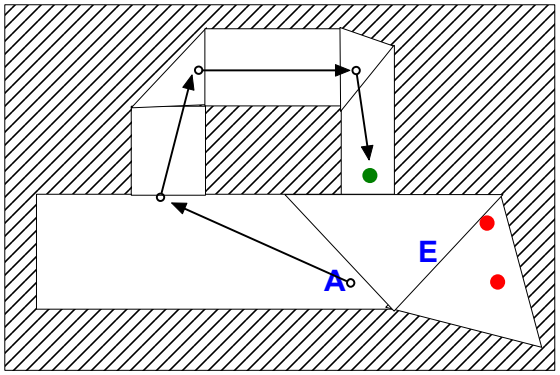
CoverNode

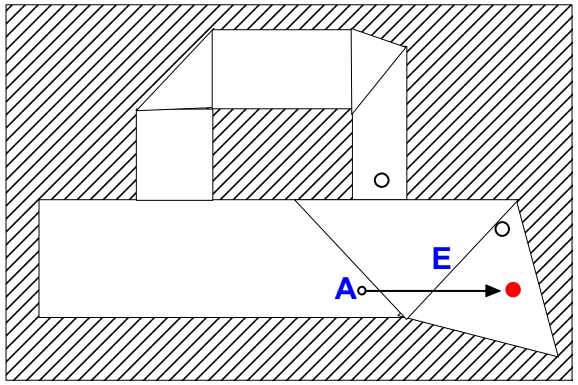


１秒に３回アップデート

PassTarget



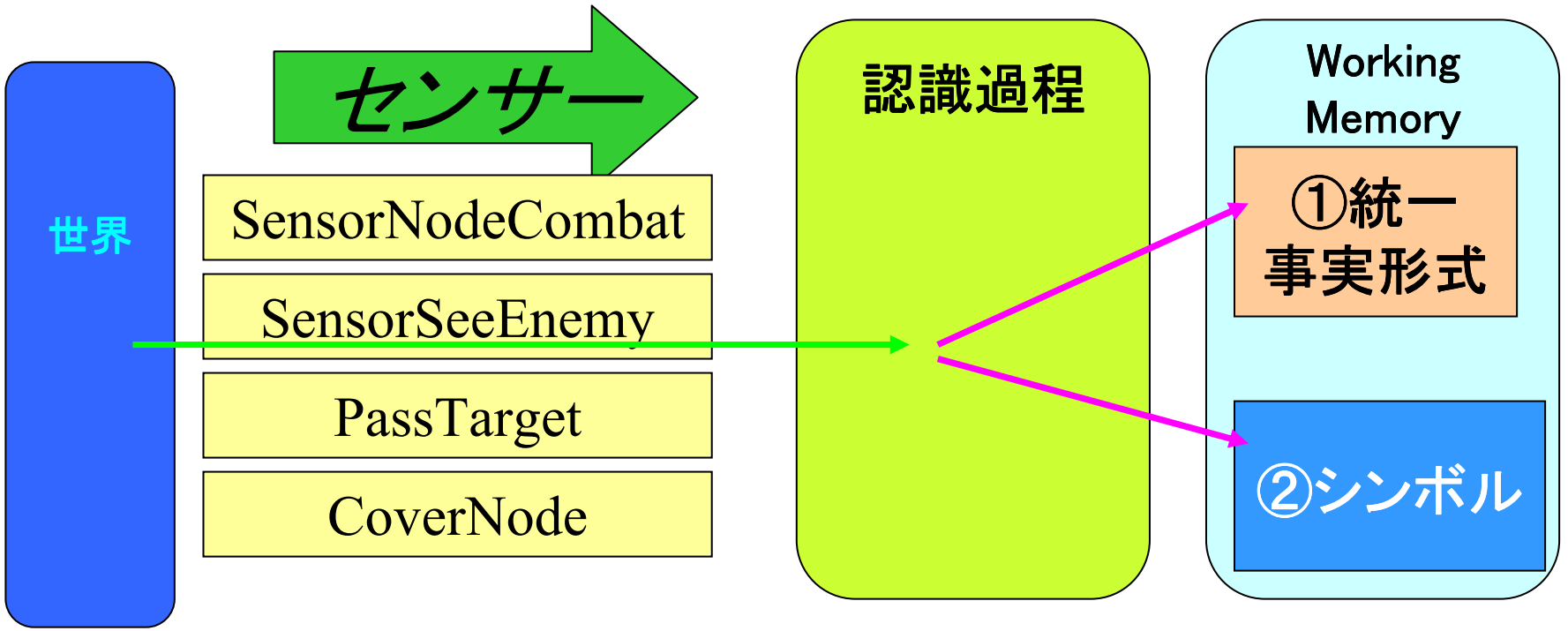




戦術ポイントまでの安全なパスを探す

F.E.A.R のCOMの感覚（センサー）

センサー ＝ 五感からの情報、及び、五感から得られるはずの情報を模擬する機能



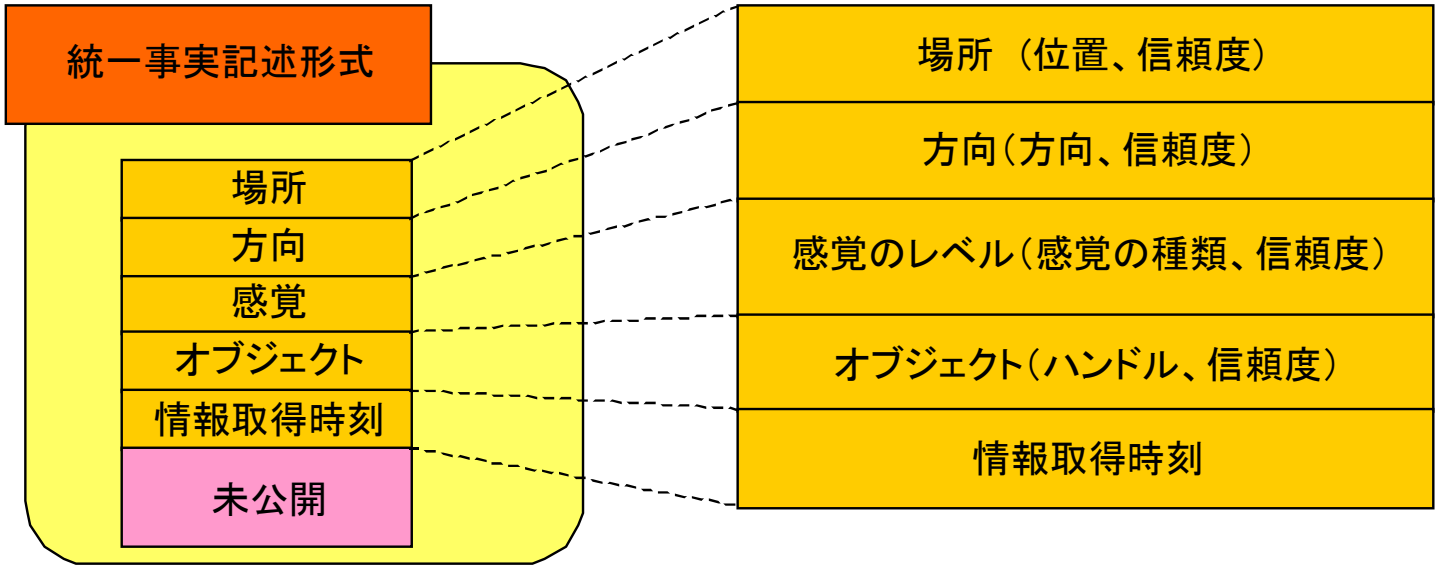
レイキャスト（視線チェック）、パス検索など重たい処理からなる関数

SensorNodeCombat　隠れる、或いは、隠れながら攻撃できる場所を探す。
SensorSeeEnemy　敵が見えるかチェック
PassTarget　戦術ポイントまでのパスを見つけ、かつそのパスが敵から安全であることをチェックする
CoverNode　隠れることができるノード

①統一事実記述形式

キャラクター | オブジェクト | 任務 | 事件 | パス | 欲求 | ノード

全て以下の形式（フォーマット）で記述する。



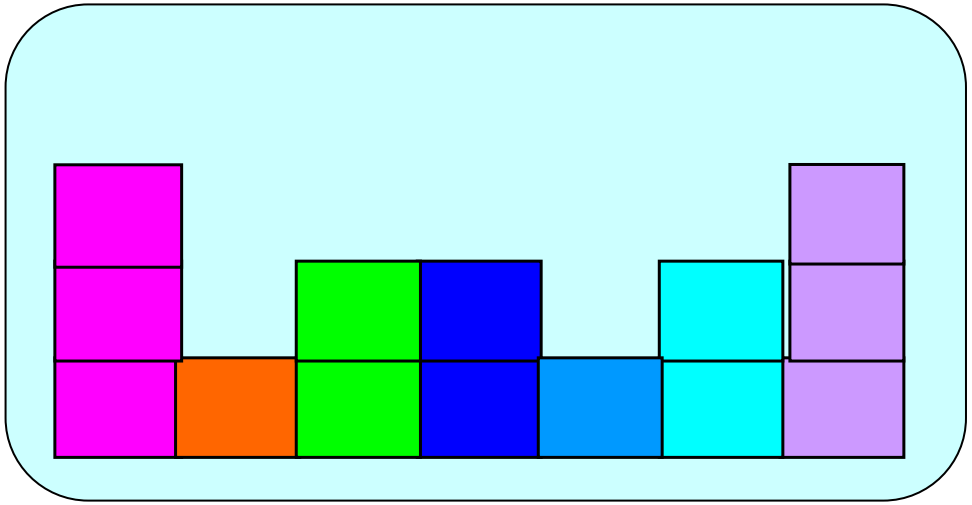

```
WorkingMemoryFact
{
   Attribute<Vector3D>      Position
   Attribute<Vector3D>      Direction
   Attribute<StimulusType> Stimulus
   Attribute<Handle>        Object
   Attribute<float>         Desire
   ...
   float              fUpdateTime
}
```

全部で本当は16個の属性がある

```
Attribute<Type>
{
   Type  Value
   float fConfidence
}
```

①統一事実記述形式

記憶領域に認識した事実をWorking Memoryへ蓄積する

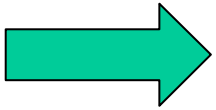


Working Memory

敵Cが時刻Mに位置Zにいた
信頼度= 0.6

敵Aが時刻Lに位置Xにいた
信頼度 = 1.0

敵Bが時刻Mに位置Yにいた
信頼度 = 0.0



ターゲットを決める

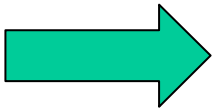
Blackboard

ターゲットはAにする

プランニングの欠点を補い
リプランニングとは？

「プランニングの最大の欠点は、
自分のプランに従わねならない」ということ

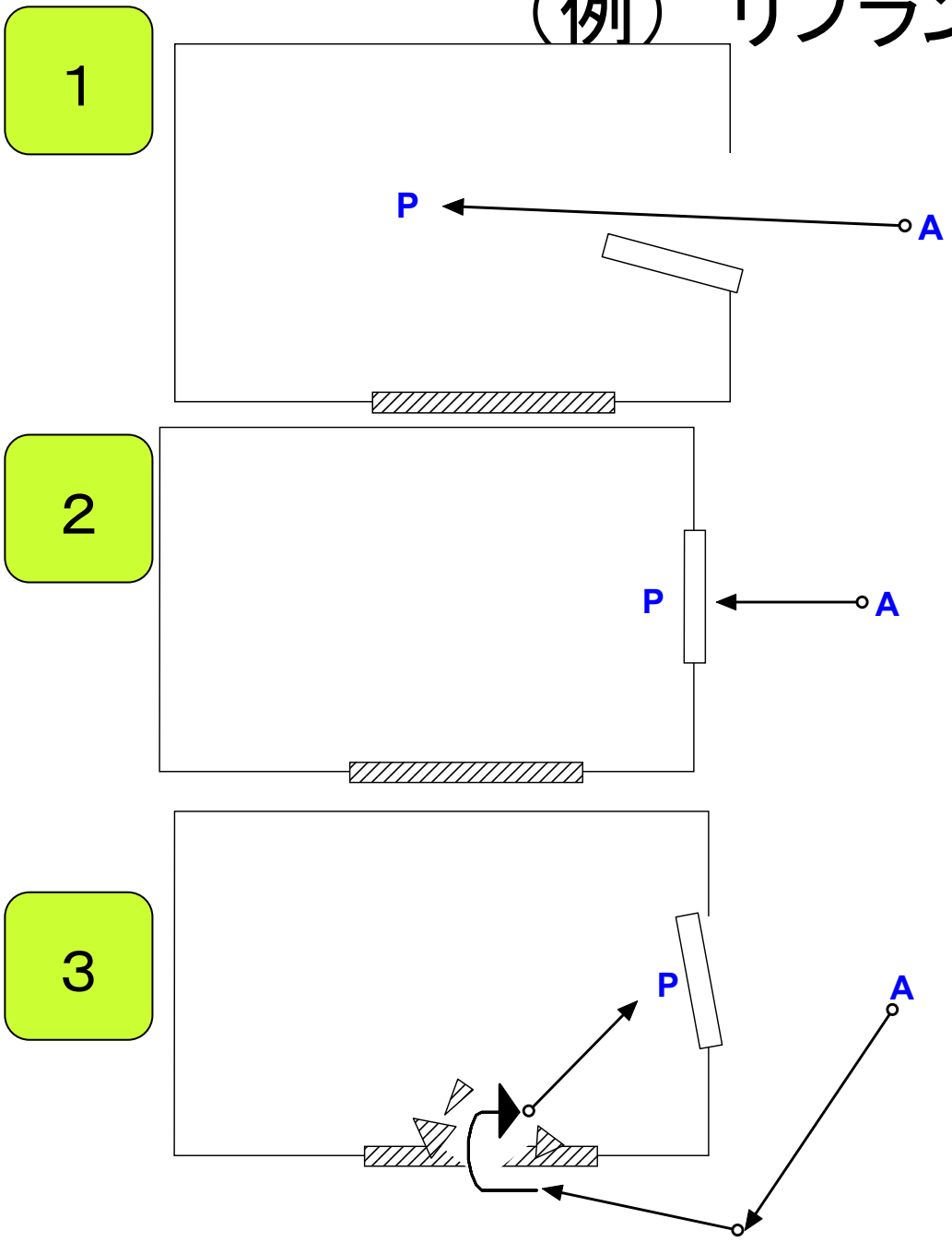
プランを実行している間に、状況が変わってしまう。



リプランニング

プランのやり直し

（例） リプランニング



Aが「Pを攻撃する」という
ゴールを選択して実行

Pへ向かう途中、Pがドア
を締める。

リプラニング

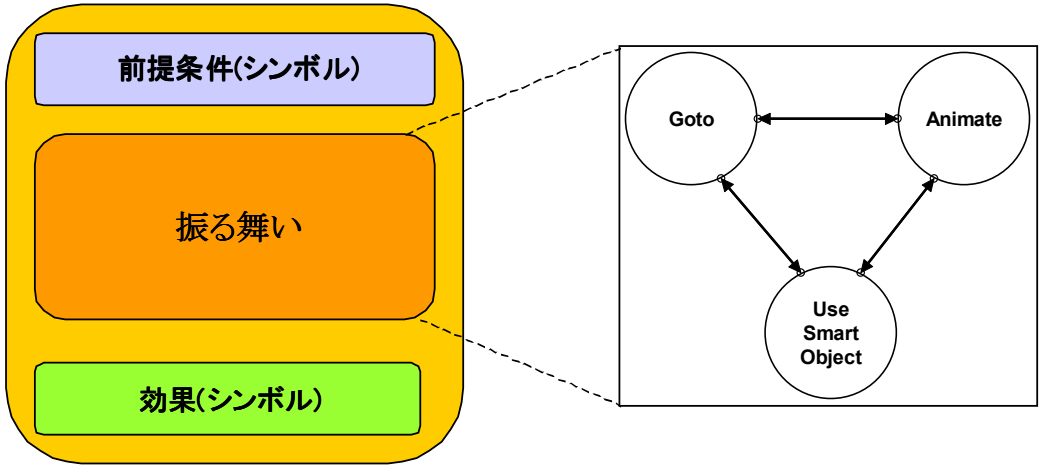
「脇のドアを壊して」「攻撃する」
ゴールを設定して攻撃

リプランニングによる動的な変化への対応

ドアを開けて
やって来ます！

ドアを内側から
抑えてやる！

F.E.A.R における行動の記述



Soldier

	Action
1	AI/Actions/Attack
2	AI/Actions/AttackCrouch
3	AI/Actions/SuppressionFire
4	AI/Actions/SuppressionFireFromCover
5	AI/Actions/FlushOutWithGrenade
6	AI/Actions/AttackFromCover
7	AI/Actions/BlindFireFromCover
8	AI/Actions/AttackGrenadeFromCover
9	AI/Actions/AttackFromView
10	AI/Actions/DrawWeapon
11	AI/Actions/HolsterWeapon
12	AI/Actions/ReloadCrouch
13	AI/Actions/ReloadCovered
14	AI/Actions/InspectDisturbance
15	AI/Actions/LookAtDisturbance
16	AI/Actions/SurveyArea
17	AI/Actions/DodgeRoll
18	AI/Actions/DodgeShuffle
19	AI/Actions/DodgeCovered
20	AI/Actions/Uncover
21	AI/Actions/AttackMelee

Assassin

	Action
1	AI/Actions/Attack
2	AI/Actions/InspectDisturbance
3	AI/Actions/LookAtDisturbance
4	AI/Actions/SurveyArea
5	AI/Actions/AttackMeleeUncloaked
6	AI/Actions/TraverseBlockedDoor
7	AI/Actions/UseSmartObjectNodeMounted
8	AI/Actions/MountNodeUncloaked
9	AI/Actions/DismountNodeUncloaked
10	AI/Actions/TraverseLinkUncloaked
11	AI/Actions/AttackFromAmbush
12	AI/Actions/DodgeRollParanoid
13	AI/Actions/AttackLungeUncloaked
14	AI/Actions/LopeToTargetUncloaked

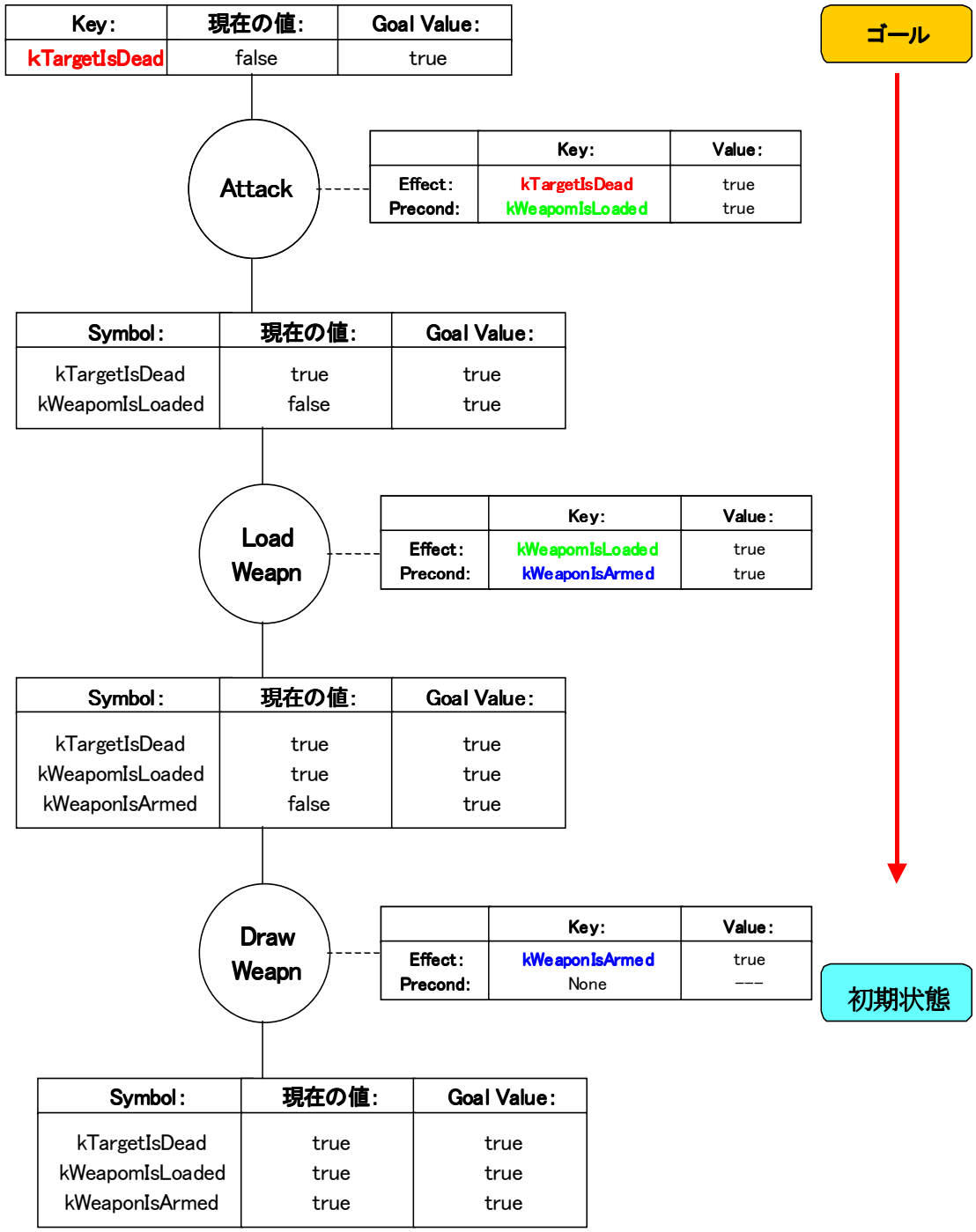
Rat

	Action
1	AI/Actions/Animate
2	AI/Actions/Idle
3	AI/Actions/GotoNode
4	AI/Actions/UseSmartObjectNode

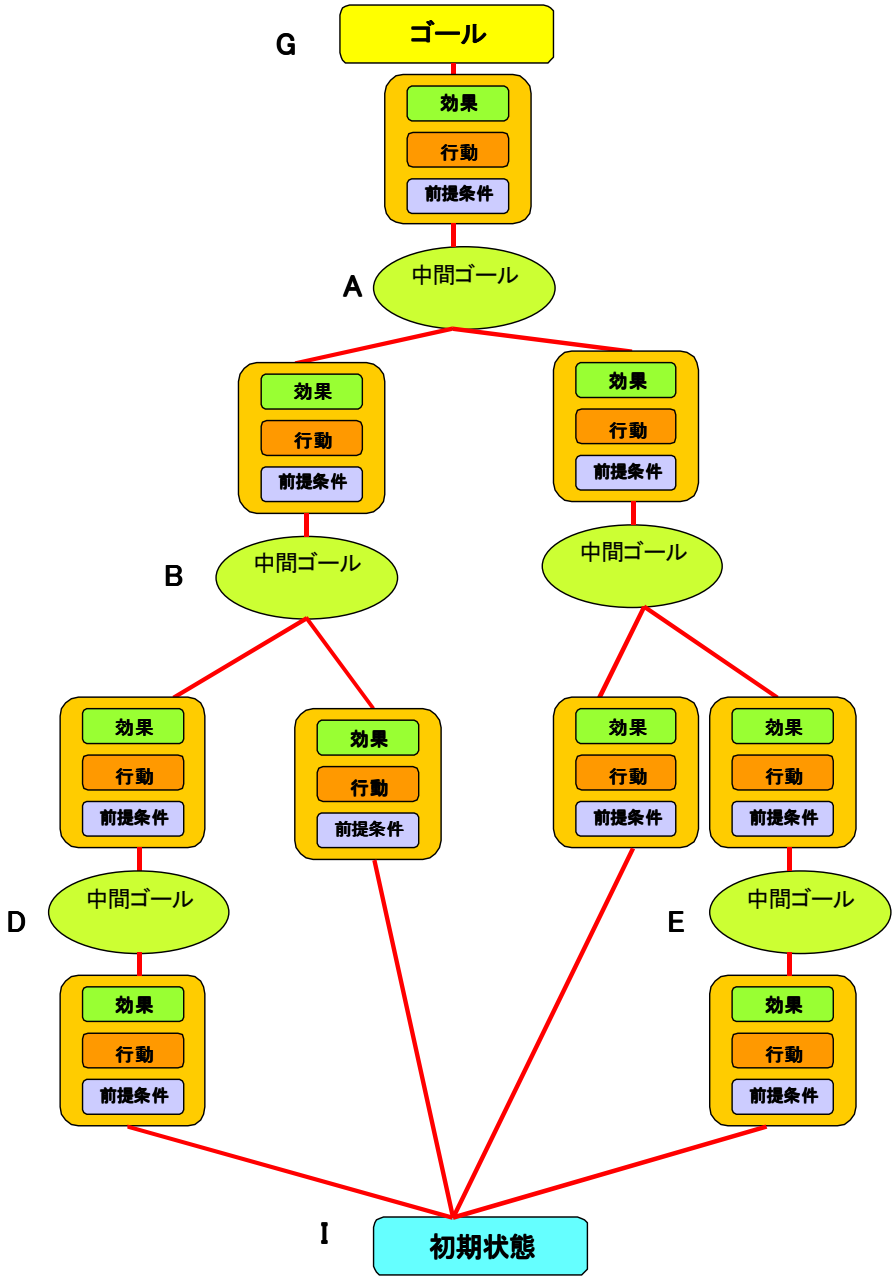
映像（移動する）

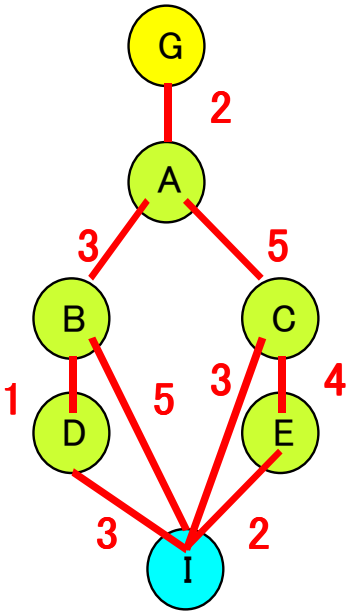
F.E.A.R

プランニング


Key:
現在の値:
Goal Value:
kTargetIsDead
false
true
ゴール
Attack
Key:
Value:
Effect:
Precond:
kTargetIsDead
kWeapomIsLoaded
true
true
Symbol:
現在の値:
Goal Value:
kTargetIsDead
kWeapomIsLoaded
true
false
true
true
Load Weapn
Key:
Value:
Effect:
Precond:
kWeapomIsLoaded
kWeaponIsArmed
true
true
Symbol:
現在の値:
Goal Value:
kTargetIsDead
kWeapomIsLoaded
kWeaponIsArmed
true
true
false
true
true
true
Draw Weapn
Key:
Value:
Effect:
Precond:
kWeaponIsArmed
None
true

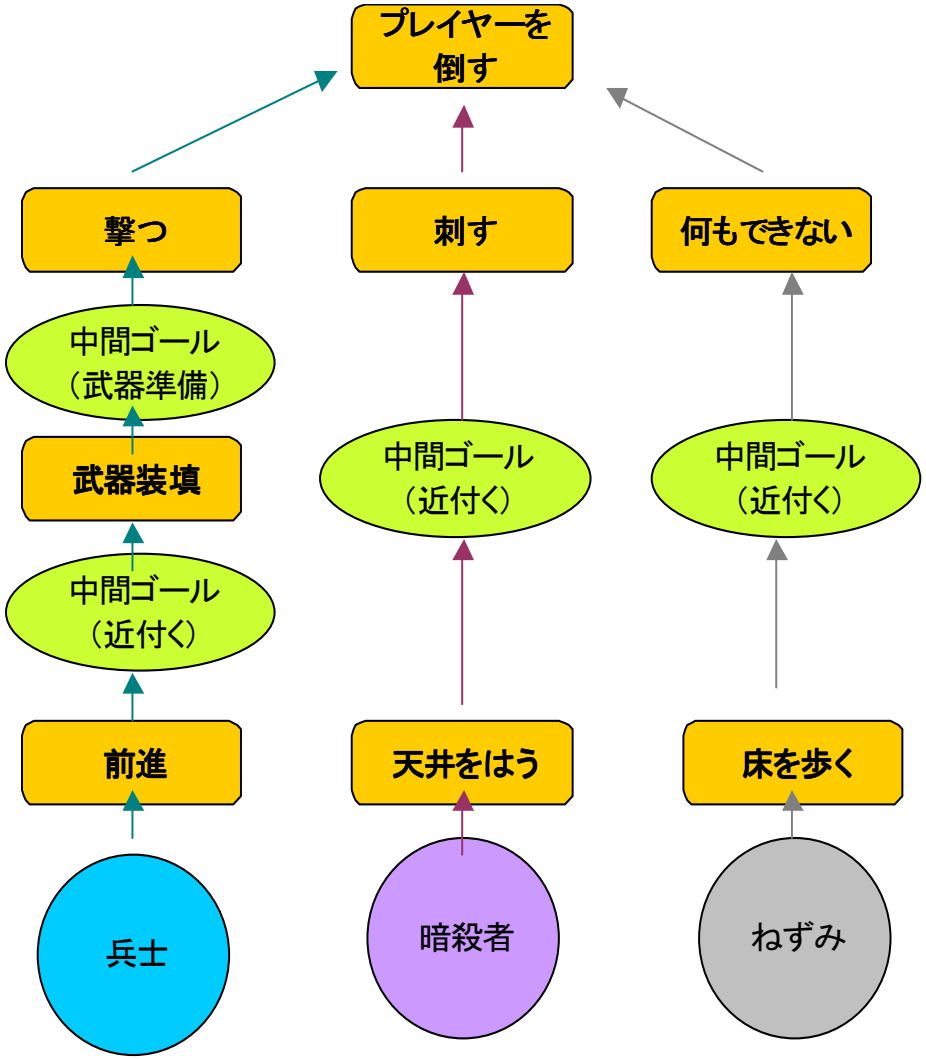
初期状態
Symbol:
現在の値:
Goal Value:
kTargetIsDead
kWeapomIsLoaded
kWeaponIsArmed
true
true
true
true
true
true





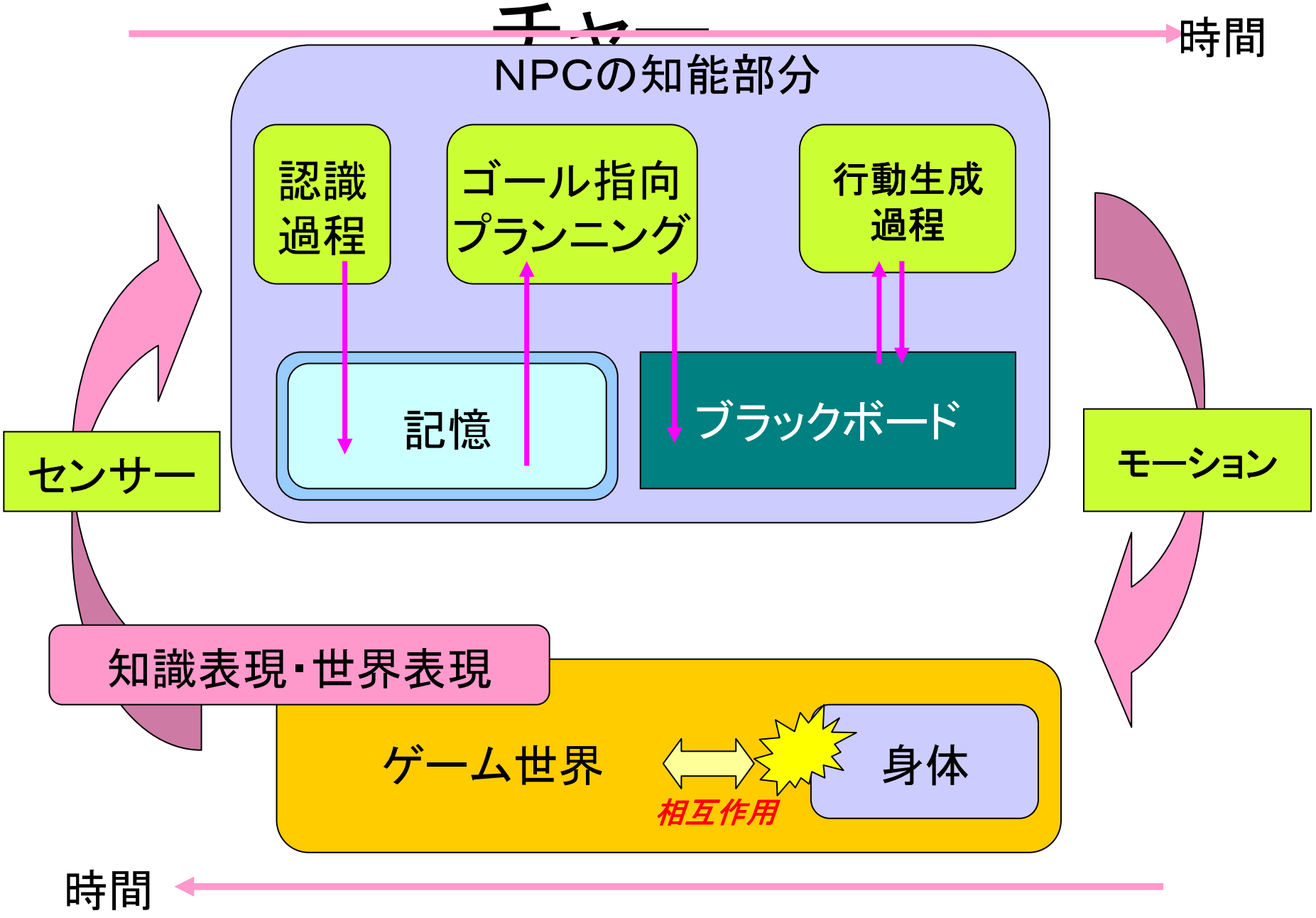
A* アルゴリズムによって検索

キャラクターの違いによって、同一のゴールに向かって異なるプランが生成される

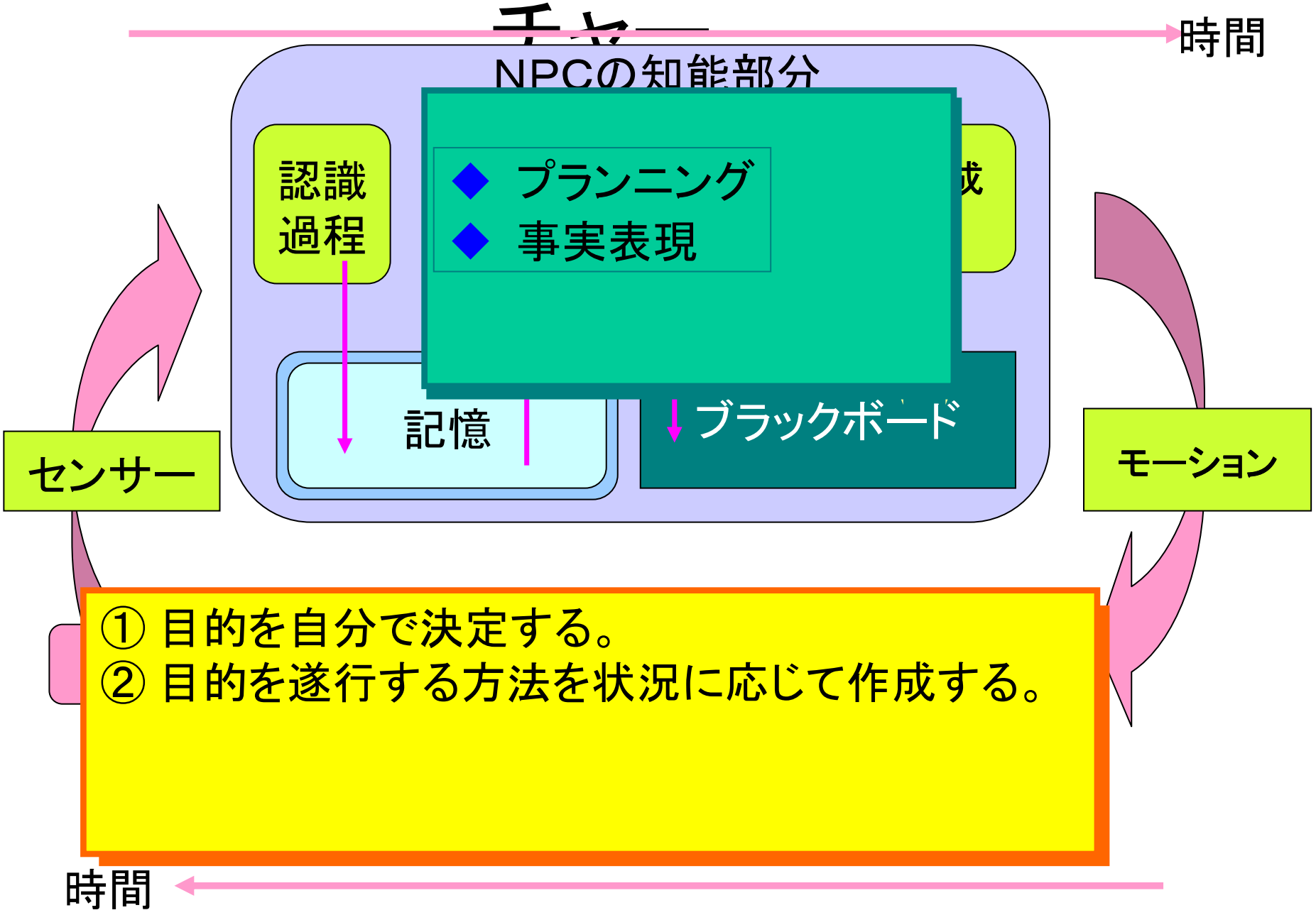


映像（移動する）

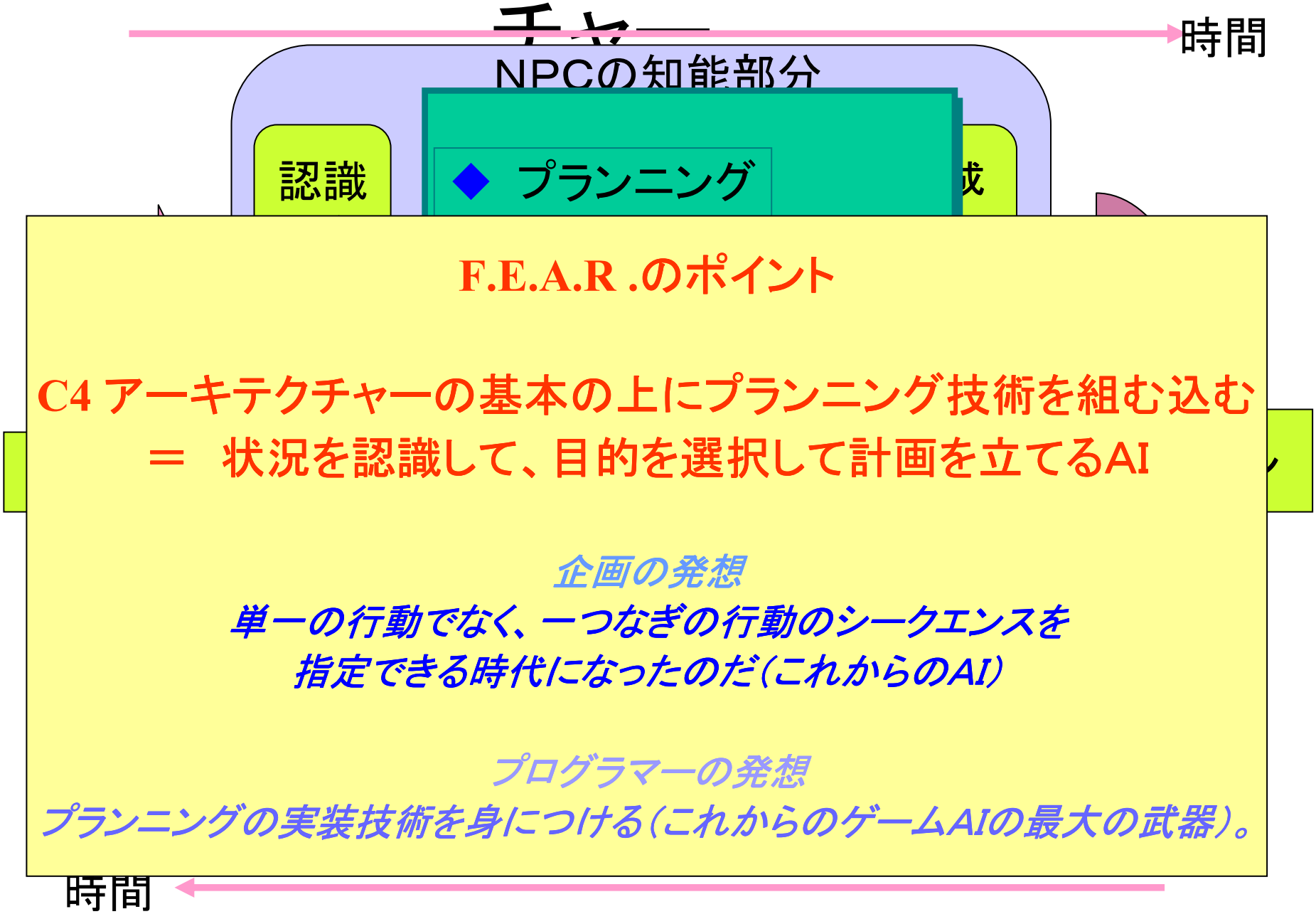
F.E.A.R NPCのAIのアーキテクチャー



F.E.A.R NPCのAIのアーキテクチャー



F.E.A.R NPCのAIのアーキテクチャー



F.E.A.R .のポイント

C4 アーキテクチャーの基本の上にプランニング技術を組む込む
＝　状況を認識して、目的を選択して計画を立てるAI

企画の発想

単一の行動でなく、一つなぎの行動のシークエンスを
指定できる時代になったのだ（これからのAI）

プログラマーの発想

プランニングの実装技術を身につける（これからのゲームAIの最大の武器）。

References for F.E.A.R.

- 論文　Orkin, J. (2006), **3 States & a Plan: The AI of F.E.A.R.**, Game Developer's Conference Proceedings.
- Jeff Orkin, Applying Goal-Oriented Action Planning to Games, AI Game Programming Wisdom 2, Charles River Media., 217-228, (2003)
- 参考文献(I) Mat Buckland, “Programming Game AI by Example”, Chapter 9, WORDWARE publishing　（第9章とそのサンプルコードはゴール指向型プランニングの優れた解説です。　教科書をお探しの方は、ゲームAIについて最良の書の一つです。推薦します。）
  オライリージャパンより「実例で学ぶゲームAIプログラミング」として翻訳が出版されています。
- 参考文献(II) Jeff Orkin’ HP http://web.media.mit.edu/~jorkin/
  （Jeff Orkin は、米におけるゲームAIにおけるゴール指向型プランニングの推進者の一人。著者のサイトに豊富な情報があります。）
- 参考文献(III) 星野 瑠美子,“F.E.A.R.のAI - 3つの状態とゴール指向プランニングシステム”
  http://www.igda.jp/modules/xeblog/?action_xeblog_details=1&blog_id=62
  (IGDA Japan サイト内、GDC2006講演の紹介）

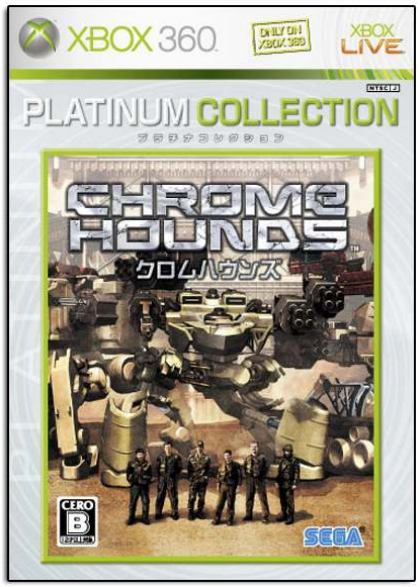
第４章　エージェント・アーキテクチャ
時間編

プランニング

(1) 連鎖プランニング（F.E.A.R.）

(2) 階層型プランニング（Chromehounds）

ChromeHounds



内容：オンライン上でロボットチーム対戦

開発元： FromSoftware

出版: SEGA

Hardware: Xbox360

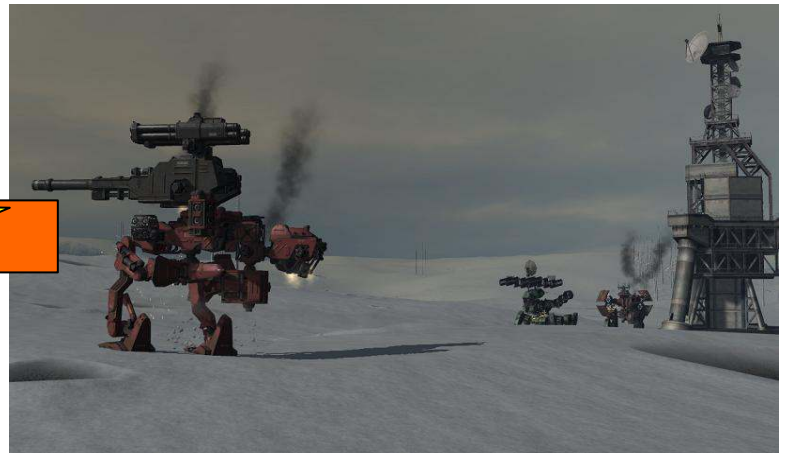
出版年： 2006年

リアルな戦場で、ロボットからなるチーム同士で戦うネットワークゲーム。
オンライン上で、プレイヤーチームと、AIチームとの対戦が可能。

ChromeHounds NPCの課題

人間の代わりに、
プレイヤーチームと戦う COM のチームを作る。

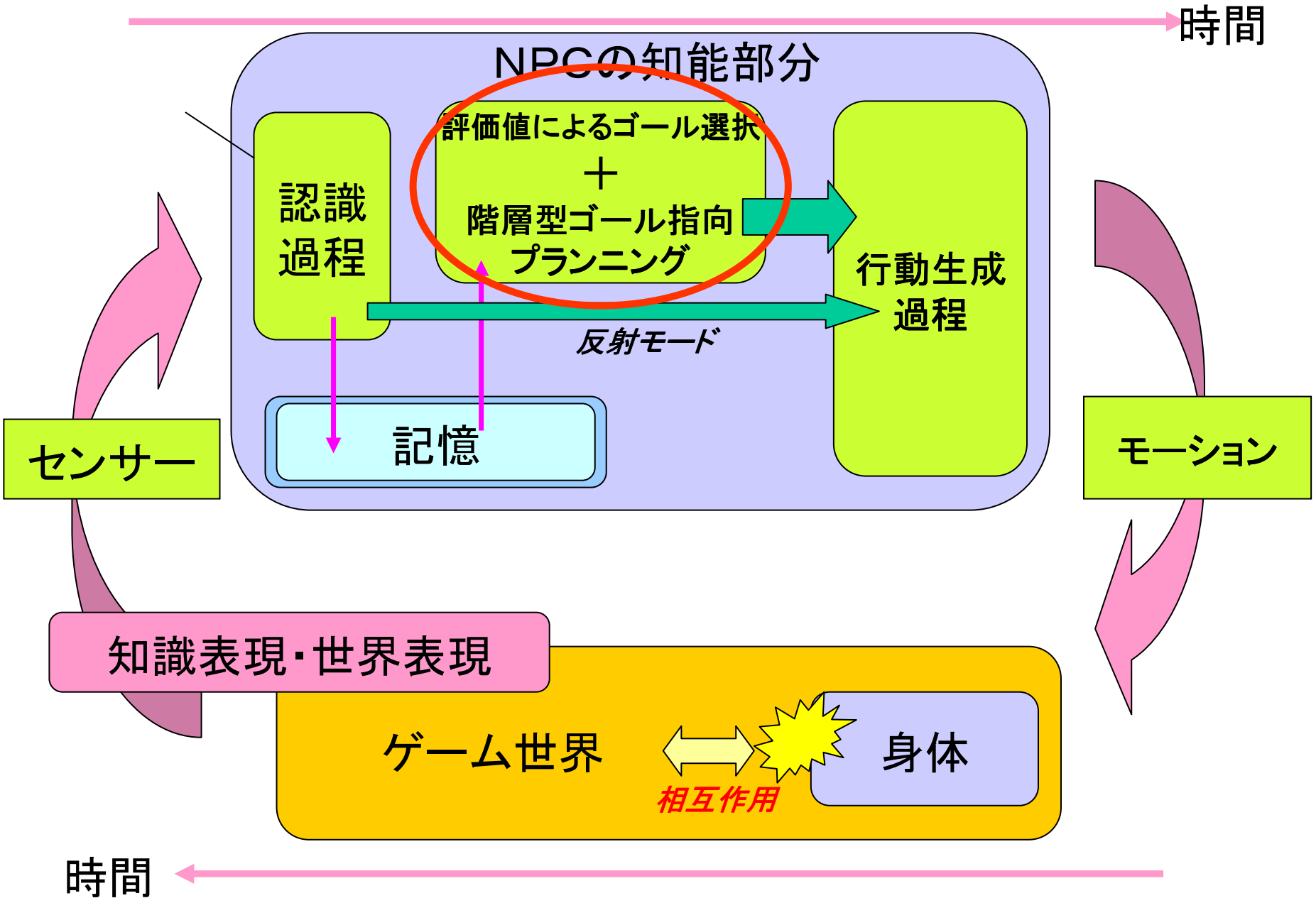
（マルチエージェント）



プレイヤーチーム（最大6名）

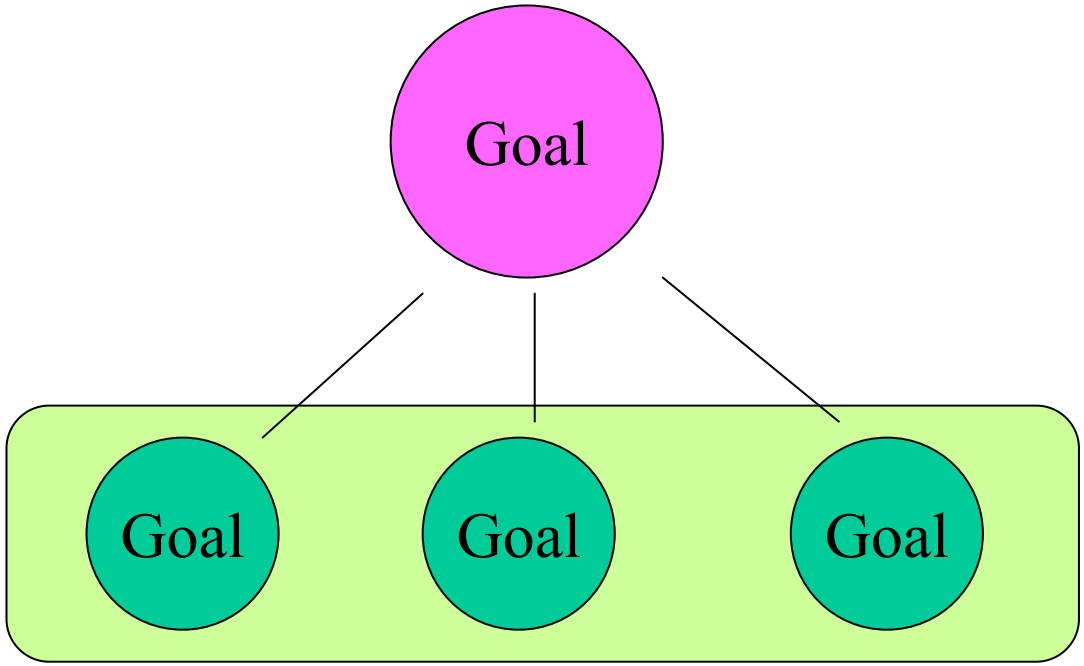
AIチーム（最大6名）

ChromeHounds NPCのAIのアーキテクチャー

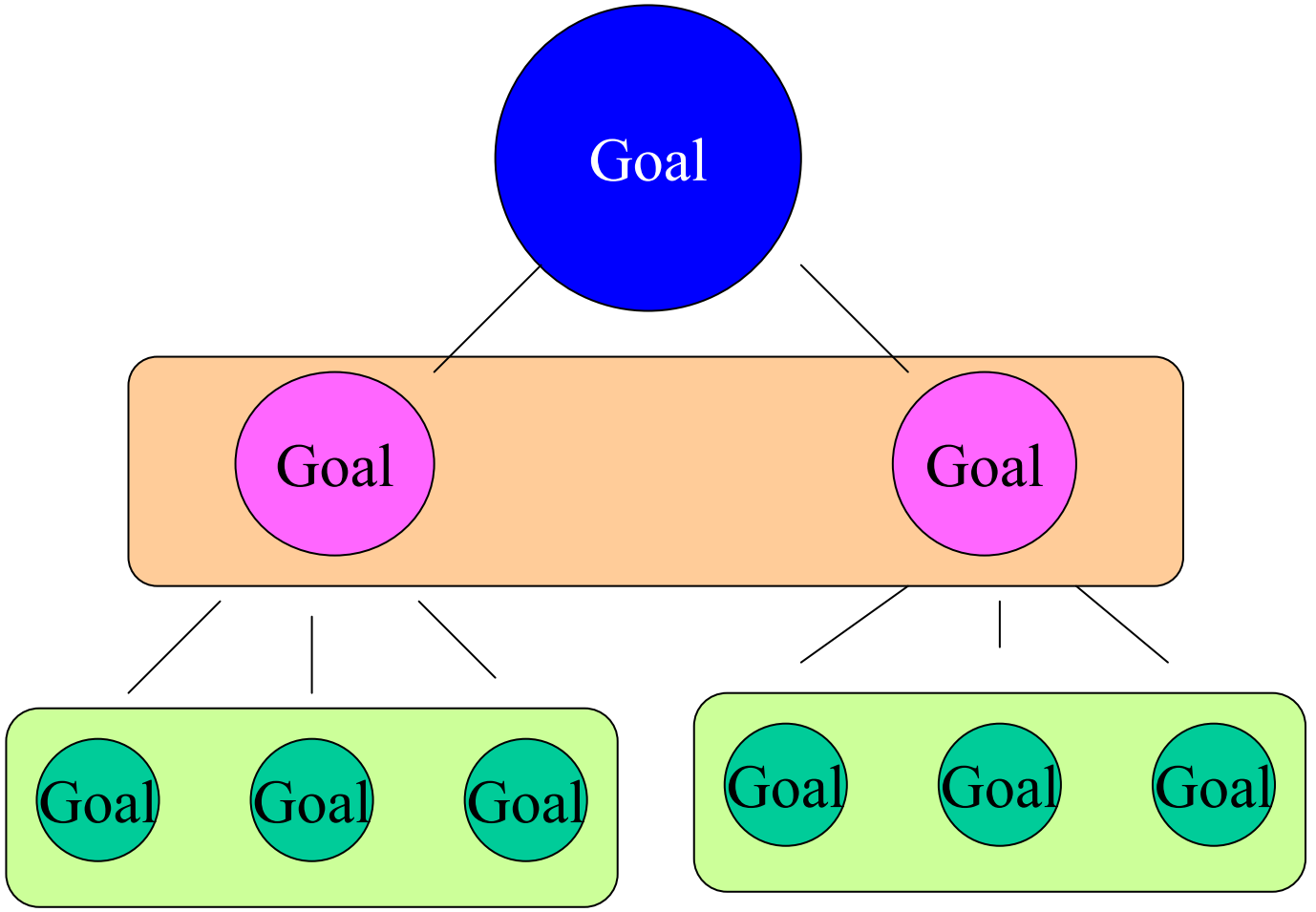


階層型ゴール指向プランニングとは？

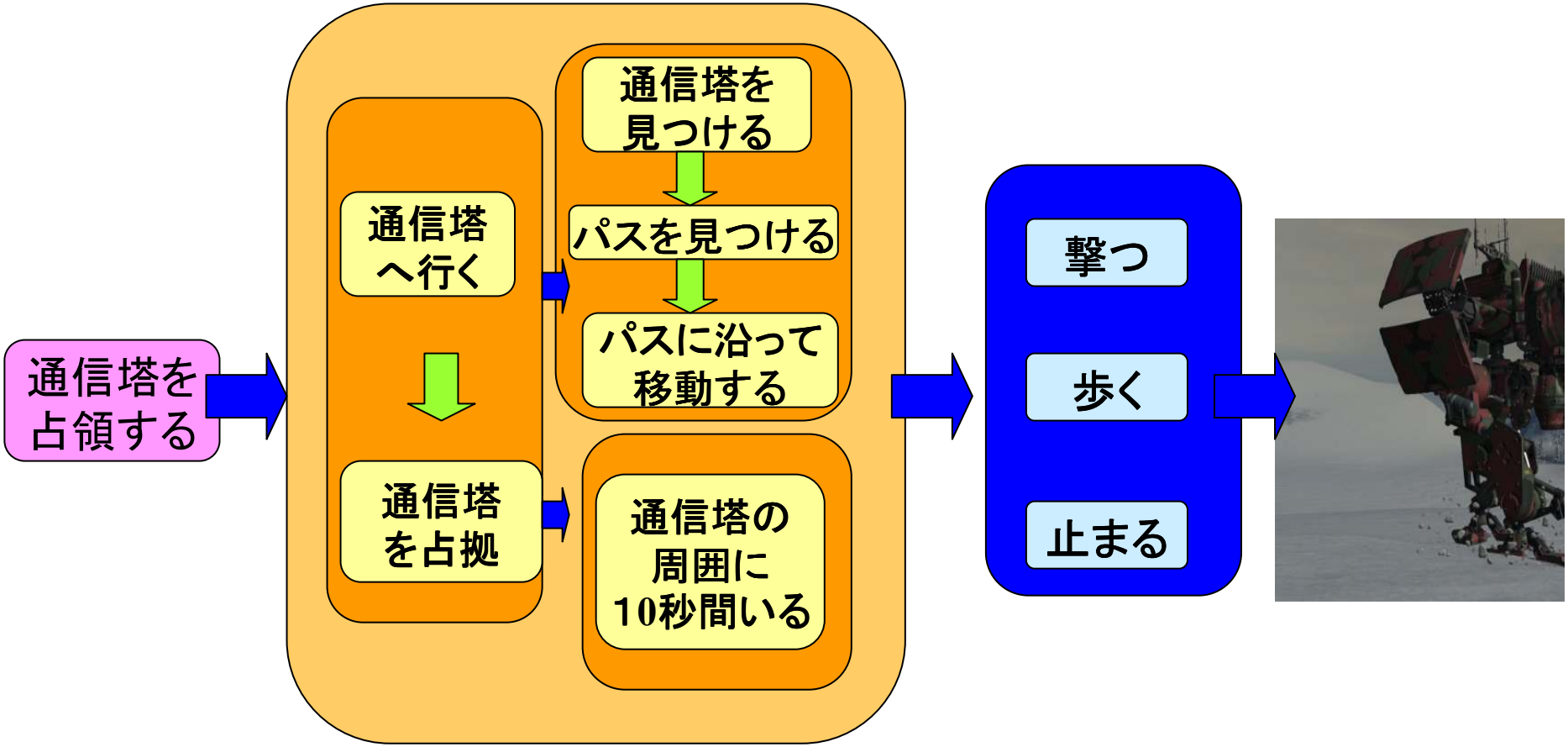
一つのゴールはより小さなゴールから組み立てられる



ゴールはより小さなゴールから組み立てられる



クロムハウンズにおける
ゴール指向型プランニング



戦術、振る舞い

操作

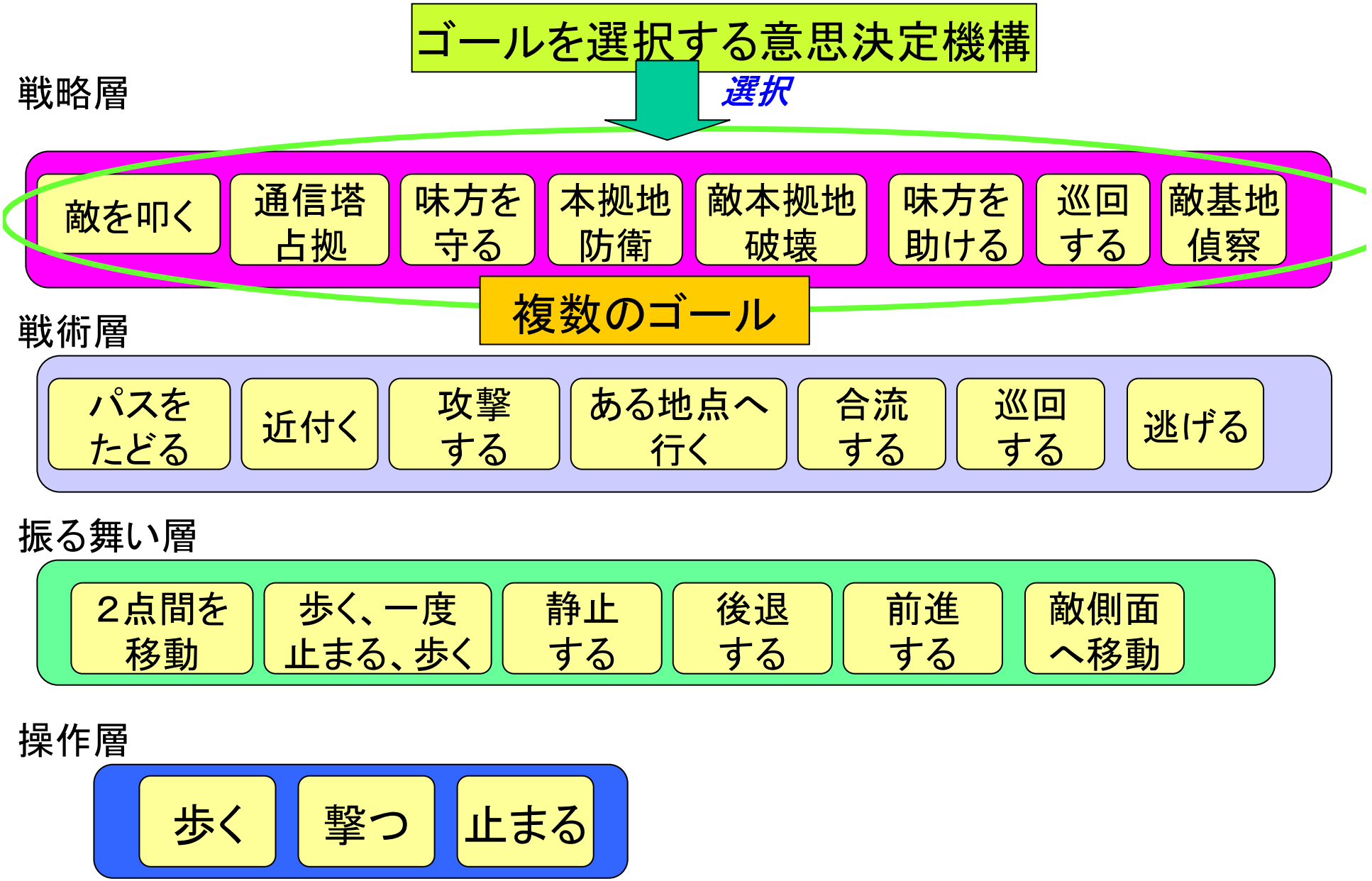
ハウンズ

ゴール指向プランニングによって
通信塔を占拠するデモ

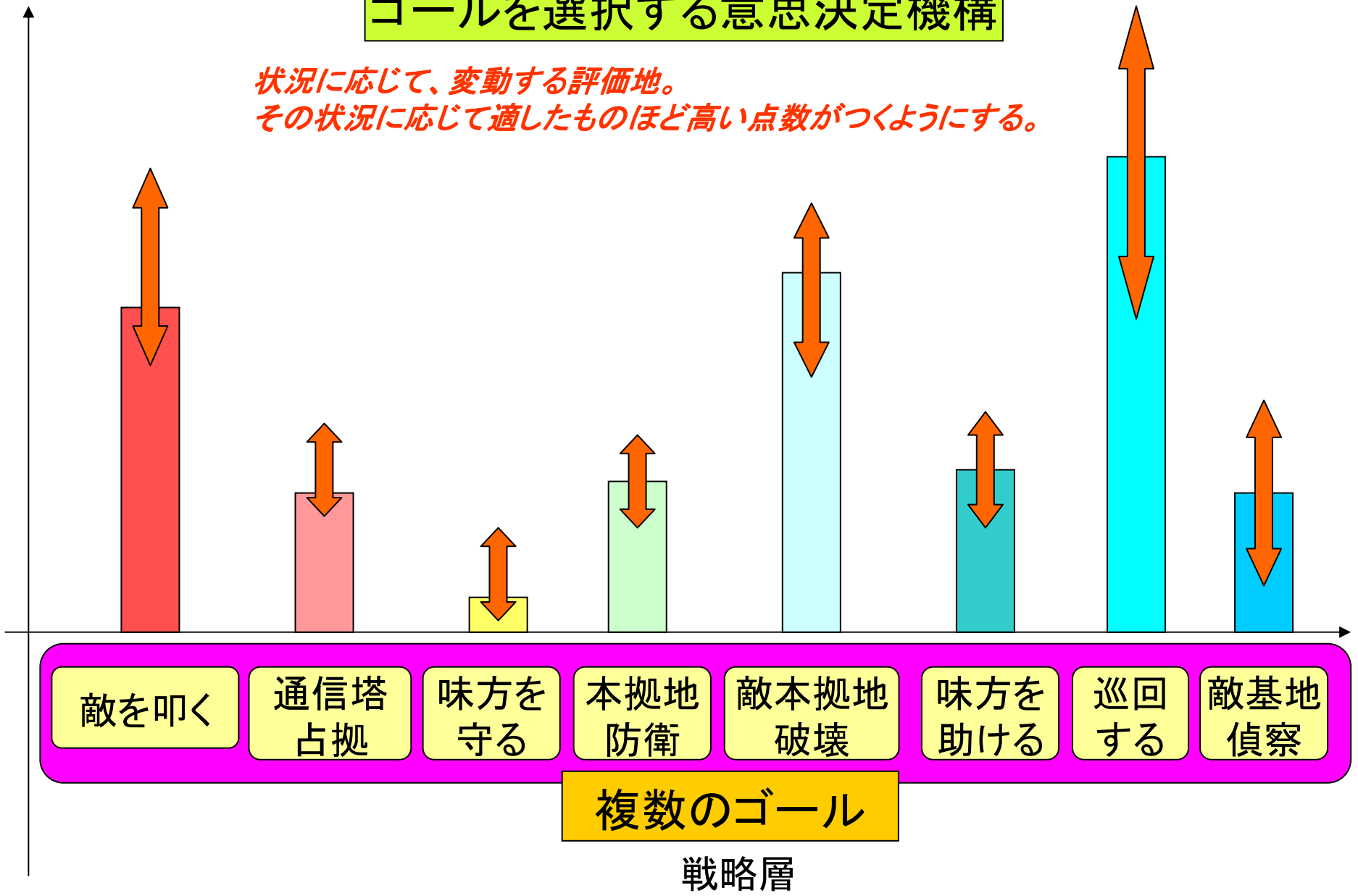
左上は階層型プランニングのゴール表示

クロムハウンズ　ゴール総合図

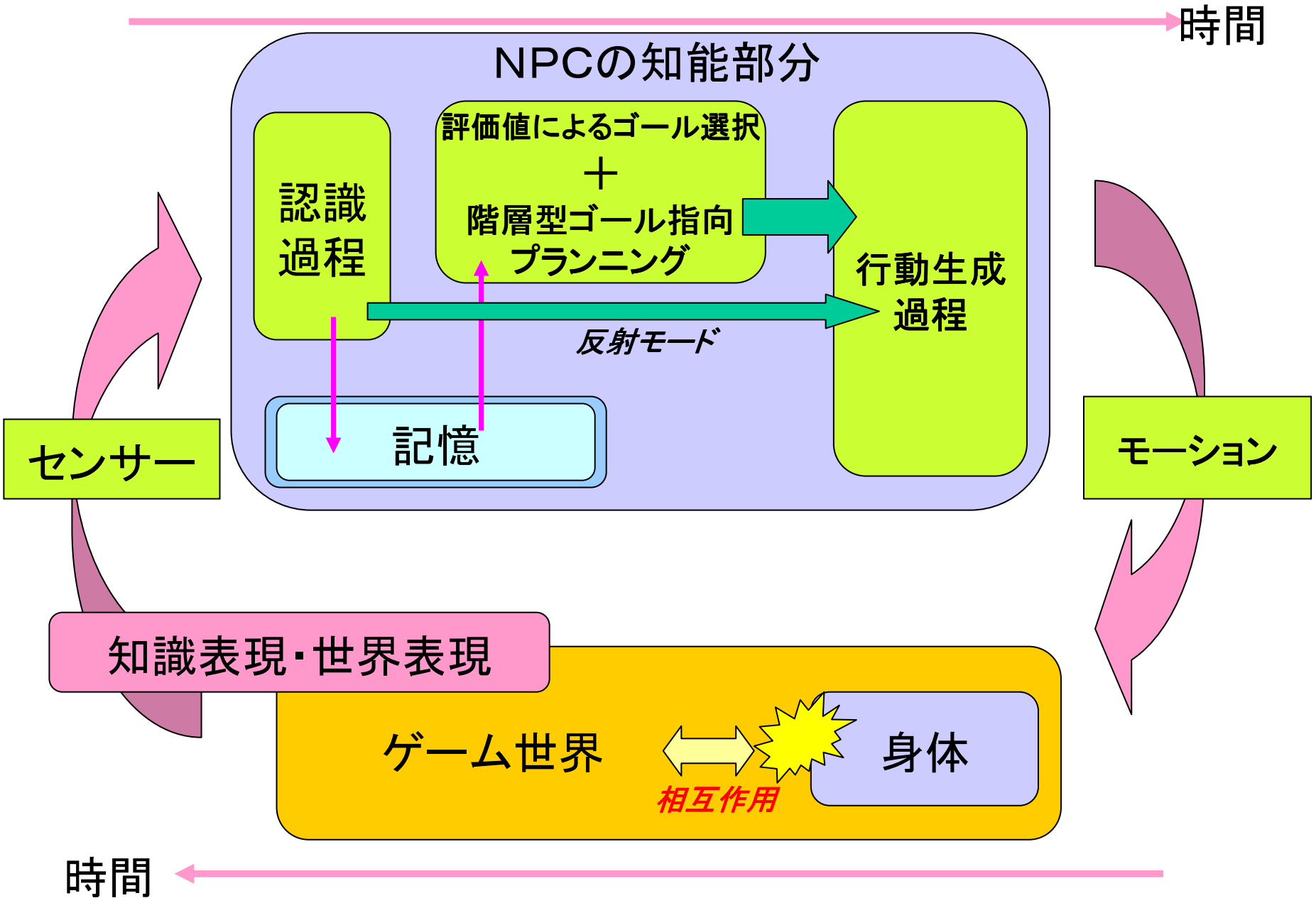
状況に応じて、戦略を選ぶ知能が必要



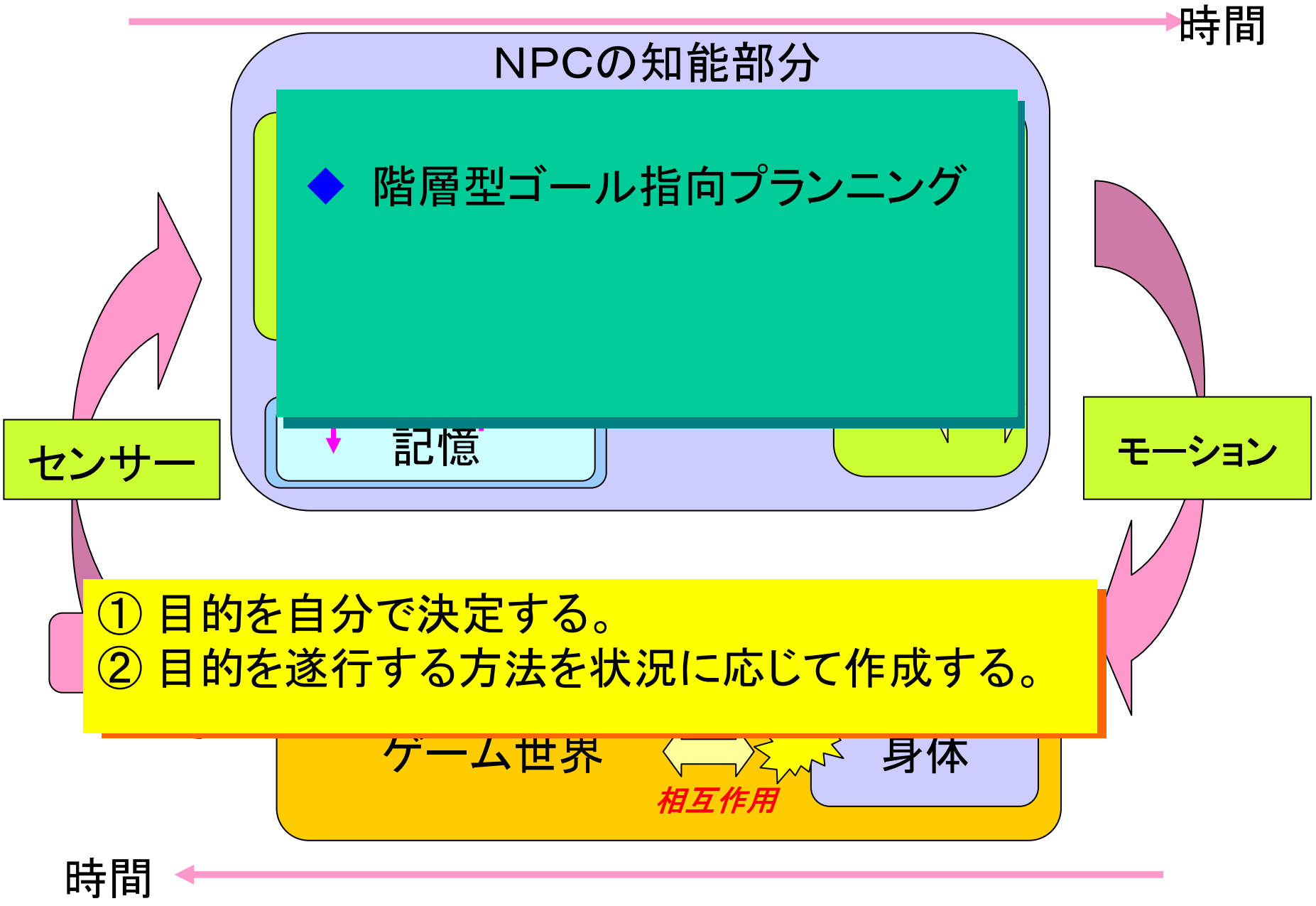
クロムハウンズ　状況により変動する評価値のイメージ



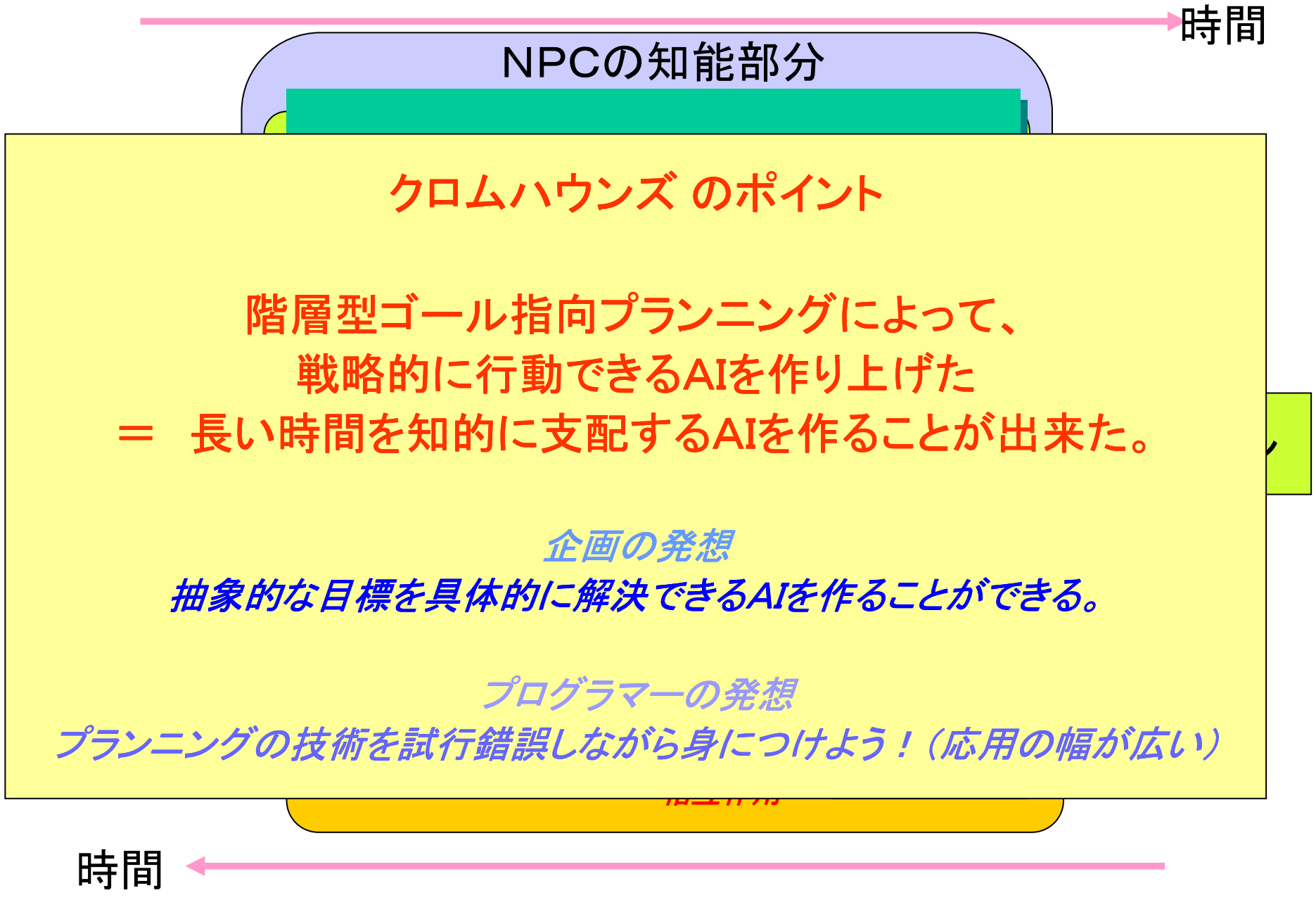
ChromeHounds NPCのAIのアーキテクチャー



ChromeHounds NPCのAIのアーキテクチャー

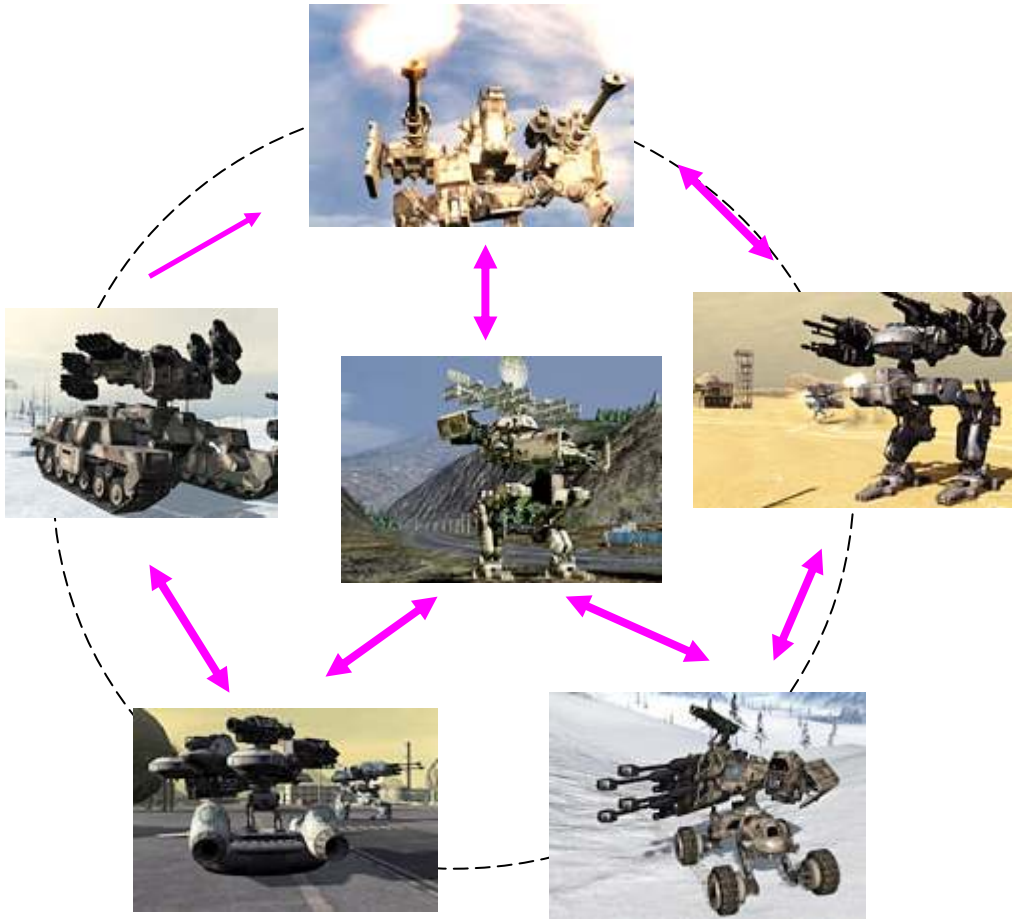


ChromeHounds NPCのAIのアーキテクチャー



マルチエージェントへ

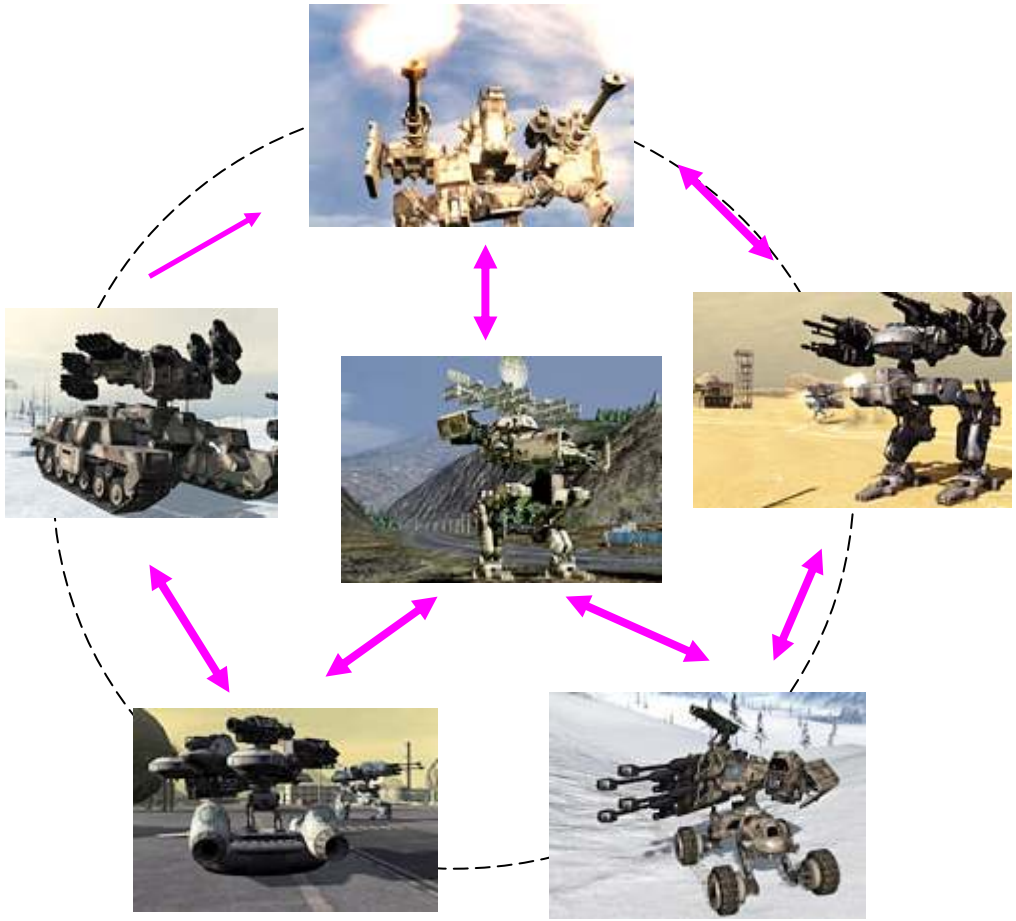
マルチエージェント技術



全体としての
新しい知能

各エージェントがそれぞれの機能を果たすことで、
全体として新しい知能を獲得すること。

集団における知性　クロムハウンズ



メンバーの維持

①護衛　②救援

ゴールによる協調

相手チームに対する
状況的優位を築く

③戦闘判断　④集中砲火

アルゴリズムによる協調

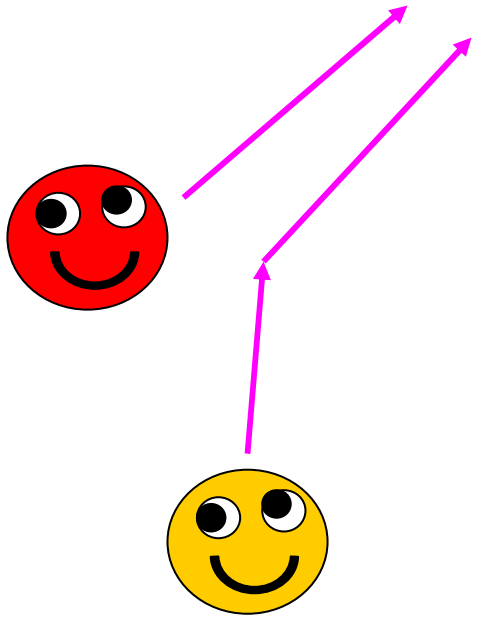
勝利のための
統制された行動

⑤チームAI

チームAIによる協調

クロムハウンズにおけるマルチエージェント技術

① **護衛**　一体のエージェントが他のエージェントと移動を共にする。

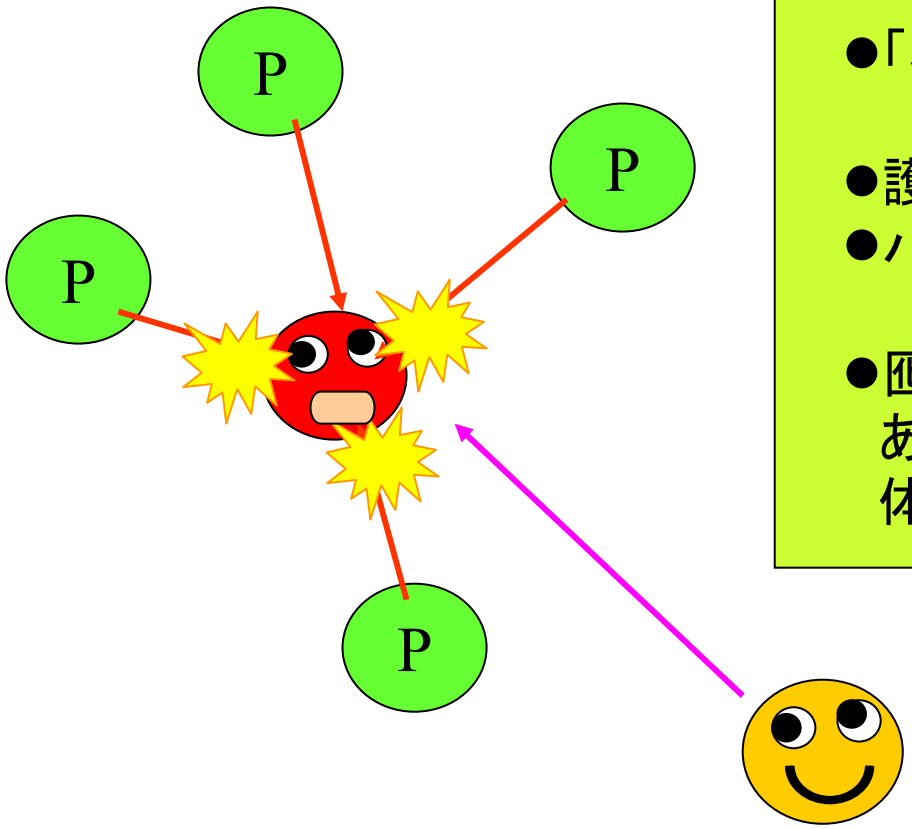


- 「護衛する」というゴールを用意する
- 護衛される対象は戦力が少ないか、
  移動速度が遅いハウンドが選ばれやすい


GOAL_THINK, 寿命 -1.000000
GOAL_PROTECT_TARGET, 寿命 62.939941
GOAL_ATTACK_TARGET, 寿命 2.959961
GOAL_ATTACK_INFIELD, 寿命 9.
GOAL_APPROCH_TARGET, 寿命 4.
GOTO_POSITION, 寿命 -1.000000
FOLLOW_PATH, 寿命 -1.000000
GOAL_APPROCH_TARGET, 寿命 16.754826
GOTO_POSITION, 寿命 -1.000000
FOLLOW_PATH, 寿命 -1.000000
0348
0048
13:02
N
W
E
S

クロムハウンズにおけるマルチエージェント技術

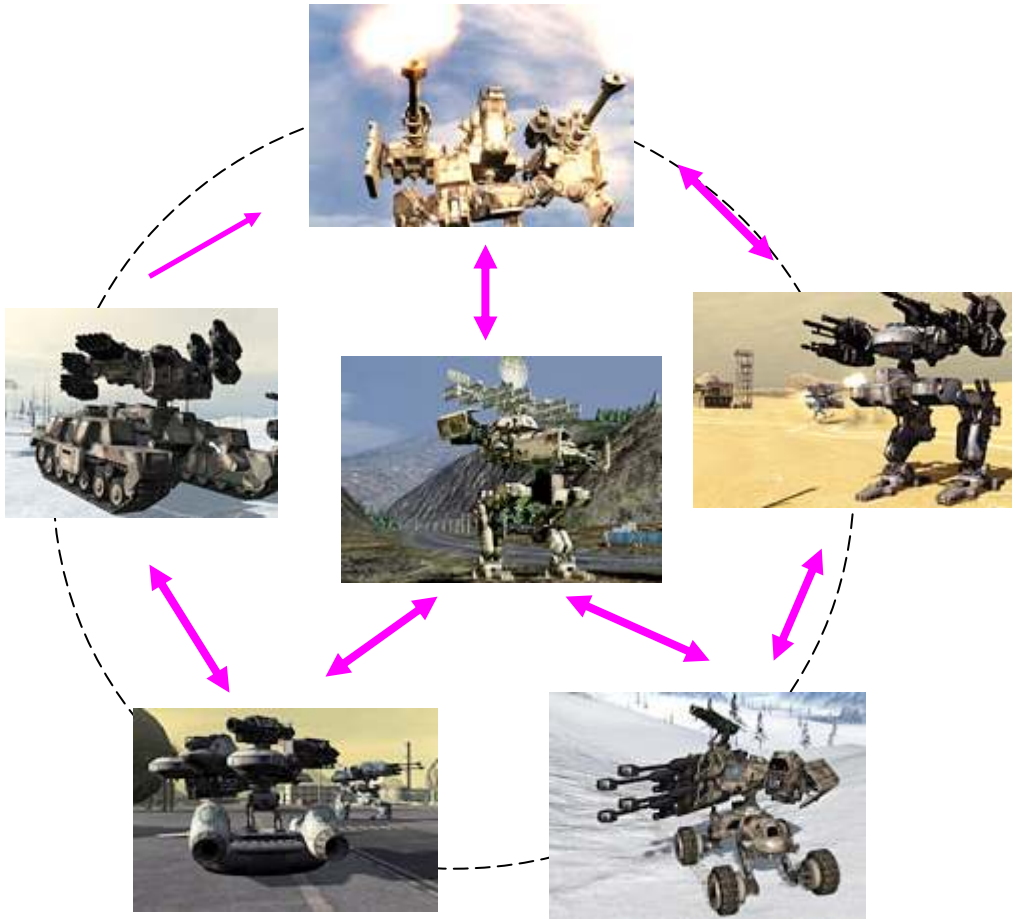
② **救援**　一体のエージェントが窮地にある他のエージェントの戦場に駆けつける。



- 「救援する」というゴールを用意する
- 護衛される対象は体力の残りが少ない
- ハウンズ
- 囮に使われる可能性があるので、あまりに遠かったり、あまりに体力が少ない場合は、救援に行かない

ALL AI SYSTEM

集団における知性　クロムハウンズ



メンバーの維持

①護衛　②救援

ゴールによる協調

相手チームに対する
状況的優位を築く

③戦闘判断　④集中砲火

アルゴリズムによる協調

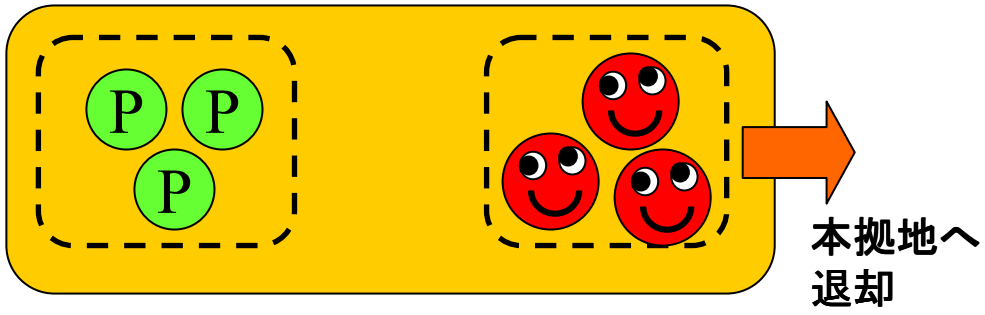
勝利のための
統制された行動

⑤チームAI

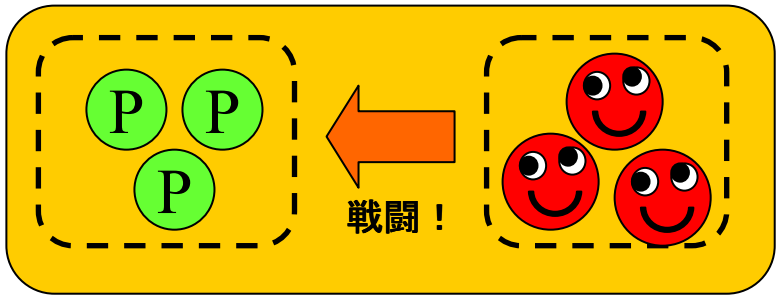
チームAIによる協調

クロムハウンズにおけるマルチエージェント技術

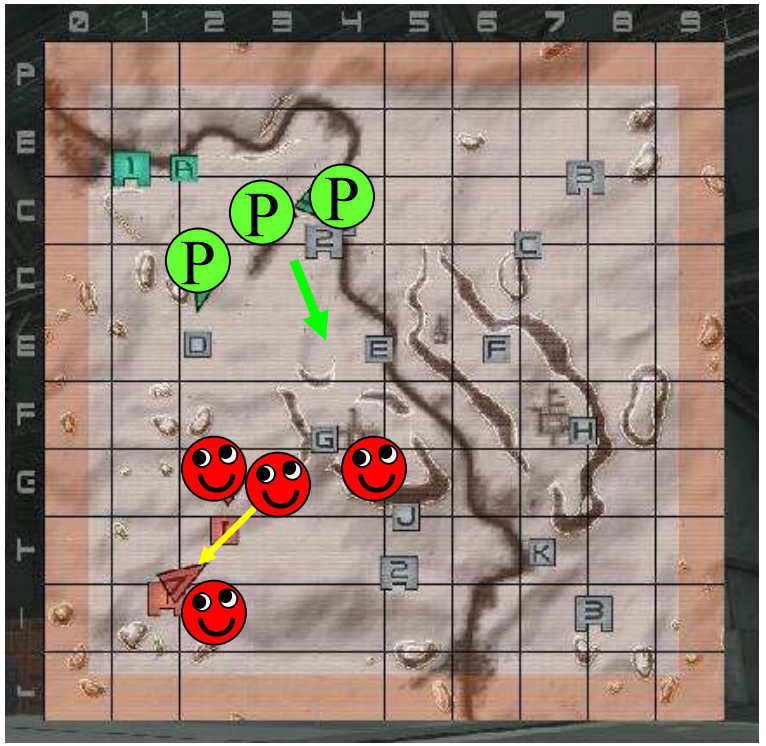
③ **戦闘判断** エージェントが周りの敵と味方の戦力を計算して戦うべきか、逃げるべきかを判断する。 **デバッグの過程で追加**



プレイヤーたちの戦力 ＞ 1．4 x エージェントたちの戦力



プレイヤーたちの戦力 ＜ 1．4 x エージェントたちの戦力



戦力比が大きい無駄な戦闘を回避し、常に相手を上回る戦力を増築してプレイヤーに対抗する

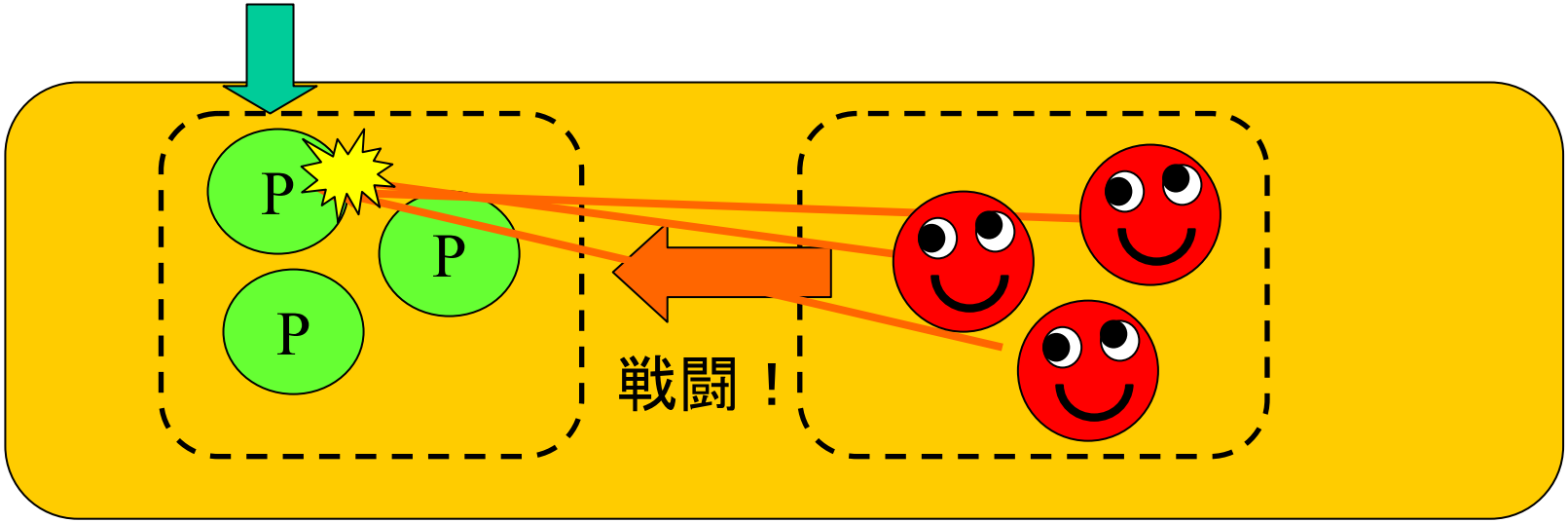
ALL AI SYSTEM　　AIチーム vs AIチーム 戦デモ

クロムハウンズにおけるマルチエージェント技術

④ **集中砲火**　複数のエージェントが複数の敵ターゲットに対し
ターゲットを統一する

デバッグの過程で追加

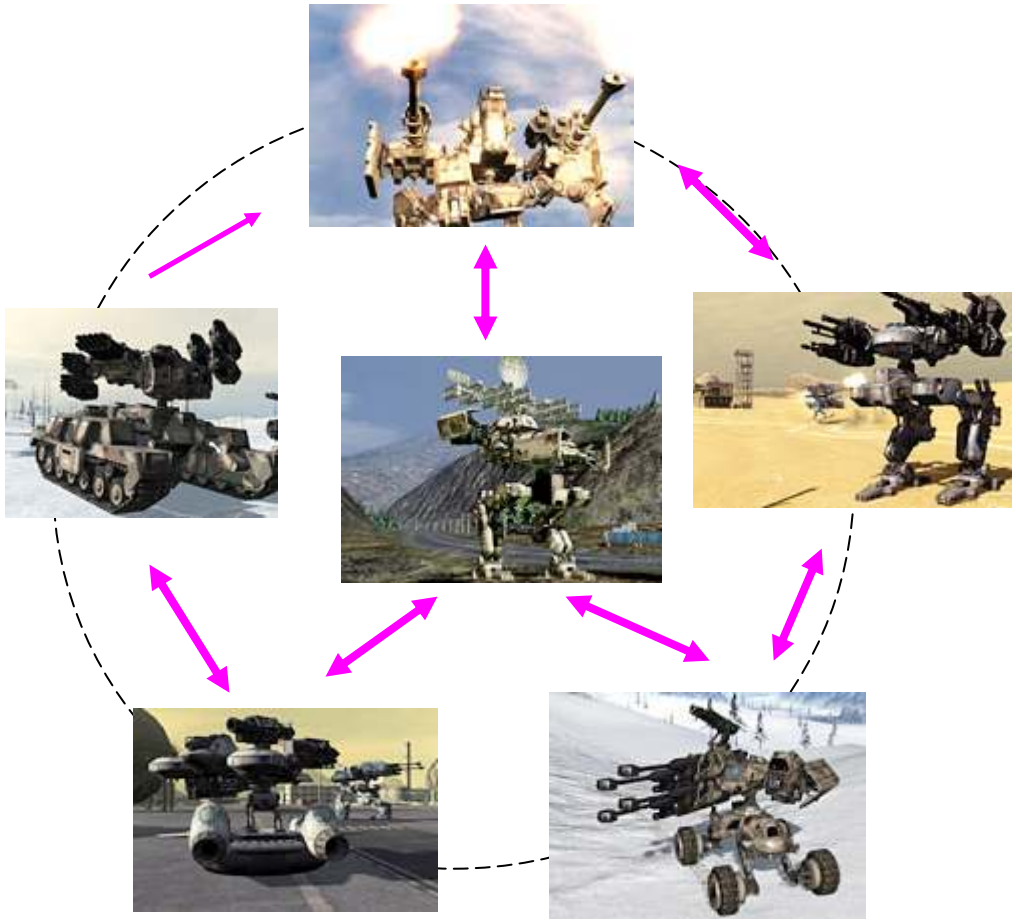
その場で戦力が最も低い敵を集中的に攻撃する



その場で戦力が最も低い敵を集中的に攻撃し
ダメージの分散を防ぐ

ALL AI SYSTEM　　AIチーム vs AIチーム 戦デモ

集団における知性　クロムハウンズ



メンバーの維持

①護衛　②救援

ゴールによる協調

相手チームに対する
状況的優位を築く

③戦闘判断　④集中砲火

アルゴリズムによる協調

勝利のための
統制された行動

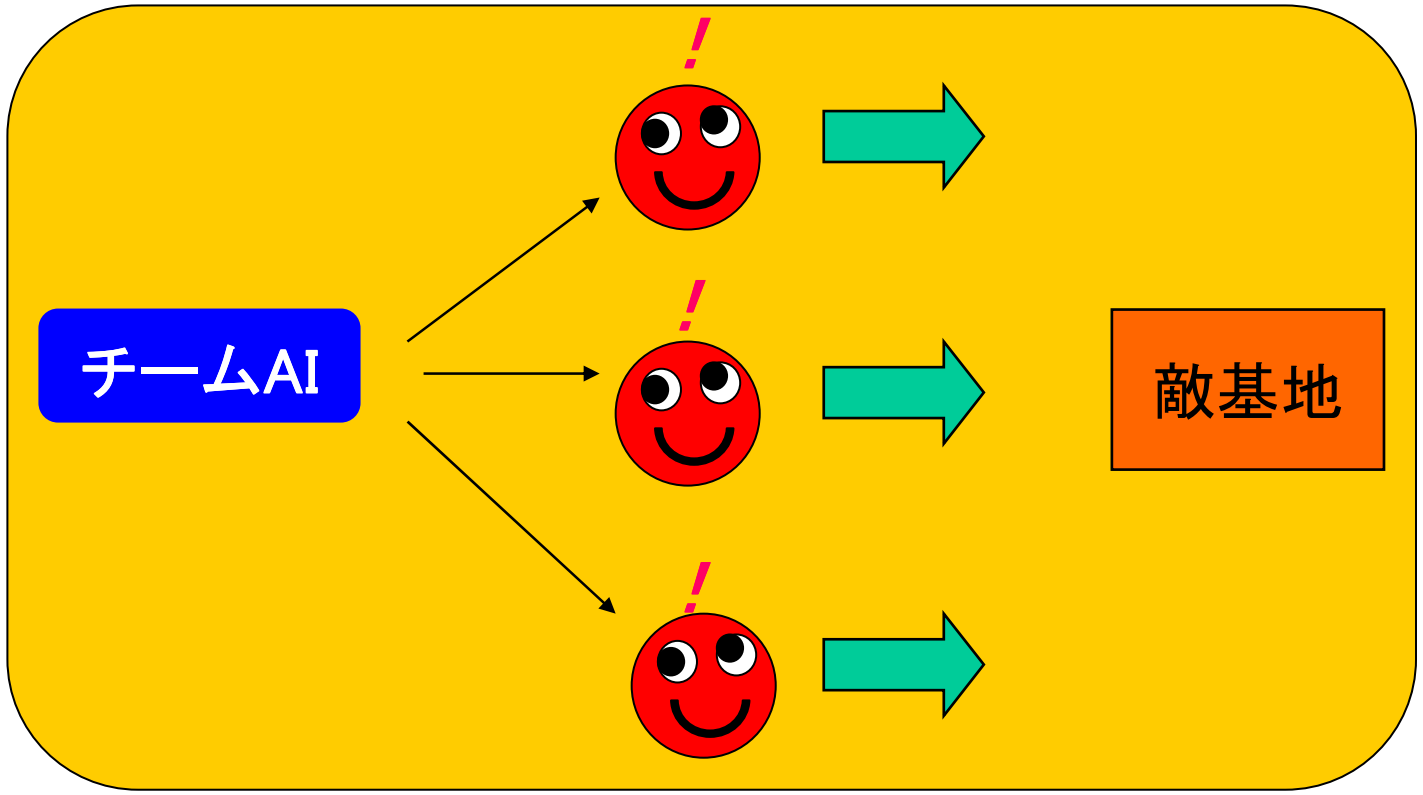
⑤チームAI

チームAIによる協調

クロムハウンズにおけるマルチエージェント技術

⑤ 勝利のための統制された行動

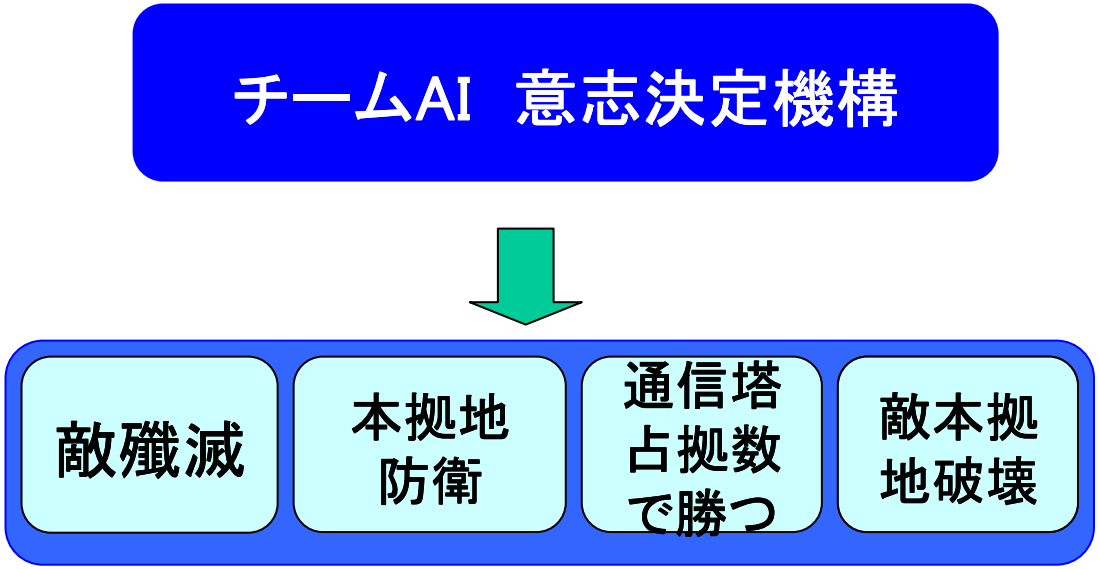
複数のエージェントがゴールを共有する



勝利のために目的を共有する

チームAIの構造

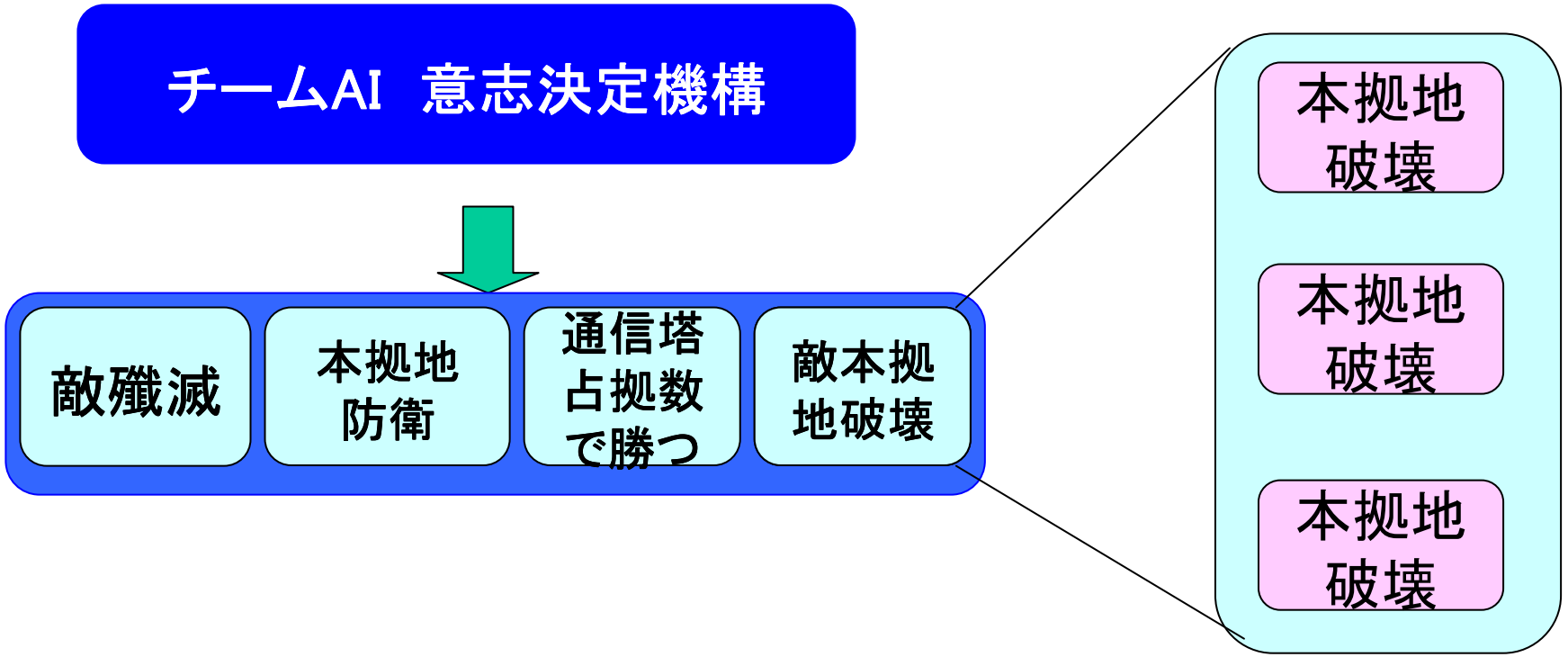
４つの戦略を持ち、ゲーム全体の状況を反映する
評価関数によって、一つの戦略を決定する。
(評価関数による意思決定 = 個体の意志決定と同じ方法)



チームとしての戦略
（＝勝利条件と同じ）

チームAIの構造

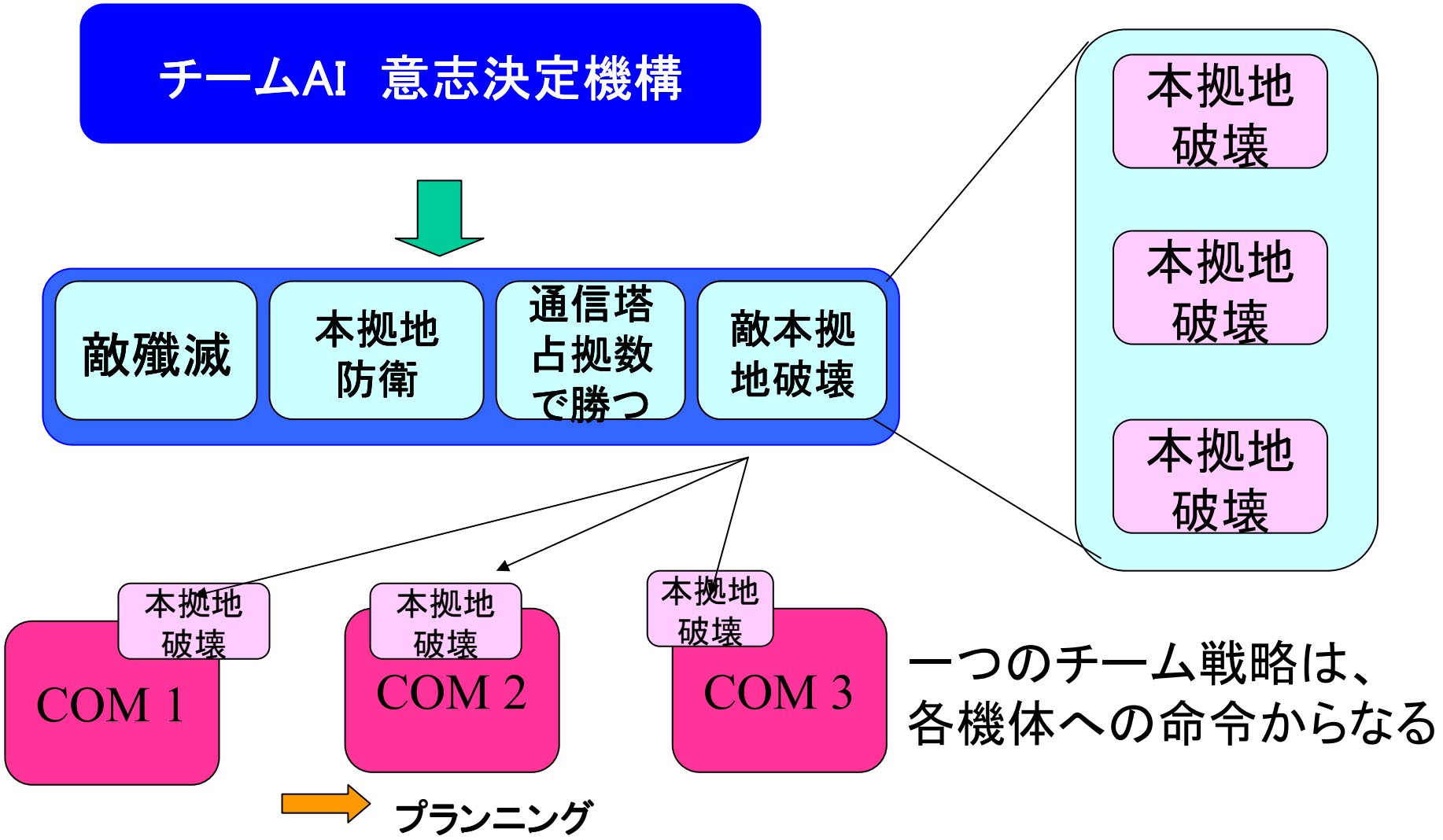
４つの戦略を持ち、評価関数によって、一つの戦略を決定する。
(評価関数による意思決定 = 個体の意志決定と同じ方法)



一つのチーム戦略は、
各機体への命令からなる

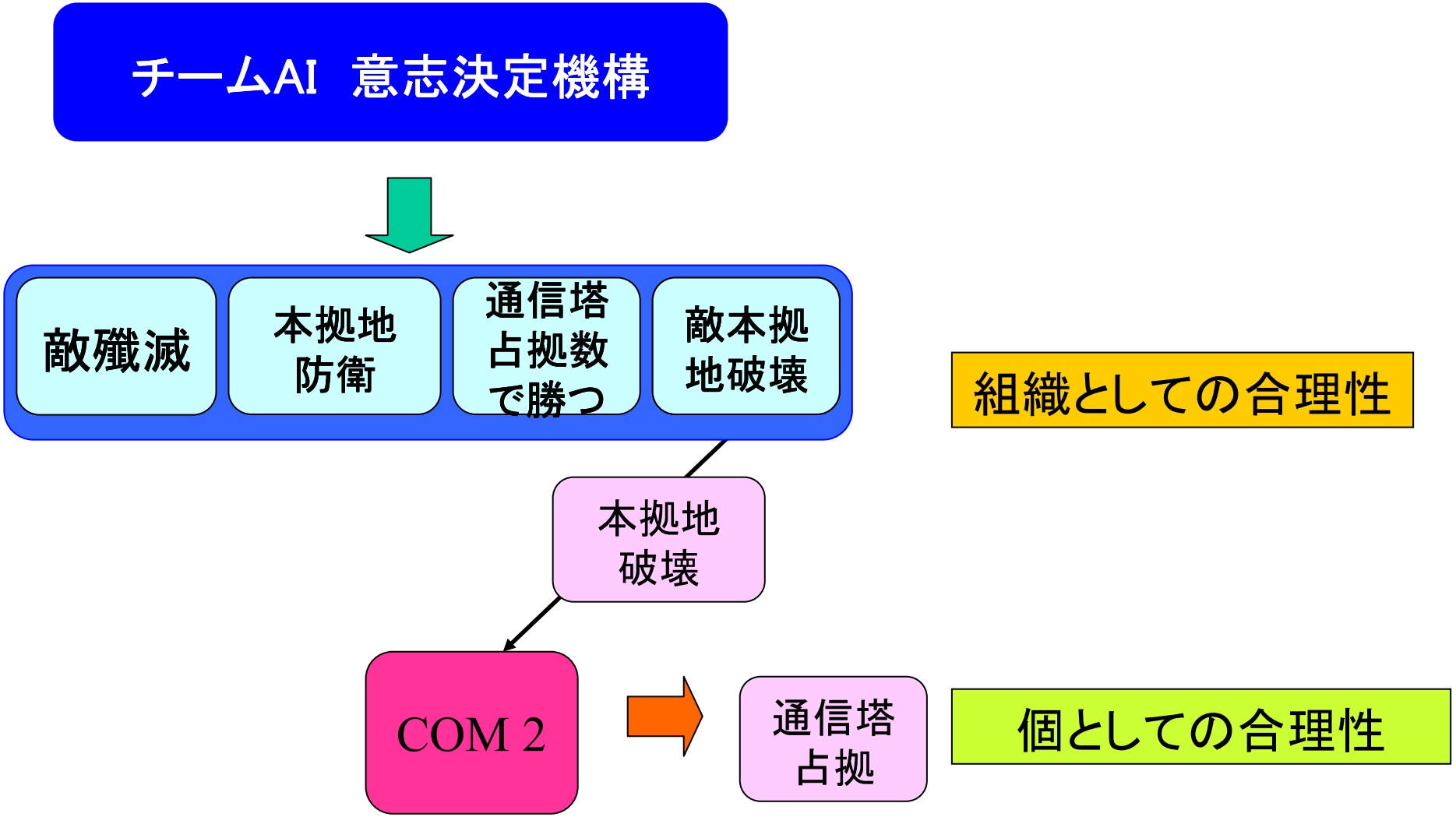
チームAIの構造　＝　ゴール指向型の拡張

COMのゴール指向プランニングの上に、チームAIを積み上げる

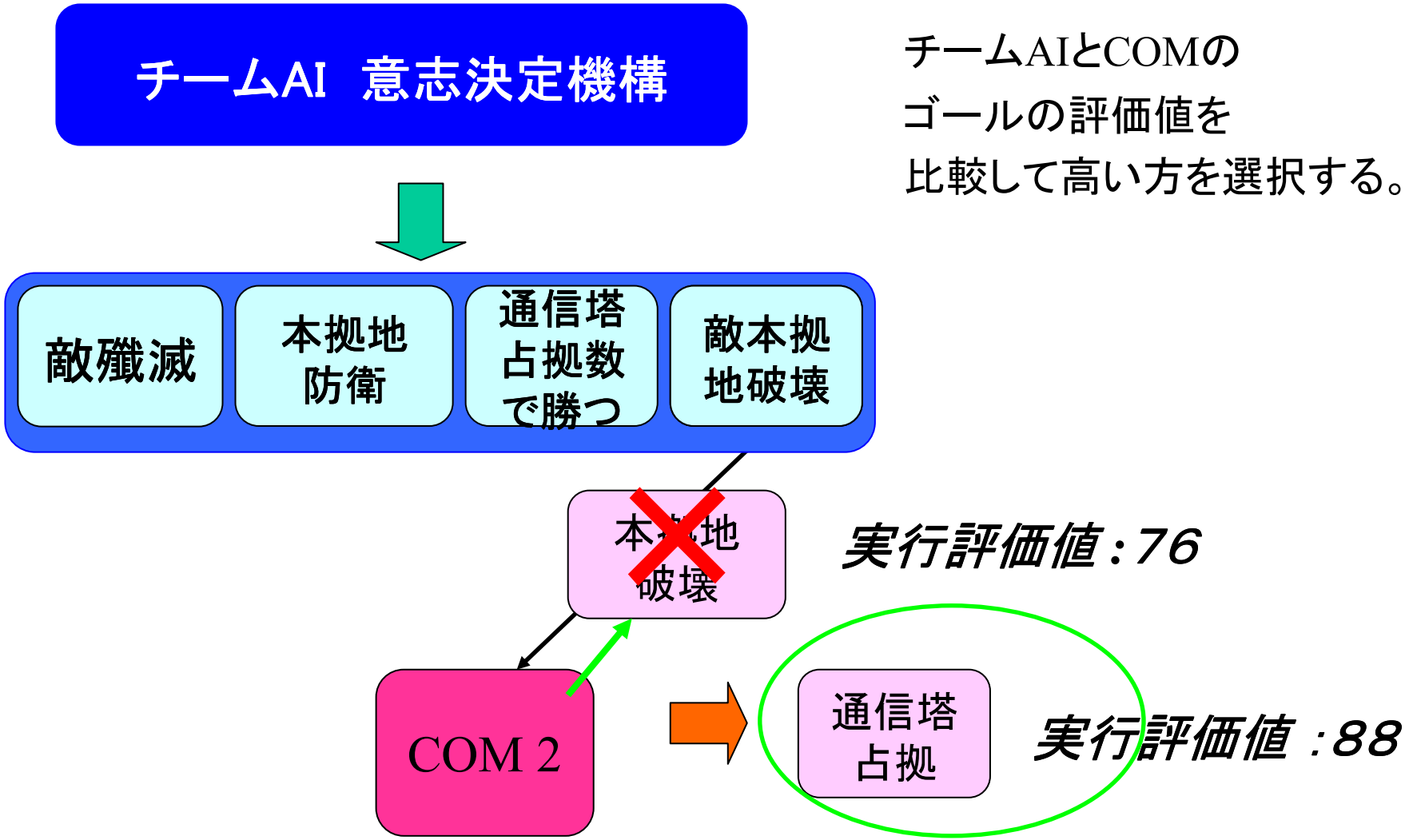


一つのチーム戦略は、
各機体への命令からなる

チームAI の意思決定と COMの判断を比較して、最終的に決定する

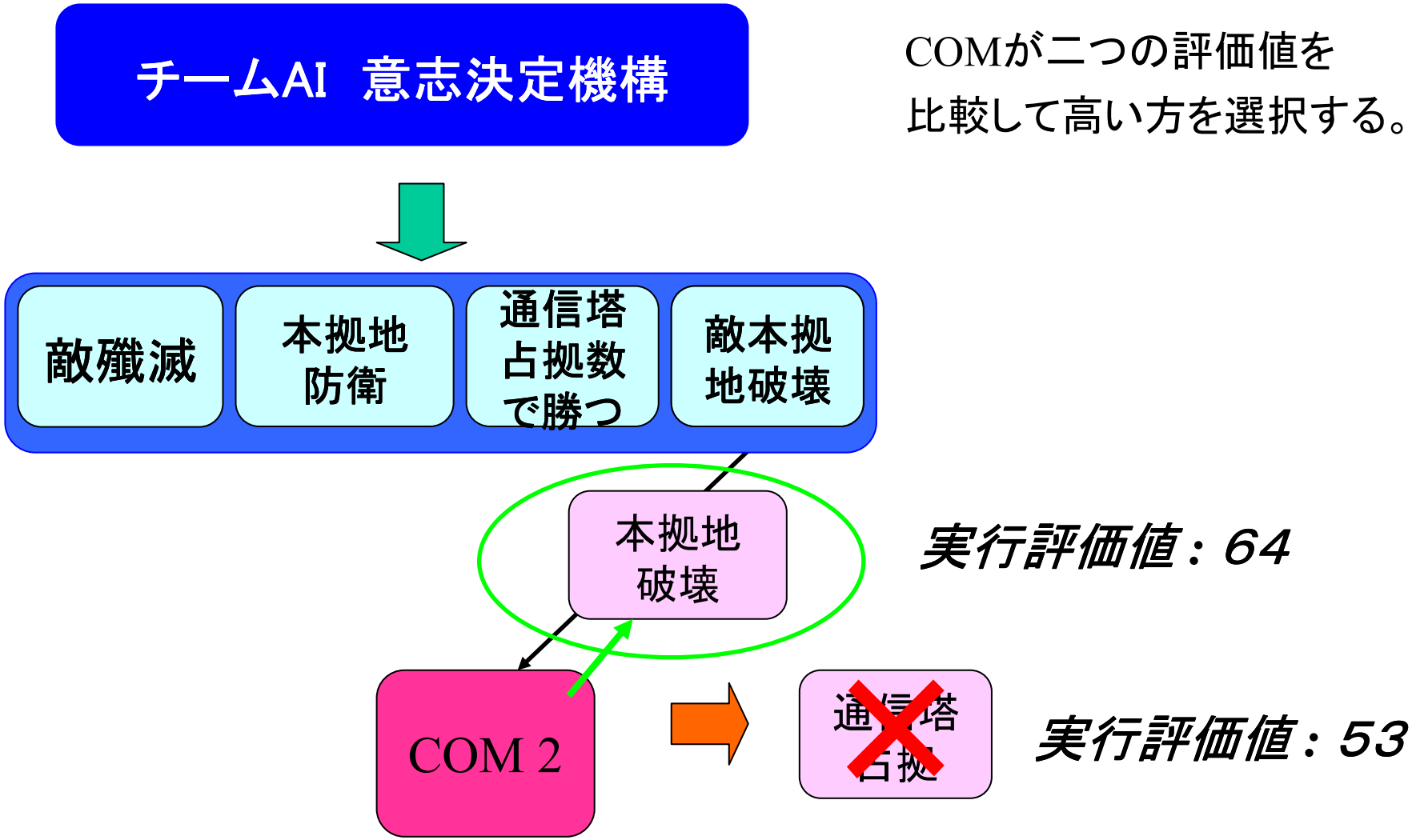


チームAI の意思決定と COMの判断を比較して、最終的に決定する



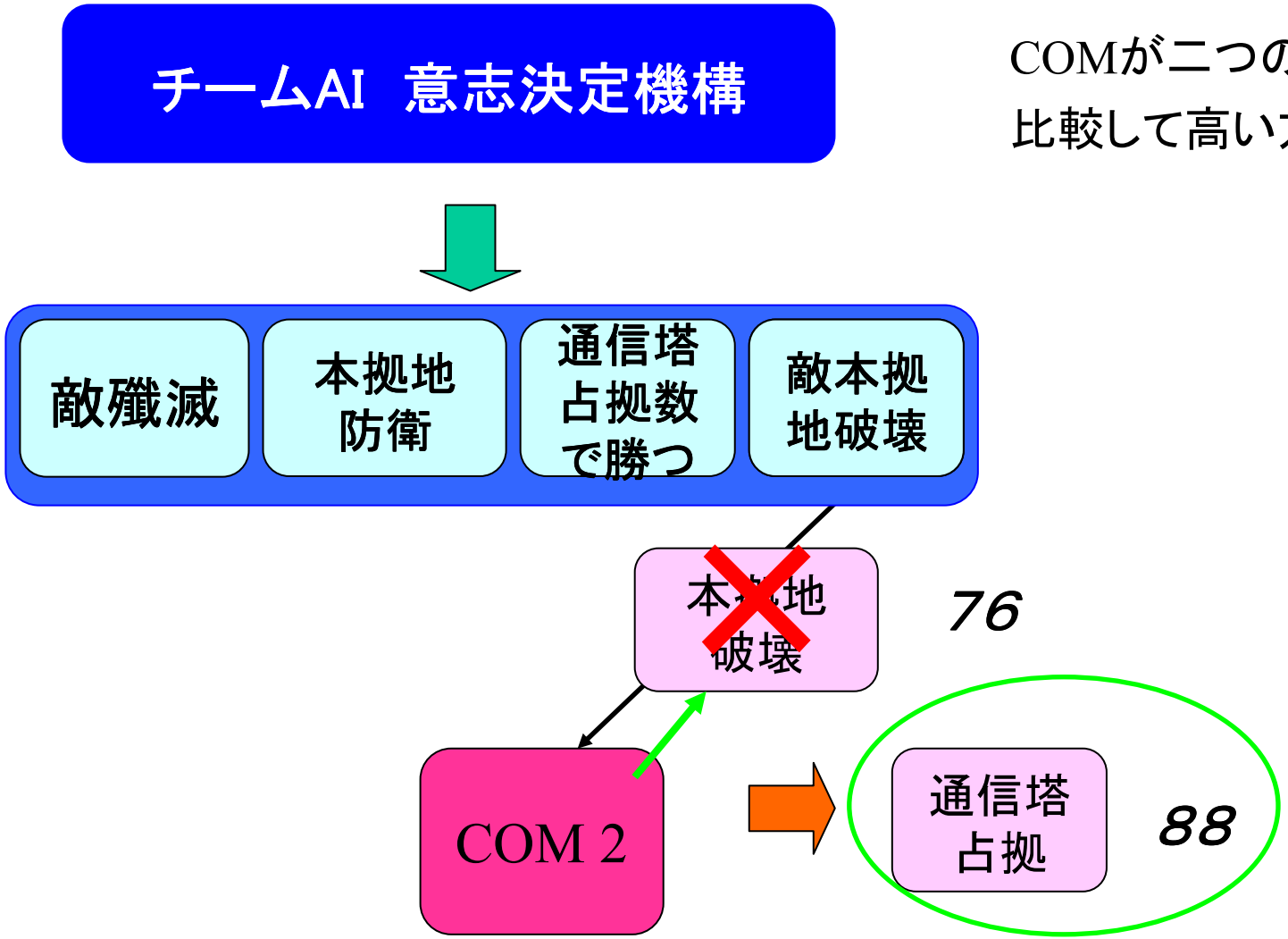
チームAIとCOMの
ゴールの評価値を
比較して高い方を選択する。

チームAI の意思決定と COMの判断を比較して、最終的に決定する



COMが二つの評価値を
比較して高い方を選択する。

チームAI の意思決定と COMの判断を比較して、最終的に決定する

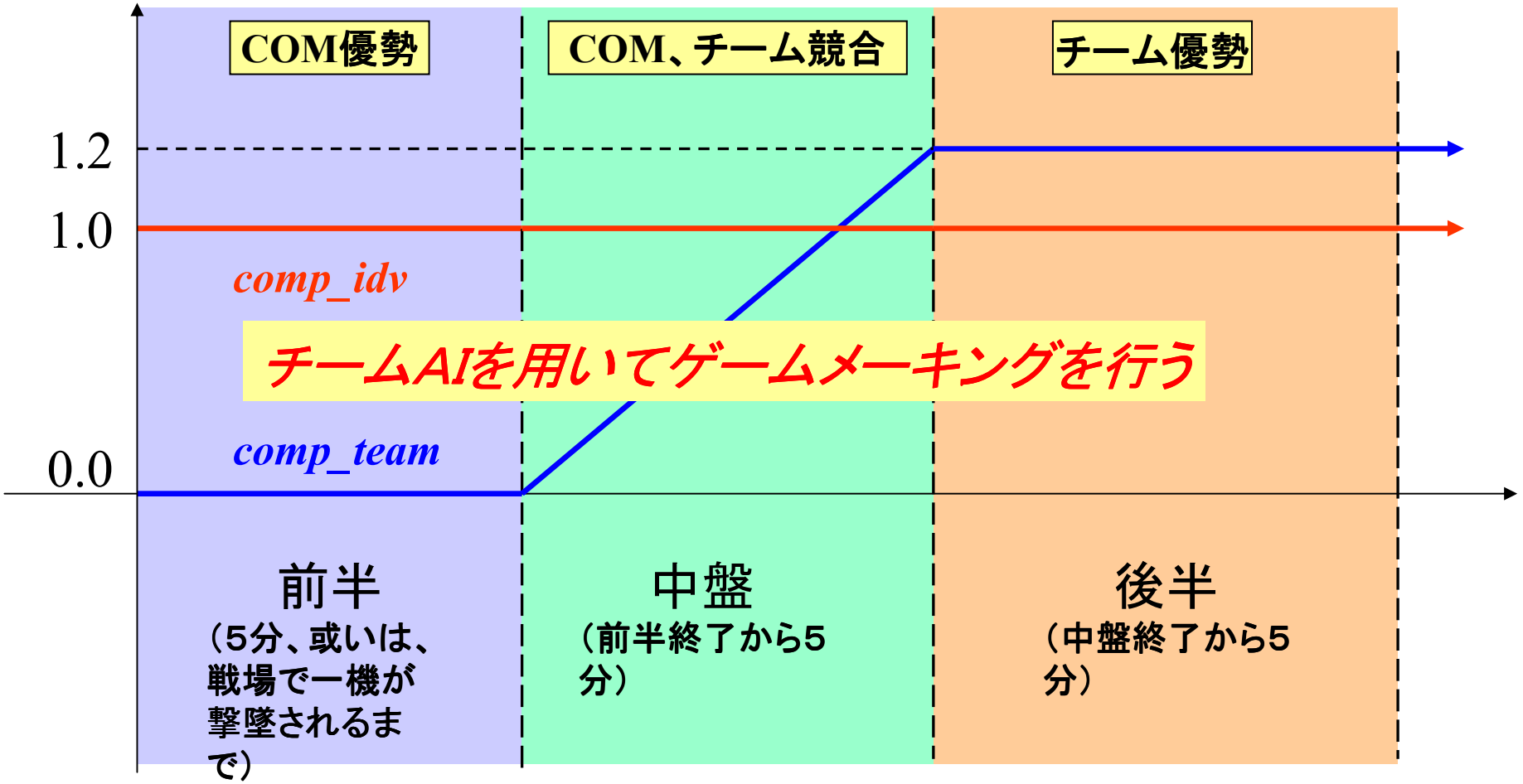


COMが二つの評価値を
比較して高い方を選択する。

チームAIの介入の仕方

前半はチームAIより個としてのAIの判断を優先、
後半はチームAIの判断を優勢にしたい。

比較のためのチームAIゴール評価値 ＝ ゴール評価値 x comp_team
比較のためのCOM AIゴール評価値 ＝ ゴール評価値 x comp_idv



チームAIのデモをご覧ください

ALL AI SYSTEM

第5章　エンターテインメントAIへ

ゲームAIの目標

①最初の目標　　時間と空間を支配できるAI

（技術的目標）

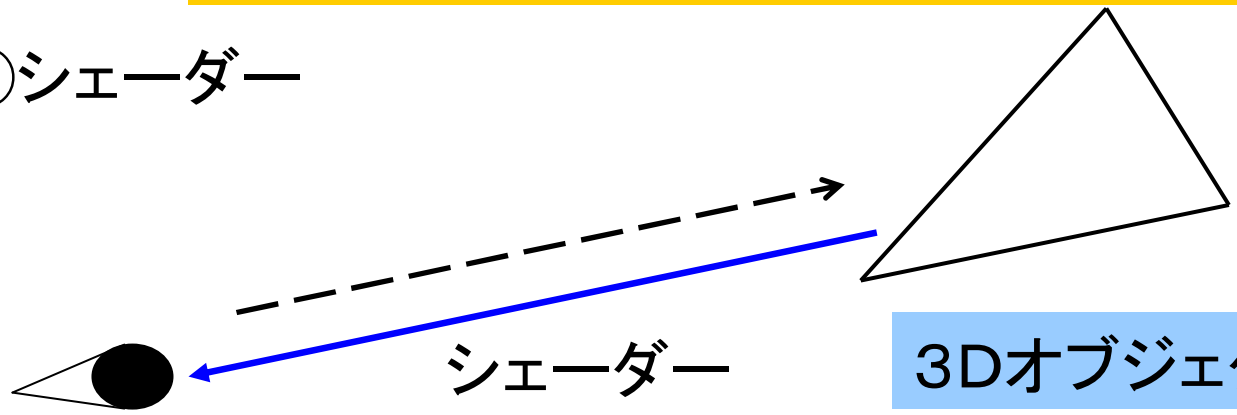
②最後の目標　　ユーザーの心理をコントロールできるAI

（エンターテインメントAIとしての目標）

見せ方

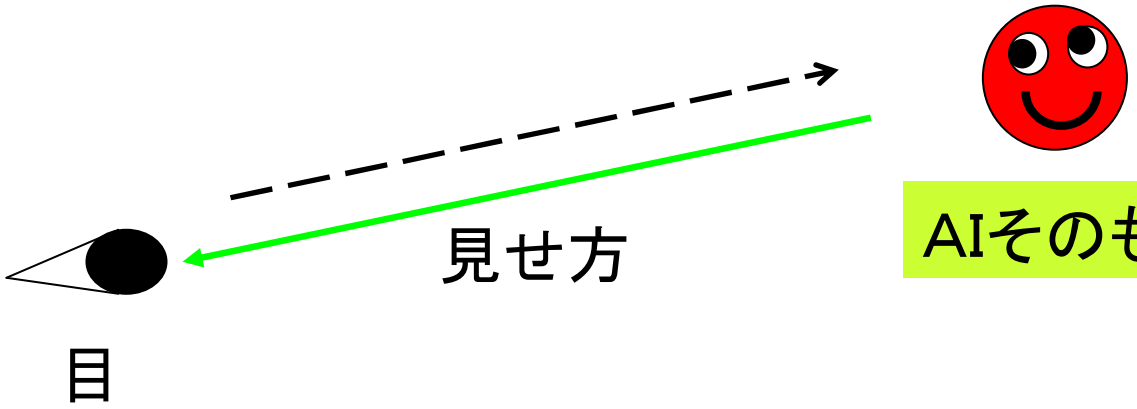
AIもまたAIとして完成しているのみならず、
ユーザーにどう映るか、までのプロセスが大切

①シェーダー



3Dオブジェクトそのもの

②エンターテインメントとしてのAI



AIそのもの

Halo1~3 を通して、AIを使って
エンターテインメントを作る、
という姿勢を見てみよう

Halo のAI

第１節　コンセプト
第２節　思考
第３節　移動
第４節　演出

Halo とは？

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

Xbox, 全米、世界を代表するFPSの一つ
(Halo 500万本　Halo2 700万　国内10万本未満）

Halo デモ

Halo AI　のコンセプト

第１節　コンセプト

第２節　思考

第３節　移動

第４節　演出

Halo AI

敵（コグナント）

グラント

ちょこまかと
動き回る。
愛嬌がある。

ジャッカル

手堅い。

エリート

大型。

味方

人間

普通の人間。

デザインと技術要件

ゲームデザイナー

デザイン

- スコープ時間　3分
- 種族による個性化
- 戦略的な動き

プログラマー

コード

- スコープ時間　30秒
- 知的な判断
- 即座の反応

Halo AI の目指したもの

① 知性

個体

- *プレイヤーがするようなことはできるようにする*
- *意図を明確にする*
- *種族としての個性を出す*

グループ

- *戦略的な目的を明らかにする*
- *種族の役割をはっきりと出す*

Halo AIの目指したもの

② インタラクティブ

印象的なAIを作る

- *プレイヤーに反応*
- *驚き、怒り, 恐れなどの感情*

馬鹿っぽさ

- *限定された知識*
- *予測される行動*

意外な行動

- *臨界点（あるところから突発的な行動）*
- *恐怖のあまり逃走, 凶暴化, 退却,*

Halo AIの目指したもの

③ **予測不可能**

予測不可能とは、ランダムのことでなく、くり返しがないこと

反射的AI

- *ゲームの中でプレイヤーは唯一の予測不可能な要素*
- *予測不可能な状況を作り出す*
- *AIがそれに反応すれば予測不可能なユニークな状況を作り出す*

アナログ情報による反応

- *位置*
- *タイミング*

スクリプトによる制御は条件文がデジタルな条件なので、
予測されやすいが、アナログ情報を参照することで、
わずかな差で条件が変わるので、予測されにくい。

技術的制約

- 20 – 25 Actors
- 2 – 4 Vehicles
- About 15% of Xbox CPU
  （pentium III 733 MHz ）

この技術的制約の上で

① 知性

② インタラクティブ

③ 予測不可能

を実現する。

①知的なAIを作る

プレイヤーに何を考えているかわかるように行動する

1) プレイヤーからわからないAIの状態はなくす

2) プレイヤーにAIの考えている情報を伝える

a) 言葉

エイリアンは喋れないけど「オウオウウオウ」みたいに反応する

b) 意図のわかる行動

「こそこそする」「走る」「隠れる」など明確な意図を持つ行動

c) アニメーション

「飛び込む」「転がる」など特殊な行動

3) AIが注意を払っている対象を、体の向き、視線、照準からわかるようにする

4) プレイヤーに対する対話・アニメーション

②インタラクティブなAIを作る

個体ごとにそれぞれ記憶を形成する

1) AIは、チートはせず自分の視覚、聴覚、触覚から情報を取得する。
（友人の位置は無条件にわかる）

2) 記憶対象の限定

完全なモデル：ゲーム内の全敵、仲間に対する全情報を保持

↓

情報が大きくなりすぎて断念

↓

そのキャラクターにとって重要なキャラについて
味方ならば3、敵ならば5つ程度の情報を保持する

オブジェクトについては、周囲や特に重要なものに対する情報を保持

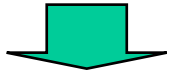

d0 01

③　予測不可能なAIを作る

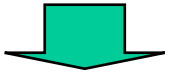
面白い行動が自然に創発されるような仕組みを作る

1) 「怒り、防御、恐れ、驚き」をファジー変数として制御していたが断念
複雑でユーザーから思考が見えない

「発見」「死」「ダメージ」「発砲」などイベントに反射して行動する仕組み



キャラクターたちが状況を認識しゲーム世界で生きているかのように動く



うまく行っているので、チーム制御はなし

Halo AI の思考の仕組み

第１節　コンセプト

第２節　思考

第３節　移動

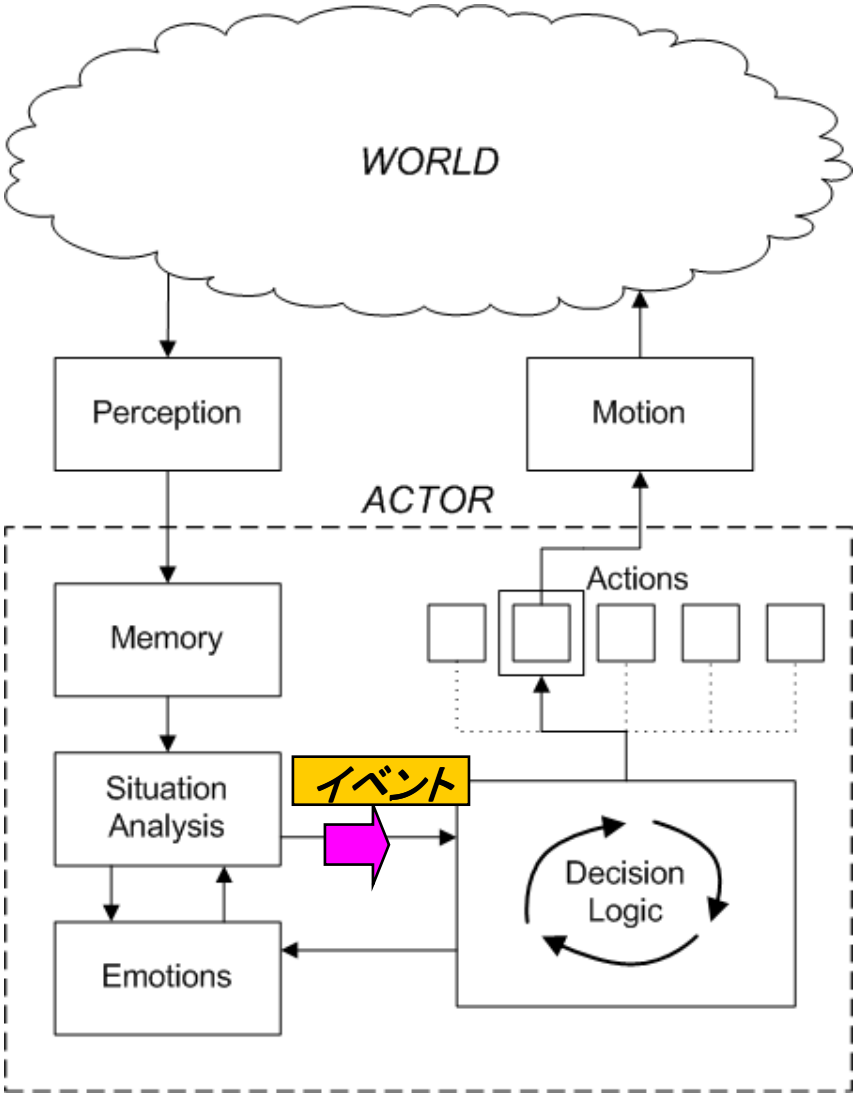
第４節　演出

Halo AI のアーキテクチャー

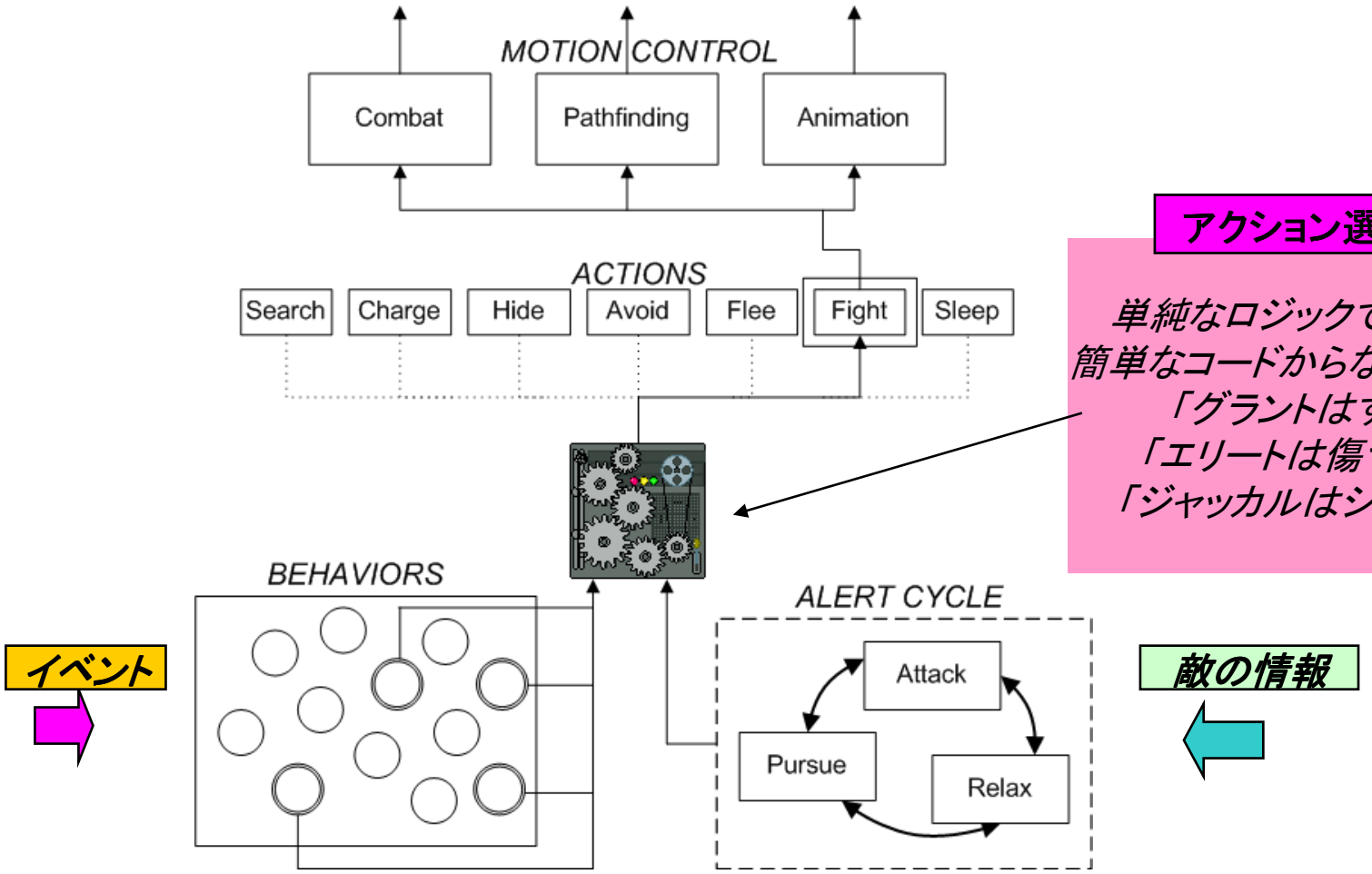
ゲーム世界とAIの間の情報の流れ
はこの層で完全にコントロールされる

取得した情報（生の情報）を解析して
イベントとして意志決定部分に渡す
（この解析は時間を要する。）

意志決定部分はアクションを選択する



Halo AI の意志決定部分



アクション選択ロジック

単純なロジックで1ページ未満の簡単なコードからなる。傾向としては、
「グラントはすぐ逃げる。」
「エリートは傷つくと隠れる。」
「ジャッカルはシールドを持つ。」

チャージ, 退却, 隠れる場所探す
グレネードを投げる、車に入る、
死体を確認

Halo AI　キャラクターの移動

第１節　コンセプト

第２節　思考

第３節　移動

第４節　演出

Halo AI　全体と個体

全体のデザイン

全体の戦闘の流れを
デザインする

この節で説明Halo AI　全体と個体

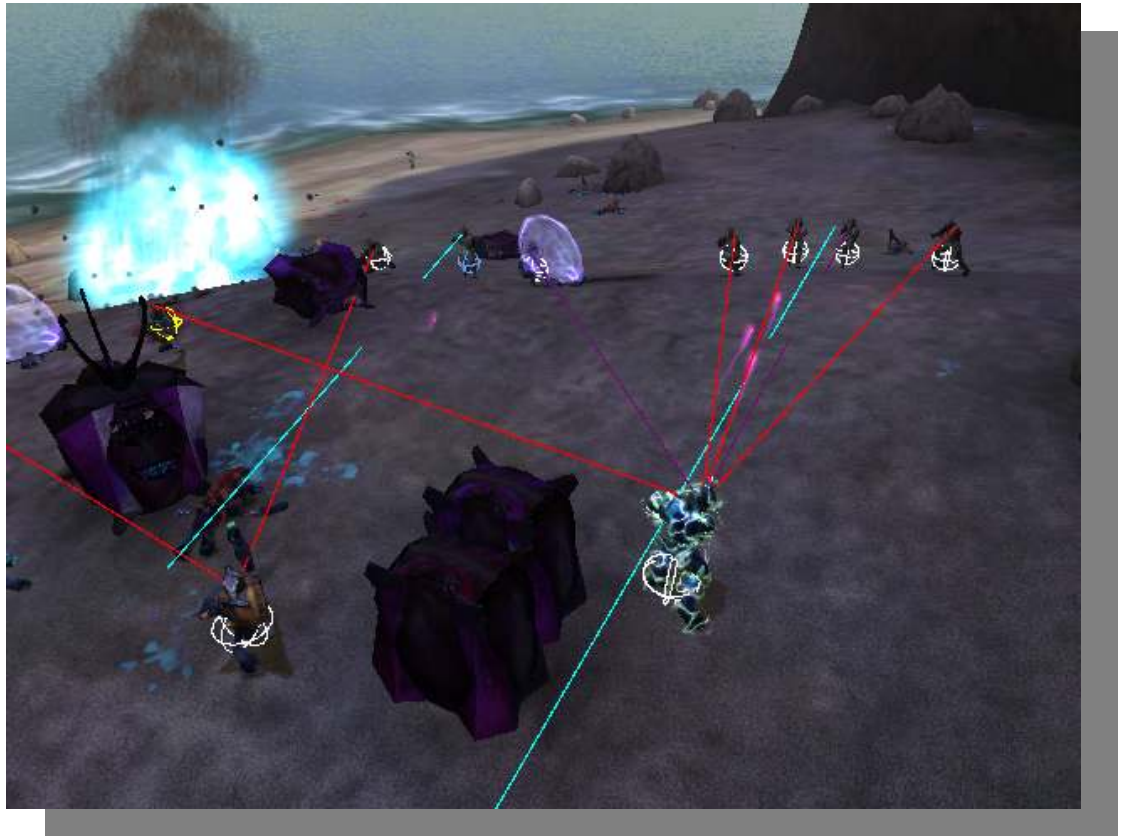
全体のデザイン

全体の戦闘の流れを
デザインする

個々のAI

行動と反応

前節で説明



ポジショニング（位置取り）

デザイナーがスカッド（チーム）ごとに使って欲しいポジションを
グルーピング（同じ色）して置いて行く

↓

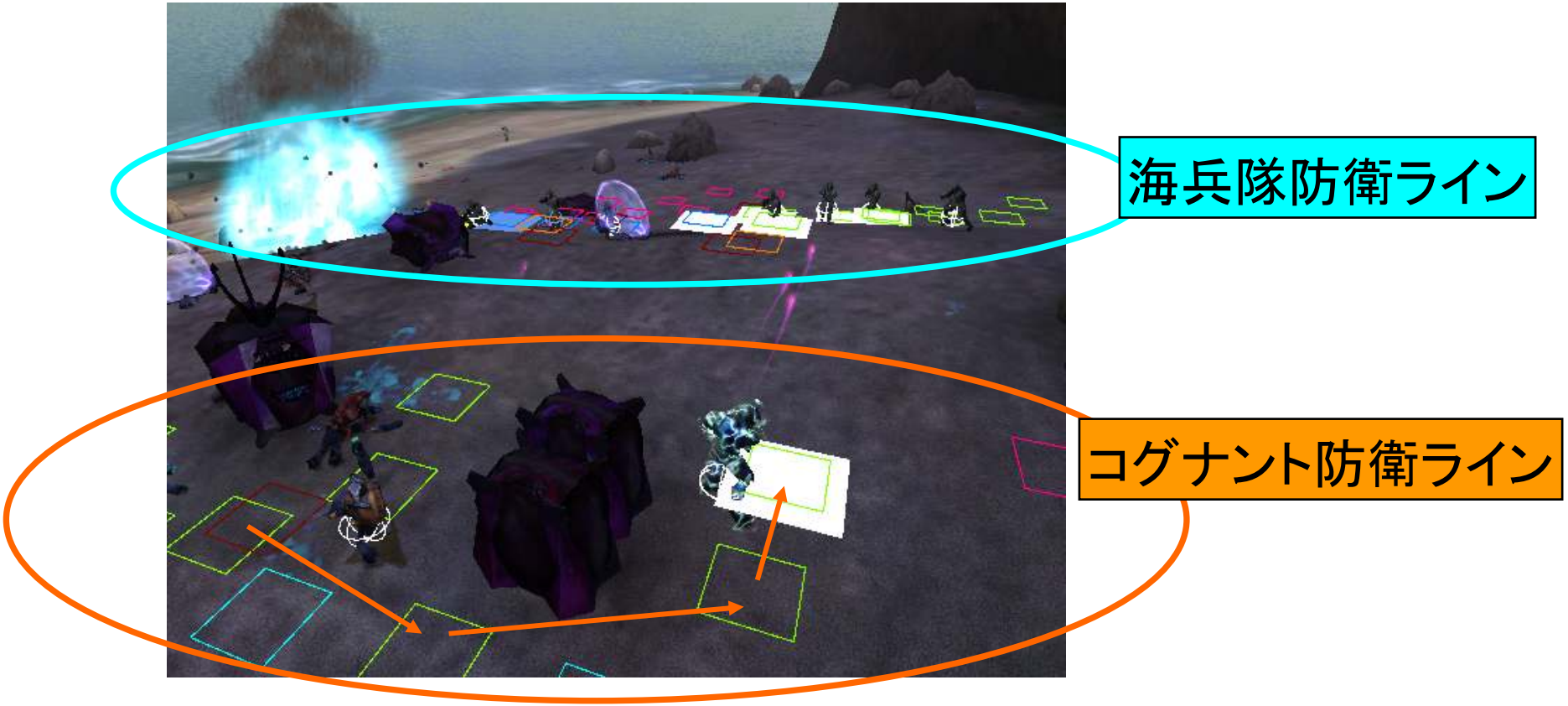
デザイナーがメンバーと出撃エリアを決めてグループを形成

↓

デザイナーがAIチームの全体的な戦闘の動きの流れを制御する

ポジショニング（位置取り）

AIたちは、自分の属するスカッドの指定されたポイント群の間を動く



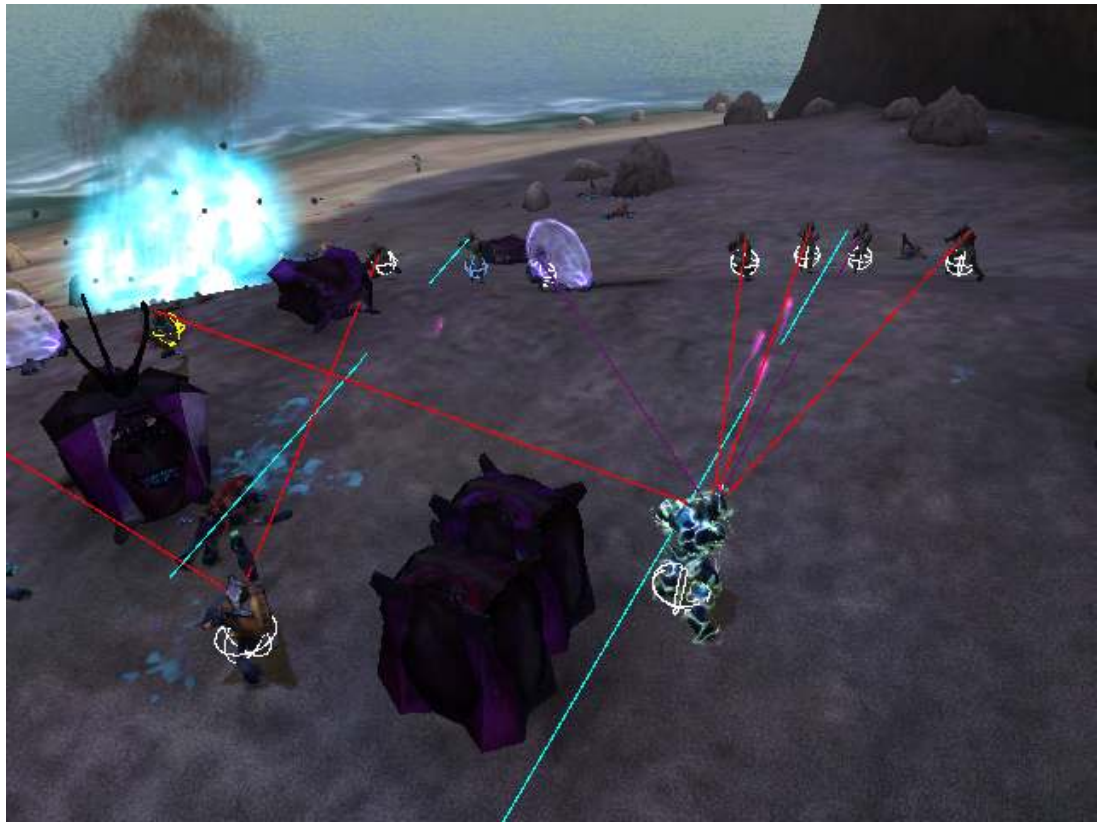
（1）攻撃、防御、後退などの単純な動作はアルゴリズムで自動制御
（2）その他の複雑な戦術的な移動はスクリプトによって制御を行う

ポジショニング（位置取り）

意志決定によってアクションが選ばれると、
各アクションに応じて指定ポジションを使った計算を行う。

（例） 隠れる…各ポイントから敵からの射線が通るかどうか計算する
策敵…最近ターゲットが観察された位置を検索

- 視線
- ターゲットまでの距離
- 隠れることが出来る場所の近さ
- 友軍、敵の位置
- グレネード、車などの情報



AIの計算負荷のうち、レイキャストが60%、
残りはパス検索と認識状態の保持

キャラクターAIの演出

第１節　コンセプト

第２節　思考

第３節　移動

第４節　演出

味方AIからの会話

（1）契機：意志決定と刺激から

(例)　傷つけらる、死ぬ、敵を見る、
グレネードを投げる、隠れる場所を探す

(2)発言内容の選択

A) 優先度、内容、ユニークネス（？）、関連性
B) ランダムにダイアログのタイプを選ぶ
C) 近い場所にいるキャラクターだけが答えられる。

（3）設定

57のイベント
166のダイアログのタイプ
12人の喋れるキャラクター
5147行の会話記録

味方AIからの会話（例）


I'd have been your daddy but that dog beat me over the fence!
Now who's going to save the human race, bright boy?
Hey mate, this one looks like your sister!
Take that skirt off and get back there and fight, soldier!
I'd love it if you'd STOP KILLING MY MEN!
Get up! Get up so I can kill you again!
Notify his next of kin, because they're next!
And my mum thought I was gonna be a doctor...
I didn't like him either, but damn!
Hey mate, I don't think they're gonna give you a medal for that one...

まとめ

企画

（1）Halo の AI　は、考え込まれたコンセプトの上に設計されている。

プログラマー

（2）幾つかの実装や技術を試しながら改善して来た。

（3）プログラマーが仕組みを作った上で、デザイナーにAIの制御を任している。

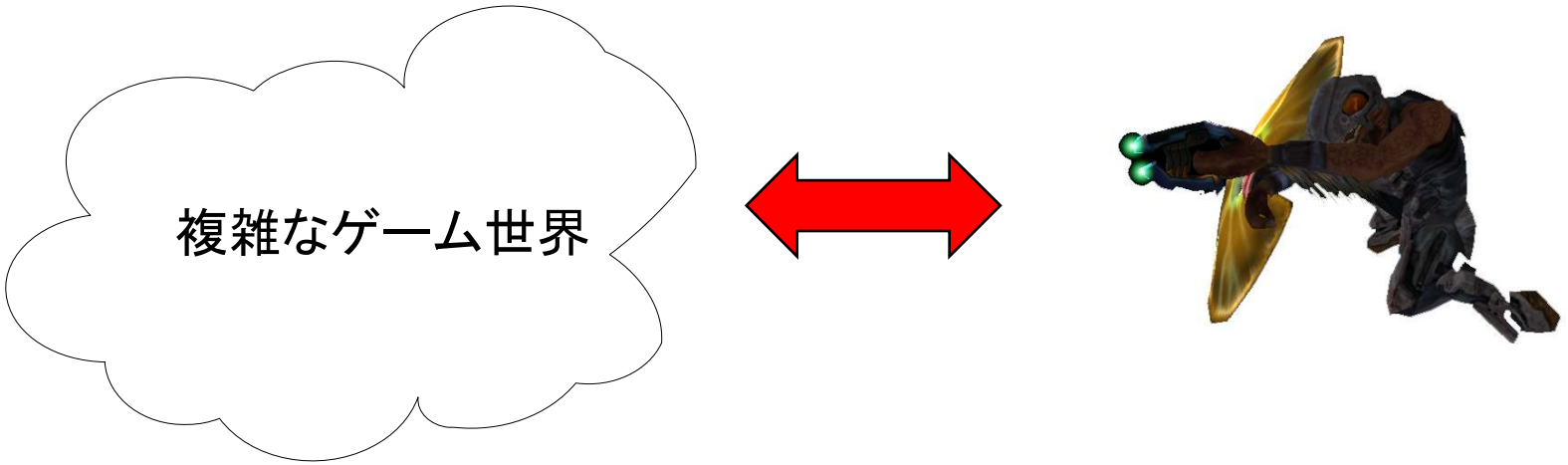
Halo2 デモ

Halo2 AI のコンセプト

第１節　コンセプト
第2節　思考
第3節　記憶
第4節　移動
第5節　デザイナーコントロール

Halo2 AI のキーワード

ゲームAIの陥りやすい問題点



複雑な状況に対応するように作っているうちに、スケーラビリティーをなくしてしまう。
（そのコードはそのAIにしか使えない）

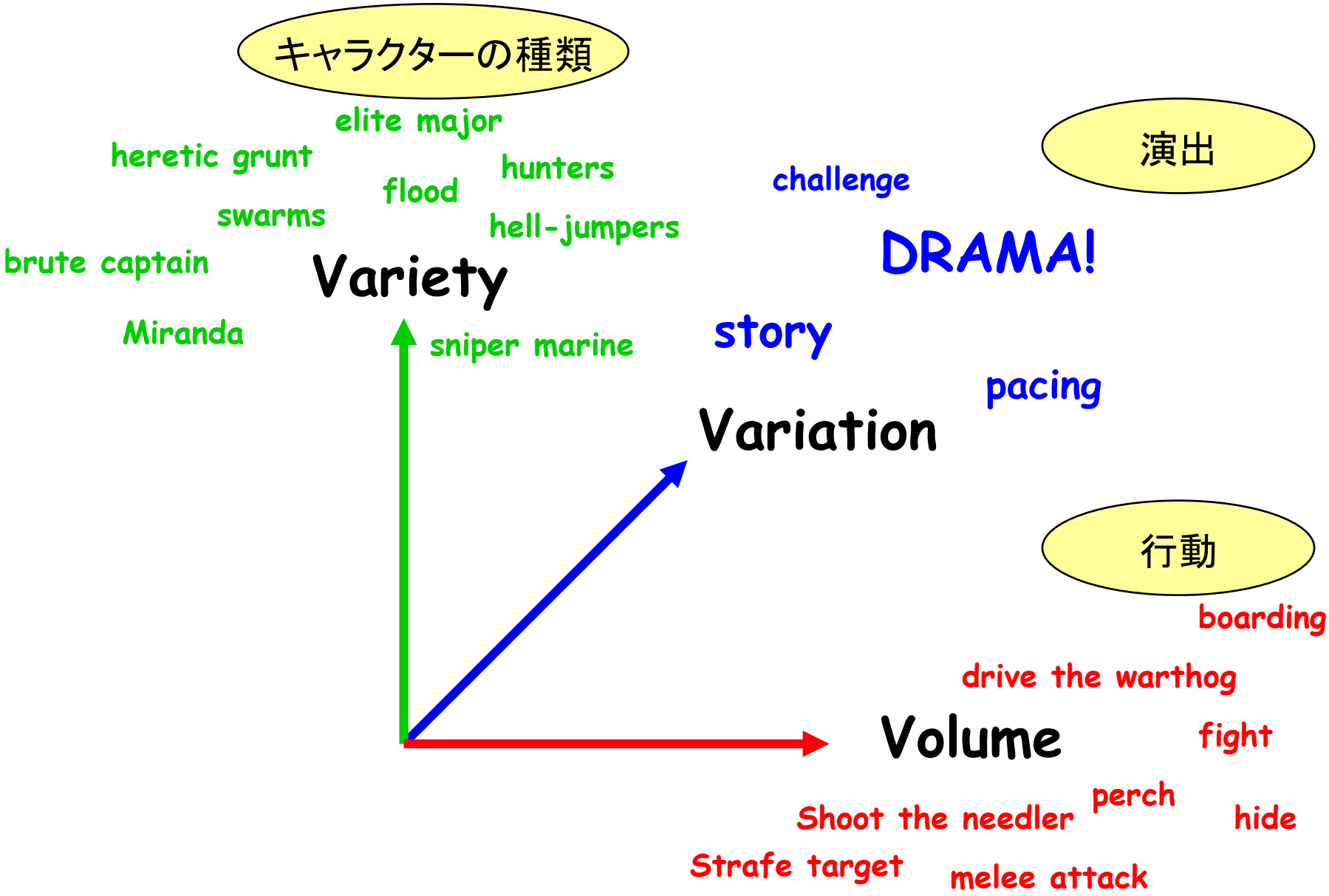
これからのAIに必要なこと

Scalable AI へ

スケーラビリティー（Scalability） ＝ 柔軟な拡張性を持つこと
簡単なAIから複雑な思考のAIまで、全て同じ仕組みの中で実現できること

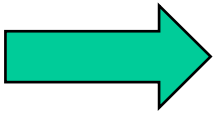
これは複雑になるゲーム世界の中でいろいろなAIを同時に実現するために必要

Halo2 AIのスケーラビリティーの3つの軸



Halo2 AIの目標

スケーラブルな意志決定を実現する



仕組み： HFSM x キャラクターによる使い方の差

Halo2 AI の思考の仕組み

第１節　コンセプト

第２節　思考

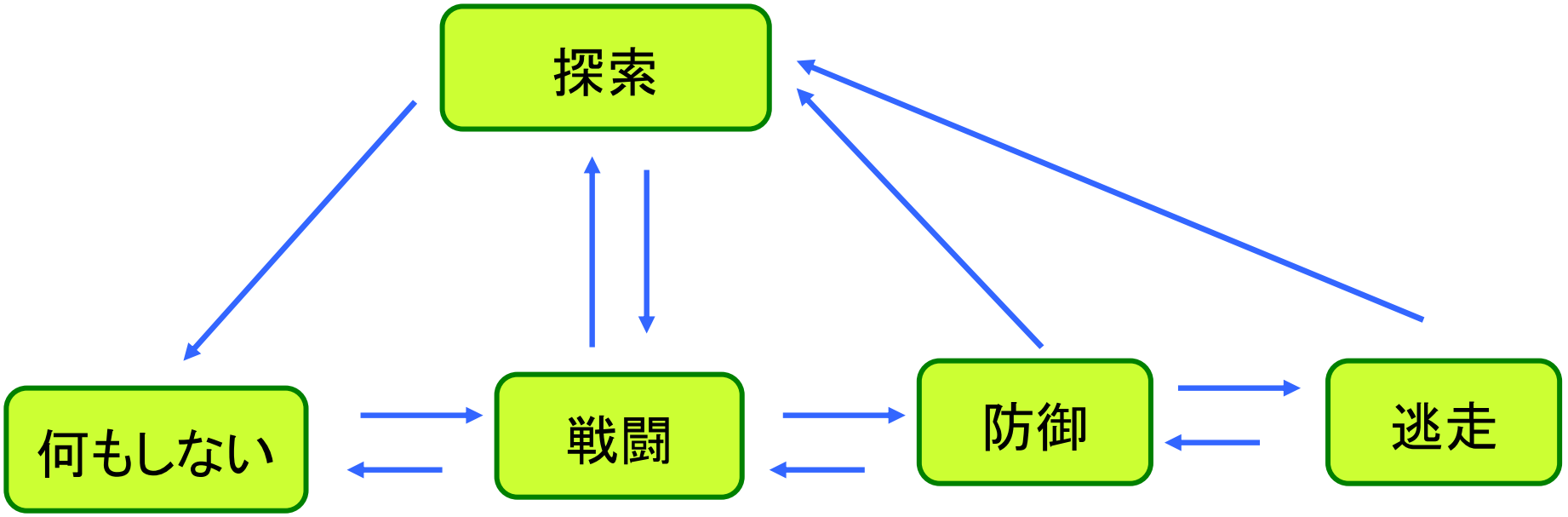
第３節　記憶

第４節　移動

第５節　デザイナーコントロール

FSMを検討する

基本戦闘サイクル



このダイアグラム自体は単純だが、遷移条件や状態からアクションを指定するロジックは複雑になり、実装は難しい。

→ 別の仕組みを探す

HFSMによる意志決定

DAG (有効非環グラフ)　一方向にノードが伸びて、ノードが環を為すことのないグラフ構造

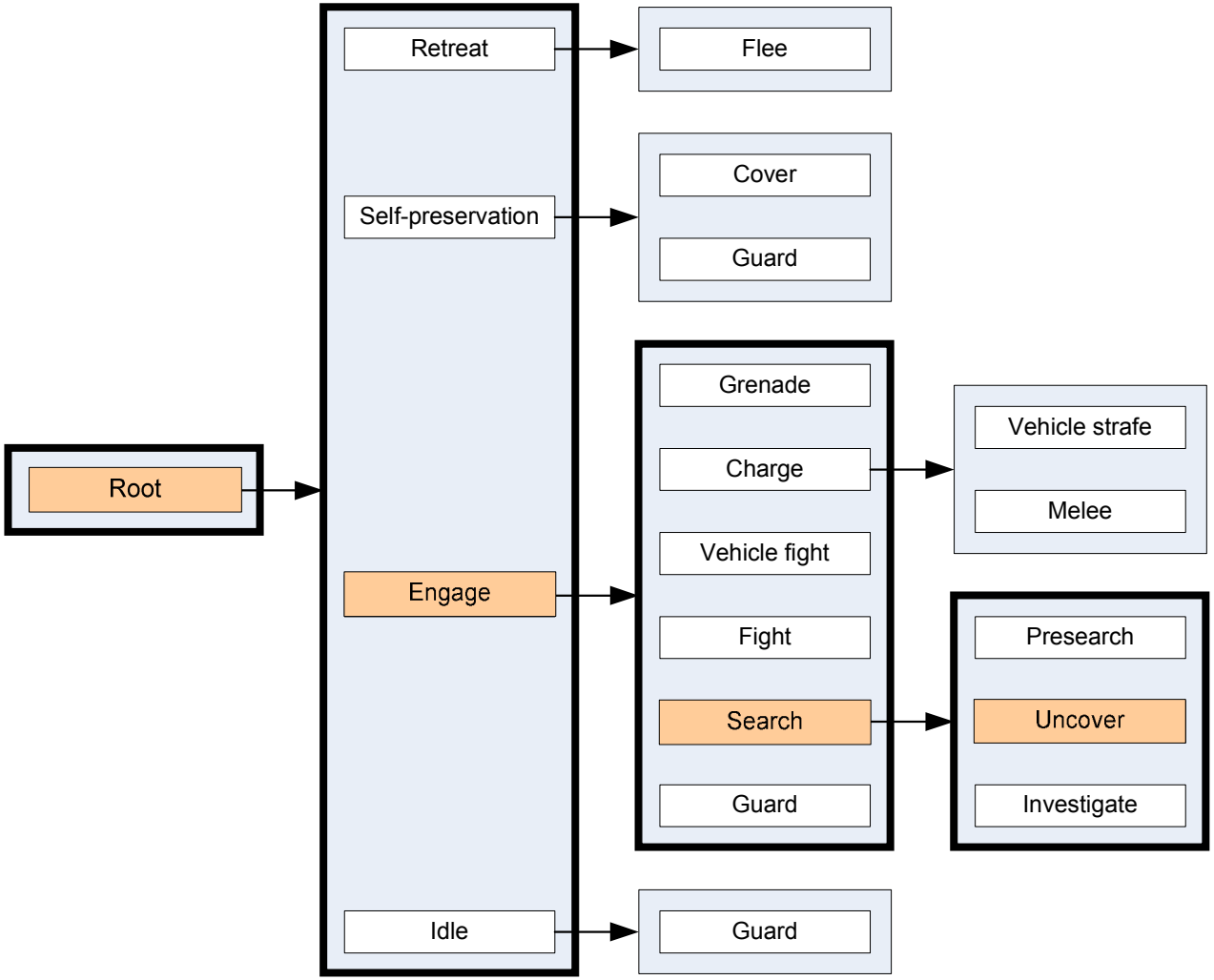
振る舞いツリー　末端のノードは全てキャラクターの行動を表現する

長所

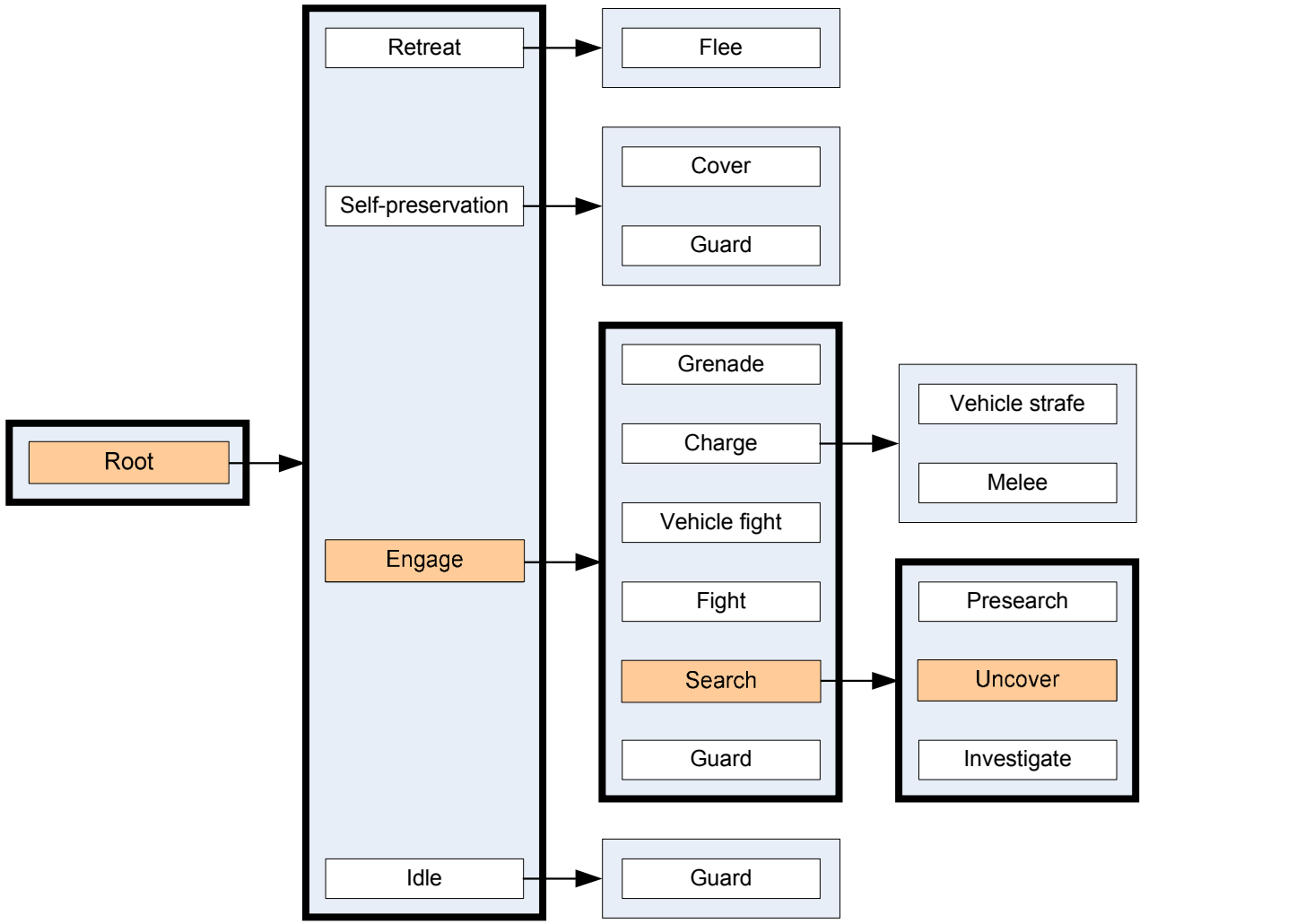
- 直感的
- モジュールに分解可
- スケーラブル

短所

- 見かけほど単純ではない
- デバッグが難しい
- メモリーを取る

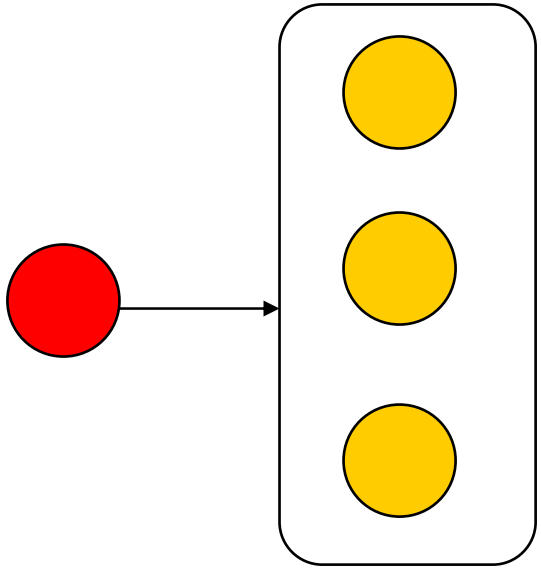


HFSM の動作原理

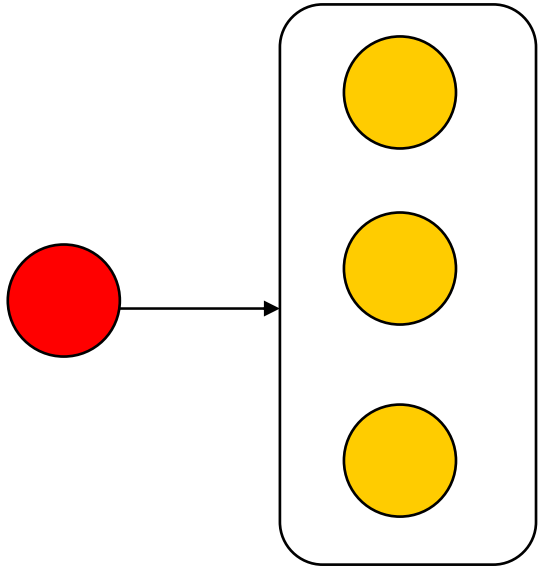


どのようにノードを選択して行くのだろうか？

ノード選択の原理



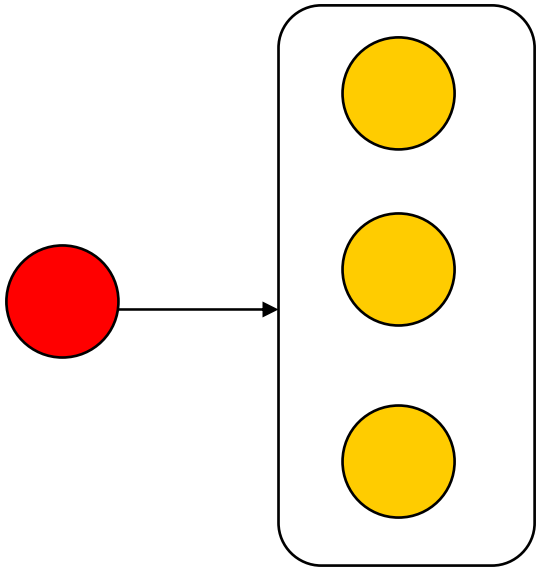
親ノードが子ノードを選ぶ

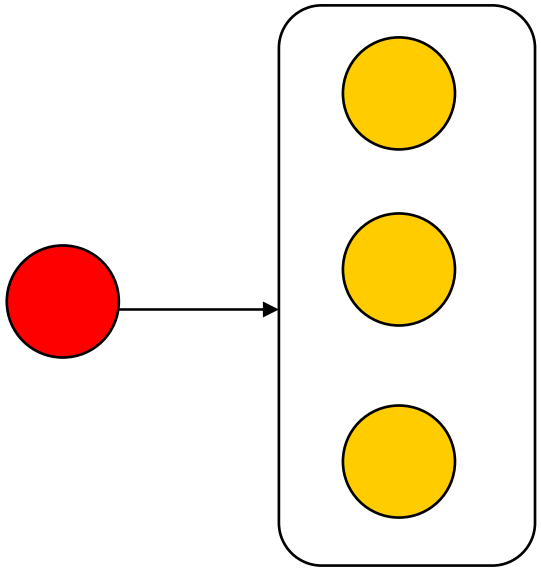


子ノード層がノードを選ぶ
(子ノード同士が争う子ノード競合モデル
child-competitive decision model)

Halo2

子ノード競合モデル





実数による評価値システム

Chrome Hounds

第３回セミナー参照

長所：状況を評価できる。欠点：評価式が複雑

バイナリーテスト & 選択アルゴリズム

Halo2

評価関数によって、行動がその状況で
実行できるかだけを判断し、実行可能な
行動の実行順序をアルゴリズムが決定

バイナリーテスト…*車に乗っている、いないか、*
敵をみつけているか、いないか複数の条件判定からなる。

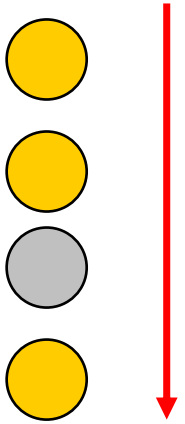
ノードの実行スキーム（アルゴリズム）

どのスキームを選ぶかは、親ノードの種類によって決定されている。

① 優先度リスト法

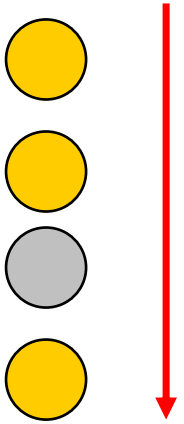
root　engage

あらかじめノードの優先度を
決めて、第１位から実行するが、
実行中のノードより高い優先度を
持つノードは、インタラプトする
ことができる。



② シークエンス法

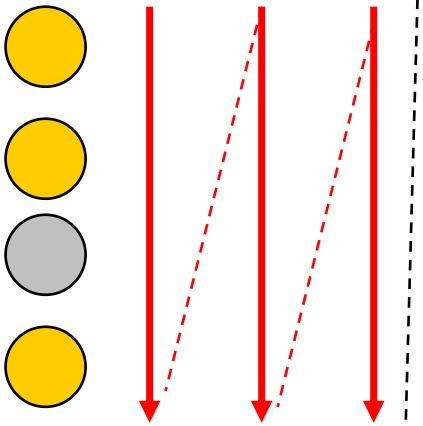
あらかじめノードの実行順序を
決めておいて、実行可能な
ノードをその順番に従って実行する



③ シークエンシャル・ルーピング法

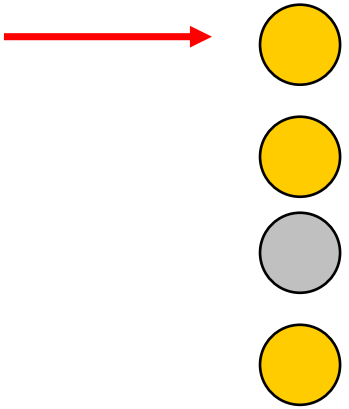
search

基本は②と同じだが、
終端まで行くと、
もう一度実行する。



④ 確率的方法

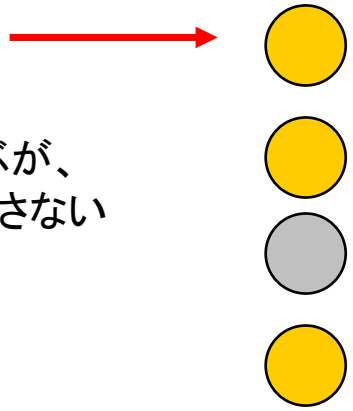
ランダムに選ぶ



ノードの実行スキーム（アルゴリズム）

⑤　オン-オフ法

postcombat



ランダム、或いは
優先リスト法によって選ぶが、
一度選んだものを選び直さない

実行順序を定義　＝振る舞いの順序を規定できる。

振る舞いの順序（常識）を規定できる。

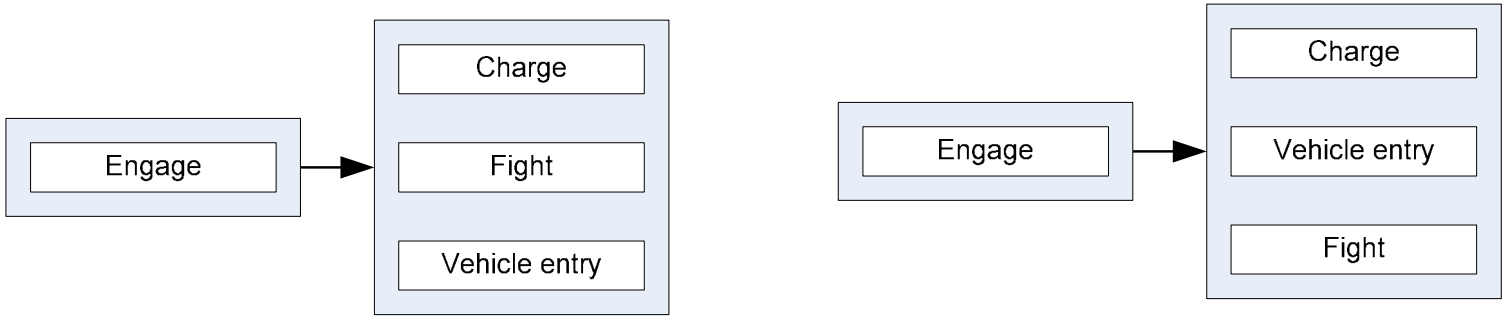
本システムの4つの問題点

① 子ノード競合モデルの問題点
② くり返し条件判定の問題
③ キャラクタータイプに特化した振る舞い
④ レアイベントの扱い

①子ノード競合モデルの問題点と解決法

問題

「優先度リスト法」で優先度を固定するが、
状況によって優先度は違うではないか？

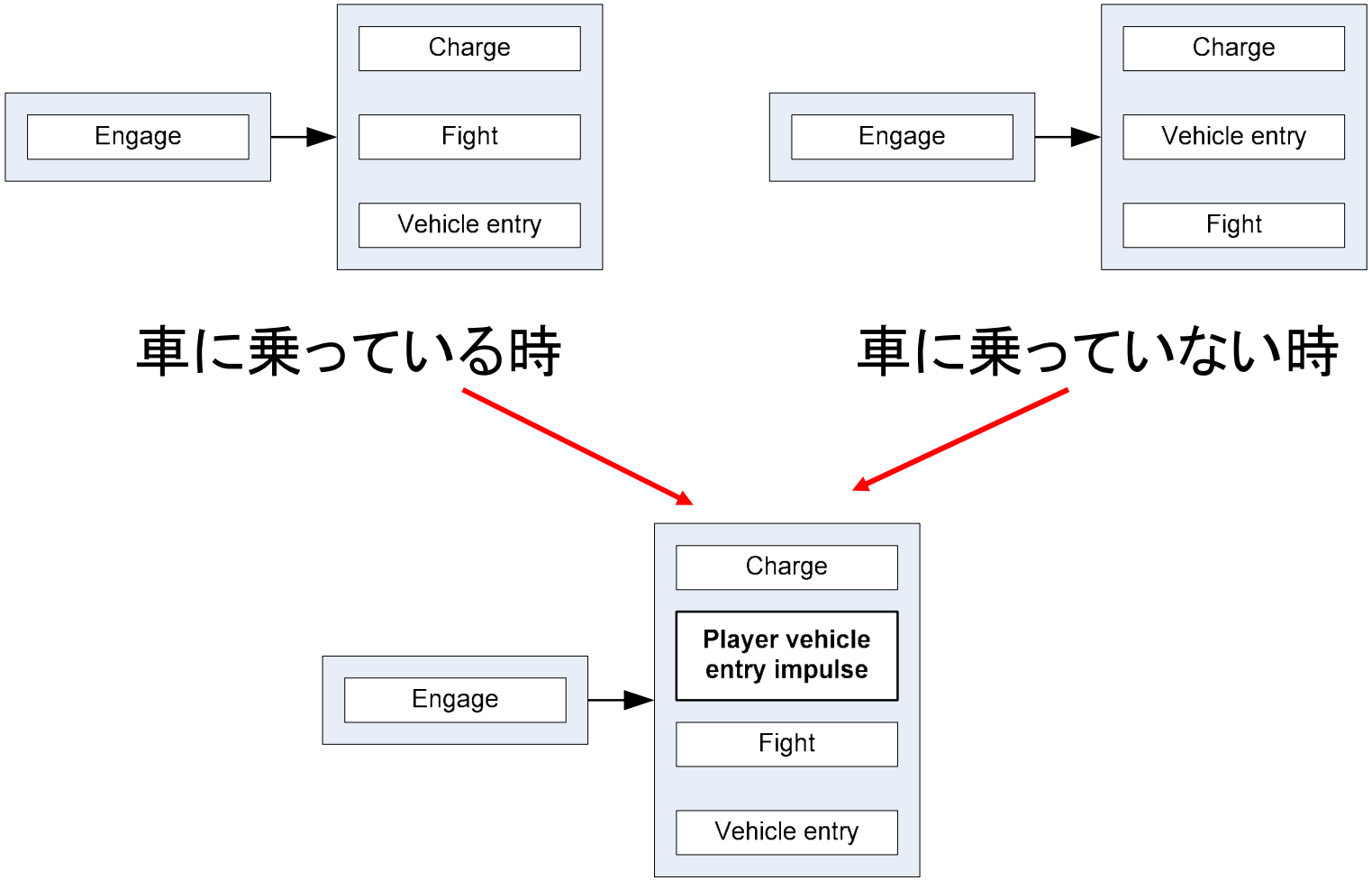


車に乗っている時と乗っていない時では、当然優先度が違う

子ノード競合モデルの問題点と解決法

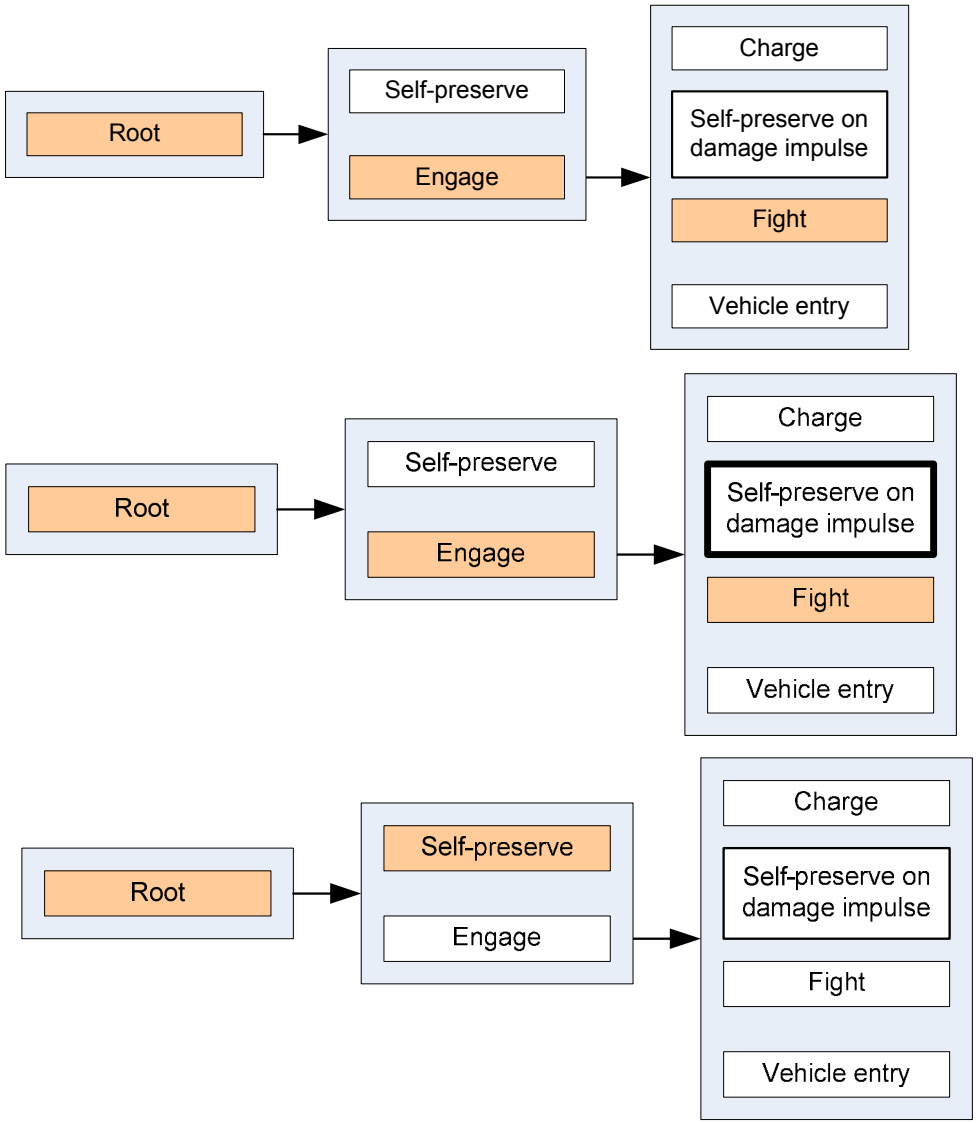
解決法

トリガー付きのモジュールを用意する



トリガー付きモジュールの他の応用法

上位レベルのモジュールへ
飛び移る（Pop）する



②くり返し条件判定の問題

問題

評価関数は、何度も何度も同じ条件を判定している
しかし、大きな条件に対しては事前に計算しておく

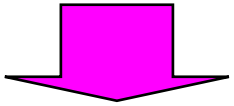
解決法

事前に実行可能性を判定するためのアルゴリズム

（1）各振る舞いの実行可能条件を、ビット列で表現する
（2）その条件を現在の状態のビット列と比較する

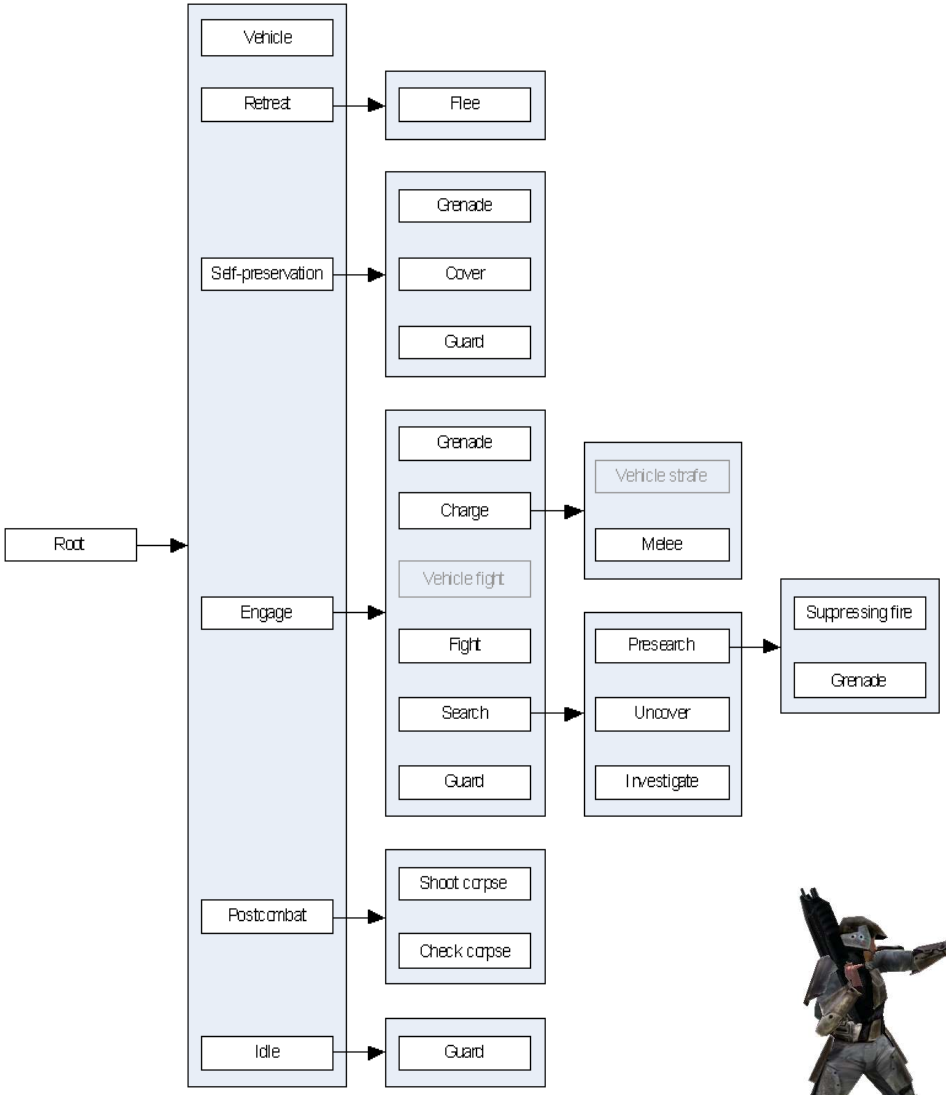
	乗車？	敵発見？
ある振る舞いの実行条件 →	0	0
条件 →	1	0

同じではない

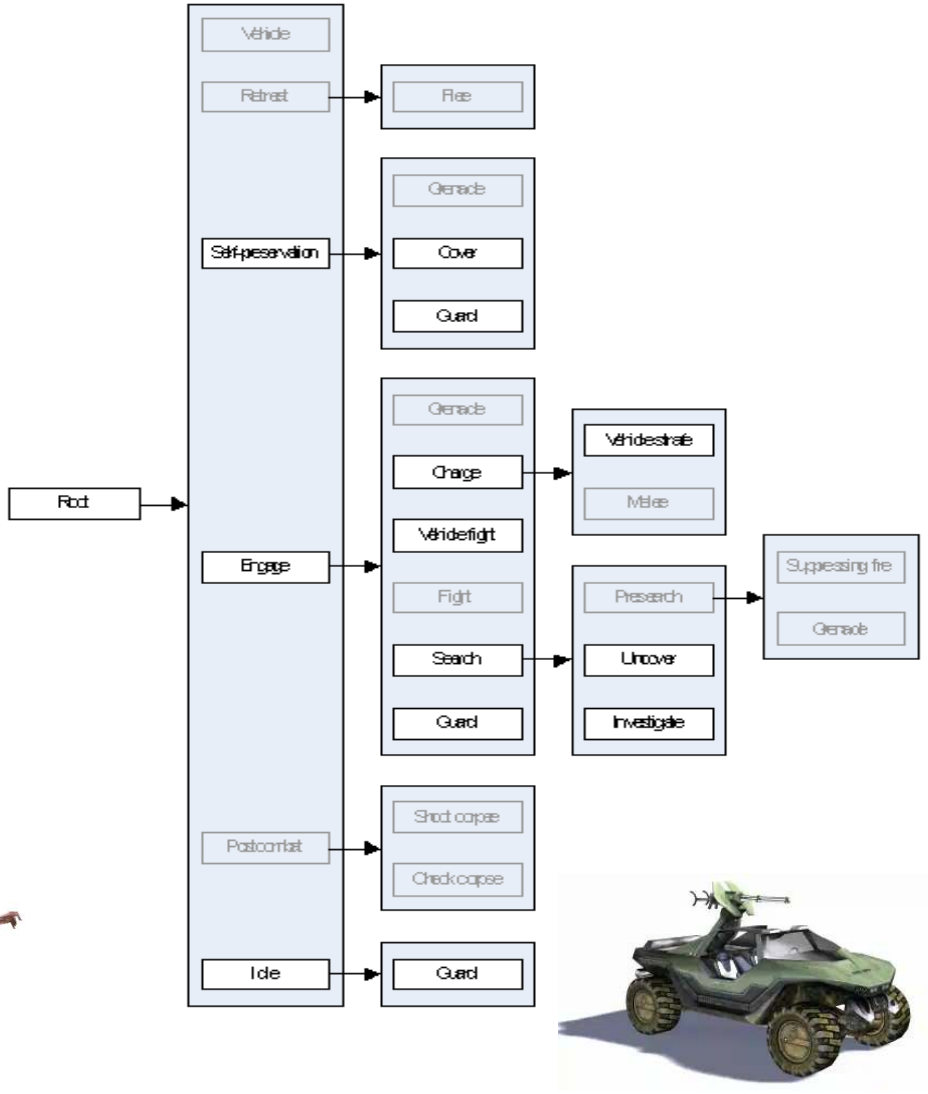


実行不可

車に対する状態による振る舞いの実行可否

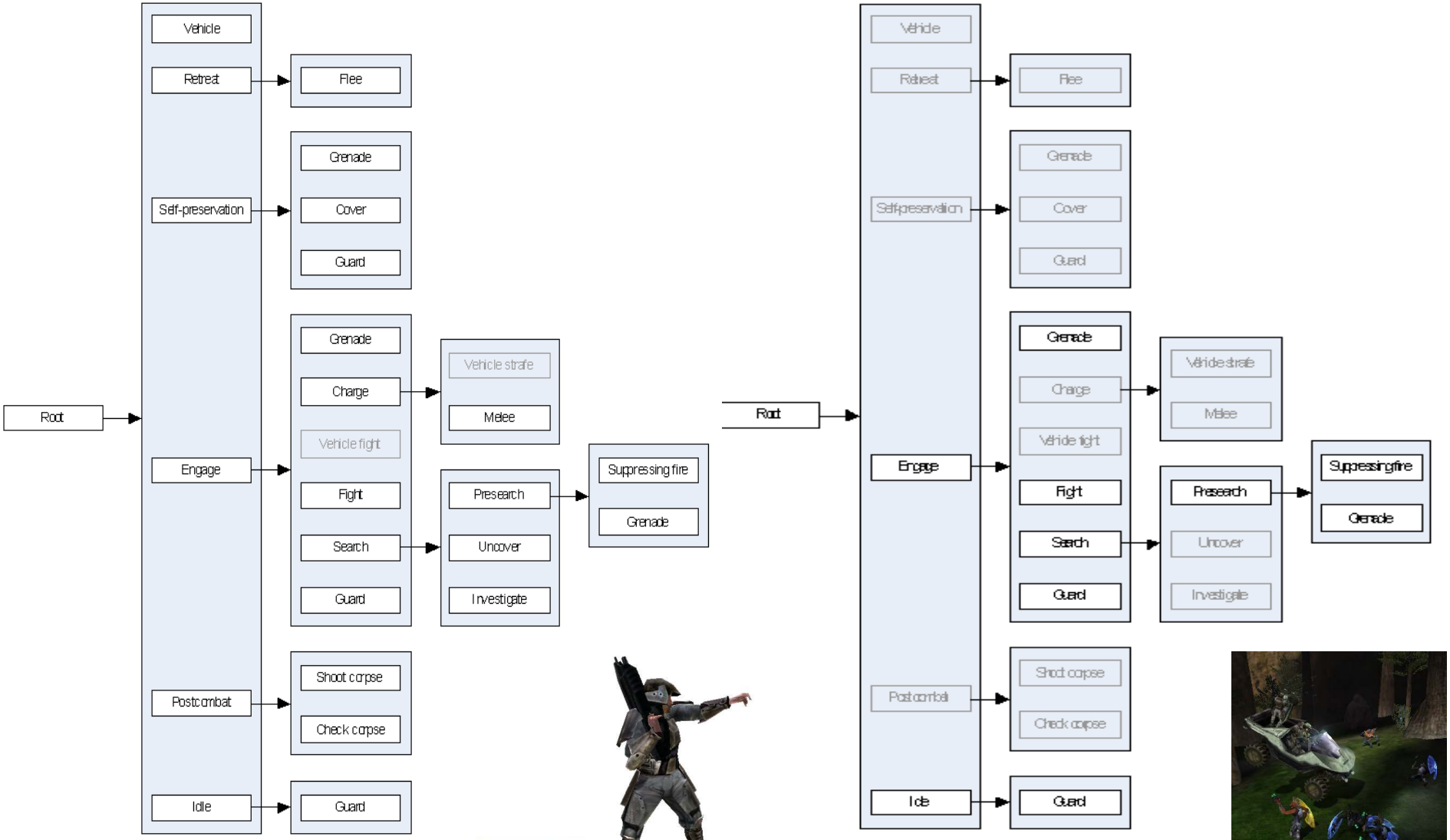


歩兵



車の運転手

車に対する状態による行動の実行可否



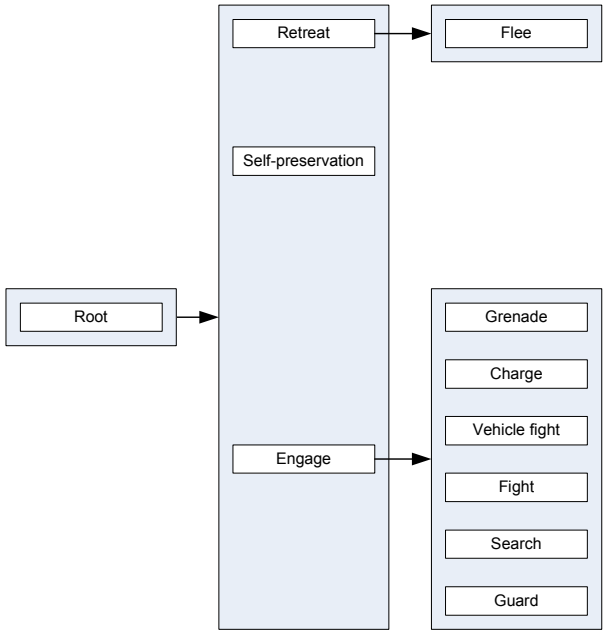
③ キャラクタータイプに特化した振る舞い

問題

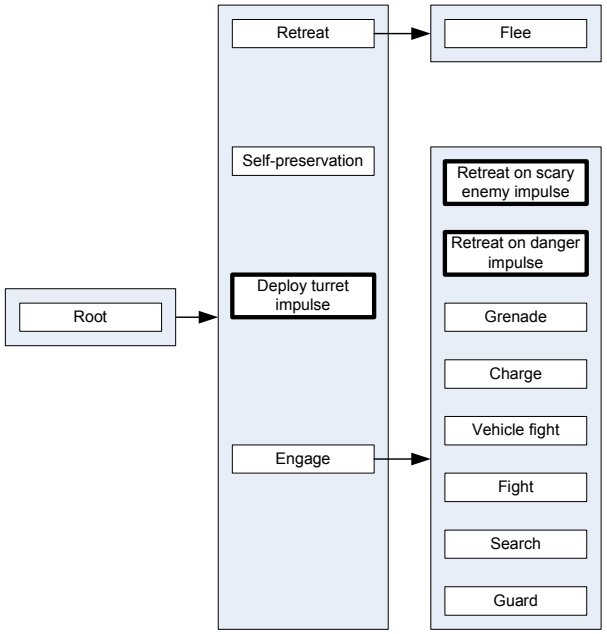
1つのタイプでしか実行できない振る舞いを何度も判定したくない

解決法

最初からFSMに加えるのではなく、自分が判定するときに加える。



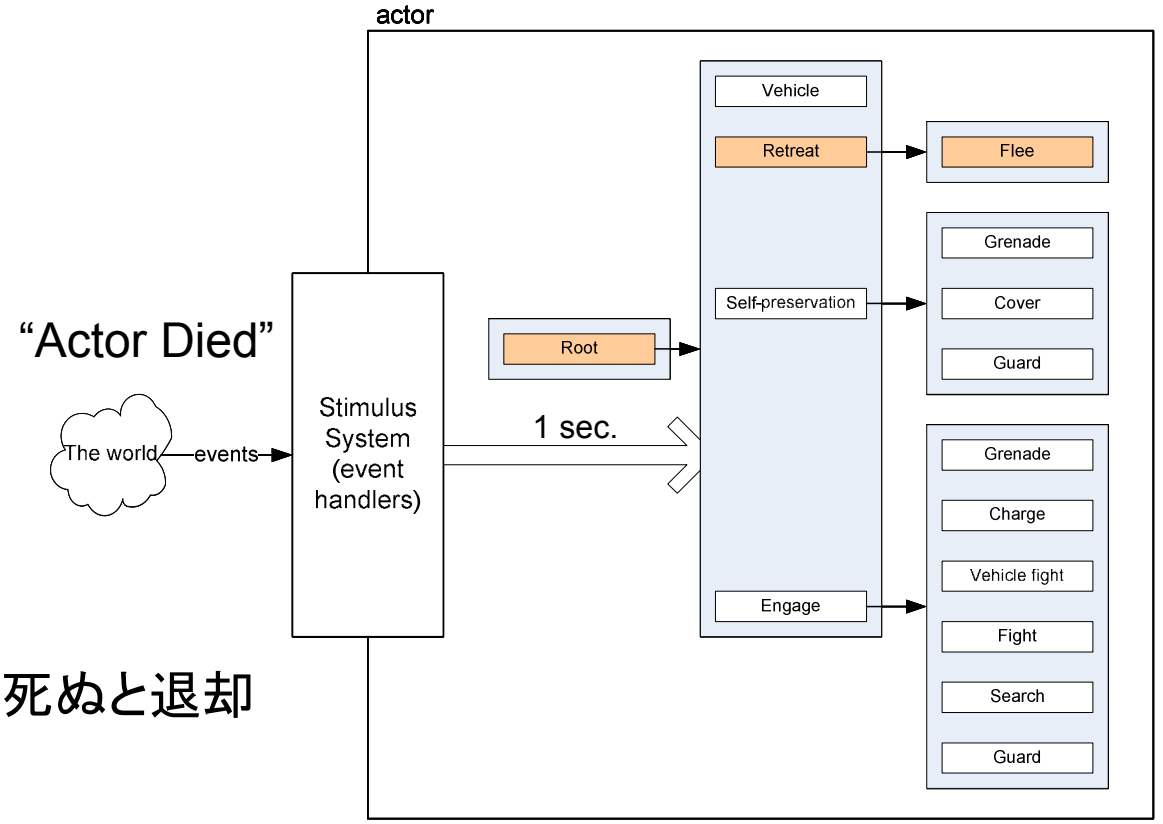
Generic



Grunt

④レアイベントの扱い

問題　滅多に起こらないイベントの判定を毎回したくない

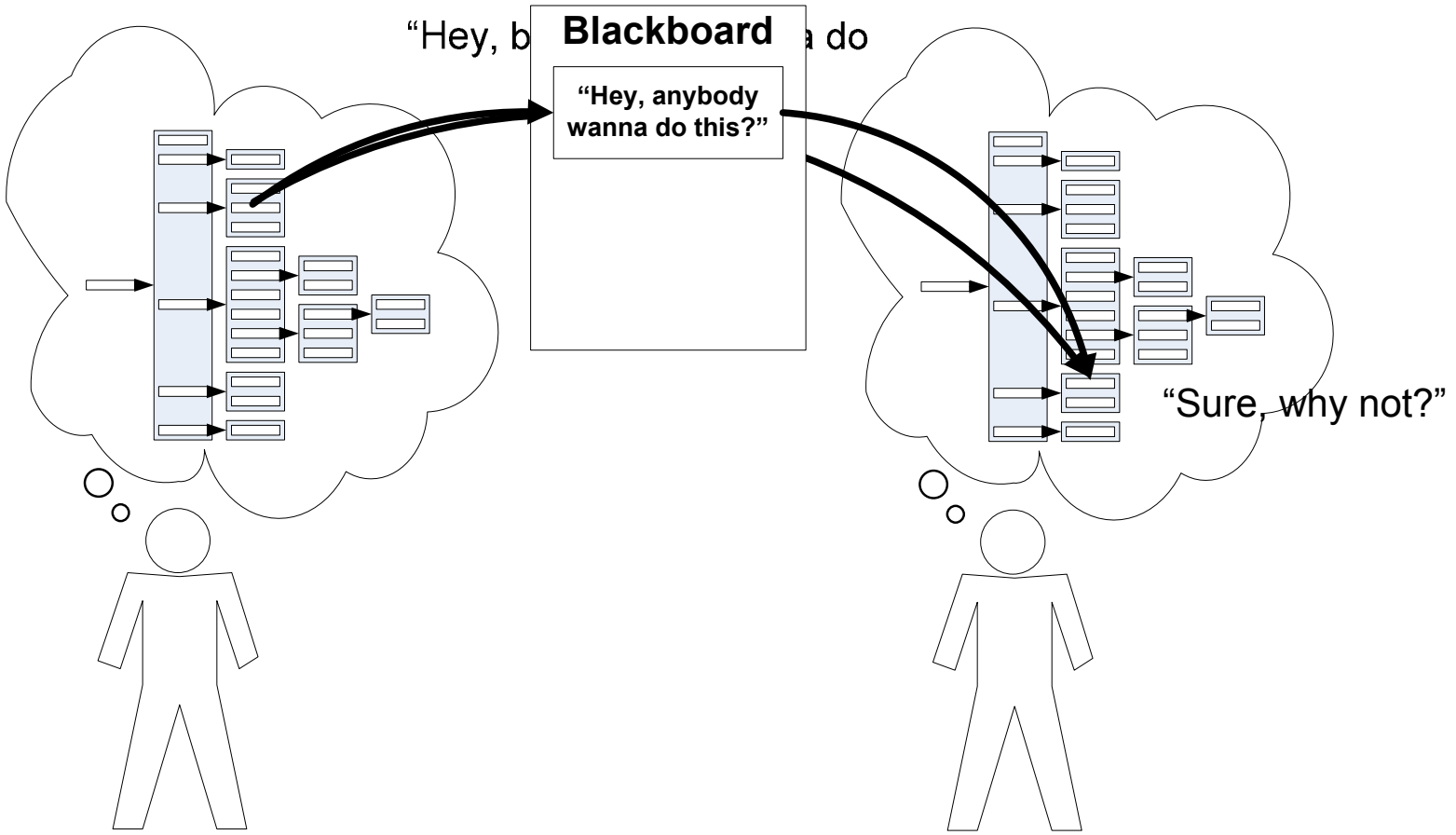


(例)グラントはリーダーが死ぬと退却

解決法

突発的な事象による振る舞い
非同期で振る舞いを割り込ませることが出来るシステムを作っておく

補足:ブラックボードによる協調



Halo2 の記憶構造

第１節　コンセプト

第２節　思考

第３節　記憶

第４節　移動

第５節　デザイナーコントロール

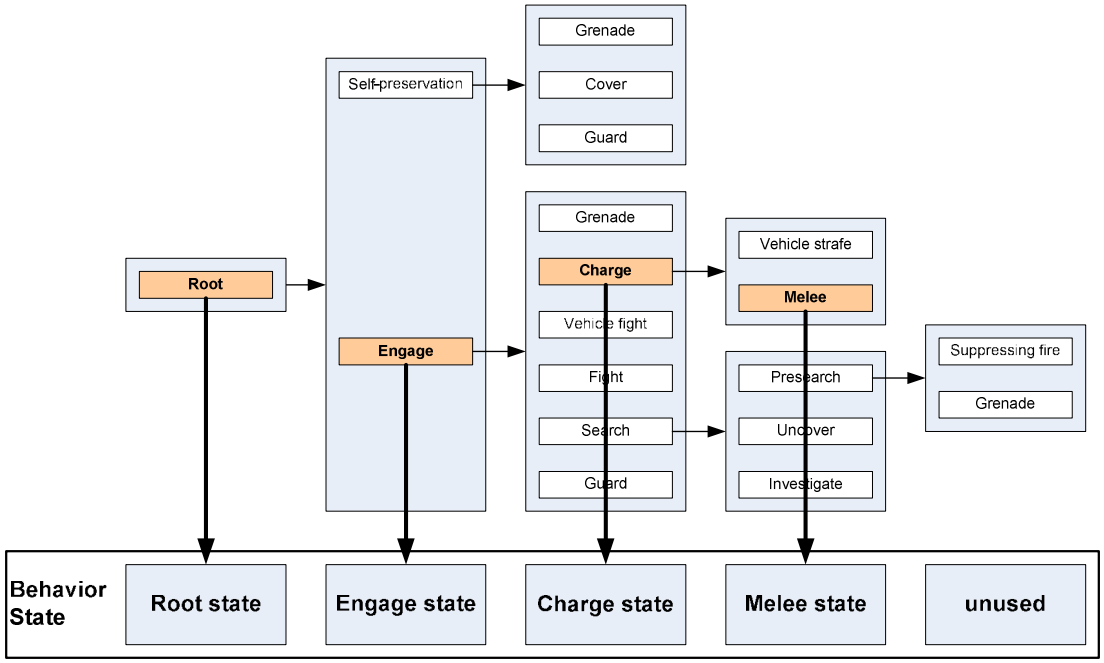
メモリーと記憶

各キャラクターについて、HFSMを保持していると、
ツリー構造の拡大に従って大きなメモリーを取ってしまう

⇒ 振る舞いのチャンクだけを保持しておく

ツリー自身は、
静的データとして
キャラクターとは
独立に存在し、
全キャラクターに
再利用される。

この部分だけを記憶

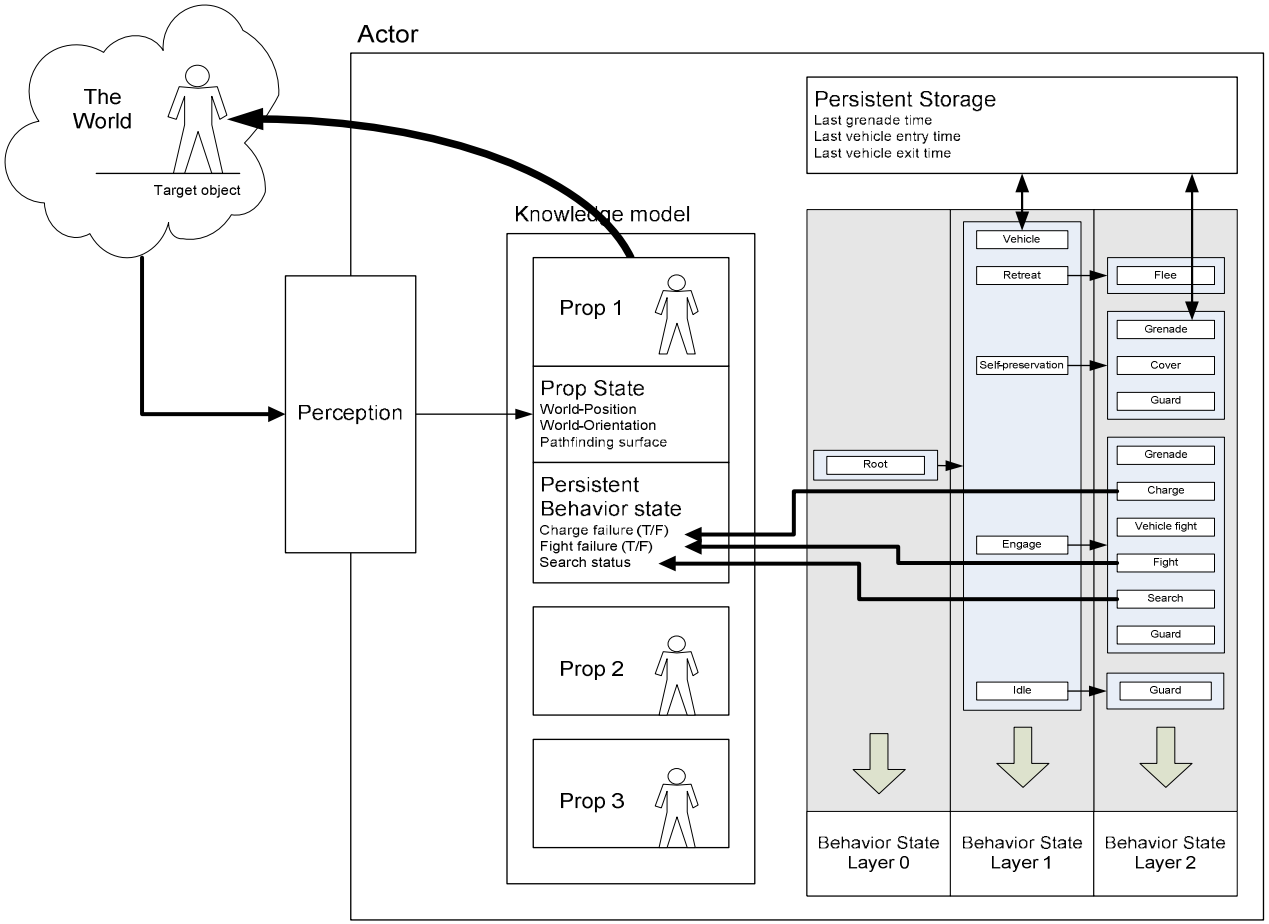


⇒ 「過去の振る舞いについての記憶が保持されない」という問題

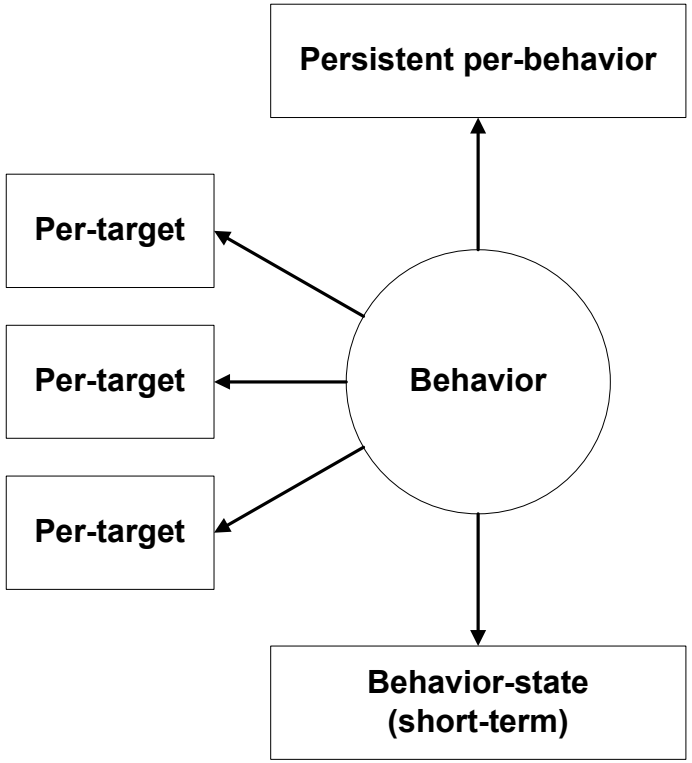
「過去の振る舞いについての記憶が保持されない」という問題

（例）過去いつグレネード弾を投げたか？
（一定時間空かないと投げることが出来ない）
索敵の結果。

➡ ツリー構造の外に各キャラクターが自分の記憶プールを持っておく



記憶モデル＝知識モデル



知識モデルが、AIの判断の基礎となる

オブジェクトについては、自分の視界を
遮った場合のみ保持する。

- 各振る舞いについて永久記憶　グレネードを投げる、車のアクション
- 各振る舞いについての短期記憶　その振る舞いが終わったときの状態
- 各ターゲット についての記憶 認識情報,最後に観測した位置, 方向
- 各ターゲットについての各振る舞いの記憶

  最後に乱闘した時間、その対象に対する索敵失敗結果、パス検索失敗結果

Halo2 AI の移動

移動のためのデータ構造

階層的なデータ構造

用途

ゾーン

↓

エリア ——— プレイグランド

↓

ポジション ——— 戦術的解析

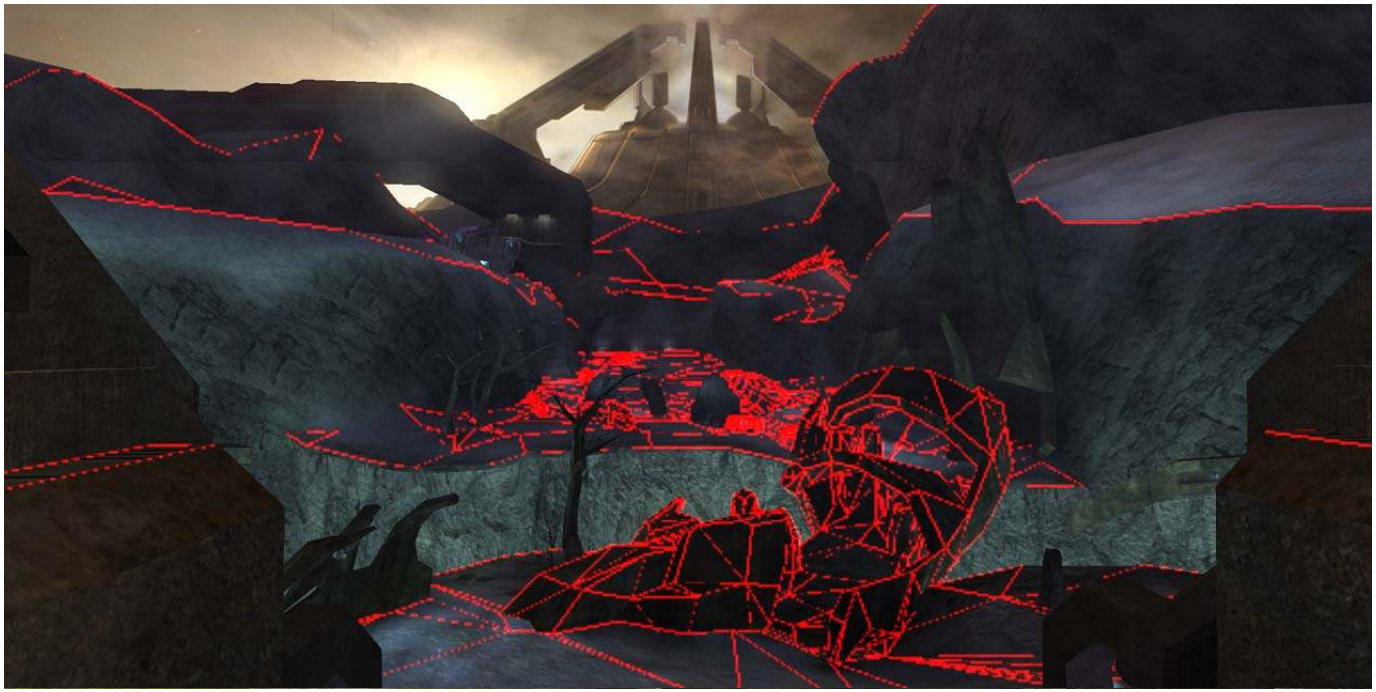
ナビゲーション・メッシュ ——— 移動

AIに何をさせたいかによって、それに適したデータ構造を
用意せねばならない。（知識表現、世界表現）

ナビゲーション・メッシュ

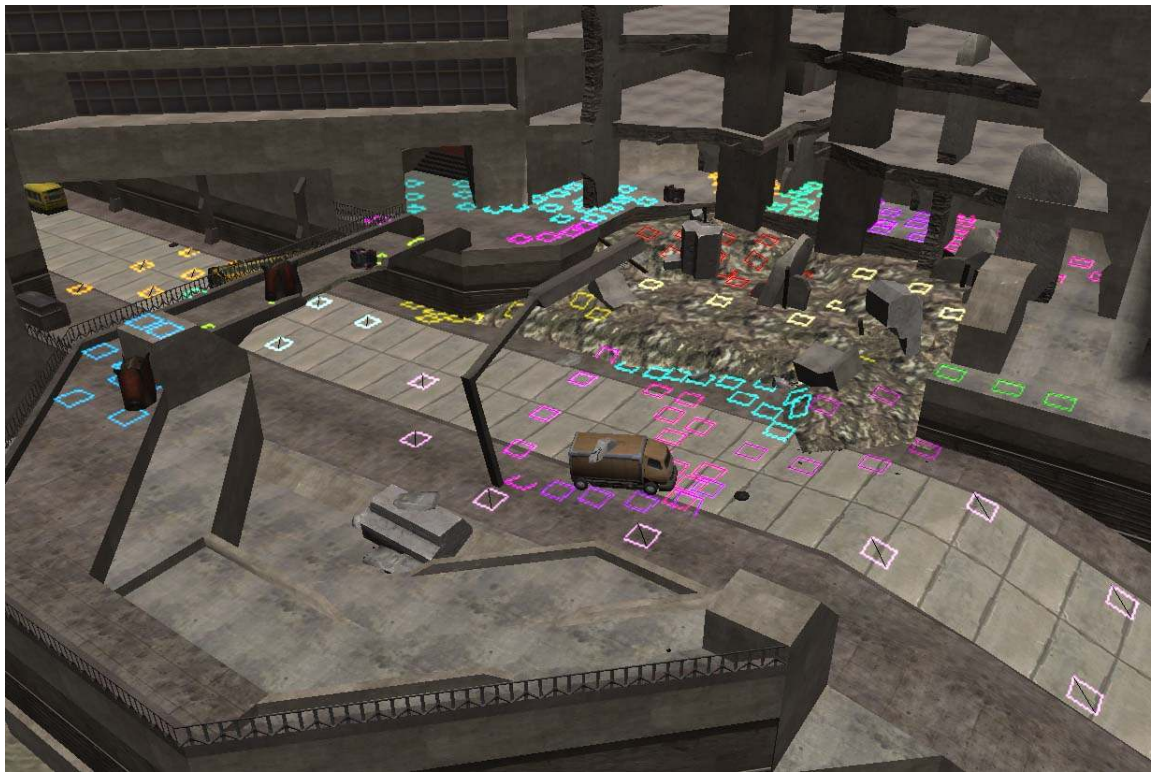
ナビゲーション・メッシュ

移動のためにマップ上に敷き詰めた三角形



パス検索を行い三角形に沿って歩くことで
任意の場所へ行くことができる

ポジション



戦術的位置を「プレイグランド」としてグルーピングしながら配置

「プレイグランド」内は、パス検索なしで自由に移動できる。

移動のためのデータ構造

階層的なデータ構造

用途

ゾーン

↓

エリア ——— プレイグランド

↓

ポジション ——— 戦術的解析

ナビゲーション・メッシュ ——— 移動

AIに何をさせたいかによって、それに適したデータ構造を
用意せねばならない。（知識表現、世界表現）

キャラクターコントロールの原理

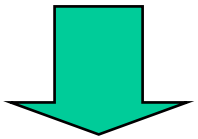
プレイグランド内はパス検索なしに自由に移動できる。
それ以外は、ナビゲーション・メッシュ上のパス検索で移動する。

デザイナーコントロール

第１節　コンセプト
第２節　思考
第３節　記憶
第４節　デザイナーコントロール

デザイナーコントロール

前節までで Halo2 AI の思考と記憶の構成について説明した



このシステムの上に、デザイナーがAIをカスタマイズして行く

デザイナーの仕事

キャラクター定義

- モデル、アニメーション
- 振る舞いのパラメーター
- 静的コントロール　バラエティー豊かであること

ステージ上で動かすためのスクリプト

- 動的コントロール　バリエーション豊かであること

静的コントロール

- キャラクタータイプ
  - エリート
  - グラント
  - マリーンズ（兵士）
- 種類
  - ゴールドエリート
  - レッドエリート
  - Honor guard elite
  - Jetpack elite
- それぞれの種族はキャラクタープロファイルを持つ

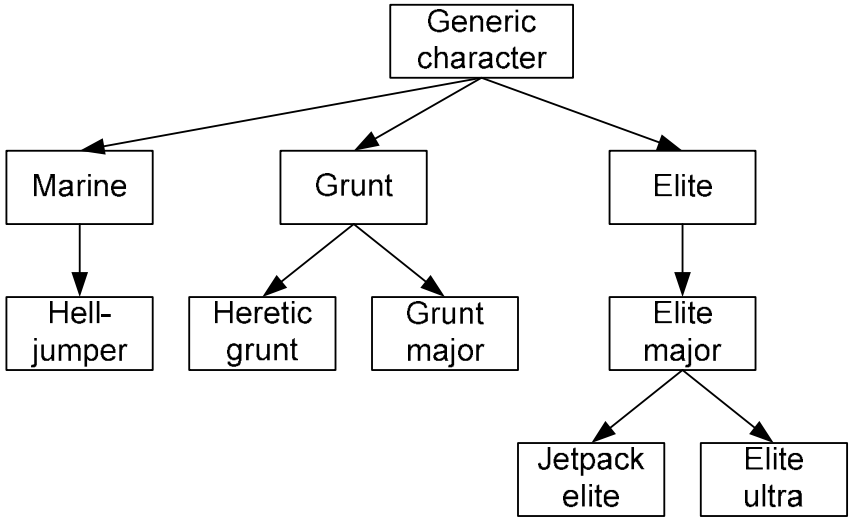
膨大なパラメーター

64(variants) x
80 (behaviors/variant) x
3 (parameters/behavior) =

15,360

静的コントロール：パラメーター調整

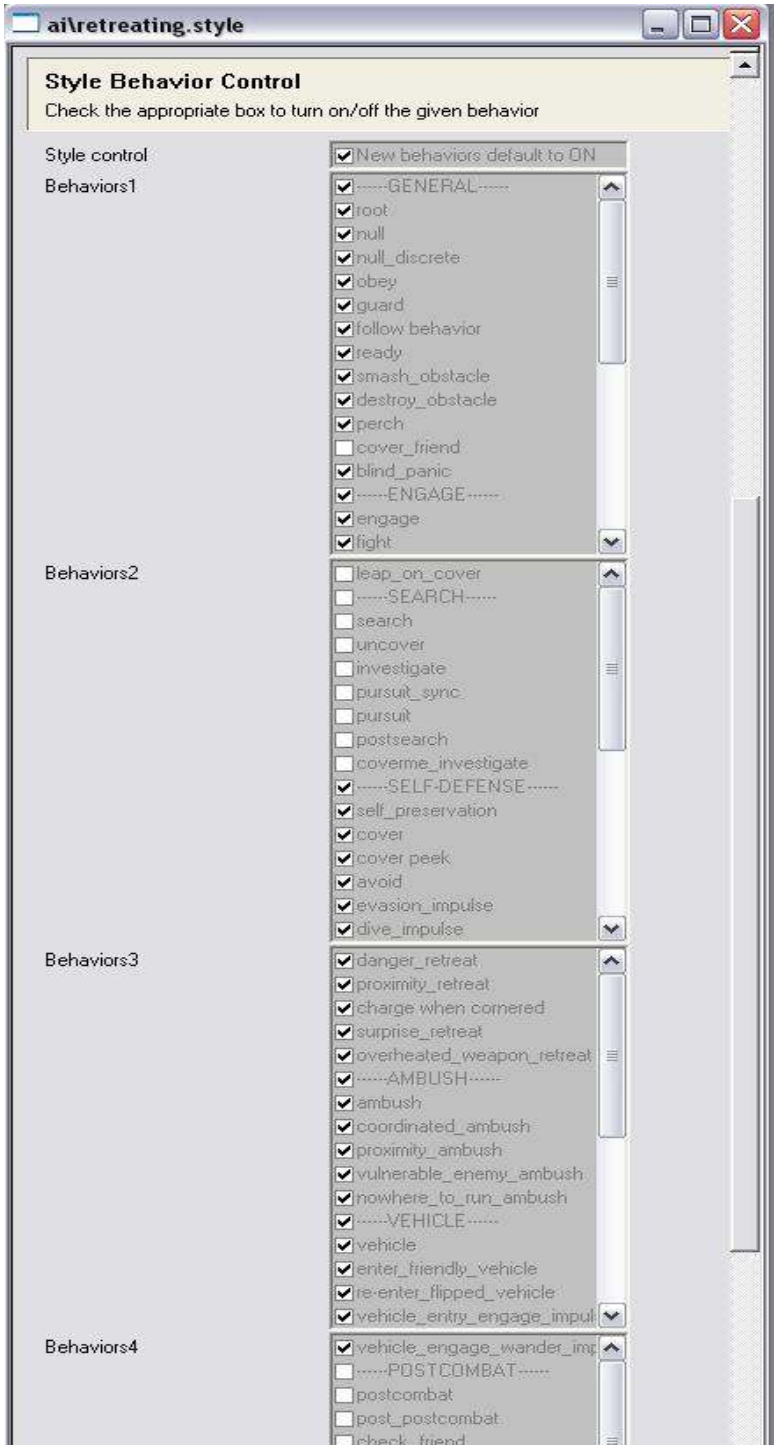
継承構造によるパラメーター定義



体力、感覚、索敵能力、使用武器などの情報を継承しながら分岐

静的コントロール: スタイル調整

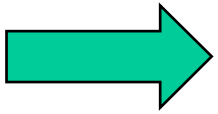
ツールによる振る舞いのマスク



動的コントロール：マップ上のAIのグループ

Halo

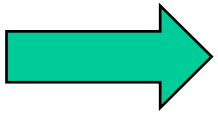
エリアとメンバーを指定してグループを形成して出撃



エリアにグループが固定される

Halo2

エリアとメンバーを分離。



グループのエリア間を移動が可能

Halo2における集団制御

Squads

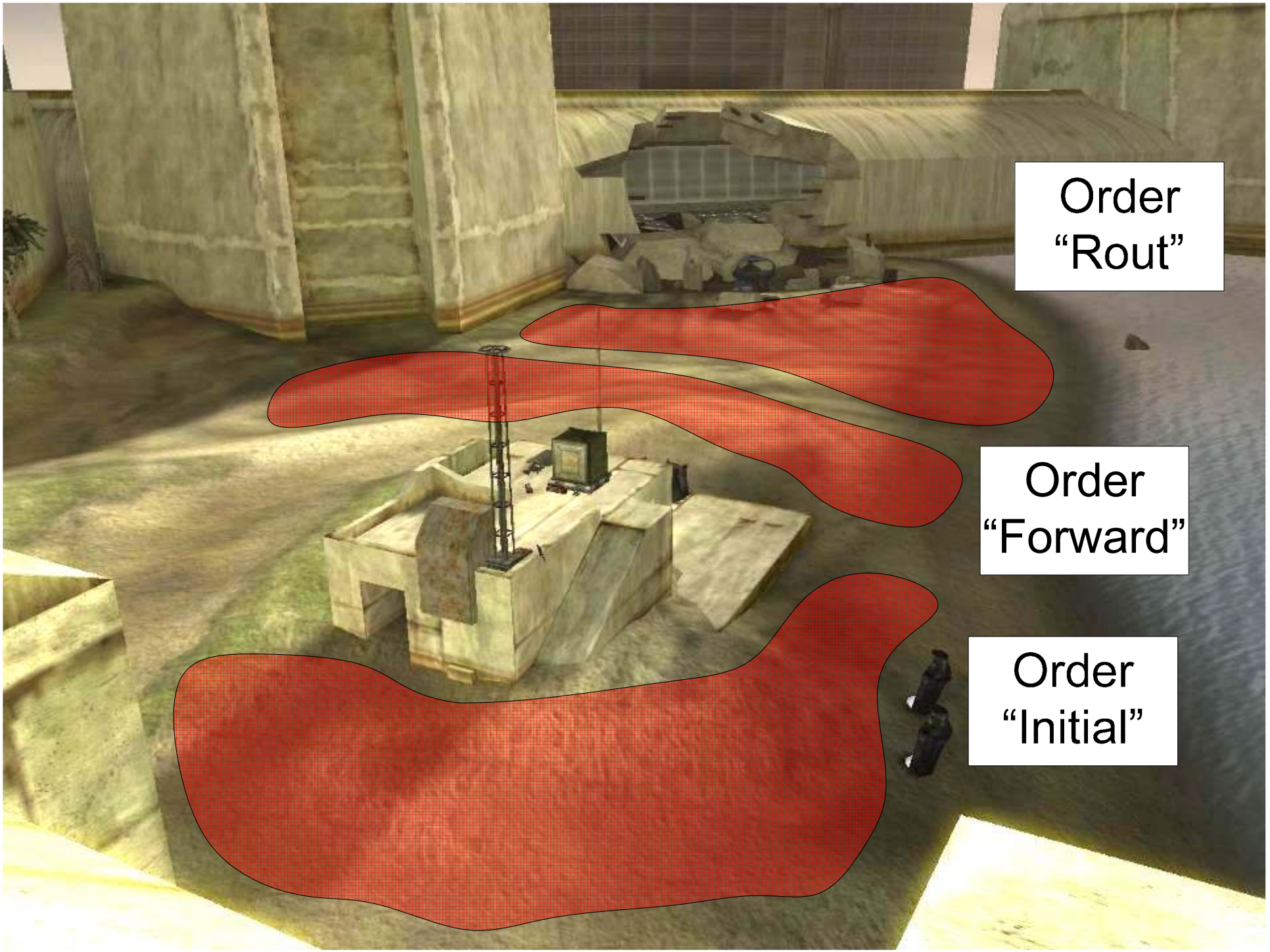
AIのグループ

Zones

エリアの集合

“Orders”

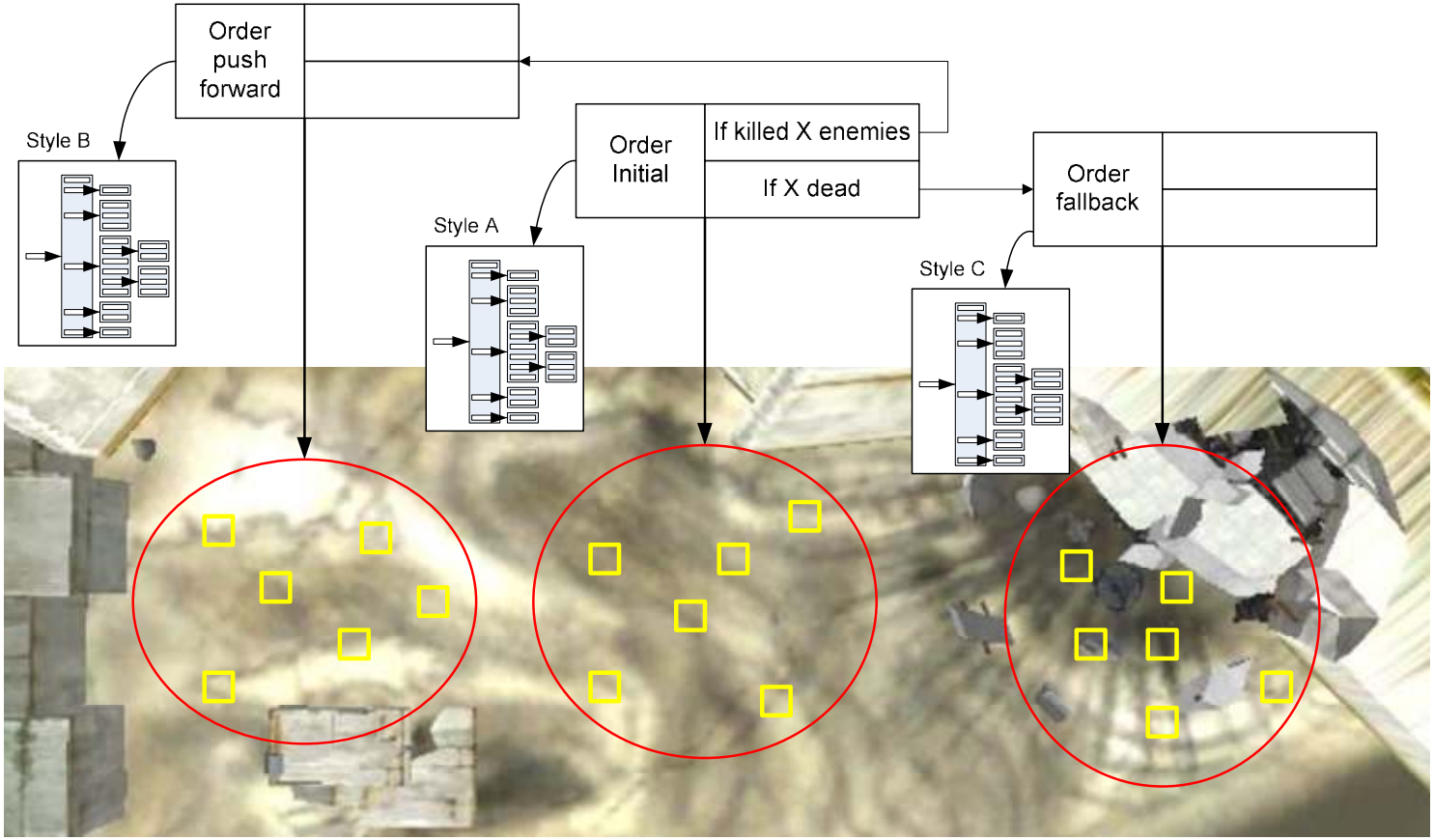
エリア間の移動を指定

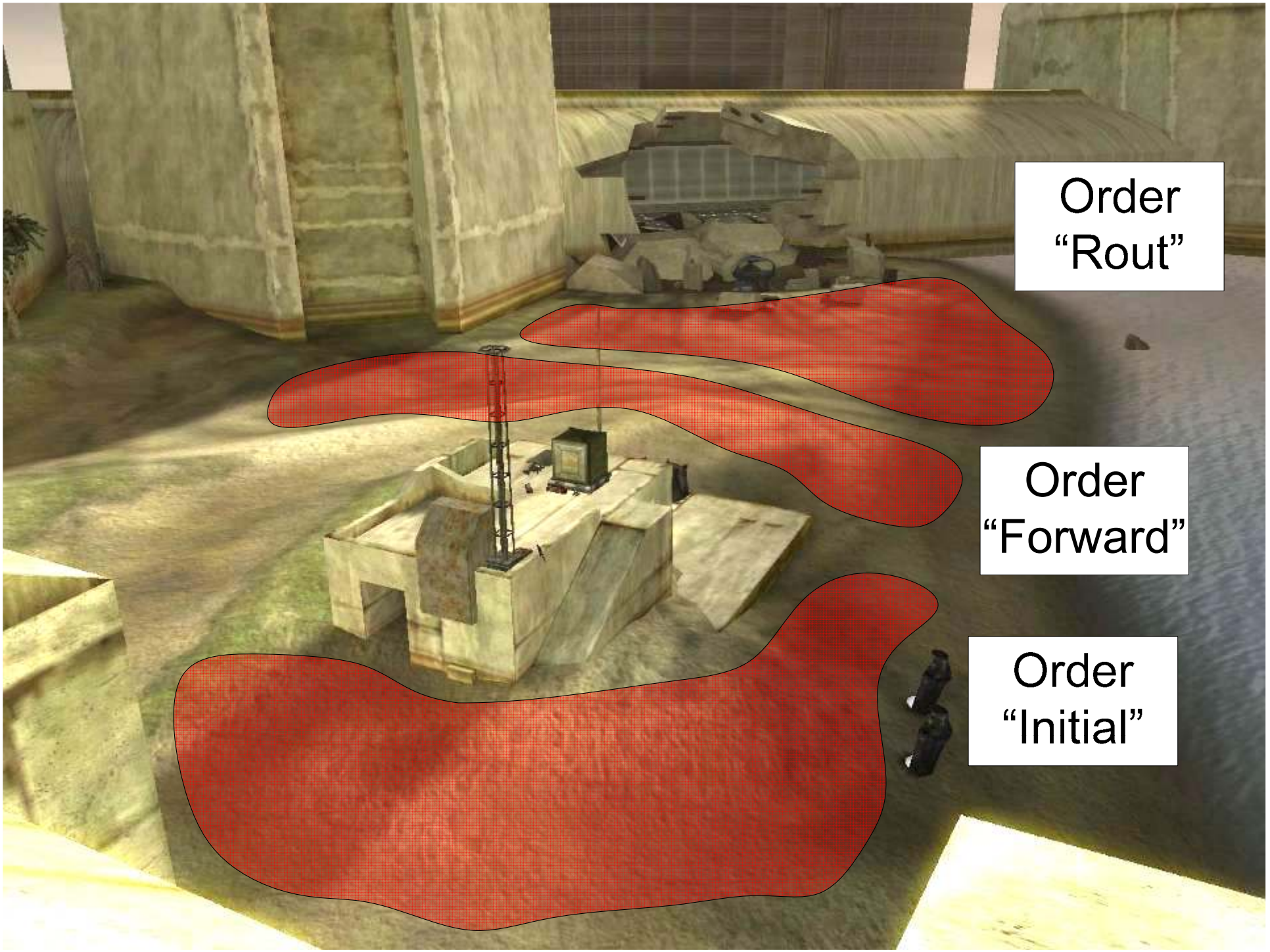

Order
“Rout”
Order
“Forward”
Order
“Initial”

オーダー&スタイル

エリアの遷移を規定

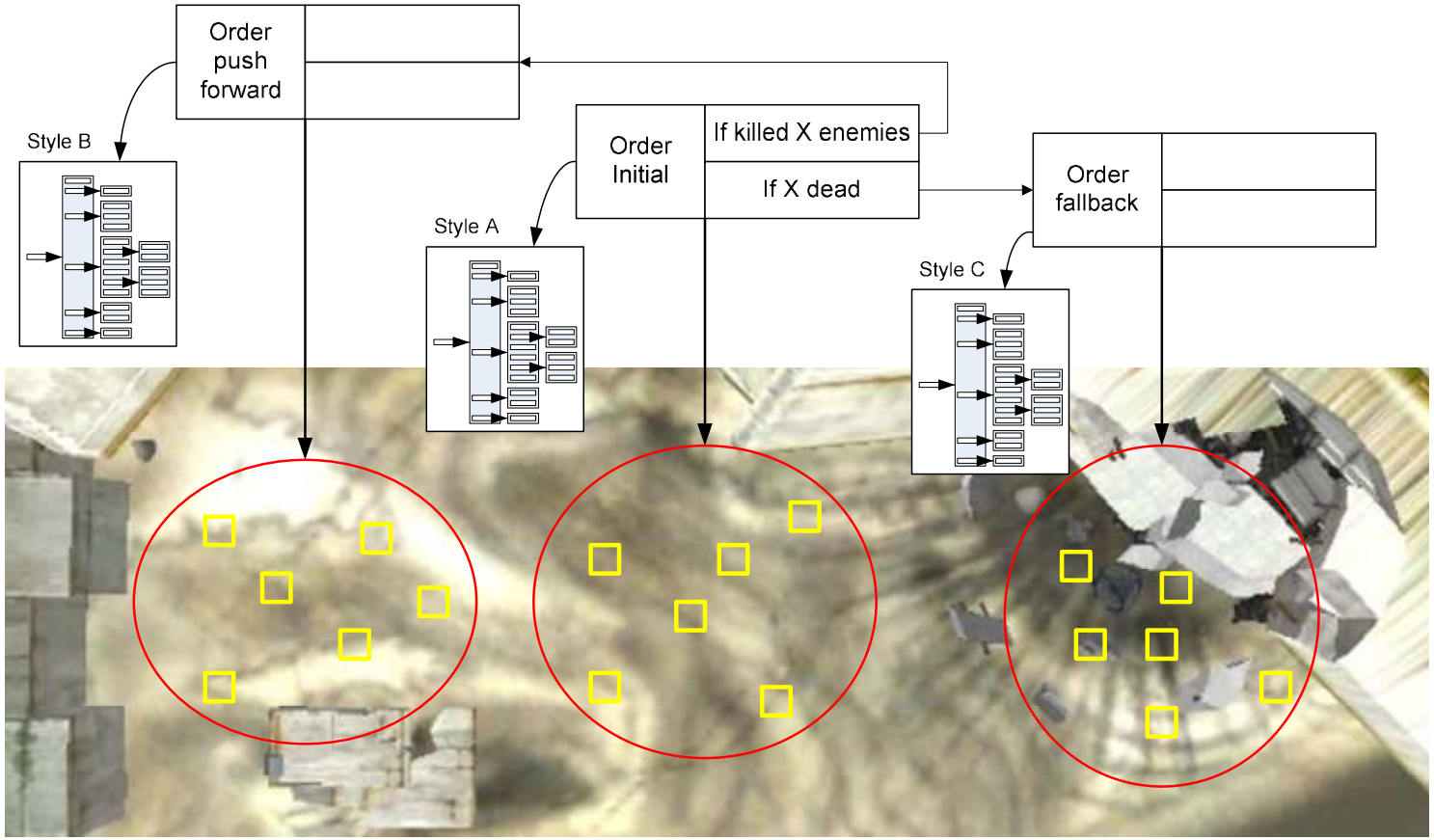
そのゾーンで許されている行動のリスト




Order
“Rout”
Order
“Forward”
Order
“Initial”

オーダー&スタイル

オーダー、スタイル、そして階層構造のデータを使った
スクリプトによって、グループとしてのAIの制御を実現する。



まとめ

（１）Halo2 AI は思考部分においてHaloの思考部分を
一新し、拡張性の高いアーキテクチャーを実現した。

（２）チーム制御では、エリアとメンバーを切り離し、
チーム単位でより自由に移動できるシステムを作った。

（３）プログラマーが仕組みを作った上で、デザイナーにAIの
カスタマイズを任せている。

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)

よりよい戦場を構築するために:
HALO 3 AI オブジェクティブ・システム

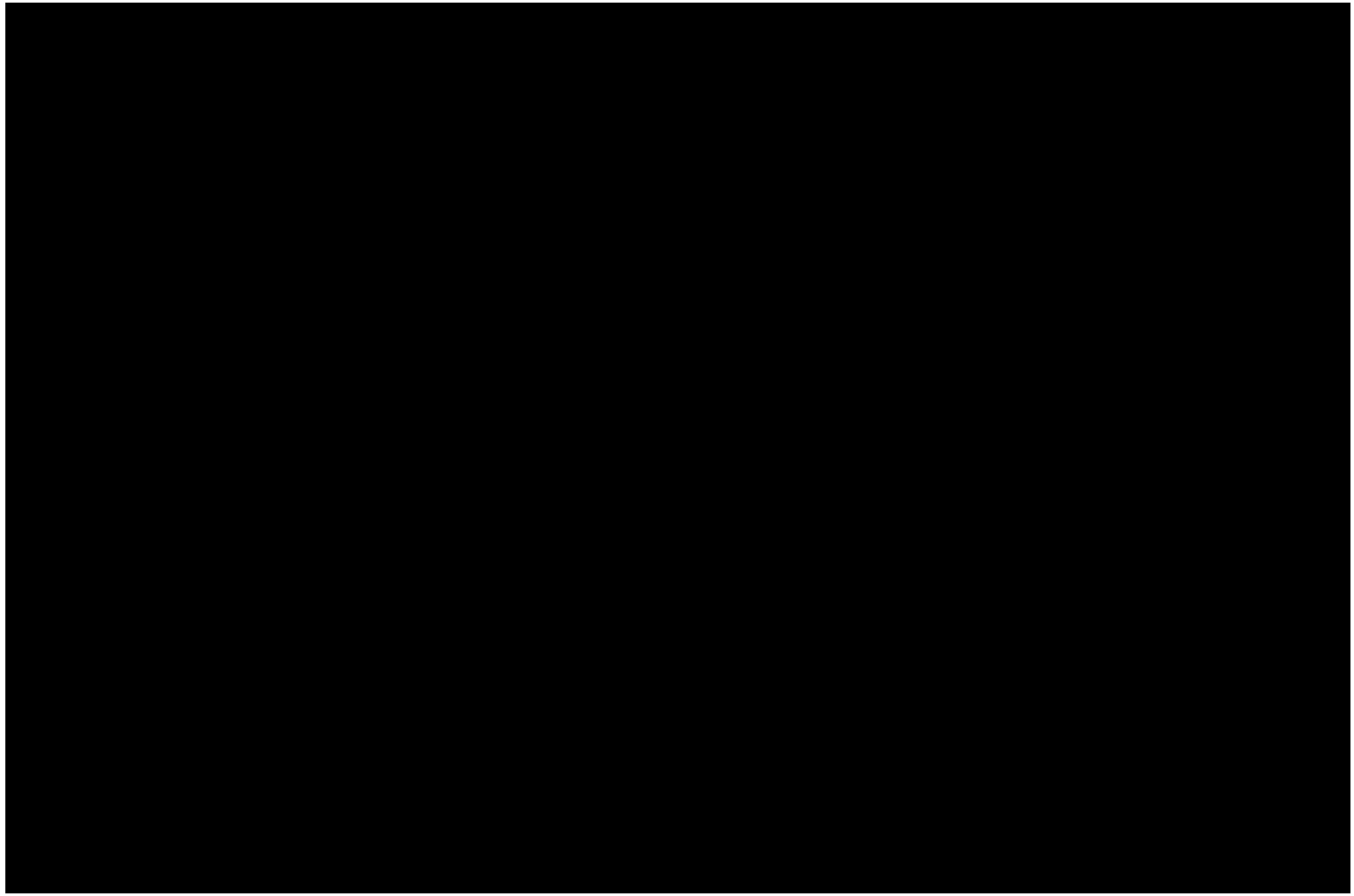
よりよい戦場を構築するために: HALO 3 AI オブジェクティブ・システム
Building a Better Battle: HALO 3 AI Objectives
https://www.cmpevents.com/GD08/a.asp?option=C&V=11&SessID=6863
講演者 : Damian Isla (AI Lead, Bungie Studios)
日時 : 2月22日（金）16時～17時
場所 : West Hall, 3004号室

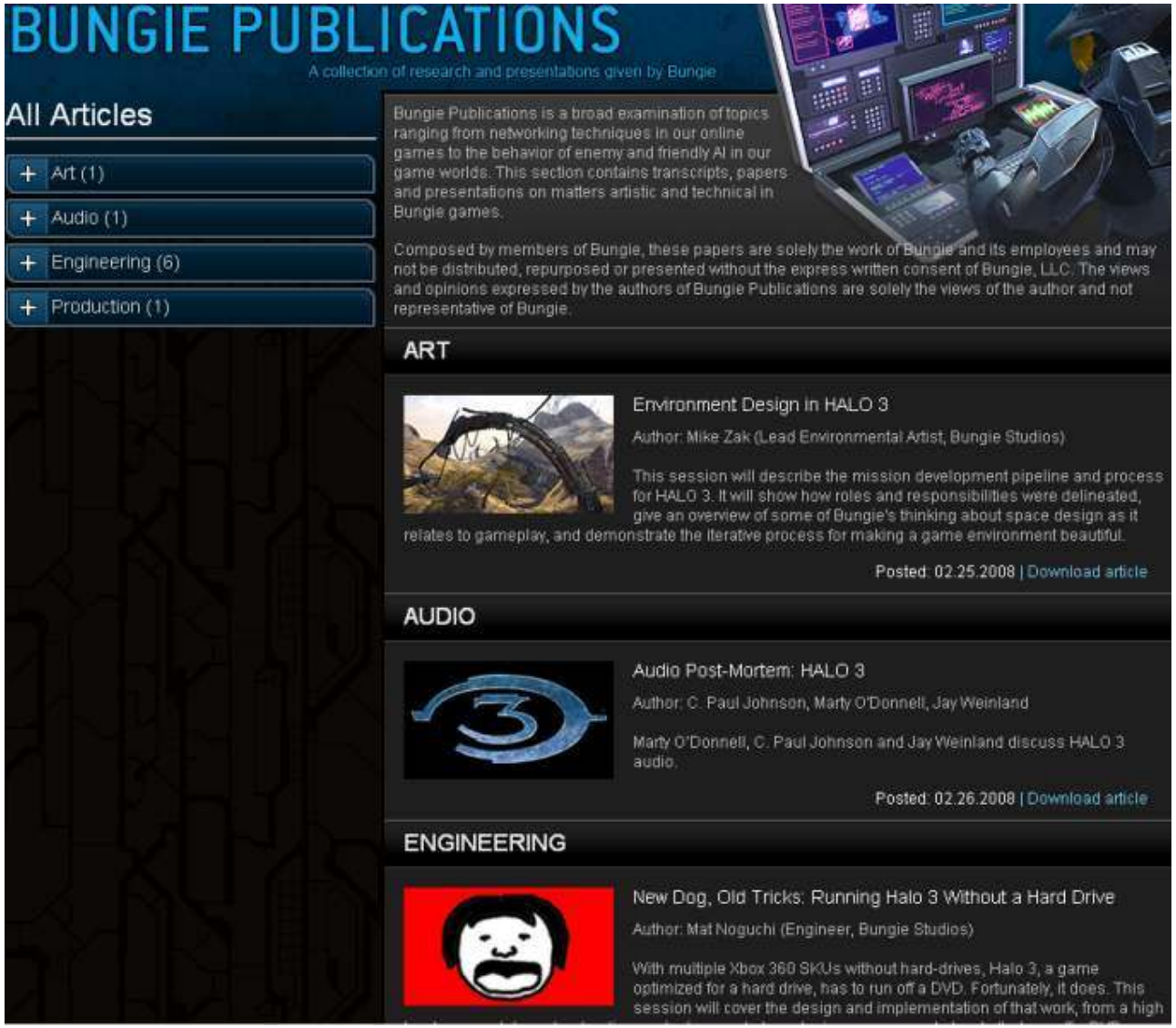
Halo3 戦場の形成

戦略性の高い地形　＋　地形や状況を認識して行動する優れたAI　＝　戦場

Mike Zak, Environment design in Halo3 (GDC2008)
http://www.bungie.net/Inside/publications.aspx

本講演





http://www.bungie.net/Inside/publications.aspx

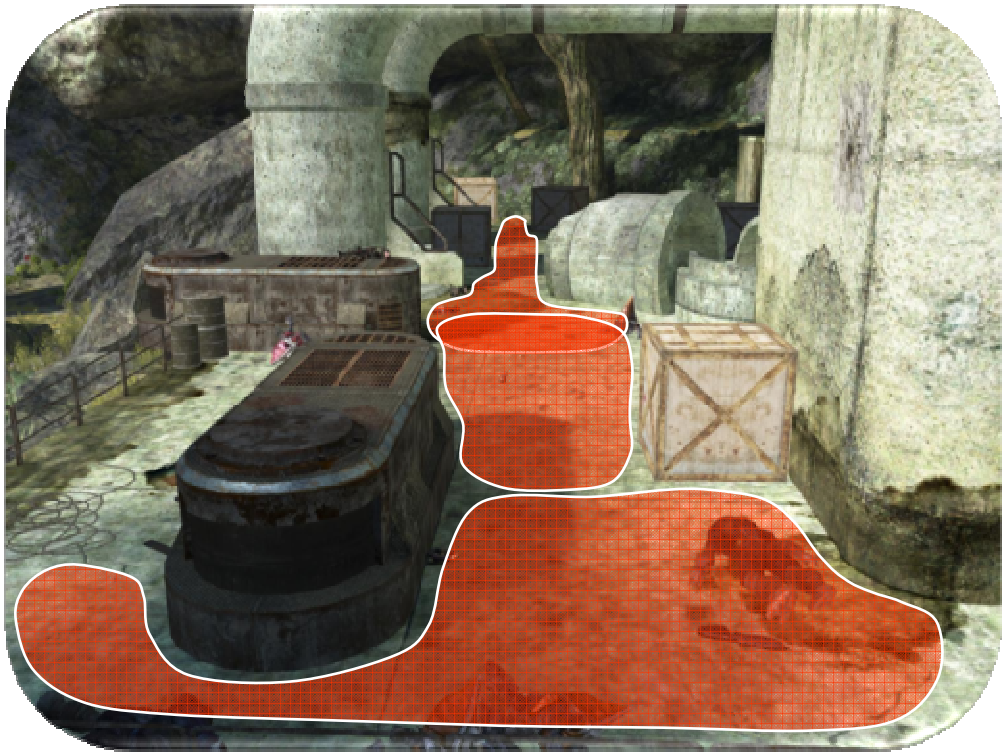
Bungie は今回はなんと全ジャンル9講演！

HaloシリーズのAIに一貫するテーマ

エンカウンター・ロジック　＆　それほど広くない領域での移動的戦闘

攻撃ロジック
- どのような編成チームで
- 何処で（場所）
- どのタイミングで

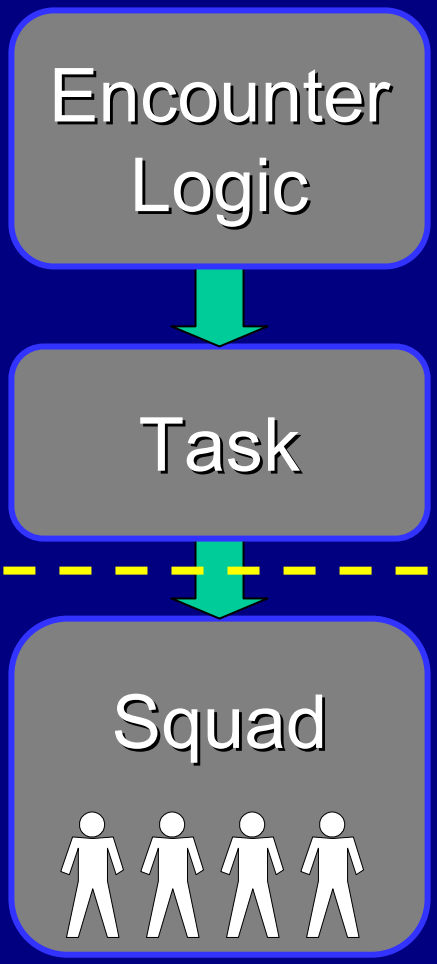
出会うか？



Halo AI の特徴
- テリトリー
- 振る舞い
  - 攻撃的
  - どうぶつかるか？
  - プレイヤーを追撃

Haloというゲームは、そういった領域をAI，環境を
丁寧に作り上げて行くことで質の高い面白さを提供するゲーム（三宅）

Halo3　AIデザインコンセプト

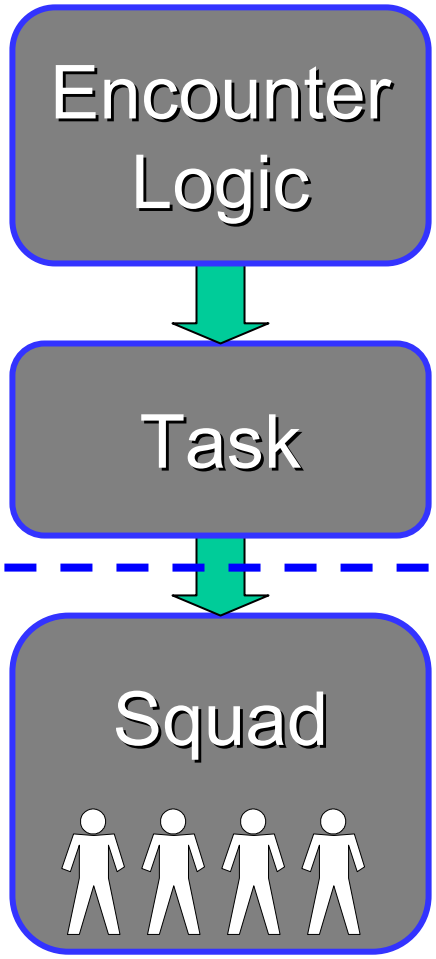


Mission-designers script sequence of tasks

~~Within the task, the AI behaves autonomously~~

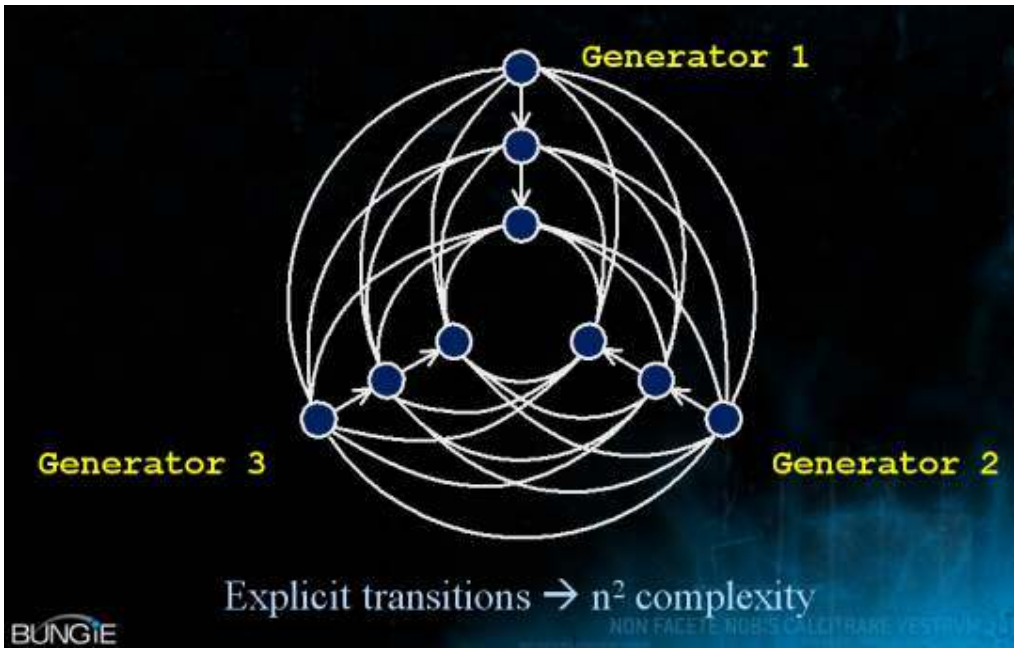
ミッションはデザイナーがスクリプトで書き、Squad にタスクを渡す。
Squadのメンバーはそこから自律的に動かす？　No！

Halo3　AIデザインコンセプト



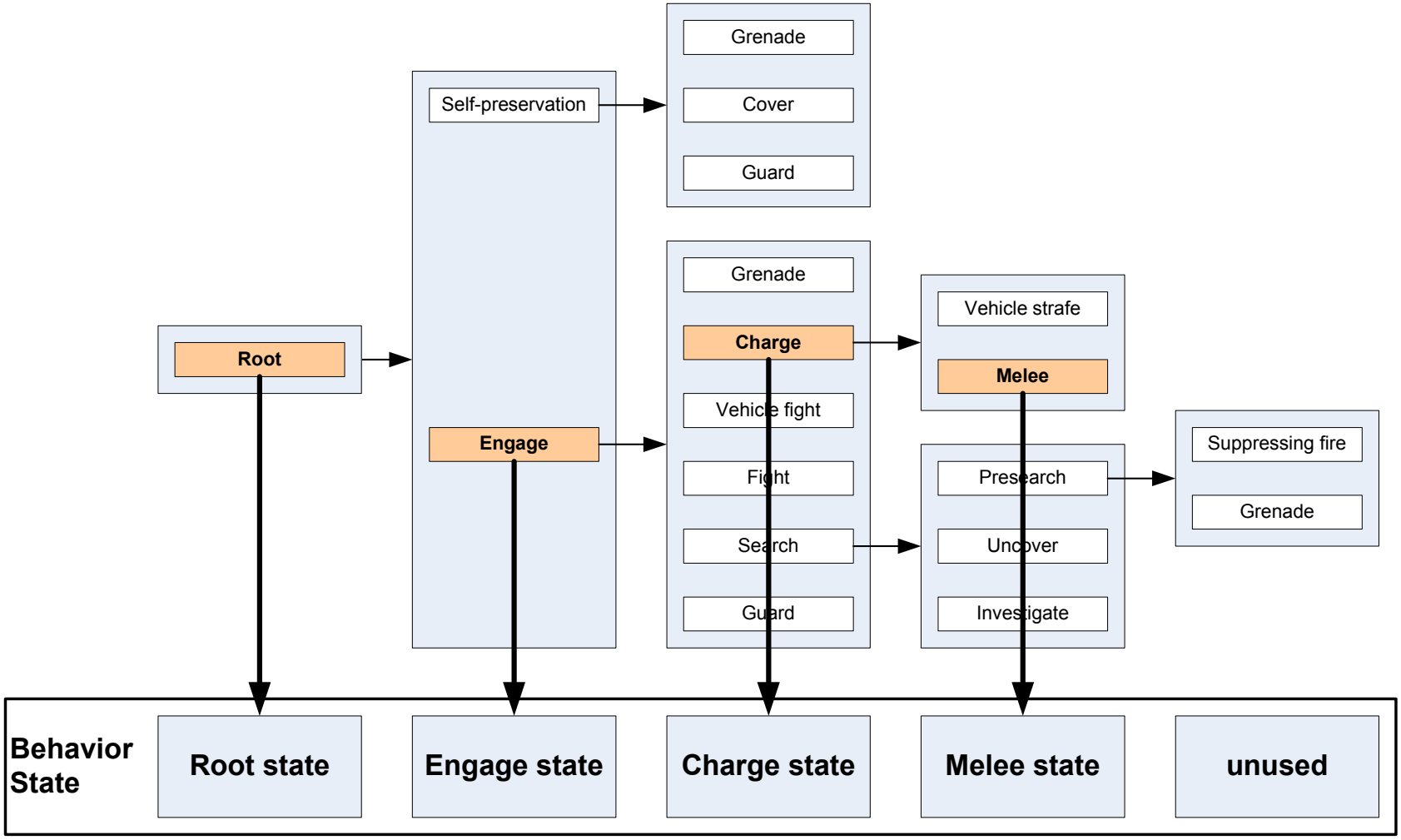
Mission-designers script sequence of tasks

FSMは？
　複雑になりがち…



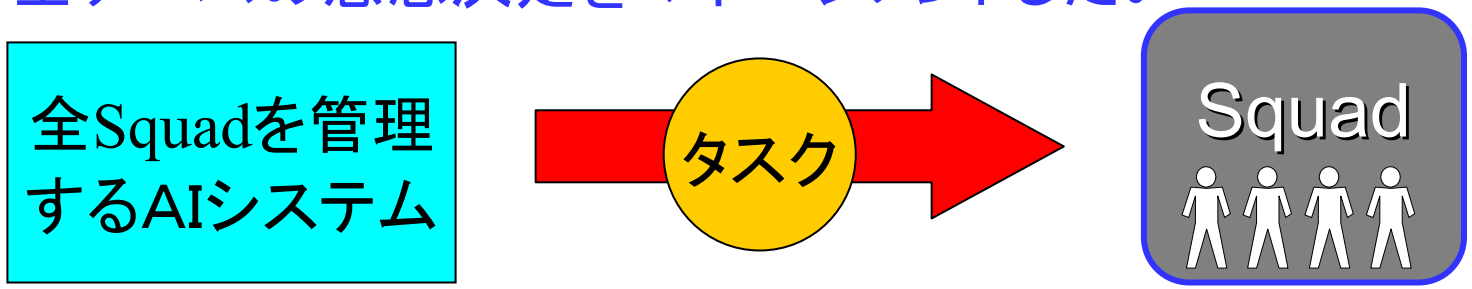
Halo2　AIデザインコンセプト

一体のAIに対して振る舞いを選択する（HFSM）

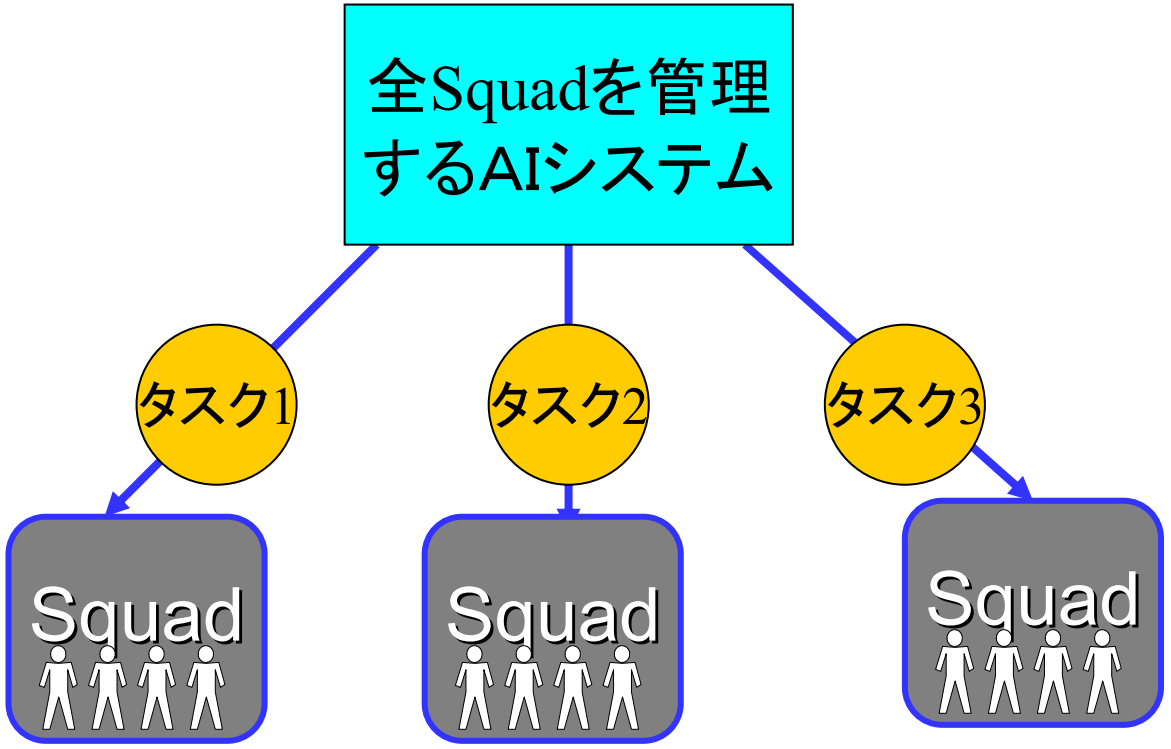


Halo3　AIデザインコンセプト

我々はチームに対して1つの戦略を選択したいわけではない
全チームの意思決定をマネージメントしたい

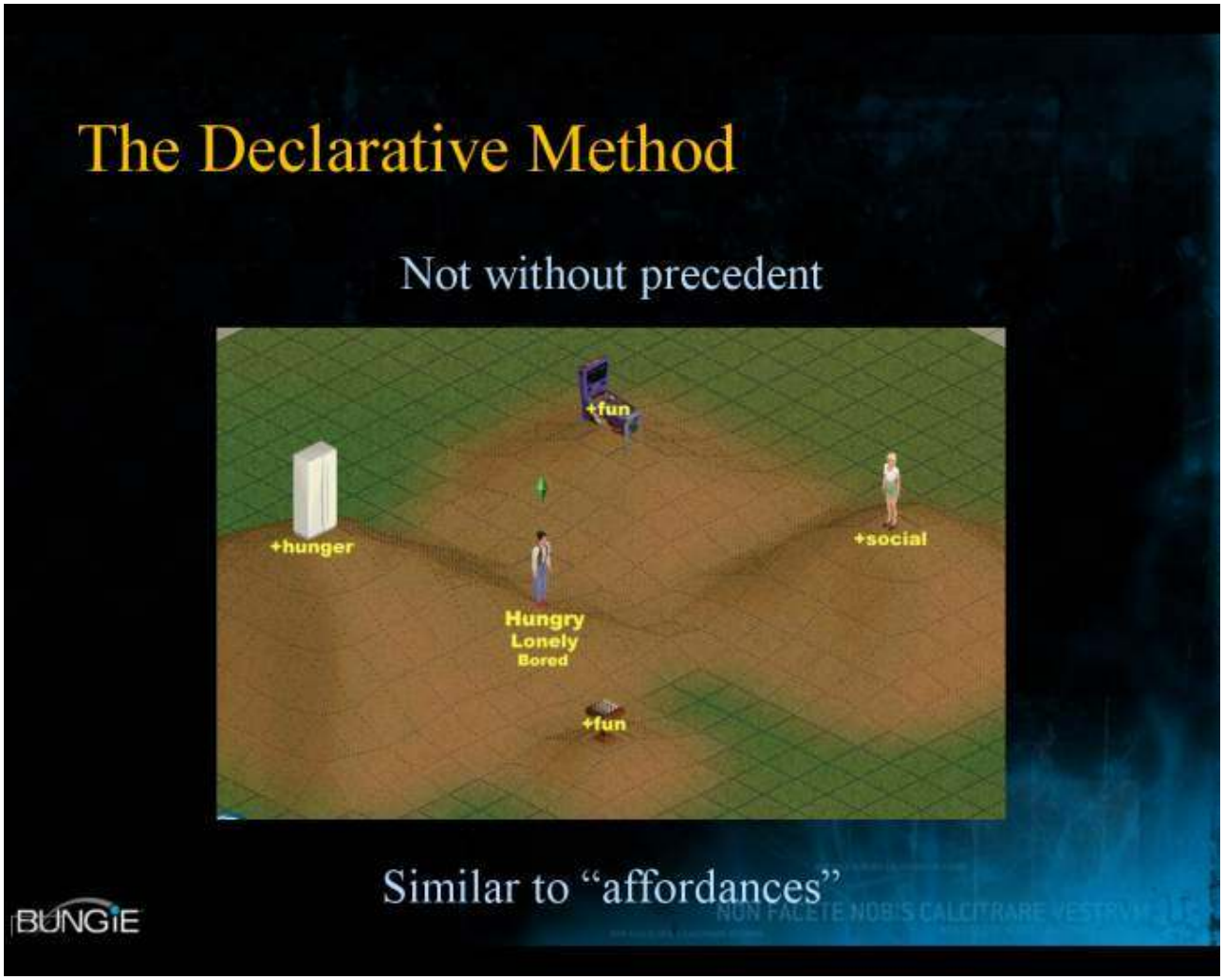


我々は複数のチームに対して戦略を振り分けたいのだ！



Halo3　AIデザインコンセプト

Declarative Method　とは？



「AIが対象（タスク）を選択する方法」ではなく、
「対象（タスク）がAIを受け入れるかどうかを選ぶ方法」

The Declarative Method

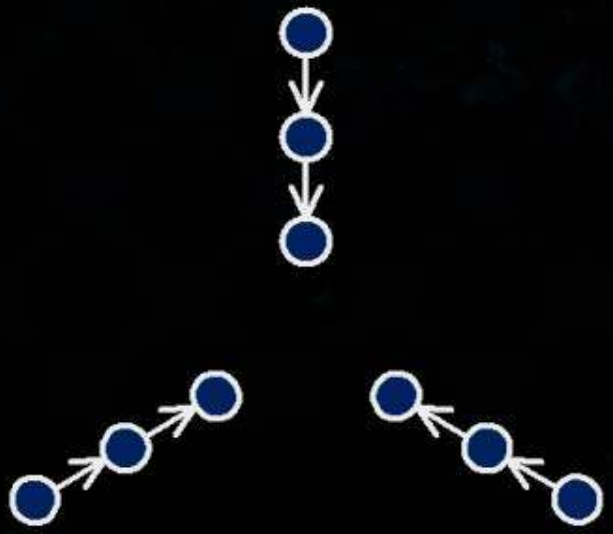
新しいアプローチ:

その環境で為すべきことを定義したタスクを用意せよ

そしてシステムにそのタスクを誰が為すべきかを判断させるのだ

複数のチームに対して戦略を振り分ける仕組み

3 Generators Revisited



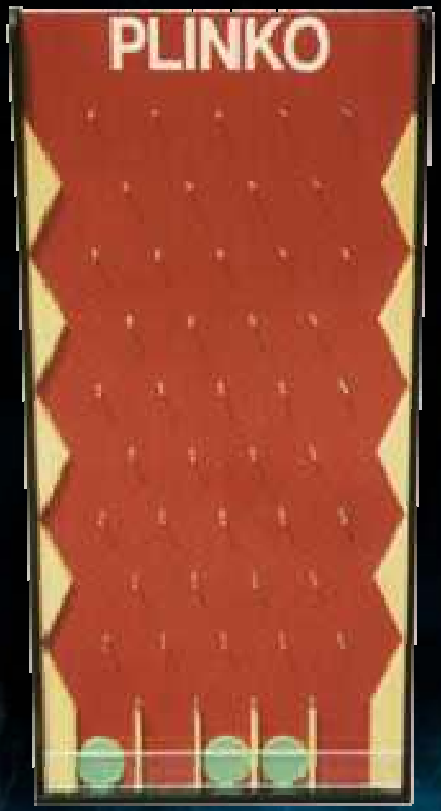
- root
  - g1_group　g1 alive　max 10
    - g1 forward　>75%
    - g1 fallback　>50%
    - g1 laststand
  - g2_group　g2 alive　max 10
    - g2 forward　>75%
    - g2 fallback　>50%
    - g2 laststand
  - g3_group　g3 alive　max10
    - g3 forward　>75%
    - g3 fallback　>50%
    - g3 laststand

タスク

Halo 3 AI Objectives System

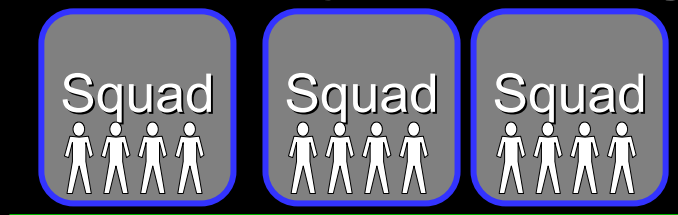
構造
- プライオリティー付きツリー構造
- タスク自身が自分を定義し評価している
  - プライオリティー
  - アクティブ、ノンアクティブの判定
  - キャパシティー（受け入れ条件、人数だけではない）

アルゴリズム
- トップから複数のSquadを流し込む
- その Squadたちが、最も重要なタスクタスクにチェックインさせる。



Basically, it’s a plinko machine
…と言われても。

Halo 3 AI Objectives Systemイメージ



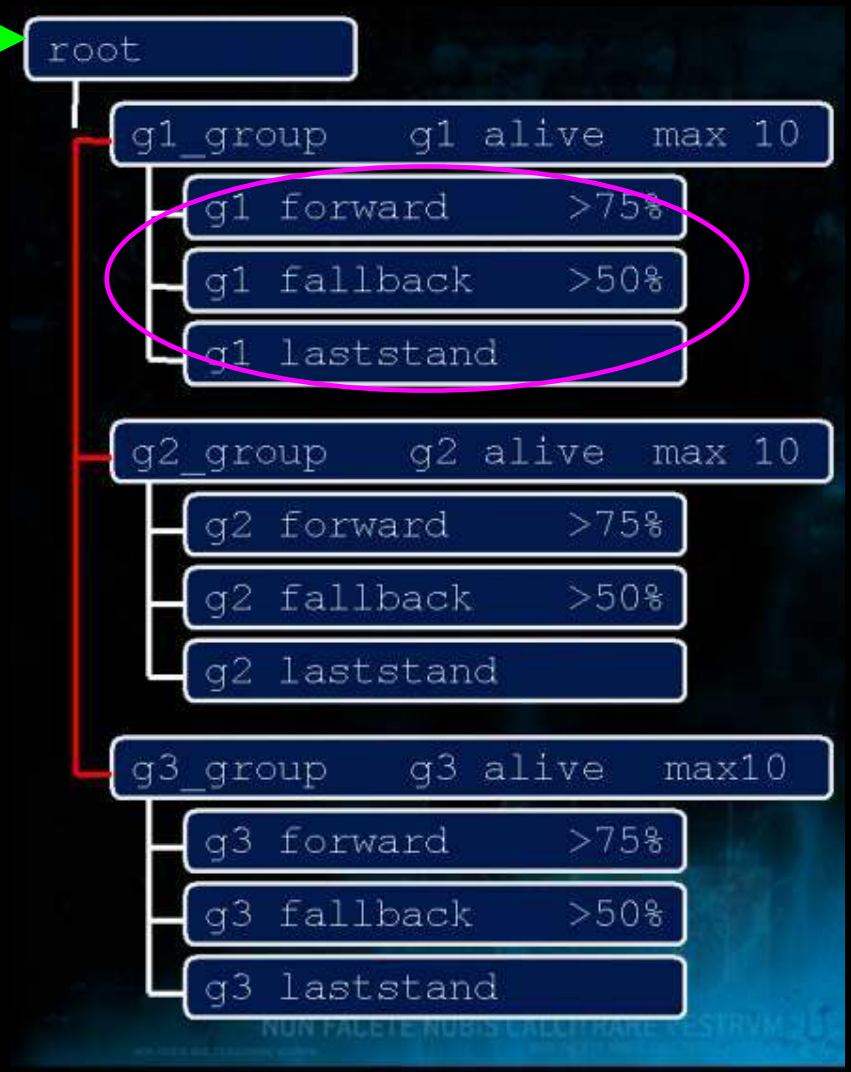
タスクが Squadを選ぶ

各タスクが自分で自分の
オンオフとプライオリティーを計算

トップ(最も高い)プライオリティーの
高いタスクを考慮する

現在、タスクを割り当てられる
Squad をリストする

タスクの受け入れ条件に合うSquad
を割り当てて行く



Halo 3 AI Objectives System 概要

Halo 3 AI Objectives System 概要

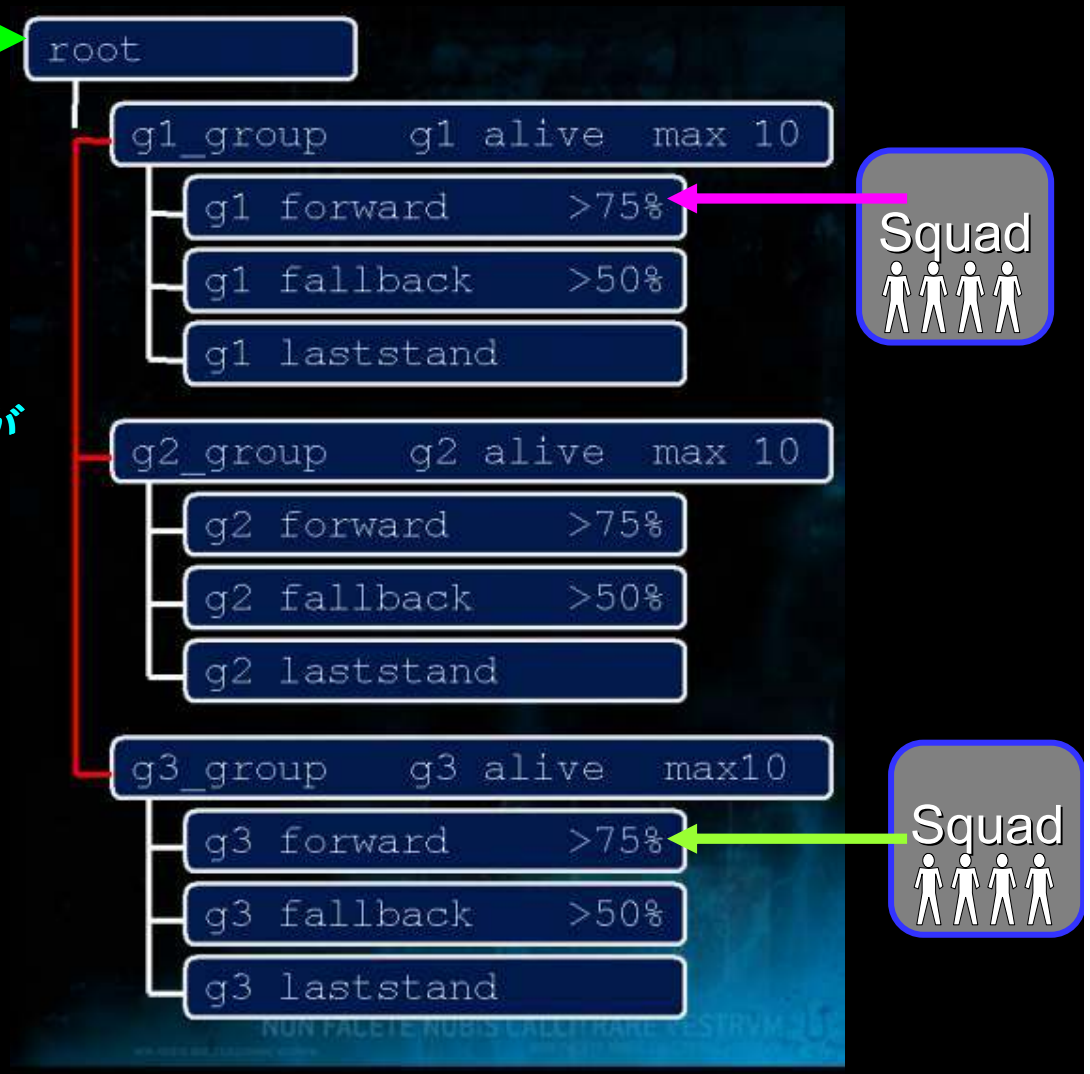
Squad

タスクが Squadを選ぶ

Squadを受け入れたタスクは
受け入れ拒否になる。

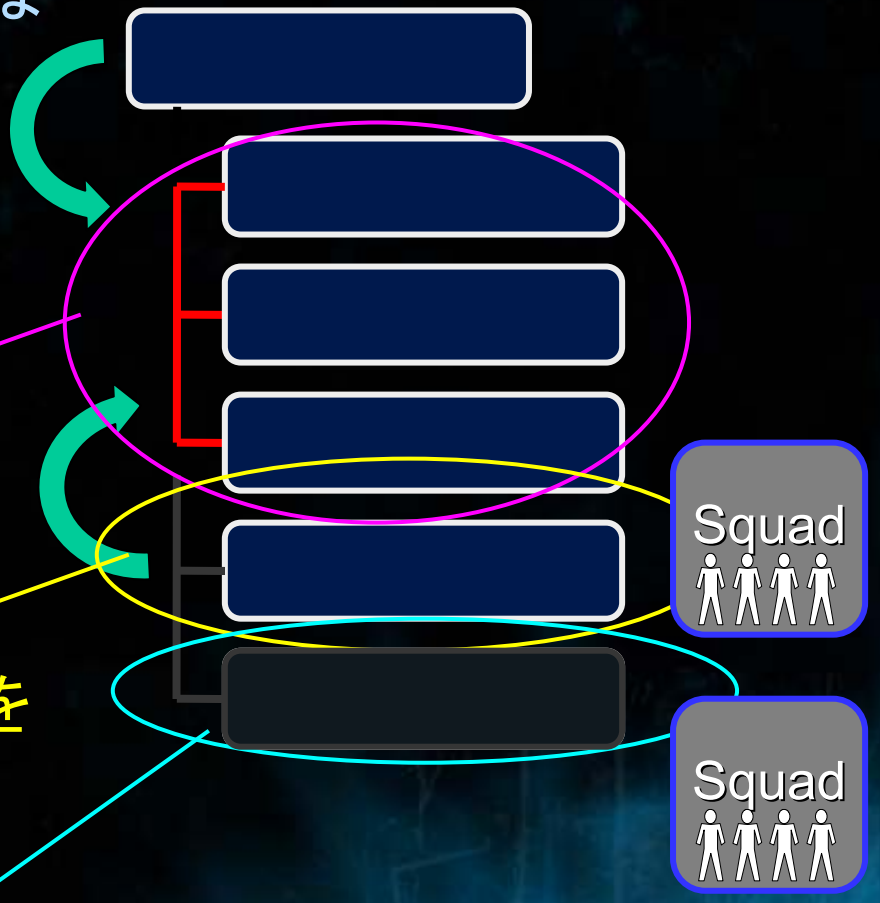
残りのタスクの中で
最もプライオリティーの高いタスクが
受け入れるSquadを選択する

後は同じ



きちんとしたアルゴリズムの説明

- アクティブな子タスクを決定する
  - この時、非アクティブになったタスクに割り当てられていたSquadは親タスクに一旦引き戻される

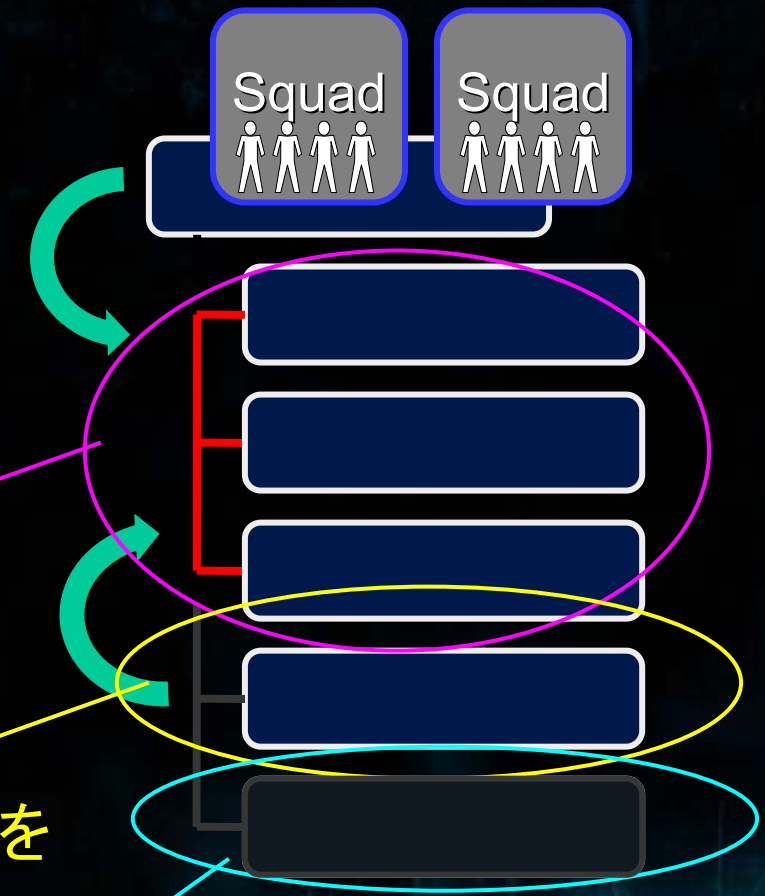


きちんとしたアルゴリズムの説明

①アクティブな子タスクを決定する

ーこの時、非アクティブになったタスクに
割り当てられていたSquadは
親タスクに一旦引き戻される

ートッププライオリティーより低いプライ
オリティーに従事しているSquadも
再割り当ての対象となる。



トッププライオリティーを
持つタスクたち

次に高いプライオリティーを
持つタスクたち

非アクティブになったタスク

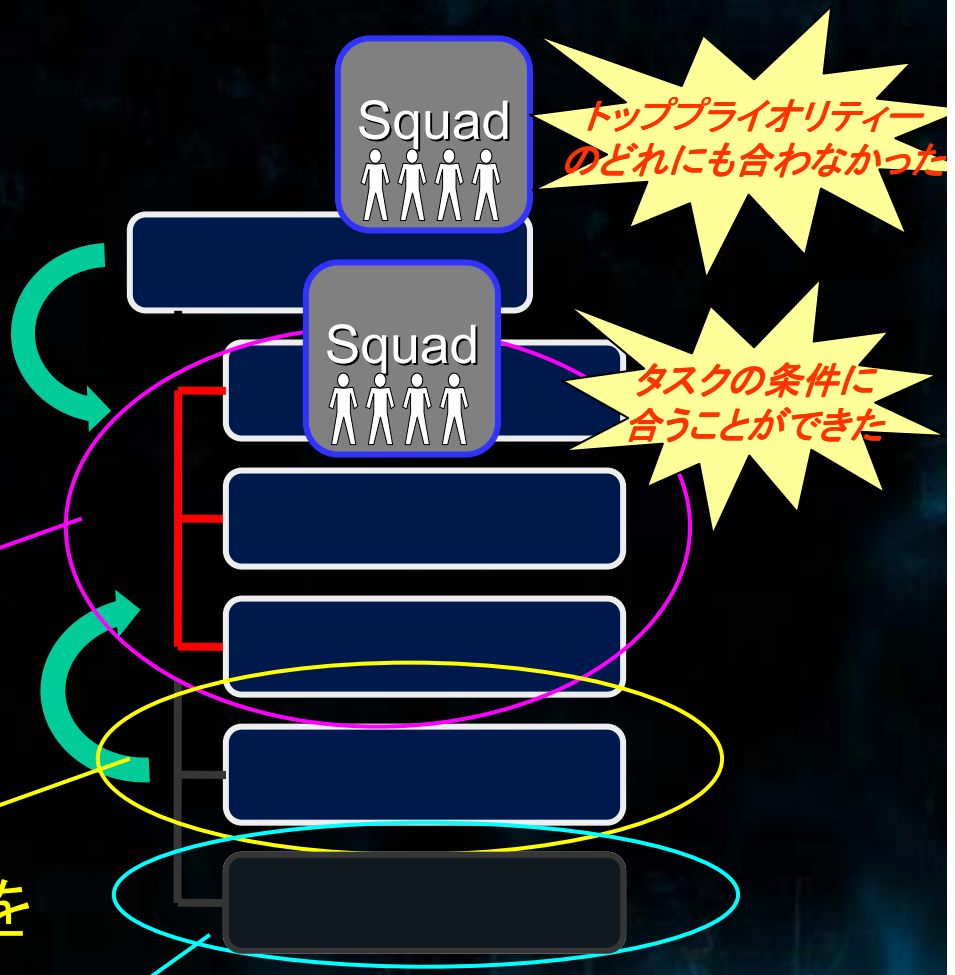
きちんとしたアルゴリズムの説明

②最高プライオリティーを持つグループを考える

③割り当て可能なSquadを集める

- 親のノードの登録されているSquad
- 上記のプライオリティーより低いタスクを実行しているSquad

④Squadをタスクに割り当てる



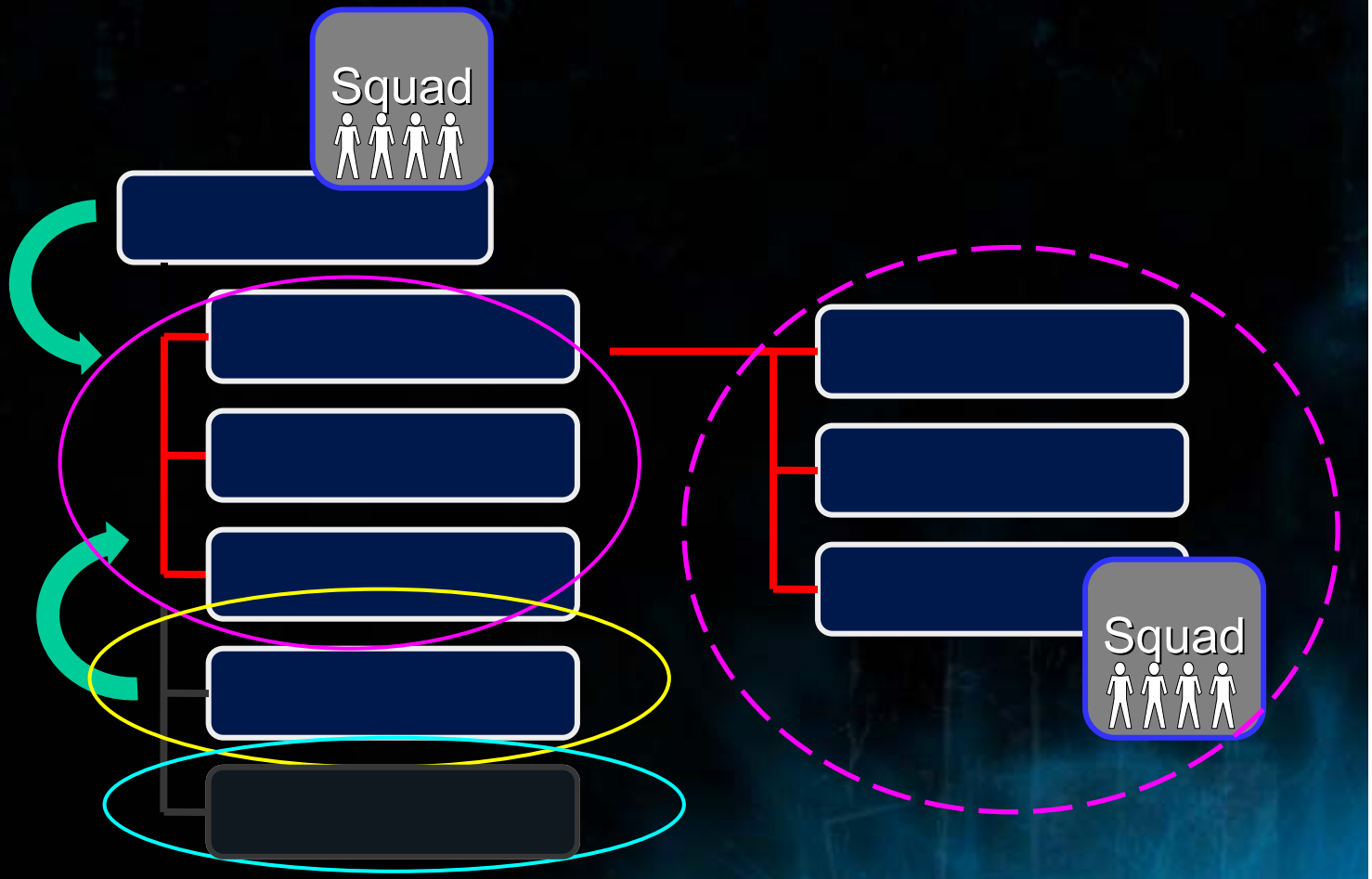
トッププライオリティーを
持つタスクたち

次に高いプライオリティーを
持つタスクたち

非アクティブになったタスク

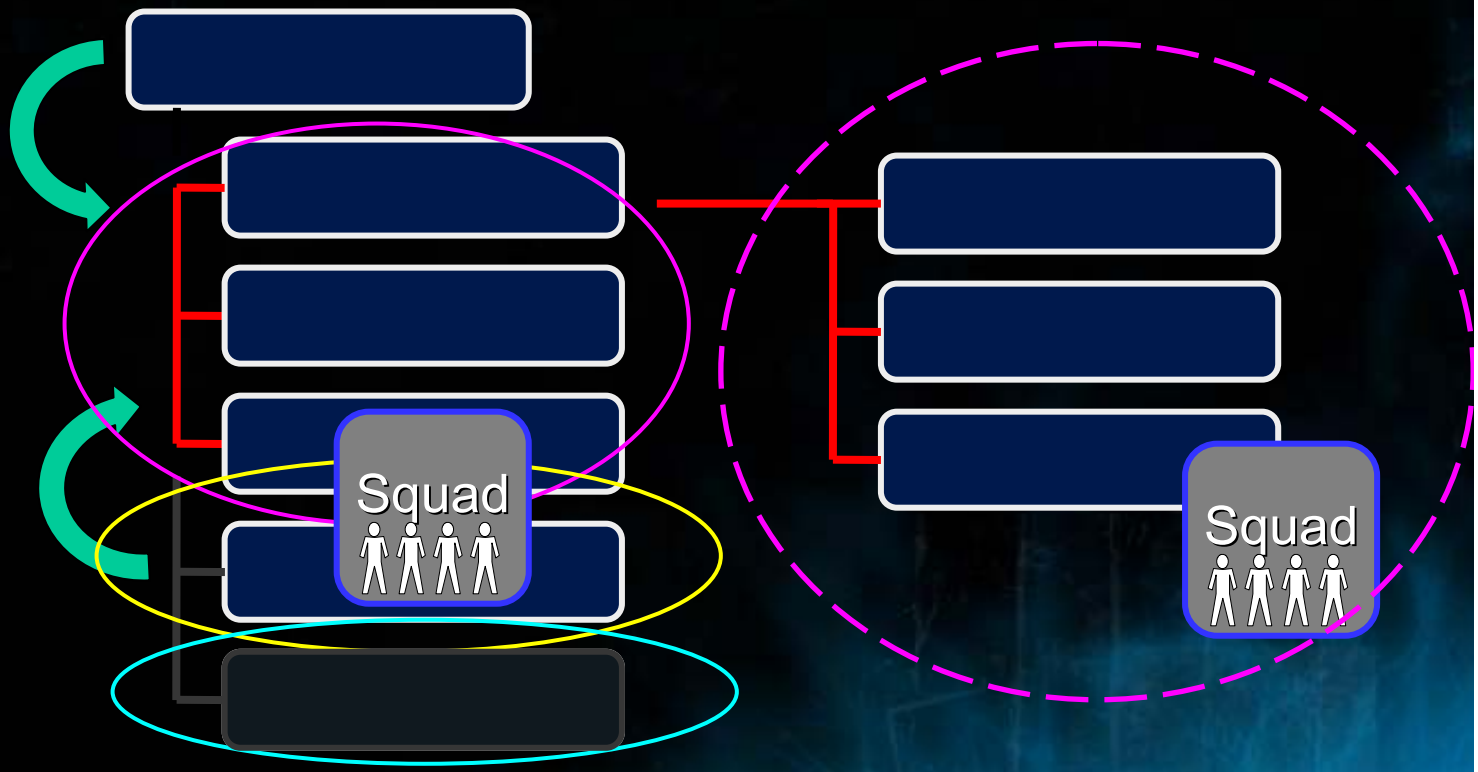
きちんとしたアルゴリズムの説明

⑤トッププライオリティーのタスクが、さらに下にタスク構造を持つなら、そのさらに子タスクの中にSquadを割り当てていく。

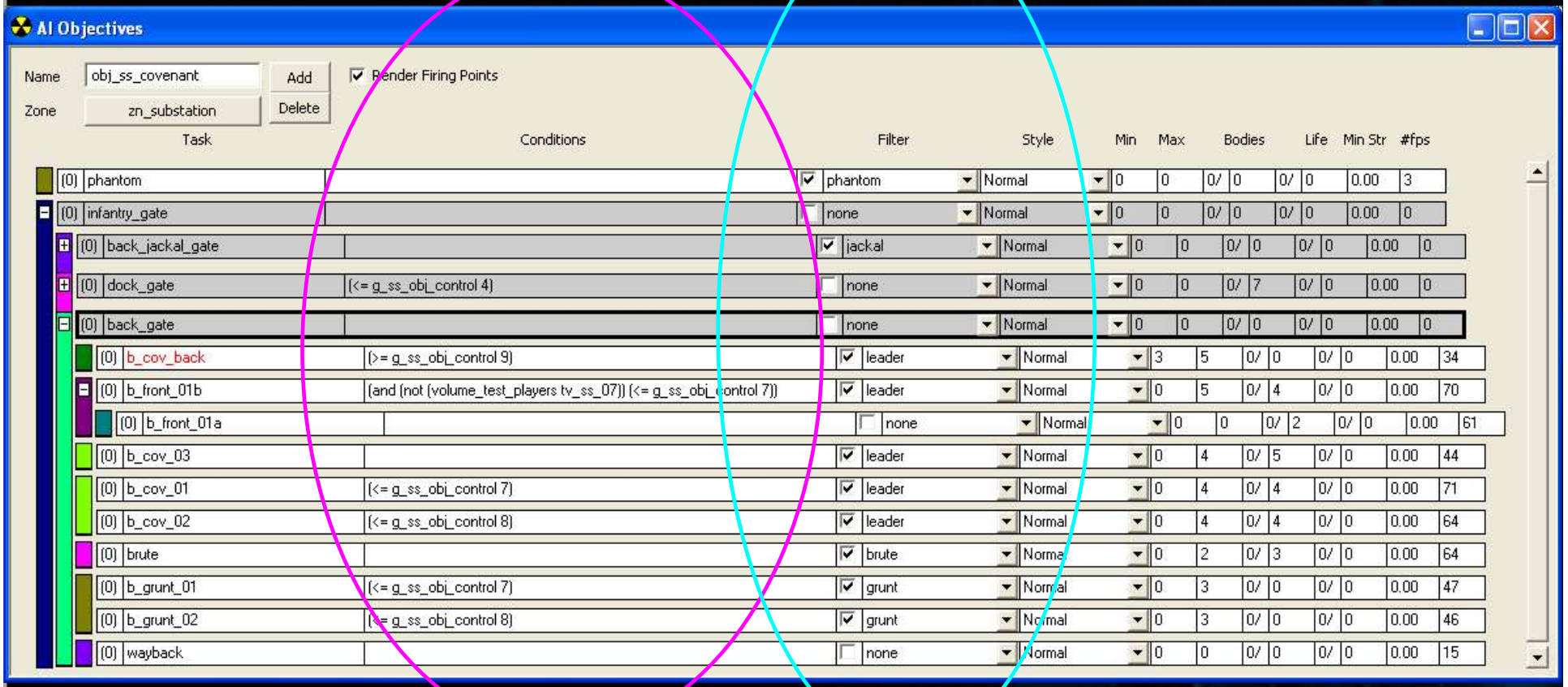


きちんとしたアルゴリズムの説明

⑤次に高いプライオリティーの高いタスクにSquadを割り当てる。
⑥これをくり返す



Designer UI



Active or inactive

種族によるフィルター（受け入れ条件）

Squad 割り当て問題

結局、この問題はn個のsquadsをm個タスクに割り当てる問題

- set ***S*** of ***n*** squads
- set ***T*** of ***m*** tasks

Now, find a mapping $F(S) \rightarrow T$

２つの問題:

1. タスクキャパシティーの制約条件問題
2. コスト最小問題 *H(F)*

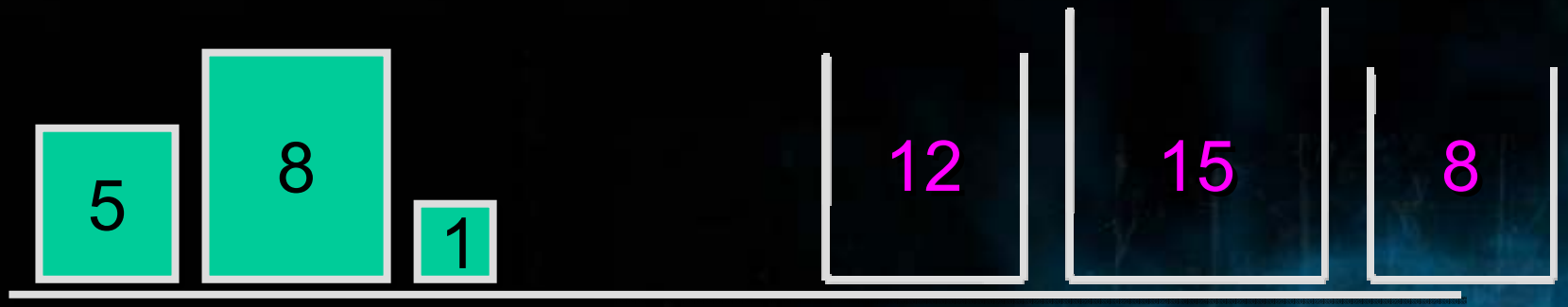
（注）コスト関数Hとは割り当てFが1つ決まったときに、
その割り当てにおいてSquadが行動するときに
かかるコスト（移動時間、戦闘負荷など）の総和を表す
即ち、H(F)が大きい割り当てはよくない割り当てである。

Squad 割り当て問題

1. タスクキャパシティーの制約条件問題

 # タスクに割り当てる Squads t ≤ capacity(t)

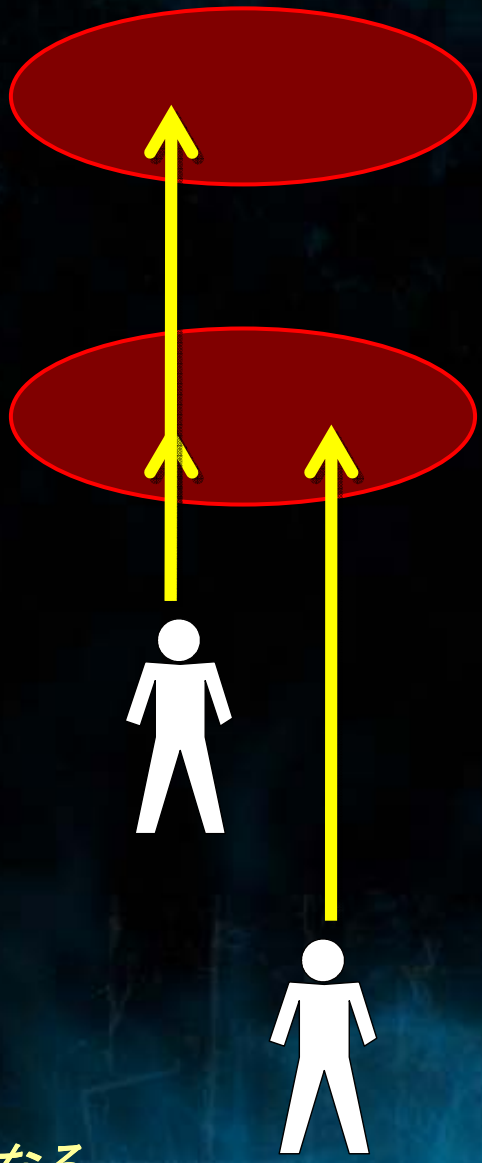
... but remember, we're bucketing by squads.



Squad 割り当て問題

2. コスト関数　*H(F)*　を最小にする

コスト関数を最小にすると
いうことは、全体のSquad
割り当てを最適化すること。



考慮事項

- 移動距離を長くしない
- *質の高い行動*
- *ツリーのバランスを取る*
- *プレイヤーの近づける*

よくないH(F)関数だとまるでまぬけなことになる。
よいH(F)を見つけることが割り当てにおいてとっても大切！

「よりよい戦場を構築するために:
HALO 3 AI オブジェクティブ・システム」
の教訓

Haloは各分野が丁寧に作り上げられている（全9講演）。
Haloは「洋ゲー」としてはとても親切な作り。

AIは、ゲーム的混沌を最後に引き受ける。
AI技術とはその情報を処理して行動を生成する技術。
そこには、Damian Islaの作家性とも言うべき個性が現れている。

ゲームAIでは、設計者の作家性さえ表現され得る。
そのためには人工知能に対する基本的な知識が必要。

（ゲームAIではなく、人工知能そのものの考え方。初歩的なところから考えて、
まとまりのある面白いシステムを作るのが、Damianの持ち味）

参考文献

①Halo3　9講演のPPT資料　http://www.bungie.net/Inside/publications.aspx

②　Bugie.net のGDC2008のHalo3関係の講演の概要集
http://www.bungie.net/News/content.aspx?type=topnews&cid=13264

③　Bungie Podcast (毎回、Bungie の開発者が登場してインタビューに答えます)
http://www.bungie.net/Inside/content.aspx?link=bungiepodcasttime

The Bungie Podcast: 08/16/07 (Halo3 AI の特集です。Damian がAIについて語ります！)
http://www.bungie.net/News/content.aspx?type=topnews&cid=12705

④Bungie’s Halo hordes a huge jump forward(Sydney Morning Herald
http://www.smh.com.au/news/technology/bungies-halo-hordes-a-huge-jump-forward/2007/09/24/1190486224797.html

⑤ゲームAI連続セミナー第４回「Halo2 におけるHFSM(階層型有限状態マシン)
http://www.igda.jp/modules/eguide/event.php?eid=42

⑥ Artificial Intelligence and Interactive Digital Entertainment 2005（講演PPT）
http://www.aiide.org/aiide2005/talks/index.html

課題

自分でゲームを一つデザインして、
そのゲームのAIを設計せよ。

ご清聴ありがとうございました。

これ以外に、意見や質問があれば、メイルかアンケートへ

y.m.4160@gmail.com

(IGDA Japan登録アドレス　yoichi-m@pk9.so-net.ne.jp　)

http://www.igda.jp