博士論文

デジタルゲームにおける

人工知能の動的連携モデルとその実装

三宅陽一郎

本 論 文 の 要 旨

本論文の目的は，デジタルゲームの新しい人工知能モデル「MCS-AI動的連携モデル」（Meta-Character-Spatial AI Dynamic Cooperative Model）を提案し，このモデルを実際のゲーム開発に導入することによって，ゲームAI技術の向上，ゲームデザインの向上，ゲーム開発効率の向上の３つがもたらされることを検証することである．

本論文はまず本モデルの背景となる知識と，これまでのゲームAIシステムの問題点を提示し，その解決策として本モデルを提案する．そして，本モデルの各要素のAIの構造，役割，機能を述べる．さらに，本モデルを既存のゲームへ適用することで，様々なゲームを解析し，対応する技術を抽出する．また本モデルがゲームデザインに関してどのような変化もたらすかを検証するために，ゲーム開発者に対する自由回答文のアンケートによる比較検証実験を行い，本モデルがゲームデザインの質を変化させることを検証する．また，本モデルを大型ゲーム『FINAL FANTASY XV』（スクウェア・エニックス，2016年）のAIシステムの基礎設計として導入した結果と効果について報告する．最後に，本モデルのこれからの研究課題を示す．

以下，本論文の内容に沿って述べる．デジタルゲームの人工知能は，ゲーム内で用いられる人工知能と，ゲーム開発において用いられる人工知能に分類されるが，ここで述べる人工知能は自律型人工知能と機能型人工知能を含む総称である．ゲーム内で用いられる人工知能は「メタAI」「キャラクターAI」「スパーシャルAI」の３つの自律型人工知能に分類される．

「メタAI」はゲーム全体を俯瞰的にコントロールするAIであり，ゲームのあらゆる要素をコントロールする．ゲームの大局的な状況変化を認識し，ゲームに影響を与えることでゲームの流れを形成する役割を持つ．「キャラクターAI」はNPC（ノン・プレイヤー・キャラクター）の意思決定を行う頭脳として，周囲の状況を知覚し，知覚から得た情報によって認識を形成し，その認識の上に意思決定行い，意思決定に従って動作を形成する．「スパーシャルAI」は「メタAI」「キャラクターAI」の複雑な地形・空間の認識や状況認識をサポートするため，地形解析・状況解析を事前に，かつゲーム動作中に動的に行い，地形・空間・状況を抽象化した知識表現として提供するAIである．また逆にスパーシャルAIから自律的にメタAIへの情報の伝達を行い，キャラクターAIへ自律的に制御をかけることもある．

この３つのAIが動的に連携する仕組みが「MCS-AI動的連携モデル」である．それぞれのAIが自律的に動作しながら，コミュニケーションとデータの受け渡しによって動的協調を行い，各AIが持つ機能が組み合わされることによって，さまざまなゲームデザインの要求を実現し，ゲームユーザーに対して多様なゲーム体験を生み出す．

メタAIは，ゲームへ作用するポイントとして「キャラクター操作」「ゲーム全体を操

作」「コンテンツ生成」がある．メタAIの作用によって，ゲームが変化し，作用による効果が生み出されることを示す．キャラクターAIは「エージェントアーキテクチャ」と「ブラックボードアーキテクチャ」，そしてキャラクターAIで最もよく使用される７つの意思決定モデルを整理し，それぞれの意思決定モデルとメタAIの関係性を説明することで，意思決定アルゴリズムへのメタAIの作用が，キャラクターAIにどのような変化をもたらすかを示す．スパーシャルAIについては「パス検索」「位置解析技術」「影響マップ」など空間解析・状況解析技術を述べ，これらの解析情報が，メタAI，キャラクターAIの認識形成，意思決定，行動形成をどのようにサポートしているかを示す．

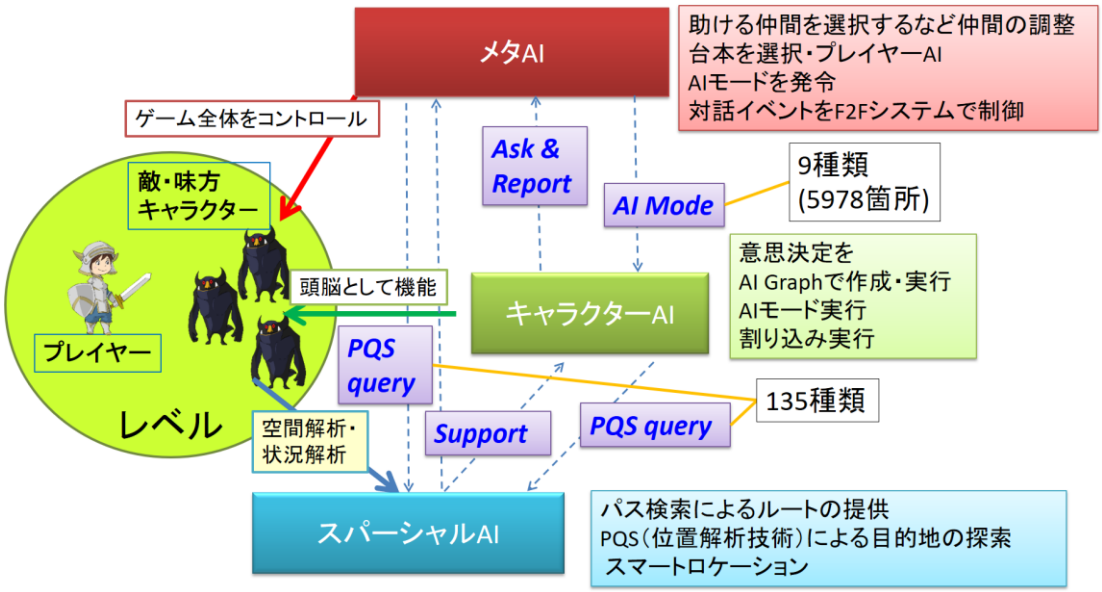
「MCS-AI動的連携モデル」はゲームAI開発の体制とコストにも影響を与える．必要と想定される各AIに属するAIモジュールの開発や，各モジュールを作成するためのツールの作成を行うことで，AI開発を実際のゲームデザインやゲーム製作が始まる以前から始めることができる．またモジュールを開発する開発者と，それを用いて実際のコンテンツを製作する開発者に仕事を分割することが可能となる．さらに初期のフレームワーク，ツール制作のコストを別にすれば，各シーンにスクリプトを準備するコストに比べて，実際のコンテンツにおける開発効率は数十倍から数百倍の効率となる根拠を示す．

「MCS-AI 動的連携モデル」から既存のゲーム捉え直すことで，各 AI における機能やアルゴリズムを抽出することができることを示す．本論文では『パックマン』（株式会社バンダイナムコエンターテインメント，1980 年）を「MCS-AI 動的連携モデル」から捉え直した結果，メタ AI の技術として「波状攻撃タイミングコントロール」「スピードコントロール」「出現数コントロール」という技術を抽出することができた．また『クロムハウンズ』（(C)SEGA / FromNetworks,Inc./ FromSoftware,Inc.，2006 年）を同様に捉え直すことで，スパーシャル AI における「地形に沿った影響マップ」「ゴールデンパス」「位置解析技術」を抽出し新しいゲームデザインを考案することができた．

「MCS-AI 動的連携モデル」がゲームデザインに与える影響について検証するために，ゲーム開発経験を持つ被験者 26 人に対して日本語による自由回答文のアンケートを行った．MCN-AI 連携モデルと MCS-AI 動的連携モデルの解説文が用意され，それぞれの解説を読んだあとに『パックマン』（株式会社バンダイナムコエンターテインメント，1980 年）の新しいゲームアイデアについて複数回答を行う．得られた文章を形態素解析し，共起ネットワークによるデータ分析を行った．結果として， MCN-AI 連携モデルの回答文が AI を取り込まないアイデアが多かった結果に比べて，MCS-AI 動的連携モデルの回答文は，より各 AI とその連携を盛り込んだアイデアとなる結果を得た．「MCS-AI 動的連携モデル」はゲームデザインを質的に変化させる効果を持つことを示した．

MCS-AI 動的連携モデルと AI Graph の関係について述べる．AI Graph は，キャラクターAI における汎用的な意思決定の設計原理であり，AI Graph Editor はその原理に従って実際に作成されたソフトウェア・ツールである．開発者は，AI Graph Editor を用

いて階層的な AI Graph を作成することで，キャラクターAI の意思決定モジュールを作成する．MCS-AI 動的連携モデルの要件を満たすために AI Graph を拡張する．メタ AI はキャラクターAI へ命令を与えるが，キャラクターAI は命令を遂行するために自律性を保ったまま行動することが求められる．そこで AI Graph を保存し，呼び出すことができると同時に部分的に書き換えることができる「アセット化・テンプレート機能」を AI Graph に拡張機能として準備する．すべてのキャラクターの最上層に共通のテンプレート化されたステートマシンを設定し，それぞれのステートをキャラクターごとに実装する．そのステートを AI モードと呼び，メタ AI は全キャラクターに共通した AI モード指定することでキャラクター全般に関する統一された命令システムを持ち，個々のキャラクターは指定された AI モードに沿って個々の能力に応じて自律的に行動する．



「MCS-AI動的連携モデル」の導入事例として『FINAL FANTASY XV』のゲームAIシステムを取り上げる．本ゲームは初期の設計段階から「MCS-AI動的連携モデル」に基づいた設計を行った（上図）．その導入手法と結果を報告し効果を量的，質的の両面から検証する．本ゲームのNPCは数十種類のモンスター，基地や拠点に配置された兵士，数名の仲間からなるが，すべてのキャラクターの意思決定に関してAI GraphとAIモードを導入した．「AI Graph Editor」を用いてそれぞれのNPCの意思決定が製作された．AI Graph は，モンスター，兵士とも３階層であるが，仲間キャラクターに関しては15階層となった．これはモンスター，兵士が数秒～数分の間，戦闘を行いゲームステージから退場することに対して，仲間キャラクターはゲーム開始時から数十時間，プレイヤーキャラクターと行動を共にすることが理由だと考えられる．メタAIとキャラクターAIの間に，9種類のAIモードが設定され実装箇所は5978箇所となった．またメタAI，キャラクターAI

に対してスパーシャルAIから提供される位置検索システムのクエリーの種類は135種類となった．これはスパーシャルAIがメタAI，キャラクターAIから要求される地形解析の多様さに対応している．このような量的な結果は，本モデルにおける3つのAIの連携のつながり方の多様性を示している．またこの他に「MCS-AI動的連携モデル」の上に新しいシステムが構築された．スパーシャルAIから場を介してキャラクターAIを制御する「スマートロケーション」，メタAIとキャラクターAIを用いてプレイヤーキャラクターの動作を補助する「プレイヤーAI」，メタAI，キャラクターAI，スパーシャルAIの3者の連携を通してプレイヤーとNPCの対面の会話を実現する「Face-to-Face対話システム」など，ゲームデザインの要求に応じて「MCS-AI動的連携モデル」の上に新しいシステムが複数実現されており，このシステムを用いてゲームデザインに質的変化がもたらされた．また開発体制としては，開発者がAI Graphを用いることで，自然にゲームAI技術を用いることになり技術の全体的向上につながった．本モデルによってAI開発チームの作業と力を集約することができ，またバグもまたツール上の不具合として収集することができたため，効率的に開発を進めることができた．『FINAL FANTASY XV』の開発を通じて，本モデルが，ゲームAI技術の向上，ゲームデザインの向上，ゲーム開発効率の向上をもたらすことを示した．

「MCS-AI動的連携モデル」はこれから学習・進化・プロシージャル技術と結びつき，ゲームから学習し，ゲームを生成する人工知能として発展して行くと考えられる．プロシージャル技術は本モデルにゲームそのものを生み出す力を与え，学習アルゴリズムは，生成されたゲームを調整する機能を持つ．MCS-AI動的連携モデルはその発展により，ゲームを生成し，調整する力を持つようになると期待される．

（要旨おわり）

目次

1. 序論

本論文全体は,「MCS-AI 動的連携モデル」の提案とその実装による効果の報告からなる.「MCS-AI 動的連携モデル」はゲーム全体を観察しゲーム全体に影響を及ぼすメタ AI，キャラクターの頭脳としてのキャラクターAI，地形・状況の空間的特性を積極的に活用するために地形・状況の特徴を抽出してメタ AI，キャラクターAI に伝えるスパーシャル AI，この 3 つの自律型人工知能が相互・動的に連携するモデルである.

本論文の目的は,「MCS-AI 動的連携モデル」の導入によって，ゲーム AI 技術の向上，ゲームデザインの向上，ゲーム開発効率の向上がゲーム開発においてもたらされることを検証するところにある.

本章では，本論文の背景と目的について述べる．第 1.1 節では本論文全体の構成について述べる．第 1.2 節では背景となる知識について述べ，第 1.3 節では MCS-AI 動的連携モデルの背景について，第 1.4 節では本論文の目的について，第 1.5 節では本論文全体の章構成と各内容について述べる．第 1.6 では本論文で「人工知能」という言葉を用いた時に指す領域について，第 1.7 章ではデジタルゲームにおけるユーザーエクスペリエンスについて述べる．第 1.8 節では本論文で用いる用語について説明を行う.

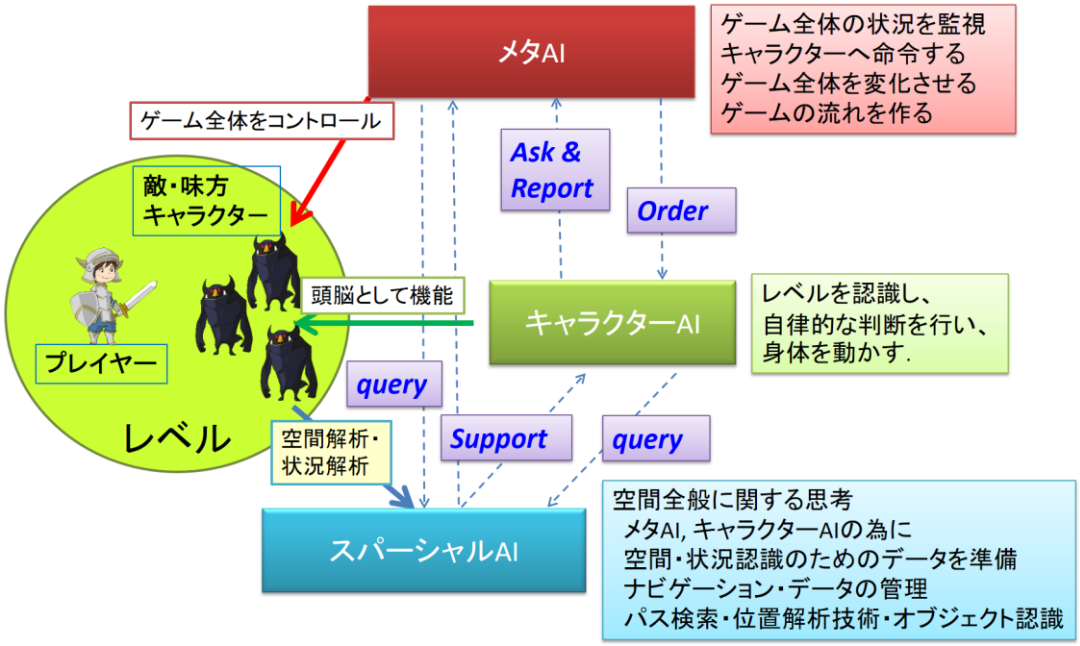


Fig. 1.1　MCS-AI 動的連携モデル

1.1　本論文の構成

デジタルゲームは産業として４０年以上の歴史を持ち，ゲームの発展とともに，そこに実

装される人工知能も共に変化して来た．デジタルゲームの大きな流れとして，RPG（ロールプレイングゲーム）に代表される「物語的ゲーム」と，キャラクターのアクション操作を主とする「アクションゲーム」の二つが存在する．前者はゲームを俯瞰的にコントロールする「メタ AI」を発展させ，後者はゲームに登場する自律型エージェントの基礎となる「キャラクターAI」と，身体的特性に応じて空間を認識する「スパーシャル AI」を発展させて来た．本論文では，両者の人工知能モデルの限界と問題点を示し，それらを解決する上位のモデルとして，3 つの AI を統合した「MCS-AI 動的連携モデル」（Meta-Character-Spatial AI Dynamic Cooperative Model）を提案する(Fig. 1.1)．

本モデルはデジタルゲーム全般に適用可能なモデルであり，近年の複雑で大規模なゲームを技術，ゲームデザイン，開発効率という 3 点において支えるべく設計されたモデルである．本論文では，本モデルがこの 3 点においてどのような効果を持つかを述べ，さらに実際に本モデルを大型ゲーム（『FINAL FANTASY XV』(スクウェア・エニックス，2016 年)）へ実装することで効果の検証を行う．また本モデルがゲームデザインにどのような変化をもたらすかについて，MCS-AI 動的連携モデルの解説を受けた被験者のアイデアと，以前の MCN-AI 連携モデルの解説しか受けない被験者のアイデアを比較する実験を行う．MCN-AI 連携モデルは MCS-AI 動的連携モデルの一つ前のモデルであり，キャラクターAI からの要求を受けてキャラクターのナビゲーションを行うナビゲーション AI と，メタ AI，キャラクターAI によるモデルである．ナビゲーション AI は自律型人工知能ではなく，またメタ AI との連携はない．詳細は後述する．

MCS-AI 動的連携モデルの実装においては，各 AI を連携させる仕組みを導入した．汎用的な意思決定システム「AI Graph」を MCS-AI 動的連携モデルの要求に合わせた拡張を行うことで MCS-AI 動的連携モデルを実装した．この拡張された「AI Graph」の仕組みについても解説を行う．

1.2 デジタルゲームの背景

本節では，デジタルゲームの背景的知識について述べる．デジタルゲームの特徴は，一つの仮想的な世界を立ち上げ，その中でユーザーを遊ばせる点である．その中でも，デジタルゲームのゲームデザインは，前述したように，大きく二つの源流が存在する．「アクションゲーム」と「物語的ゲーム」である．アクションゲームは現実に似た疑似的な物理法則のある世界の中で，プレイヤー・キャラクター（Player Character）とノン・プレイヤー・キャラクター（Non-Player Character，一般に NPC と略される）を動かすのに対して，物語的ゲームは，世界設定された世界の中で，ゲームシナリオに沿ってキャラクターを動かすことで物語を体験する．1970 年代，80 年代のデジタルゲームの黎明期においては，一人，或いは，二人で遊ぶゲームがほとんどであった．そこで，アクションゲームでは敵キャラクターとして，或いは仲間キャラクターとしての NPC の人工知能が必要とされ，物語的ゲームで

は，キャラクターを物語に沿って動かし，世界を変化・更新しながら，物語を進行する人工知能が必要とされた．

アクションゲームでは，環境を認識し，動的にその身体を動かす「キャラクターAI」（Character AI）が必要とされる．また，身体動作を形成するための空間・地形の特徴を抽出する「ナビゲーション AI」(Navigation AI)が「キャラクターAI」の動作生成をサポートする．一方で，物語の中で特定の役割（ロール）を演じながらプレイする RPG は，TRPG（テーブルトーク RPG）を起源の一つとする．TRPG は，ゲームマスターと呼ばれる担当者が存在し，物語とゲームステージ，ゲームの展開（敵の登場など）を作り，設定パラメータ，キャラクターの種類と登場タイミングを決定する．ゲームマスターはプレイヤーに決められた項目にパラメータを振らすことでキャラクターを作らせた上で，ダイスなどを振らせてゲームプレイを進める．RPG における主要な人工知能とは，「ゲームマスター」としての人工知能である．これは「メタ AI」（Meta-AI）と呼ばれ，ゲームを生成しコントロールする権限を持つ．「メタ AI」という言葉が最初に登場するのは 2005 年のウィル・ライトの講演 [1]であり，それ以来「ユーザーを含めたゲーム全体の状況を認識し，目的に沿ったゲームシステム（ゲームの流れ，ゲームデザイン，レベルデザイン）の整合性，難易度を実現するために，ゲーム全域（キャラクターやオブジェクトの配置，動作，イベント，地形）を自律的システムへと変化させる」として発展してきた歴史を持つ [2]．詳細については第 6 章で述べる．また，同時に RPG の中では活動するキャラクターAI が必要である．アクションゲームでは物理的空間の中で巧みに物や地形を用いながら運動するキャラクターが必要で．一方 RPG などでは，台詞を喋り演出的な動きをする「役者」として演じさせるために，「メタ AI」から制御する．たとえば，特定の場面で特定の台詞を言わせるように命令し，特定の場所に移動せよと命令する，ようにである．

このように，二つのゲームの潮流から 3 つのゲーム AI の要素「メタ AI」「キャラクター AI」「ナビゲーション AI」が発展した [3]．後述するが，「ナビゲーション AI」はより一般に空間解析・状況解析を行う「スパーシャル AI」へと発展する．

なお，本論文で述べる AI 連携モデルには名称がついているものがなく，整理のためすべて著者が名称をつけたものである．

1.3 MCS-AI 動的連携モデルの背景

本節では，「MCS-AI 動的連携」を提案する背景について説明する．ゲーム開発は，1994 年以降のゲームの 3D 化において，3D 空間でキャラクターを動かす，というこれまでにない課題に直面し，この課題はゲーム AI 技術に「レベルスクリプト AI」「キャラクターAI」「ナビゲーション AI」の技術的分化をもたらした（Fig. 1.2）．

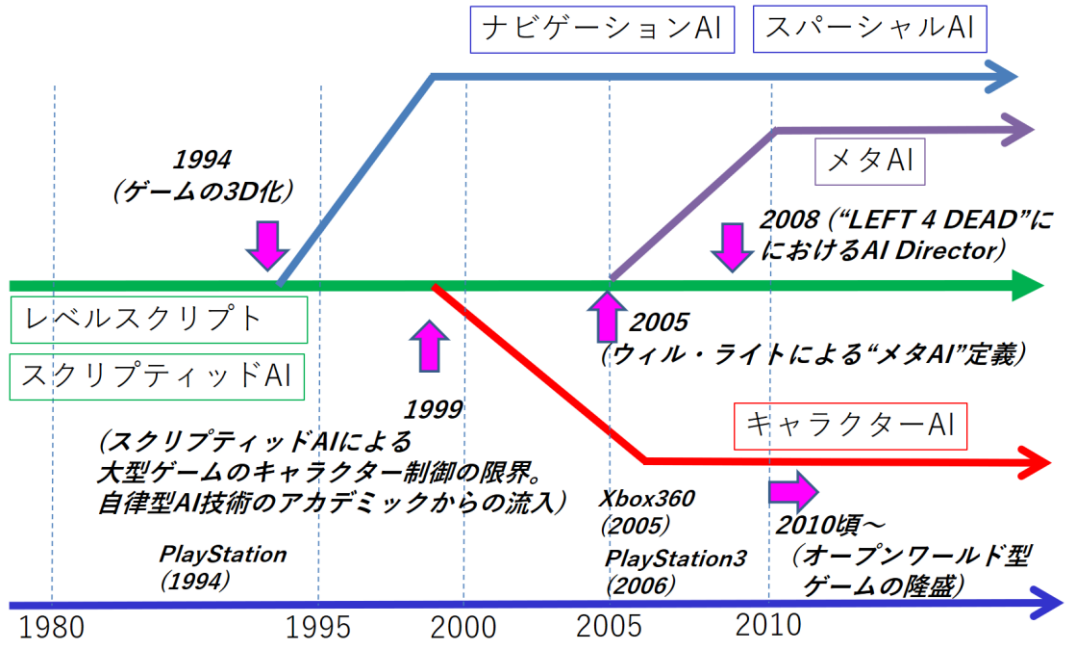


Fig. 1.2　ゲーム AI の歴史

「レベルスクリプト」(Level Script) は，キャラクターを含むゲーム状況をコントロールする外部スクリプトである．一般的なスクリプト言語を用いる場合や，各開発において独自のスクリプト言語を製作する場合もある [4]. スクリプト言語が用いられることが多いためこのように呼ばれるが，実際はプログラムとスクリプト双方の混在したシステムの場合もある．ゲームによってはスクリプト言語を用意せず，プログラムとデータだけの場合もある．特に，レベルスクリプトの内，キャラクターを操作する部分を「スクリプティッド AI」(Scripted AI) と呼ばれる．スクリプティッド AI もまた，プログラムとスクリプト双方の混在したシステムの場合もあり，ゲームによってはスクリプト言語を用意せず，プログラムとデータだけの場合もある．この「レベルスクリプト」,「スクリプティッド AI」からなるモデルを LS-Model (Level Script-Scripted AI Model) と呼ぶ (Fig. 1.3)．LS-Model は，デジタルゲームの最小モデルであり，現在においても，このモデルは主に小型のゲームで使用され続けている．LS-Model は，歴史的にも，ゲーム AI システムの出発点である．

「キャラクターAI」の発展は，キャラクターの自律性の発展である．キャラクターAI の発展の契機は，1994 年前後にゲームが 3D 化し，キャラクターが 3D 空間を解釈しそこで運動を自発的に構成する必要に迫られた事態による．当初はそれでもスクリプティッドな AI で対処できるようにゲームのステージ側を単純にすることで対処していたが，2000 年前後のゲームの大規模化と共に本格的な自律的 AI 技術の導入が避けられなくなった．そこで，主にロボティクスの自律的 AI 技術などがゲーム産業に導入されることとなった．しかし，キャラクターの自律性の発展は，やがてキャラクターをゲームのコンテクストに沿わせることを次第に困難にさせて行った．その場の状況に適応した行動を取ることと，大きな物語

の中で役を演じることは相反する知的能力を要するからである．そこで「レベルスクリプト」は「メタ AI」となり，キャラクターの自律性と演技性のバランスを取らせる役割を持つようになった．またメタ AI の中でも戦闘におけるキャラクターAI の俯瞰的なコントロールに特化した「AI ディレクター」が「LEFT 4 DEAD」（Valve Software，2008 年）で導入され，さまざまなゲームで応用されることとなった［2］．詳細は第 6.3.1 項で述べる．さらに，ゲームの大規模化とジャンルの越境化は，ゲームステージの複雑化をもたらし，「ナビゲーション AI」は「スパーシャル AI」となり，より空間の複雑性を取り込み抽出する，一般的な空間認知を担う人工知能となった．

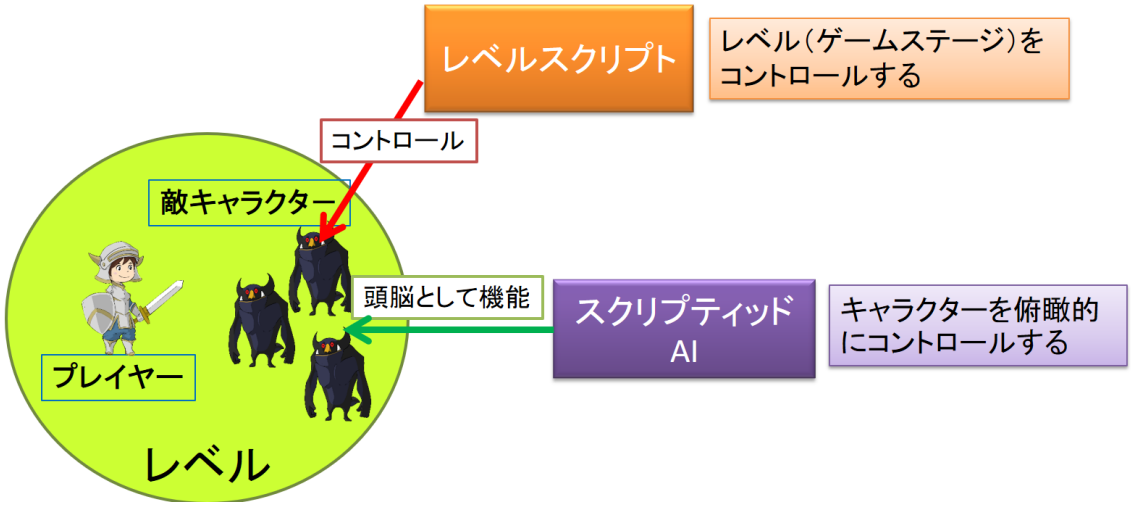


Fig. 1.3　LS-Model

本論文において「MCS-AI 動的連携モデル」を導入する背景には，２つのゲームデザインの発展がある．一つは，アクションゲームと物語的ゲームが，「PlayStation2」の後期，「PlayStation3」（ソニー・インタラクティブエンターテインメント，2006 年），「Xbox360」（マイクロソフト，2005 年）以降，つまり，おおおよそ 2004～2006 年前後には融合したことである．アクションと物語双方を持つゲームが 80 年代，90 年代を通して存在していたが，いずれかに力点が置かれたゲームであった．しかし「PlayStation3」「Xbox360」以降ではアクションシーンとシームレスに物語を展開する映像を展開することができるようになった．また数分の長尺のものから，数秒のものまで自在に映像や劇シーンを挿入できるようになった．たとえば『アンチャーテッド エル・ドラドの秘宝』『アンチャーテッド 黄金刀と消えた船団』『アンチャーテッド －砂漠に消えたアトランティス』（2007，2009 年，2011 年，SIE, Naughty Dog）などはアクション部分と演出部分がシームレスに行き来できるようになっている．「アクションゲームでありながら RPG」，「物語的ゲームであるが，アクションゲームの要素が多いゲーム」，「リアルタイムストラテジーゲームでありながら重厚な物語があるゲーム」，など，ジャンルを超えてアクションゲームと物語的ゲームが融合し始めた．これは，ゲーム機のハードウェアの性能の向上に伴い，一つのゲームジャンルに制限す

る必要がなくなったことが背景にある．

「MCS-AI 動的連携モデル」を導入するもう一つの要因は，ゲームのオープンワールド化である．オープンワールドとは，シームレス（途中にローディングの待ちが存在しない）に広大なゲームフィールドを移動できるゲーム世界のことである．たとえば，『Fallout 3』『Skyrim』（Bethesda Game Studios, 2008, 2011 年）や『グランド・セフト・オート IV』シリーズ（Rockstar Games, 2008 年）は数十 km 四方のゲーム世界を旅するゲームとなっている．このようなゲームでは，そのマップの広さゆえにレベルデザインを精緻に作り込むことが難しくなる．敵キャラクターを配置し，レベルの地形やギミックを作り込むことに加えて，プレイヤーがどちらからやって来るか，どこが戦闘の場所になるかわからないからである．そこで動的にゲーム状況に対応する「自律的なキャラクターAI」，そして，その場の空間を動的に利用するための「スパーシャル AI」，そして，物語としてのゲームの流れを作り続ける「メタ AI」の連携が必要となった．この動的連携システムが「MCS-AI 動的連携モデル」である（Fig. 1.1）．次節からさかのぼって「MCS-AI 動的連携モデル」の起源である 2 つのゲームの流れ「アクションゲームにおける人工知能モデル」（第 1.3.1 項）「物語的ゲームにおける人工知能モデル」（第 1.3.2 項）について詳説し，両者の統合モデルとして，「MCS-AI 動的連携モデル」（第 1.4.1 項）を紹介する．

1.3.1 アクションゲームにおける人工知能モデル

本項では，アクションゲームにおける人工知能モデルについて述べる．アクションゲームにおける人工知能の発展はレベルデザインの発展と共にあった．大規模化・複雑化して行くレベルデザイン，さらにレベルデザインの 2 D から 3 D への変化が，キャラクターAI を，スクリプト言語で外部制御する「スクリプティッド AI」から「自律型エージェント」への変化を促すこととなった．この変化の歴史については，第 3.3 節で解説し，自律型エージェントとしてのキャラクターAI の詳細は第 7 章で解説する．

2D ゲームではキャラクターの移動領域を指定する．固定パスを指定するなど，ほとんど「ナビゲーション AI」技術が用いられることはなかったが，3D ゲームでは 3D 空間内の経路が複雑となり．キャラクターのパス検索技術が必須となった．ただ，1994 年頃から 2004 年頃までの「ナビゲーション AI」の黎明期には，3D ゲームにもかかわらず，ナビゲーション AI のないゲームが混在し，一時的にゲーム産業全体において，キャラクターの振る舞いの質が低下した時期であった．海外・国内を問わず，FPS や RTS ゲームでは，キャラクターが障害物で足止めをされる事象が多く見られた．そこで，この時期の Game Developer Magazine（UBM Technology Group）や Game Programming Gems（Charles River Media, 2000 年）ではキャラクターを複雑な空間に移動させるための技術について記事が多く展開された [5] [6]．これにはゲーム・ハードウェア容量の問題もあり，「ナビゲーション AI」技術はメモリ・リソースが比較的潤沢に使える PC ゲームからその導入が始まり，

「PlayStation2」(ソニー・インタラクティブエンターテインメント，1999 年) の後期 (2004 年前後) に至ってナビゲーション AI が多く導入されるようになった (Fig. 1.4)．ナビゲーション AI は，さらにマップの広大化，地形の複雑化にキャラクターを順応させるために，より一般に，キャラクターの身体的特徴に応じて地形を解析してその特徴を抽出する「スパーシャル AI」(Spatial AI) へと発展していく．この技術的詳細については第 8 章で説明する．

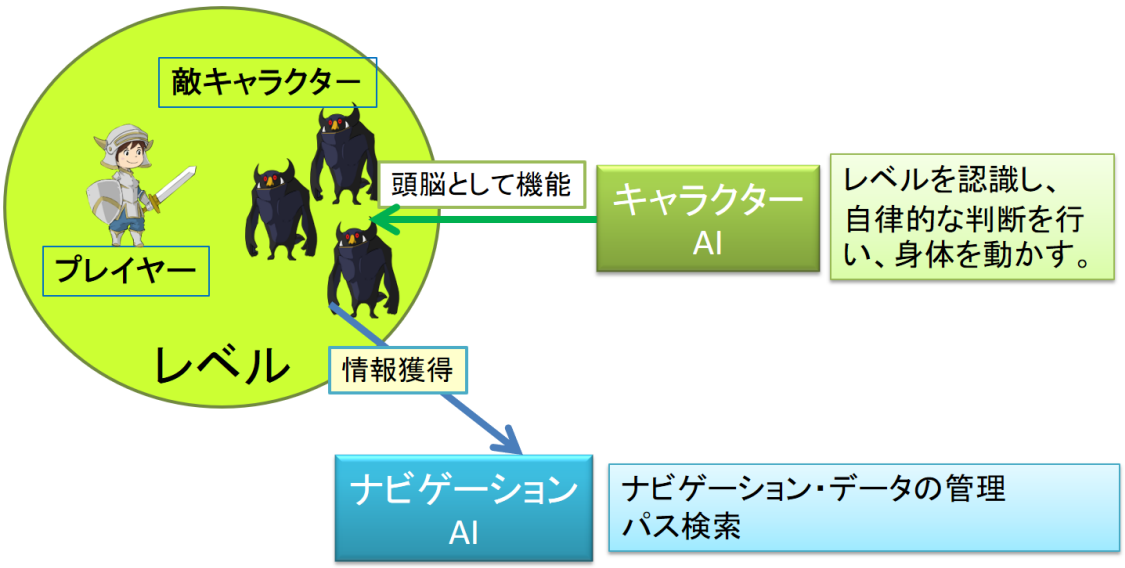


Fig. 1.4　アクションゲームの人工知能モデル

1.3.2　物語的ゲームにおける人工知能モデル

本項では，物語的ゲームにおける人工知能モデルについて述べる．物語的ゲームは，ゲームのステージの総数，敵キャラクターの数，シナリオの長さを拡大することで発展してきた．物語的ゲームの人工知能が直面する問題は，多岐にわたるゲームの要素 (シナリオ，グラフィック，戦闘，会話，音楽など) を組み合わせて，物語に沿った状況を作り上げることである．物語的ゲームにおける AI の役割は，ゲームを俯瞰的にコントロールする立場である．このような制御は，前述したように「レベルスクリプト」と呼ばれる (Fig. 1.5)．「レベルスクリプト」は，あらかじめ決められた状況を各要素への指定により実現するものである．たとえば，キャラクターはこの座標へ移動せよ，プレイヤーがこの場所に来たら岩が転がる，などである．さらに物語的ゲームの場合は，そこで物語上の演出コントロールが加わることとなる．プレイヤーと仲間の NPC たちがお城の出発の前で数十秒の会話を繰り広げていたら，新しい NPC がやってきて仲間に入れて欲しい，などというシーンの構築である．

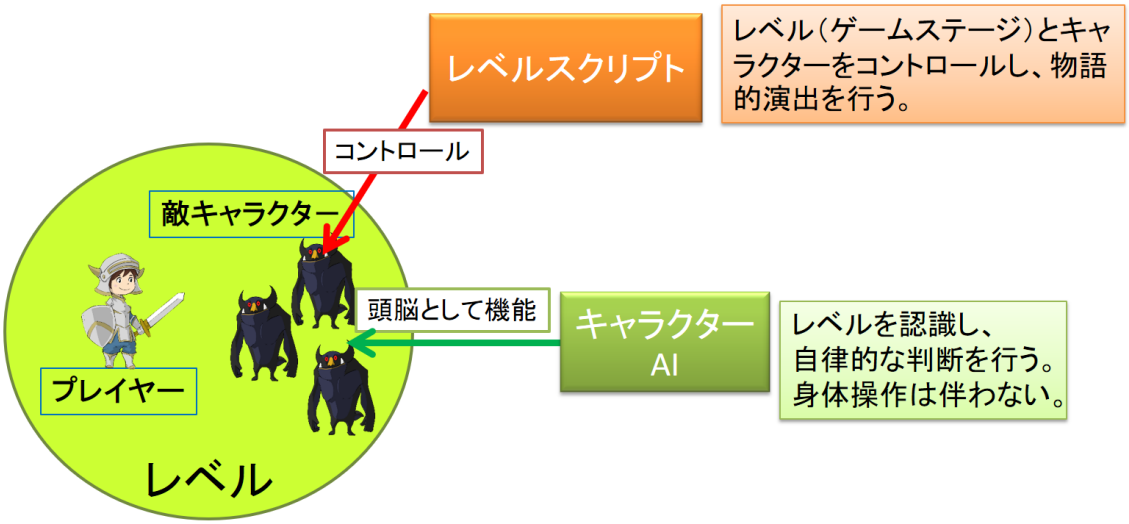


Fig. 1.5　物語的ゲームにおけるメタとキャラクターAI

しかし，レベルスクリプトは，ゲームステージが大規模化すると，イベント（戦闘や会話）が始まる場所が想定できなくなり，複数の要素（敵，地形，天候，状況）が動的に絡み合った状況に対応する必要に迫られるようになる．そこで，「スクリプティッド AI」は「キャラクターAI」へと発展していく（Fig. 1.5）．このキャラクターAI は，アクションゲームにおけるキャラクターAI と違い，身体的運動要素が弱く，論理的な判断をするキャラクターAI である．たとえば選択する魔法を選ぶなどである．

レベルスクリプトはゲーム状態を動的に認識しゲームに影響を与える「メタ AI」へと変化していく．「メタ AI」の成立にはさまざまなタイトルを経た長い年月が必要であり，この詳細については第 6 章で解説する．

第 1.3.1 項で「アクションゲームにおける人工知能モデル」，第 1.3.2 項で「物語的ゲームにおける人工知能モデル」について詳説した．次節で両者の統合モデルとして「MCS-AI 動的連携モデル」を導入する．

1.4　本論文の目的

本論文はデジタルゲーム AI における課題を新しいモデルによって解決することを目的とする．デジタルゲームにおける人工知能には以下の 3 つの問題点がある．

(1)　AI 技術の向上：基礎技術
(2)　デザインの向上：製品開発
(3)　開発効率の向上：開発技術

それぞれについて説明する．第一にゲームにおける AI 技術の基礎技術の向上である．ゲームの規模と複雑性が増す中で，ゲームデザインから要求される AI 技術の水準が上がっている．そこで求められているものは，拡大し複雑化するレベルデザインの中で，ゲーム開発者が，各場面にキャラクターがするべき動作を指定するのではなく，キャラクター自身が状況を認識し自律的に活動すること，第二に，その自律型キャラクターたちを用いてゲームのシーンを構築する，というゲームデザインの要求を満たすことである．そのためには，ゲームデザイナーが AI 技術を調整する仕組みが必要となる．第三に，ゲーム AI システムの開発には，ゲームコンテンツに依存しない一般的な人工知能技術の開発と，各ゲームコンテンツに開発する部分が必要である．特にエンジニアの仕事と，具体的にコンテンツを作るゲームデザイナーの仕事を効率的に連携・協調させる仕組みが必要である．ゲームデザイナーが高いレベルのゲーム AI システムを構築できるために，作り込むためのツールと，そこから利用できる AI 技術が必要である．

上記の 3 つの課題をより詳細に述べると以下のようになる．

(1)　AI 技術の向上：キャラクターAI とゲームの状況を認識しコントロールする方法
(2)　デザインの向上：ゲームデザイナーが人工知能を活用できる仕組み
(3)　開発効率の向上：複数の開発者が協調できる仕組み

である．これらの要求を満たす技術フレームとして本論文では「MCS-AI 動的連携モデル」を提案する．このモデルは上記の 3 つの問題を以下のような形で解決する．

(1) AI 技術の要素を 3 つの独立した自律型 AI の連携システムとする
(2) ゲームデザインを 3 つ AI の要素に分解し組み合わせる仕組みとする
(3) AI の各要素の実装が再利用可能な汎用モジュール化され蓄積される

MCS-AI 動的連携モデルは，これを解決する汎用的な AI システムである．それぞれについて以下のように解決する．より具体的な説明をすると，

(1)　キャラクターに対してはキャラクターAI を，状況・環境全体に対してはスパーシャル AI がコントロールし，この 2 つをメタ AI からコントロールする．
(2)　ゲームデザインの要件をメタ AI，キャラクターAI，スパーシャル AI の技術要素として分解し，組み合わせることで設計する．
(3)　メタ AI，キャラクターAI，スパーシャル AI をそれぞれ独立に担当の開発者が開発する．各 AI において要素技術がモジュールとして蓄積され，コンテンツに応じて，それらのモジュール群が組み合わされる．

本論文の目的は，上記のような背景があるとき，デジタルゲームの新しい人工知能モデル「MCS-AI 動的連携モデル」を提案することにある．「MCS-AI 動的連携モデル」は，3 つの自律型 AI が独立して動作しつつ，互いに連携することで，様々な効果を生み出す．MCN-AI 連携モデル以前のようにお互いの動きを静止しながら連携するではなく，自律的に動作しながら，その都度必要な状況を継続的に生成し続けるモデルである．それぞれが自律的に継続しつつ協調する点は，変化し続ける状況の中で互いに噛み合わない状況を生む可能性がある．変化する状況の中でもロバストに自らゲームを変化し続ける効果を持ち続ける必要がある．そこでお互いが管理する時間的・空間的領域を持ち，ゲーム状況の変化を認識しつつ効果を及ぼすシステムとなることが求められる．

以下に概要を説明するが，本モデルの詳細については第 5 章，検証例については第 9，12 章で述べる．

以下，第 1.4.1 節において，本モデルの説明を行い，第 1.4.2 節では(1)(2)について述べる．(3)については，第 9 章で過去のタイトルの分析を行う．また本論文全体にわたって，本モデルの利点を解説して行く．

1.4.1 MCS-AI 動的連携モデル

「MCS-AI 動的連携モデル」(Fig. 1.1) は，アクションゲームにおける人工知能モデルと，物語的ゲームにおける人工知能モデルを統合したモデルである．歴史的には，第 4.3 章で述べる「LCN-AI 連携モデル」(Level-Character-Navigation-AI Cooperative Model)，「MCN-AI 連携モデル」(Meta-Character-Navigation-AI Cooperative Model) を経て統合される (Fig. 1.6)．「LCN-AI 連携モデル」は「レベルスクリプト」でゲーム全体を制御しながらキャラクターAI に命令を出し，キャラクターAI は「ナビゲーション AI」を用いてパス検索を行い，ゲーム内を移動し活動する．この「レベルスクリプト」が「メタ AI」へと発展したモデルが「MCN-AI 連携モデル」である．「メタ AI」がゲームを動的に制御し，キャラクターAI に命令を出し，キャラクターAI は「ナビゲーション AI」を用いてパス検索を行い，ゲーム内を移動し活動する．ここから「ナビゲーション AI」が「スパーシャル AI」へと発展したモデルが「MCS-AI 動的連携モデル」である．「MCN-AI 連携モデル」は歴史的に形成されてきたモデルであるが，「MCS-AI 動的連携モデル」は本論文で提案する筆者の提案する新しいモデルである．ただ，このようなモデルの変遷はこれまでにまとめられたことはなく各モデルの命令は筆者によるものである．

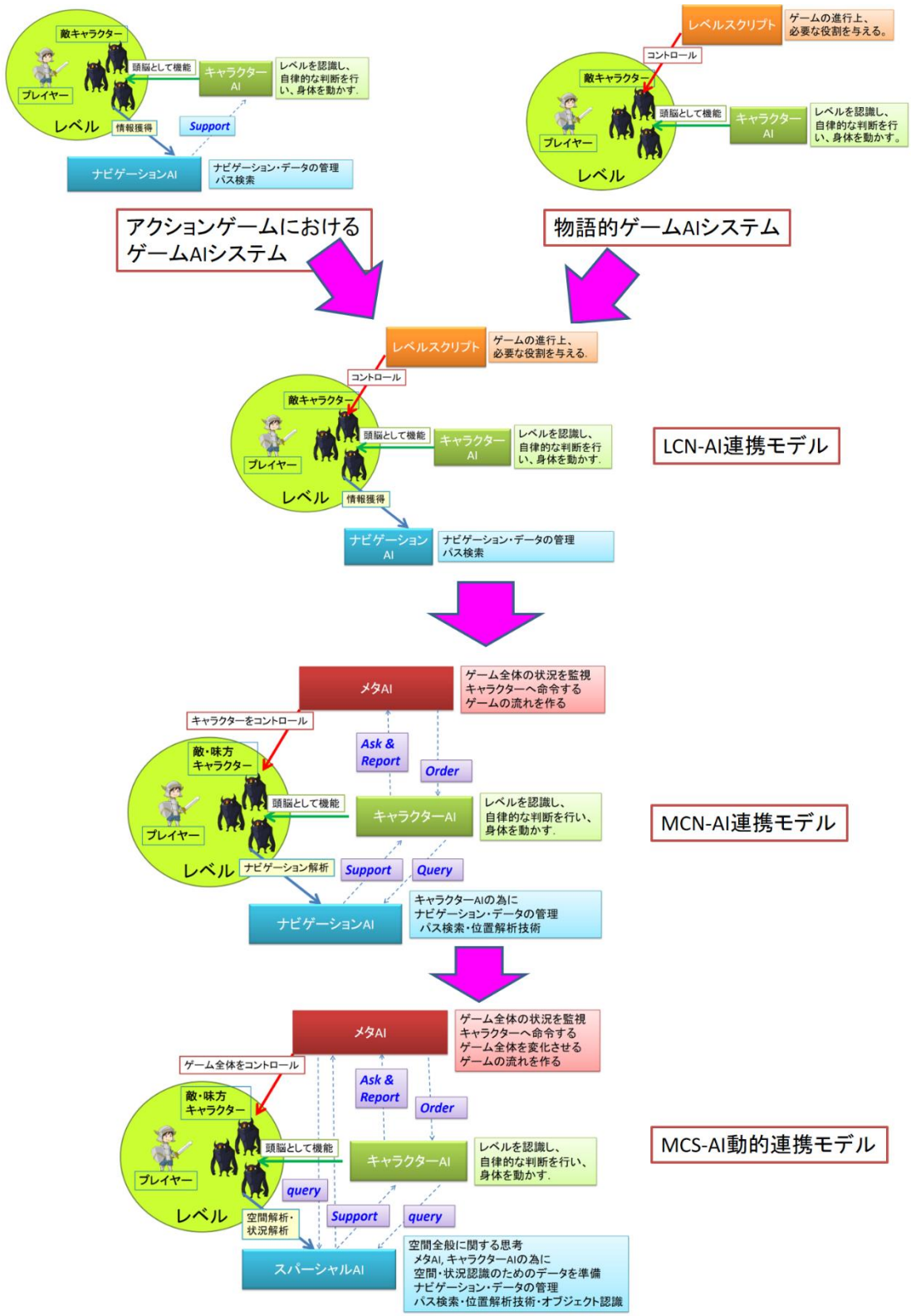


Fig. 1.6　MCS-AI 動的連携モデルの形成

「MCS-AI 動的連携モデル」の要素は，ゲームを俯瞰的に観察し，ゲームのあらゆる要素を通じてゲームをコントロールする「メタ AI」，キャラクターの頭脳として「キャラクターAI」，そしてこの 2 つがさまざまな地形のステージで動作するためのサポートをする「スパーシャル AI」である．ここで「キャラクターAI」は「自律型エージェント」としての人工知能であり，そのため「メタ AI」はその指令だけをキャラクターAI に伝えれば，あとは「キャラクターAI」が自律的に判断し実行する．「スパーシャル AI」は「ナビゲーション AI」をさらに一般化し自律化させた AI であり，パス検索のみならず，地形の特徴をキャラクターの身体的特徴に応じて抽出し空間認識をサポートする．たとえば，目的と身体運動能力に応じた目的位置の算出などである．

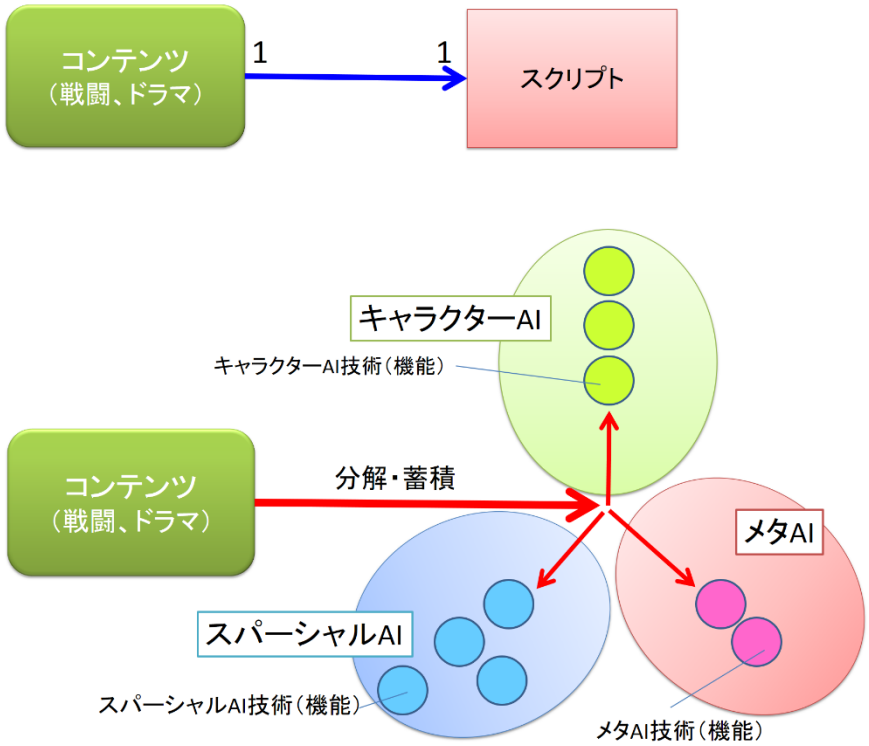


Fig. 1.7　レベルスクリプト（上図）と MCS-AI 動的連携モデルの違い（下図）

この 3 つの AI は，ゲーム実行中に動的に連携する．スパーシャル AI は静的な地形データから抽出したデータを用いつつ，自律的にゲーム実行中に動的に地形を解析し，メタ AI，キャラクターAI が必要とする空間的情報をリアルタイムに両者に提供する．ナビゲーション AI のように，メタ AI，キャラクターAI からの要求を受けて地形を解析する場合もある．キャラクターAI は，自律型エージェントとしての人工知能であり，自分自身の感覚によって環境を認識し，意思決定し，身体を動かす．しかし，キャラクターたちの自律型エージェントとしての振る舞いは，局所的なものの集合であり，個として局所的状況に合った動きであったとしても，全体としてはゲーム全体の状況やストーリーに合わせられていない．そこ

で「メタ AI」は全体を調整し，自律的にゲーム全体としての展開の流れを作り出す AI である．キャラクターAI を始め，ゲームの全要素（天候，地形，台詞，イベントなど）に指示を出すことによって，ゲームが必要とする状況を動的に作り出す．

「MCS-AI 動的連携モデル」の連携の要件は「それぞれの AI が自律化しつつ連携し，お互いの動きを止めない動的なシステム」ということである．「MCS-AI 動的連携モデル」の動的，という言葉は，それぞれの AI が継続的に動作し連携するために用いた．「MCN-AI 連携モデル」の場合，ナビゲーション AI は自律的 AI ではないため，メタ AI，或いはキャラクターから要求が来るまでは静止している．「LCN-AI 連携モデル」以前では，レベルスクリプトが動作している間は他の AI は処理を止めるため，それぞれの AI がスイッチングする形であり動的連携ではない．3 つの AI が自律的 AI となり動作することで初めて動的に連携する．

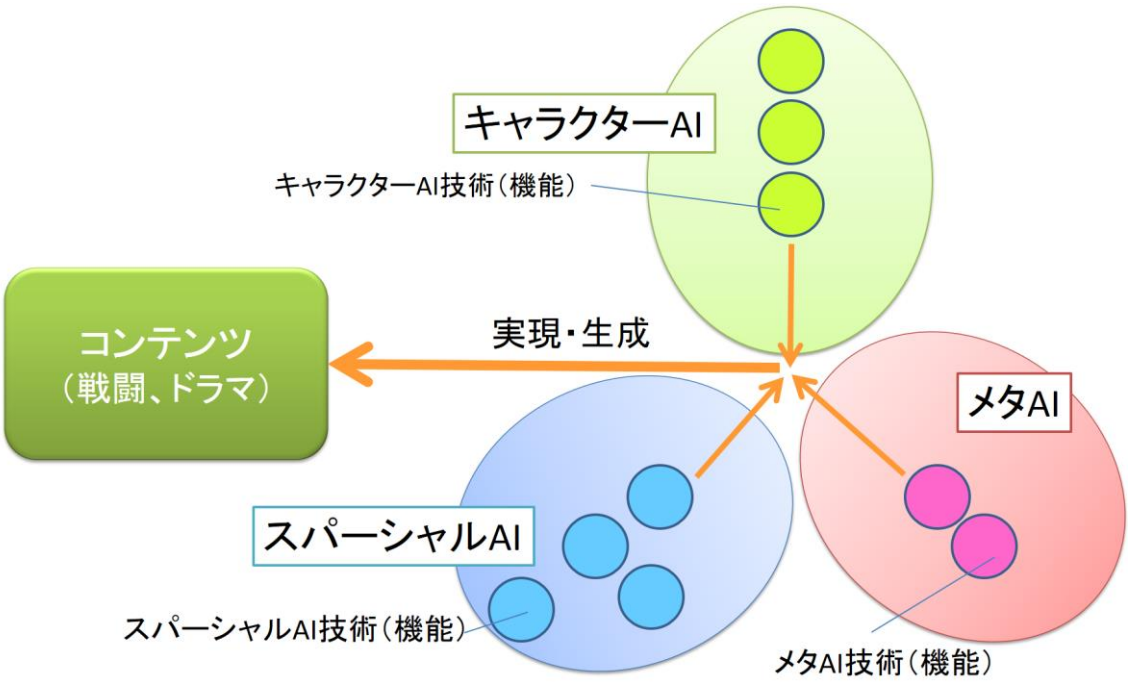


Fig. 1.8　MCS-AI 動的連携モデルからのコンテンツの実現・生成

1.4.2　　MCS-AI 動的連携モデルの利点

「MCS-AI 動的連携モデル」は，ゲーム全体の AI システムを提供する．ゲームの各要求に対して「レベルスクリプト」（各シーンに対して，シーンを構成するために記述されるスクリプト），或いは「スクリプティッド AI」はコンテンツ（戦闘，ドラマ）をスクリプト言語で記述する．基本的に作成するコンテンツの数に比例してスクリプトの数が増加して行く．一方，MCS-AI 動的連携モデルの利点は，ゲームから要求される要件を，3 つの AI の

技術要素に分解しモジュールとして実装することで，それぞれの技術をゲーム内で再利用可能なモジュールとして蓄積するところにある（Fig. 1.7）．ある程度の各 AI の機能の蓄積が終わると，あとは，それらの機能を組み合わせるだけで，様々な AI 要件を実現できるようになるところにある．

この再利用性と組み合わせによる開発によって，ゲーム AI 開発は効率化される．特に，大規模なゲームの開発の場合には，開発前半で蓄積された各 AI の機能によって，中盤〜後半以降，それらの組み合わせで，ゲームからの要求を実現できる（Fig. 1.8）．MCS-AI 動的連携モデルは，ゲーム開発全般の効率化を促進する．

1.4.3 MCS-AI 動的連携モデルと AI Graph

MCS-AI 動的連携モデルの一つの実装例として，第 11 章で「AI Graph」を提示する．「AI Graph」は汎用的な意思決定を中心とした AI 構築システムであり，MCS-AI 動的連携モデルとは独立に開発されていた．しかし，MCS-AI 動的連携モデルを実装するために，機能を拡張することで，MCS-AI 動的連携モデルを実現するシステムとなった．これについては，第 12 章で述べる．

1.5 本論文の章構成と各内容

本論文の全体の章構成を以下に示す（Fig. 1.9）．第 1 章では，本論文で提案する「MCS-AI 動的連携モデル」の背景と，全体の構成を説明する．第 2 章ではこのモデルのデジタルゲーム全体における位置づけについて述べる．第 3 章では，本モデルが立脚する「デジタルゲームの構造」を説明する．第 3 章は，既知のデジタルゲームの技術をまとめた章であり飛ばして読んでも構わない．第 4 章では，これまでのゲーム AI の限界と問題点を提示し，「MCS-AI 動的連携モデル」を構築する必要性を述べる．具体的には本モデル以前の「スクリプティッド AI」「キャラクターAI」「ナビゲーション AI」による「LCN-AI 連携モデル」について説明する．第 5 章では，第 4 章の問題を解決する「MCS-AI 動的連携モデル」を新しく提案する．本モデルは「メタ AI」「キャラクターAI」「スパーシャル AI」の動的連携モデルの概念モデルである．第 2 〜 4 章は前提知識と問題提起であり，第 1 章から第 5 章へ直接進んでも構わない．

第 6，7，8 章において，それぞれの AI について解説を行う．第 6 章では，「メタ AI」について説明する．「メタ AI」はゲーム産業の中で「AI Director」と共に育まれた概念であり，現在も発展中の概念であるから，その歴史的形成を含めて，再定義を行う．第 7 章では，「キャラクターAI」について意思決定アルゴリズムを中心に解説する．第 3 章で基本的事項を解説しているため，「メタ」「スパーシャル AI」との関わりを中心に解説を行う．第 8 章では，「スパーシャル AI」について解説する．ゲームの発展と共に，より深い地形・環境・

空間理解が必要になった結果，蓄積されて来た知見を整理する．

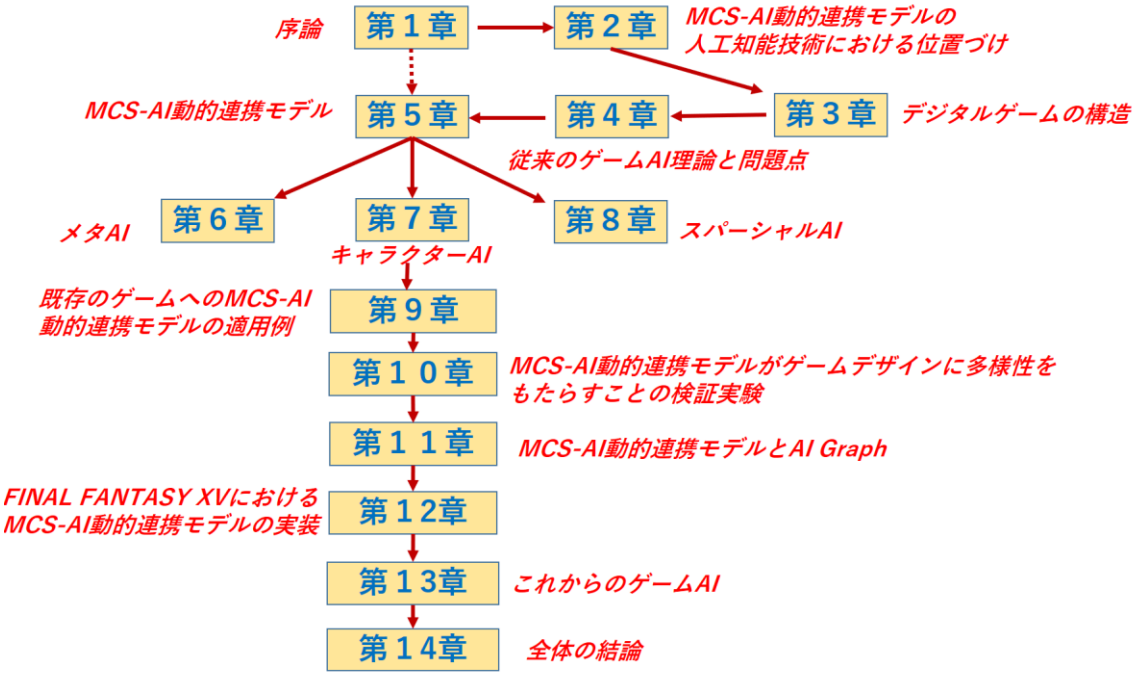


Fig. 1.9　全体の章構成

第 9 章では，「MCS-AI 動的連携モデル」を既存のゲームに適用し，いかに当てはまるかを検証する．具体的には二つのゲーム『パックマン』（株式会社バンダイナムコエンターテインメント，1980 年）と『クロムハウンズ』（(C)SEGA / FromNetworks,Inc./FromSoftware,Inc., 2006 年）において検証する．第 10 章では，「MCS-AI 動的連携モデルがゲームデザインに多様性をもたらすことの検証実験」を行った結果について報告する．第 11 章では MCS-AI 動的連携モデルにおける意思決定のシステム「AI Graph」について述べる．第 12 章では，「MCS-AI 動的連携モデル」を実際に設計モデルとして実ゲーム制作に採用した『FINAL FANTASY XV』（スクウェア・エニックス，2016 年）の実例について，その実装結果を示す．第 13 章では，これからゲーム AI が取り組んでいく「学習・進化アルゴリズム」「自動生成技術」（プロシージャル技術）について説明する．この二つはゲームの外の AI として，導入が始まっており，将来的には「メタ AI」が取り込むことで，ゲームそのものをある程度，自動生成・自動調整することが可能となる．第 14 章では全体の考察と展望を行う．

本章では，本論文全体の前提と主題を解説した．第 1.2 節ではデジタルゲームそのものの背景を紹介した．第 1.3 節では本論文が提案する「MCS-AI 動的連携モデル」の背景を示した．第 1.4 節として本論文の目的を示し，第 1.5 節で本論文全体の流れを示した．次章から，上記の章構成に従い解説して行く．

1.6　本論文における「人工知能」の定義

本論文において，人工知能という言葉でさし示す人工知能の型は 2 つある．

自律型人工知能　自ら「認識して」「意思決定を行い」「活動する」人工知能
機能型人工知能　知的機能を有するソフトウェア，プログラム

そこで「〜AI」という言葉は，2 つの意味で使われる．「メタ AI」「キャラクターAI」「スパーシャル AI」は自律型人工知能に相当する．「ナビゲーション AI」「会話 AI」「プロシージャル AI」は機能型人工知能に相当する．

ここで，本書で用いる「AI」という言葉を含む用語について，その発祥と学術，産業における使用状況についてまとめておく（Table 1.1）．

MCN-AI 連携モデルは 2 つの自律型人工知能「メタ AI」「キャラクターAI」と機能型人工知能「ナビゲーション AI」を組み合わせたモデルである．一方，MCS-AI 動的連携モデルは 3 つの自律型人工知能「メタ AI」「キャラクターAI」「ナビゲーション AI」を組み合わせたモデルである．

「将棋 AI」「囲碁 AI」「チェス AI」という場合には，人間のプレイヤーと同じ立ち位置でゲームをプレイする人工知能，という意味で AI という言葉が使われる．つまり手を選択する人間の知的活動を模したソフトウェアとしての AI である．盤を認識し，思考し，駒を動かすことから，これらは自律型人工知能に分類される．デジタルゲームにおいて特定のゲームのタイトルと AI を組み合わせる言葉の事例は，世界的に知られたタイトルに限られる．たとえば『テトリス』（アレクセイ・パジトノフ，1984 年）に対して「テトリス AI」 [7]，『ぷよぷよ』（コンパイル，1991 年）に対して「ぷよぷよ AI」 [8]である．これらは「将棋 AI」「囲碁 AI」「チェス AI」と同様にデジタルゲームをプレイする自律型人工知能という意味である．

ゲームタイトルに実装される人工知能を「ゲームの中の AI」，ゲーム開発に実装される人工知能を「ゲームの外の AI」と呼ぶ．この「ゲームの中の AI」「ゲームの外の AI」は，それまで適切な用語がなかったため著者の造語である．ゲーム産業内においても，学術的発表においても共通して用いている．これらは総称であるから，その中には自律型人工知能も機能型人工知能も含まれる．たとえば，通常，ゲームをデバッグや品質を確かめるのはテスターと呼ばれるゲームをテストする人間であるが，この作業を AI が代替する場合は「自動デバッグ AI」「テスターAI」と呼ばれる．品質保証（Quality Assurance）の AI 全般は，新しい分野であるため，一般的な用語はなく，著者は「QA-AI」と著者は呼んでいる．これらは「ゲームの外の AI」に含まれる自律型人工知能である．

Table 1.1　本論文における AI を含む用語の発祥と学術・産業における使用状況

用語	発祥	学術における使用状況	ゲーム産業における使用状況	型
ゲーム AI	学術	90 年代までは囲碁 AI・将棋 AI などボードゲームの AI を指す用語であったが，2000 年以降はデジタルゲームを含む一般的な用語に変化した.	左に同じ	自律型・機能型双方を含む総称
デジタルゲーム AI	学術	デジタルゲームの人工知能	左に同じ	上に同じ
メタ AI	産業	少なくともゲーム産業と同じ意味では用いられていない	一般的に用いられている	自律型
キャラクターAI	学術	デジタルゲームに登場する NPC の頭脳	左に同じ	自律型
ナビゲーション AI	学術	ロボット・車・ドローン・仮想空間のキャラクターなど, 一般に移動を制御する人工知能	左に同じ	機能型
スパーシャル AI	三宅	自律的に空間・状況解析を行う AI として定義		自律型
MCN-AI　連携モデル	三宅	既存のデジタルゲーム AI の仕組みを指す言葉として定義. モデルは良く使われるが，名前は付けられていなかった.		連携システムの名称
MCS-AI動的連携モデル	三　宅た	新しいデジタルゲーム AI のシステムとして定義.		連携システムの名称
-　　モデル	三宅	本論文中の「-AI モデル」という名称は筆者が命名した．元からあるモデルであるが，名称がないので名付けたものである.		連携システムの名称
会話 AI	学術	自然言語による会話機能	左に同じ	機能型
プロシージャル AI	学術	自動生成技術	左に同じ	機能型
ゲームの中の AI	三宅	使用されない	ゲームの実行中に用いられる人工知能技術全般の総称	自律型・機能型双方を含む総称
ゲームの外の AI	三宅	使用されない	ゲームの開発中に用いられる人工知能技術全般の総称	上に同じ

1.6.1　学術とゲーム産業における「AI」という言葉の用い方

ゲーム産業では「AI」という言葉が，上記の意味とは別に，NPC や敵キャラクターという意味で使われることがある．これはゲーム産業特有の使い方で特に正確さがあるわけではない. NPC や敵キャラクターは「CPU」と呼ばれることも，表記されることもある.「AI」や「CPU」という言葉によって「自動的に動く」ということをユーザーに知らせるために使われる．また結果として，ゲーム開発の現場でも使われている．本論文では「AI」という言葉をこのような意味で使うことはない.

1.7　デジタルゲームにおけるユーザーエクスペリエンス

ユーザー持つゲーム経験のことを「ユーザーエクスペリエンス」（User Experience，UX と略される）と言う. UX は様々な分野で使われる用語であるが，本論文では「ゲーム体験」という意味において用いる．誤解のないように「ユーザーエクスペリエンス」という言葉を使わず，「ユーザー体験」に統一するものとする.

「ユーザーエクスペリエンス」は多義に使われている言葉であり，その意味はいくつかに分類される [9].

(D1) その会社との全般的なあらゆるインタラクション
(D2) ユーザーの内的状態の連続的の変化
(D3) その製品とのあらゆる感性・感情経験の集合
(D4) その製品やサービスとのインタラクションから得られる価値
(D5) そのデザインから得られる経験の質

本論文では「ゲーム体験」は「ゲームをプレイするというインタラクションから得られる感性・感情経験，内的な変化，プレイ全体を通じての経験」と定義する．上記の分類では (D2)(D3)(D5)が該当する．特にゲーム AI は，メタ AI によるゲームの変化や，NPC との継続的なインタラクションによってユーザーの心理に対して連続的に影響を与えることから，(D2)に相当するユーザーの心理的な状態変化をもたらす．たとえば，第 6 章で述べるようにメタ AI とキャラクターAI との連携によってユーザーの緊張をアップダウンさせる，プレイヤーに対する難易度をコントロールするなど，一定時間に渡るユーザーの心理状態の変化を及ぼす．また，第 12.3 節の FFXV の例では，仲間キャラクターとのインタラクションの中から，次第に仲間キャラクターに対する親近感を得て行く実例について述べている．キャラクターによるユーザー体験の形成である [10].

デジタルゲームでは，ユーザーに対してプレイの目的がはっきりと提示される．たとえば，ドラゴンを倒す，宝物を得る，などである．ただ，ユーザーがその目的を達成するプロセス

を通じて，どのようなユーザー体験を得るか，ということはユーザーに対して明確に示されているわけではない．ゲーム開発者は，そのようなユーザー体験が形成されるように，ゲームとゲーム AI のデザインと設定を行う．しかし，ユーザーの行動は自由で完全に予測ができないため，MCS-AI 動的連携システムは，このユーザーの行動の変化に柔軟に対応しながら，目的であるユーザー体験の形成を目指す．たとえば，「爽快感のある戦闘」を実現したい場合，ユーザーがうまく戦闘していない場合は，それを検出し，仲間キャラクターからのサポートを手厚くし，円滑に戦闘プレイが流れるようにキャラクターの行動をメタ AI が調整する．たとえば攻撃の強さを弱める，ユーザーが逃走できるコースを残す，などである．

1.8　レベル，レベルシステムなどの言葉の使い方

「レベルデザイン」における「レベル」はゲームのそれぞれのステージを指し，地形やキャラクター，マップ上の仕掛けまで含んだ全体を意味する．キャラクター，オブジェクト，地形など，ゲームを構成する要素はエンティティと呼ばれ，エンティティが集まってレベルが構成される．NPC の配置・動作，地形，オブジェクト配置全体をデザインすることをレベルデザインと呼ぶ．またレベルシステムとはレベルを制御するプログラムを指す．

2. MCS-AI 動的連携モデルの人工知能技術における位置づけ

本章では，「MCS-AI 動的連携モデル」が，人工知能技術分野の中で，どのような位置づけにあるかを述べる．第 2.1 節では，デジタルゲーム AI が人工知能技術全般の中での領域を明らかにする．第 2.2 節では，デジタルゲーム AI の全体像を紹介する．第 2.3 節では「MCS-AI 動的連携モデル」が，どのような階層の技術に属するのかを説明する．第 2.4 節では「MCS-AI 動的連携モデル」の特徴と優位性について述べる．第 2.5 節では本モデルの限界と発展について述べる．学習・進化アルゴリズムもまた「MCS-AI 動的連携モデル」から活用される技術ではあるが，これまで別の用途，ゲームデザインや品質保証と言った分野に使用されて来た．これについて第 2.6～2.8 節で述べる．第 2.6 節では「学習・進化アルゴリズム」，第 2.7 節では「ディープラーニングの応用」について，第 2.8 節では「ゲームの品質保証の AI」について述べる．

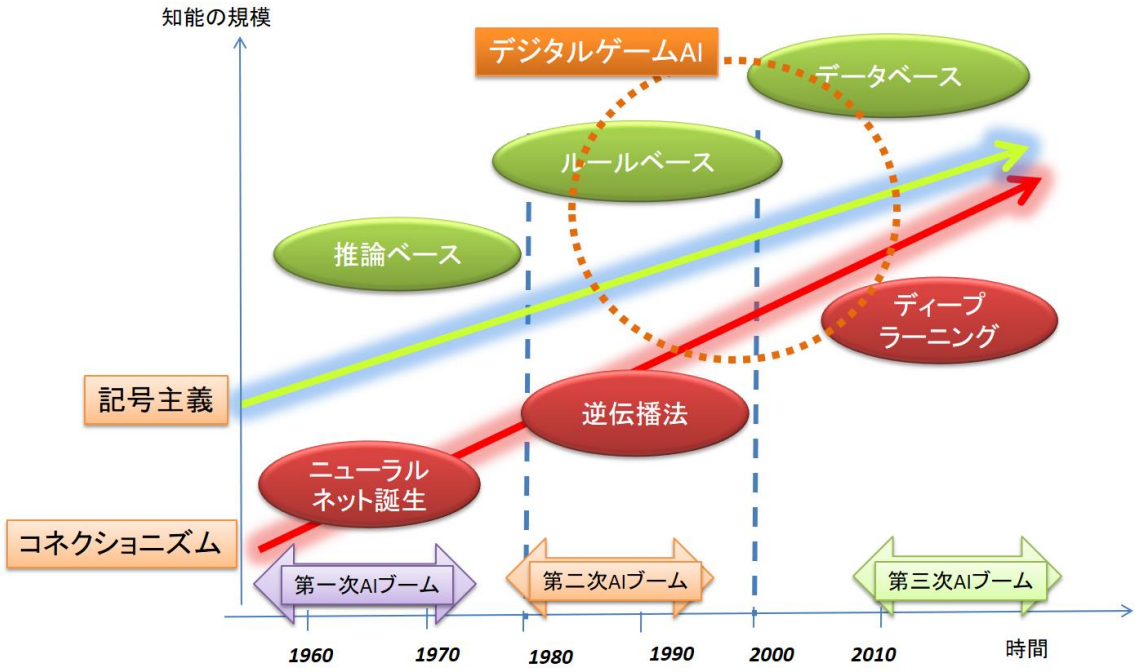


Fig. 2.1　人工知能の歴史とデジタルゲーム AI

2.1　デジタルゲーム AI の技術領域

本節では，人工知能技術の中のデジタルゲーム AI の立ち位置について説明する．人工知

能には，大きくシンボルを基礎とする「記号主義」（シンボリズム）型人工知能と，ニューラルネットを基礎とする「コネクショニズム」型人工知能がある．デジタルゲーム AI は双方の技術をまたいで使用する（Fig. 2.1）．また学習を使う方法と学習を使わずに人の手でルールや知識の形をシンボルで準備する方法があり，双方を使い分ける．またロボティクス分野はゲーム AI 同様にインタラクティブでリアルタイムかつ身体を持つシステムであるため，ロボティクス分野の AI から技術を援用する場合も多い［11］．

デジタルゲーム AI は，80 年代から現在に至る人工知能，ロボティクス AI の流れを吸収しつつ発展した．2010 年までは記号主義の方に重心が置かれていたが，2015 年以降，2020 年に至るまでコネクショニズムと機械学習の方へ重心を移しつつある（Fig. 2.2）．

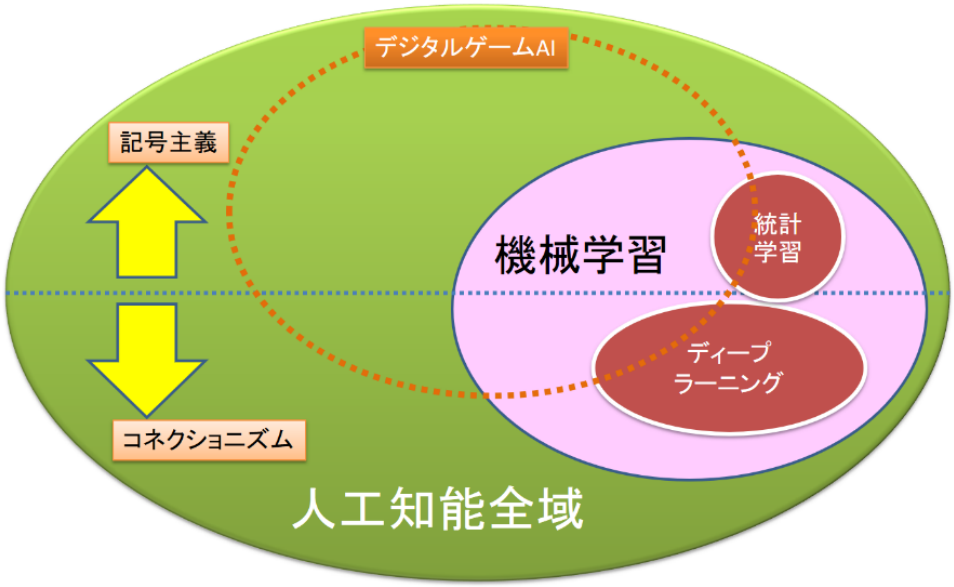


Fig. 2.2　人工知能全体図とゲーム AI の領域

2.2　デジタルゲーム AI の全体像

本節ではデジタルゲームが内包する領域の説明を行う．デジタルゲーム AI が意味をするところは大きく 2 つあり，ゲームをプレイする人工知能という意味と，デジタルゲームの中のようにゲームの構成要素としての人工知能という意味がある．将棋や囲碁，チェスなど研究で良く取り上げられる題材としてボードゲームの AI は，主に前者のゲームをプレイする人工知能であるが，デジタルゲームはそもそも開発の初期ではゲームそのものがなく，人工知能をゲームの構成要素として構築しながらゲーム全体の製作に貢献する．

デジタルゲームは仮想的な空間を舞台にするが，ゲーム産業はエンジニアとアーティストの協力によって，広大で複雑なゲームステージ（ゲーム空間とそこに置かれるオブジェクト，ギミック．レベルデザインとも言う）を作り出すことに長けている．その仮想空間を舞台として人工知能を製作する．そこでデジタルゲームは人工知能の実験場を提供する機会

でもある．デジタルゲームの複雑度に応じてデジタルゲーム AI のより高度なクオリティが必要とされる．そこで，デジタルゲームには人工知能技術の導入が特にこの 20 年間継続されて来た [12] [13] [14] [15] [16] [17] [18].

デジタルゲームの AI 分野全体は，ゲームタイトルの中でリアルタイムに動かす「ゲームの中の AI」，ゲーム開発やサービスのバックエンドで動く「ゲームの外の AI」がある（Fig. 2.3）.「ゲームの中の AI」はゲームそのものに実装される自律型・機能型人工知能の総称である.「ゲームの外の AI」はゲーム開発に用いられる人工知能の総称であり，機能型人工知能がほとんどである.

「ゲームの中の AI」については，特に 1994 年（本格的にゲームの 3D 化が始まった年）から 2012 年(ゲームステージのオープンワールド化が始まった年)までの急速の進歩の後，2017 年までにゲームのオープンワールド化に対応し，それ以降はニューラルネットをはじめとする学習アルゴリズムの導入が行われた [3] [19] [20].「ゲームの外の AI」に関しては，ゲームエンジンやサーバーログによって収集したデータの解析や，自動デバッグ，品質保証のための AI（QA-AI），自動ゲームバランシングが注力されて来た [21] [22] [23].

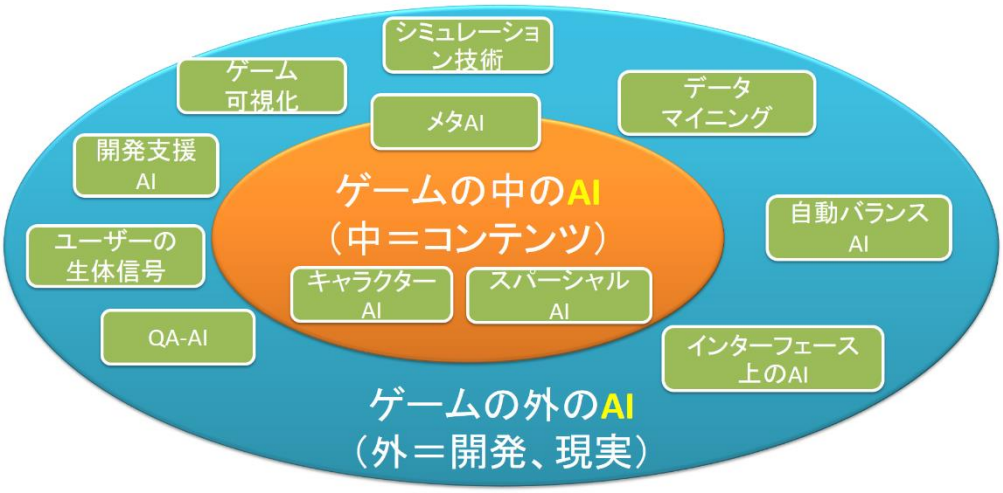


Fig. 2.3　ゲームの中の AI，ゲームの外の AI

2015 年以降では，ゲームのパラメータやデバッグ，QA の分野，つまりゲームの調整分野にラーニング技術が使われつつあり，さらに自然言語処理がたどたどしくではあるが，導入されつつある.「ゲーム品質保証のための AI」について第 2.8 章で述べる.

「MCS-AI 動的連携モデル」はこの「ゲームの中の AI」の 3 つ「メタ AI」「キャラクター AI」「スパーシャル AI」を動的に連携するモデルである.

2.3　デジタルゲーム AI の階層構造

本節ではデジタルゲーム AI 技術全体が持つ階層構造について述べる．ゲーム AI は，通

常のソフトウェア設計と同様に，小さなスケールから大きなスケールに向かって構造化され多層構造を取る（Fig. 2.4）．最下層にあるのは，アルゴリズム・レイヤーである．ここで，記号主義型とコネクショニズム型の双方のアルゴリズムの実装がある．記号主義的な「意思決定アルゴリズム」，「ニューラルネットワーク」，「遺伝的アルゴリズム」，「自動生成技術」（プロシージャル技術）などがここに属する．記号主義的な「意思決定アルゴリズム」については第 7.3 章で，学習・進化技術については，この後，第 2.6 章で述べる．プロシージャル技術については第 13 章で述べる．

第二層はモジュール・レイヤーである．モジュールは単体，或いは複数のアルゴリズムを組み合わせて実装される．「センサー・モジュール」「意思決定モジュール」「認識モジュール」など人工知能の部分となるモジュールがアルゴリズムから形成される．

第三層はアーキテクチャ・レイヤーである．複数のモジュールをアーキテクチャの上で組み合わせる．アーキテクチャの役割は，モジュールを組み合わせることで，知的機能を実現することである．「エージェント・アーキテクチャ」（第 3.3.1 項で説明）や「ブラックボード・アーキテクチャ」（第 3.3.2 項で説明），「サブサンプション・アーキテクチャ」（第 3.5.3 項で説明）などが用いられる．

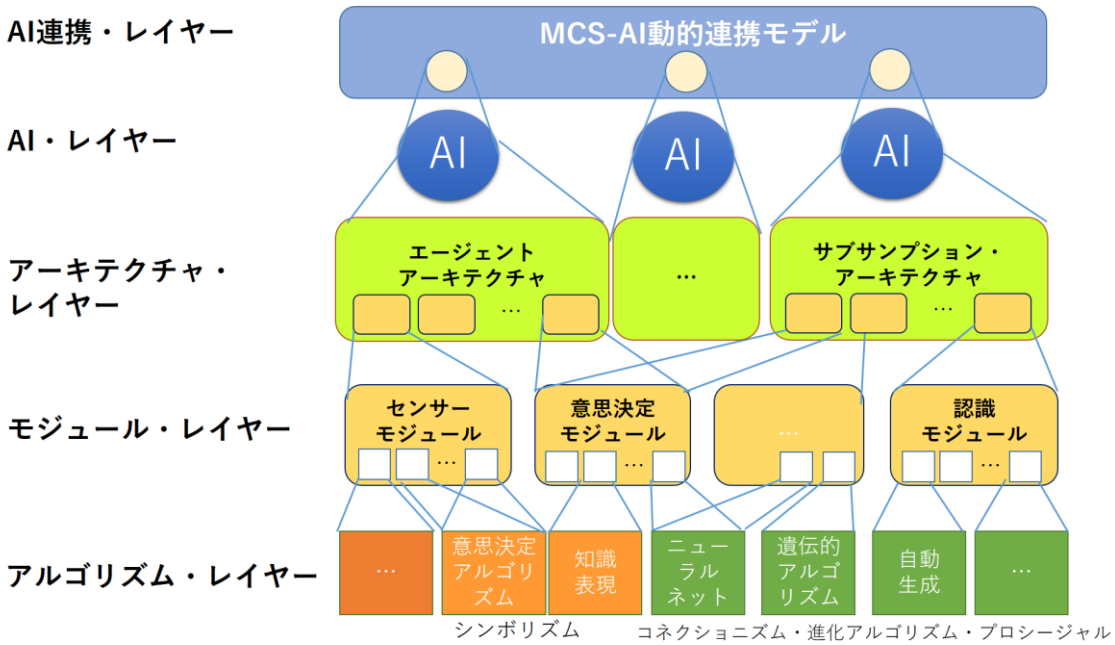


Fig. 2.4　ゲーム AI の多層構造

第四層は AI・レイヤーである．ゲームの環境世界と「アーキテクチャ」をつなぐことで，「アーキテクチャ」はそれぞれゲームにおける AI となる．「メタ AI」「キャラクターAI」「スペシャル AI」はこのレイヤーに属する．

第五層は AI 連携層である．AI を複数，動的に連携させて一つのシステムを作り出す層である．「MCS-AI 動的連携モデル」は，この層に属する．AI を複数連携させるという意味に

おいて，マルチエージェント・モデルもこの層に属する．

アーキテクチャ・レイヤーまでは既知の知見である．AI・レイヤーにおいて，キャラクターAI は，エージェント AI の一形態であり，スパーシャル AI は，ナビゲーション機能・空間認識・状況認識を集合させた AI である．メタ AI はデジタルゲームの歴史の中で形成されて来た AI である [24]．類似の概念としては「ファシリテーター」があるが，これは複数のエージェントを協調させる役割のことである [25]．デジタルゲームにおいても AI 連携・レイヤーにおいては，キャラクターの協調は「マルチエージェント・モデル」として知られているが，MCS-AI 動的連携モデルは，エージェントではなく，3 種の異なる AI の連携システムである．

このように，「MCS-AI 動的連携モデル」は，アルゴリズムレベルからアーキテクチャレベルまで，人工知能技術をまとめ上げて活用する技術である．また「MCS-AI 動的連携モデル」は AI 連携モデル・レイヤーの仕組みであると同時に，ユーザー体験を形成するシステムである．

現状（2020 年 3 月において）は，ほとんどの場合，MCS-AI 動的連携モデルはアルゴリズムレベルでは，シンボリズム（記号主義）のアルゴリズムを基礎にしており，コネクショニズムや学習・進化・プロシージャル技術のアルゴリズムを用いる場合が少数である．その理由は，これらのアルゴリズムを用いると，階層構造が作りにくくなること，と，計算結果が予想しにくく，MCS-AI 動的連携モデルが不安定になるからである．学習・進化・プロシージャル技術のアルゴリズムの MCS-AI 動的連携モデルへの取り組みは，本論文でもいくつの例を紹介するが，今後の課題として残されることになる．これについては，第 13 章で述べる．以下に，学習・進化アルゴリズムのデジタルゲームへの応用をまとめる．

2.4 MCS-AI 動的連携モデルの特徴と優位性

本節では MCS-AI 動的連携モデルが持つ特徴を，他のモデルと比較することで示すことで，MCS-AI 動的連携モデルの優位性を示す．特に，メタ AI，キャラクターAI，スパーシャル AI との関係について考察する．第 2.4.1 項では，メタ AI の機能を，他の AI に置き換えた場合に，どのようなメリットが損なわれるかを述べる．第 2.4.2 項では，スパーシャル AI を他の AI に置き換えた場合に，どのようなメリットが損なわれるかを述べる．これによって，それぞれの領域で最も発展した要素を組み合わせたモデルが，MCS-AI 動的連携モデルであることを述べる．

2.4.1 メタ AI 部分を他の AI に置き換えた場合

MCS-AI 動的連携モデルのメタ AI は，メタ AI 以外の可能性としてファシリテーターAI，チーム AI，AI Director などが考えられる．ここで，MCS-AI 動的連携モデルと，そのメタ

AI をファシリテーターAI，チーム AI，AI Director に入れ替えたそれぞれのモデルとの比較を行うことで，本モデルの優位性について述べる．ファシリテーターAI は，各キャラクターAI を調停する役割を持つ．ゲーム内でキャラクター同士の関係の調停を行う．チーム AI は，それに属するキャラクターについて管理を行う．各キャラクターから報告を聴き，命令を与える．ファシリテーターAI はあくまでキャラクターの調整モジュールとして存在するが，チーム AI はチームリーダーとして各メンバーを統率する [25] [26]．AI Director は，ゲーム状況やユーザー状態を認識し，各キャラクターに指示を与える．メタ AI の影響先をキャラクターに限定した AI である [27]．AI Director，メタ AI はゲームの外にある情報まで使用するが，ファシリテーターAI，チーム AI はあくまでゲームの中の AI として実行される．

次にキャラクターAI において上位 AI を持たないマルチエージェントの場合を考える．マルチエージェント・システムは各キャラクター間のコミュニケーションによる協調が考えられる．この場合，一つの方向にキャラクターを促すことは不可能ではないが難しくなる．

メタ AI は，ゲームの外の情報，ユーザーの生体情報やこれまでプレイ履歴などまで含めてゲームの流れを作ることが可能であり，この点はマルチエージェント・モデル，ファシリテーターAI，チーム AI，AI Director と異なる点である．また，ファシリテーターAI，チーム AI，AI Director は，ナビゲーション AI，スパーシャル AI との連携を持たない．これはファシリテーターAI，チーム AI，AI Director が常にキャラクターAI への抽象的な命令を行うだけであることに起因する．メタ AI に至って，ナビゲーション AI，スパーシャル AI との関係が確立し，ナビゲーション AI，スパーシャル AI の機能がメタ AI 内に活かされる．

このように，ゲーム要素を制御する AI は，チーム AI，ファシリテーター，AI ディレクター，メタ AI として拡張され，より大きな範囲を動的に制御できるようになった（Table 2.1）．

Table 2.1 ファシリテーターAI，チーム AI，AI Director メタ AI，マルチエージェント・モデルの比較

ゲーム要素を制御する AI	影響範囲	スパーシャル AI との関係
メタ AI (MCS-AI 動的連携モデル)	ゲーム全体 プレイヤーの心理	全キャラクターの空間認識に必要な情報を提供する自律的連携
AI ディレクター	キャラクターのみ プレイヤーの心理	プレイヤーの経路予測・位置解析のクエリー利用
ファシリテーター	キャラクターのみ	なし
チーム AI	チームのキャラクターのみ	なし

2.4.2　スパーシャル AI を他の AI に置き換えた場合

MCS-AI 動的連携モデルにおいて，スパーシャル AI を他の AI に置き換えたモデルについて考察する．スパーシャル AI はナビゲーション AI の発展形であり，その発展の仕方には段階があり，各段階において他の AI との関係が異なる．まず各キャラクターがパス検索としてナビゲーション機能を持つ場合，これはアルゴリズムやライブラリのレベルでありAI としては独立していなかった，次にナビゲーション AI として独立する場合に，メタ AI，キャラクターAI とメタ AI との AI 同士の連携となった．さらにスパーシャル AI は自律化し，サポートに関係なく自律的に環境を探索・解析する AI となり，同時，キャラクターAI，メタ AI へのサポートを行う．パス検索モジュールから始まり，サポートを行うナビゲーション AI，さらに自律型のスパーシャル AI へと発展することで，より大規模に，より詳細に空間的特徴を提供できる AI となった（Table 2.2）．

Table 2.2　空間に関する AI の他の AI との関係性の変化

空間に関する AI	キャラクターAI との関係	メタ AI との関係
スパーシャル AI	自律型 AI 同士の連携	自律型 AI 同士の連携
ナビゲーション AI	サポート	サポート
パス検索モジュール	キャラクターAI に属する	モジュールを利用

このように，３者の AI の連携には，さまざまな連携があるものの，それぞれが自律性を獲得しながら，それぞれの軸で扱う領域を伸長して行った．各領域の組み合わせの事例が存在する（Fig. 2.5）．『Killzone 2』（Guerrilla Games，2009）は「チーム AI」「自律型キャラクターAI」「ナビゲーション AI」の組み合わせによって，個々のキャラクターの状況に応じた知的な動きを実現しつつ，チームとして統率の取れた動きを実現している．『Halo2』（Bungie，2004 年）は「メタ AI」「自律型キャラクターAI」「ナビゲーション AI」の組み合わせによって，FPS のゲームジャンルにおいてキャラクターの知的な動きが高く評価され，質の高いゲームプレイによって人気を博した．『LEFT 4 DEAD』（Valve Software，2008 年）は「AI ディレクター」「自律型キャラクター」「ナビゲーション AI」の組み合わせによって，ゲームの緩急のリズムを操作するシステムを作った．それぞれの領域で最も発展した要素を組み合わせたモデルが，MCS-AI 動的連携モデルである．

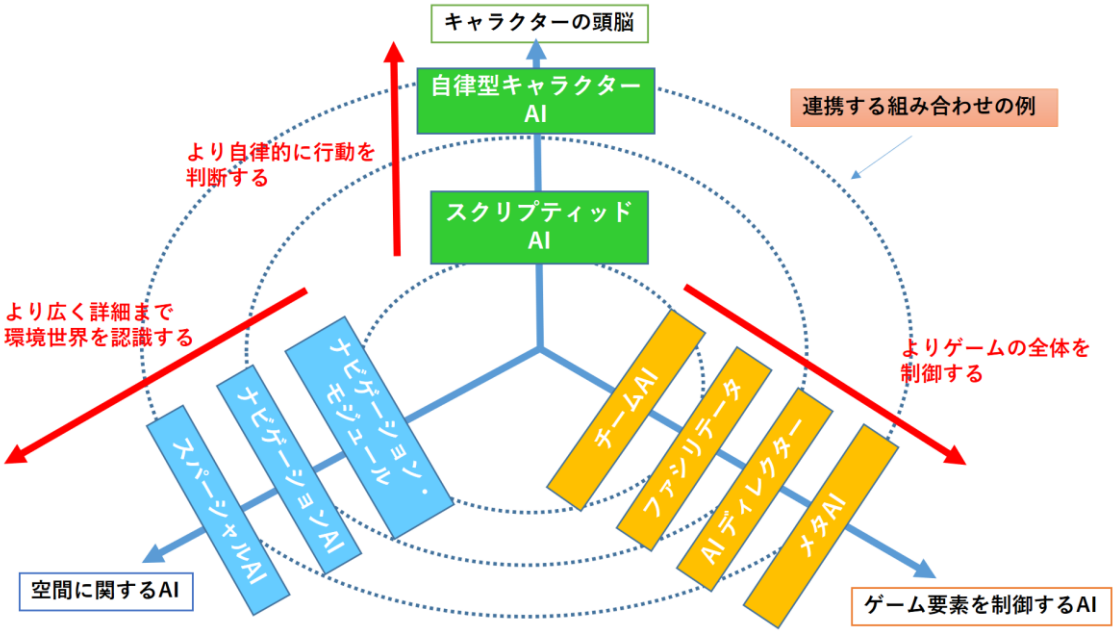


Fig. 2.5　各 AI 領域の発展と組み合わせ

2.5　MCS-AI 動的連携モデルの限界と発展

本節では，MCS-AI 動的連携モデルの発展とその限界について述べる．まず MCS-AI 動的連携モデルは，ゲームのオープンワールド化の拡大に対して構築されたモデルである．少なくともオープンワールドまでのゲームについては適応できるモデルである．次にオープンワールド化した以降のゲーム，つまり現実空間を用いるタイプのゲームについて考察する．

メタ AI の限界は二つあり，一つはプレイヤーという人間を完全にはモデル化できないこと，もう一つはゲームが発展して現実空間や現実のユーザー情報を使ったゲームへと発展して行くときに，現実世界を完全にもまたモデル化できない，という点である．メタ AI は，現実とゲーム世界の中間にあるが，現実世界に対する人工知能の認識の弱さがそのままメタ AI の弱さになる．たとえば AR 空間での現実空間の要素を完全に取り込むことはできず，またソーシャルゲームにおけるように現実のユーザー同士の関係を認識してゲームに取り込むことができない（Fig. 2.6）．

ゲームが現在のように閉じられた世界の中で起こる限り，不確定要素は人間の行動だけである．しかし，ゲームが拡張し，現実世界を舞台にするようになると，メタ AI は現実世界と直面することになる．しかし，メタ AI が現実世界を把握する力は限られており，このモデルの限界を示すことになる．スパーシャル AI もまた，現実空間ではその認識に限界が出てくる．すると，キャラクターAI についてもまた，その認識が確実なものから確率的なものへと変化せざるを得ない．

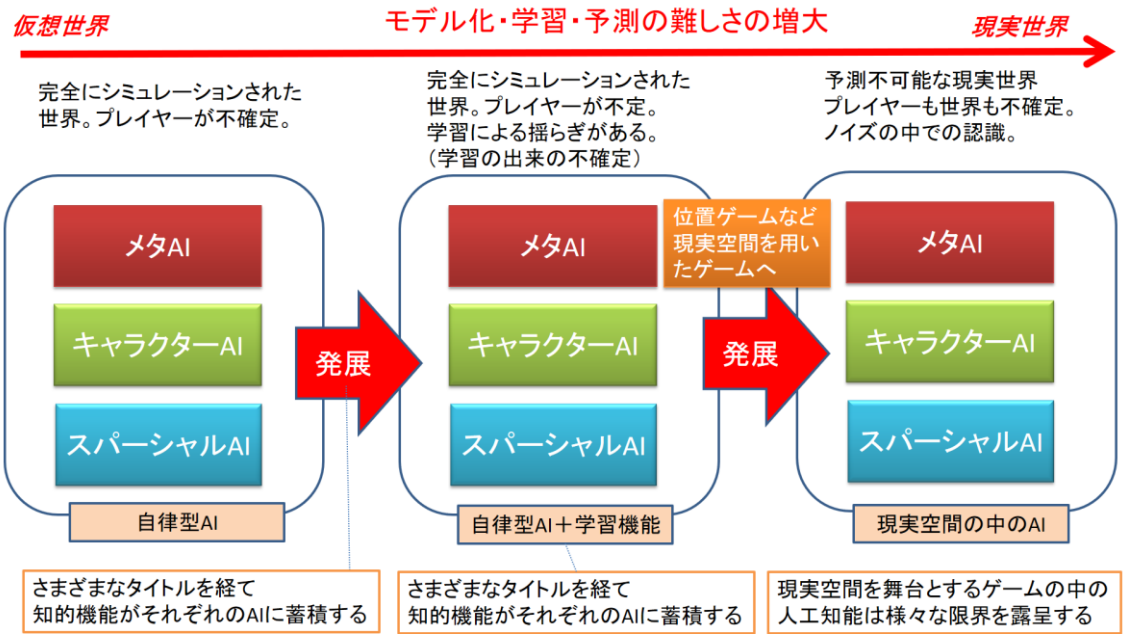


Fig. 2.6　3 つの AI の限界

ここからは，MCS-AI 動的連携モデルの発展について述べる．ゲーム AI 技術は，第 2.1 節で述べたように，機械学習を取り込みつつ発展している．ゲーム AI の各モジュールは自律化と共に，学習機能を持つようになると予想される．キャラクターAI は，強化学習などを利用してゲーム世界の試行錯誤の中から意思決定を学習するようになる．例えば，マイクロソフト・リサーチによる格闘ゲーム『Teo Feng』（Studio Gigante, 2003 年）を用いた研究では，Q ラーニングを用いて，プレイヤーとの戦闘の中から最適な攻撃の選択を学習し，その学習に基づいた意思決定が行われる [28]．このような学習事例については次の第 2.6 節で後述する．スパーシャル AI は，キャラクターをレベル上で自在に動かすことで，キャラクターの身体とその運動に応じた環境情報を抽出するようになる．また，それぞれの AI には，それまで各ゲーム AI に開発されてきたモジュールを蓄積されることになる．メタ AI もまた，ゲーム全体の状況とアクションの結果から学習することができる．

2.6　学習・進化アルゴリズム

本節では，学習，進化，プロシージャル技術のデジタルゲームへの応用について全体像を述べる．学習，進化，プロシージャル技術（自動生成技術）は，ゲームコンテンツを動的なものから動的なものへ変化させる手法である．通常，ゲームはデータとプログラムからなり，データはゲーム開発中に作成するものであるが．これらの技術は，これまで固定された部分を，インタラクティブで動的なシステムに変化させる．これらの 3 つの技術を用いるためには，ゲーム開発の設計レベルで十分に練られていることが前提であり，正確に組み込まれ

た時には，ゲームに大きな変化をもたらす．

デジタルゲームにおいて「学習・進化アルゴリズム」が組み込まれたタイトルは多くない（Table 2.3）．1995 年から 2020 年まで年間数例程度である．開発工程に組み込まれる例は増加しているが，タイトルそのものに入れることはゲーム全体を不安定にさせる代償となる導入のメリットを出すことが難しいことが理由として推測される．以下の節でそれぞれの技術のデジタルゲームへの導入について説明する．「プロシージャル技術」については第 13 章で述べる．第 2.6.1 項ではニューラルネットワークのゲームについて，第 2.6.2 では強化学習について，第 2.6.3 項は進化アルゴリズムについて，第 2.6.4 項はモンテカルロ木探索について述べる．各項は独立した項である．

Table 2.3　学習・進化アルゴリズムのゲームへの代表的な応用事例

年代	タイトル	手法と適用先
1996年	『Creatures』（Millennium Interactive）	ニューラルネットワークによるプレイヤーの指示からの学習
1997年	『がんばれ 森川君2号』（muumuu、SIE）	ニューラルネットワークによるキャラクターAIの学習
1998年	『アストロノーカ』 （muumuu，スクウェア・エニックス）	遺伝的アルゴリズムによる敵AIの進化
2000年	『ここ掘れ！プッカ』（muumuu，SIE）	ニューラルネット，逆伝播法
2003年	『くまうた』（muumuu，SIE）	歌詞／メロディのプロシージャル生成
2004年	『Tao Feng』（Studio Gigante）	Q-Learning（強化学習）による敵AIの対プレイヤーの適応進化
2005年	『Forza Motorsport』（Turn 10 Studios）	機械学習（統計学習）によるプレイヤーの挙動の学習によるドライビングAI（Drivatar）の生成
2010年	『Supreme Commander 2』 （Gas Powered Games）	ニューラルネットワークによる敵選択
2010年	『Galactic Arms Race』 （Evolutionary Games）	2Dシューティングゲームにおけるcontent-generating NeuroEvolution of Augmenting Topologies（cgNEAT）による武器自動進化
2014年	『Killer Instinct』 （レア社，Iron Galaxy Studio）	ケースベースストリーゾニングによるプレイヤーの挙動を学習したキャラクターAIの生成
2015年	『Fable Legends』（Lionhead Studios）	モンテカルロ木探索によるチームの挙動デザイン
2018年	『Race for the Galaxy』 （Temple Gates Games）	ニューラルネットワーク，TD学習によるキャラクターAIの生成
2019年	『Blade & Soul』（NCSOFT）	ニューラルネットワーク、強化学習によるキャラクターAIの学習
2019年	『サムライスピリッツ』（SNK）	ニューラルネットワークによるプレイヤーの挙動を学習したキャラクターAIの生成

2.6.1　ニューラルネットワーク

デジタルゲームにおけるニューラルネットワークの導入は，キャラクターの学習において導入が検討される場合が多い．ニューラルネットワークはセンサーからのインプットと，アウトプットを身体運動に直結させることで，環境からの学習をモデルで行うことができる利点があるが，直接的なコントロールや微調整によるカスタマイズができないために，ゲ

ーム開発に適さない場合が多い．またニューラルネットワークの学習はノード数に応じたメモリと計算リソースを必要とするために，ゲームに組み込むにはニューラルネットワークの計算量の見積もりと管理が必要となるために，他の AI 技術と比較して優先度が低くなる．

しかし，1990 年代後半になると，ゲーム機の性能がそれなりに上がり，ニューラルネットワークを用いた人工知能アルゴリズムが使われるようになる．『Supreme Commander 2』（Gas Powered Games，2010 年）では，キャラクターにパーセプトロン型ニューラルネットワークが埋め込まれている（Fig. 2.7） [29] [30]．入力は周囲の敵の情報であり，出力はどの敵を攻撃するか，という判定である．数体の敵に囲まれた時，どの敵を攻撃するか，という問題はルールベースで記述すると少し複雑過ぎる．そこでニューラルネットの入力に数体分の敵の体力，スピードなどを入れて，出力として「最も弱い敵を倒す」「最も近い敵を倒す」などを決定する．バックプロパゲーション法（逆誤差伝播法）によって 1 時間学習させる．

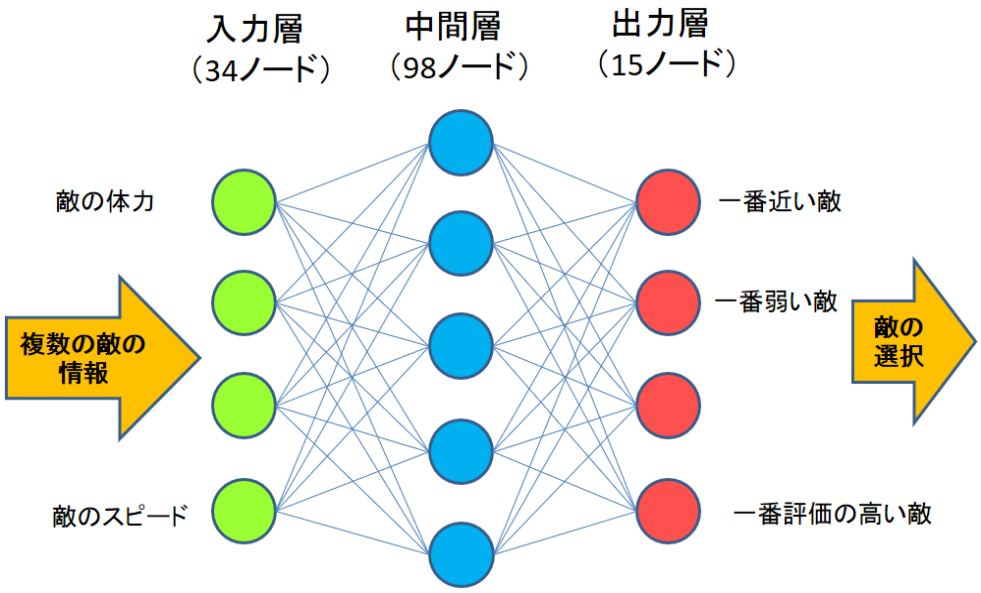


Fig. 2.7　『Supreme Commander 2』におけるニューラルネットワーク［30］

『Creatures』（Millennium Interactive，1996 年）は，数千ノードの大規模なニューラルネットワークを持つクリーチャーに物の使い方を教えるゲームである［15］[16]．このゲームは「ノルン」と呼ばれるキャラクターをユーザーの手で育てていくゲームであり，ゲーム内のオブジェクトの名前，行為の名前を学習一度学習すると，言葉を指定すれば特定の行為を行わせることができるようになる．ノルンの頭脳は，数千ノードからなるニューラルネットワークからなり，ニューロンを集めた集合体を「ローブ」と呼び，ローブとローブのそれぞれのニューロンを接続する形になっている（Fig. 2.8）．ユーザーがオブジェクトをノルンに見せると，それに対しノルンの注意ローブが反応し，ノルンはこの時，この対象に対す

る名前を言う．或いは，こちらが見せた行動に対して動作の名前を言う．正解を言った時に，ユーザーはマウスでなでてあげ，間違ったにはたたくことで，ゲーム内の名詞と動作の言葉をノルンは学習して行く．

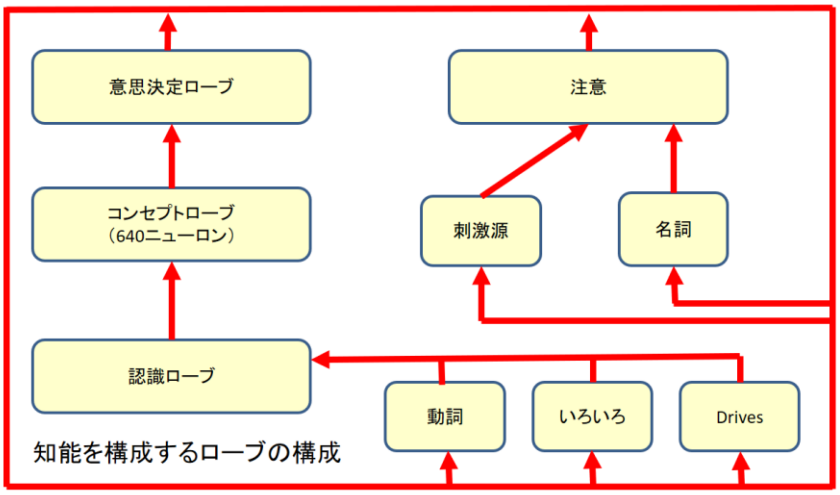


Fig. 2.8　『Creatures』ローブ（ニューラルネットワーク）全体図［31］

『がんばれ森川君 2 号』（muumuu，1997 年）は，キャラクターが発見した対象に行動を指定することで，キャラクターが対象から感じる感覚に応じた良い行動を取れるようにしていく．逆伝播法を用いたニューラルネットワークによる学習である［32］.

『ここ掘れ！プッカ』（muumuu，2000 年）は，プレイヤーが発掘してほしい石の種類を報酬によって教えていくと，次第に自分で判断できるようになります．逆伝播法用いたニューラルネットワークによる学習である［32］.

『Black & White』（Lionhead Studios，2001 年） では，ユーザーがクリーチャーをしつける．クリーチャーのＡＩはニューラルネットが入っており，学習によってクリーチャーが正しい動作を覚えてくれるようになる．クリーチャーの知能の全体は BDI モデルとなっている．さまざまな欲求が搭載され，ニューラルネットワークを通して意思決定を行い．その行いに対してユーザーが時にはプラスの報酬，時にはマイナスの報酬を与え，逆伝播法によってニューラルネットワークが変化して行く［33］.

『Blade & Soul』（NCSOFT, 2019 年）では，キャラクターの意思決定にニューラルネットワークを用いて，100 ミリ秒ごとに体力，距離，技発動までのクーリングタイムなどをセンサリング（インプット）し，アウトプットとして移動，技，攻撃などの行動を決定する．また，単に学習するだけはなく，攻撃的（Aggressive），中立的（Balanced），防衛的（Defensive）など戦闘スタイルを学習するために，各スタイルに合った「体力比」「距離の取り方」になっているかどうかを判定し，追加報酬（ペナルティ）を設定する［34］［35］.

このようにニューラルネットのゲームへの応用は一つのモジュールの中のピンポイント

の使用であった．それは安定性からも計算量からも最適な使い方であった．

2.6.2　強化学習

強化学習は環境の中で行為に対する報酬を設定し，設定した報酬に合った行為を学習して行く手法である．特に強化学習のアルゴリズムの 1 つである Q-Learning は時間的推移のある連続的な時間過程におけるキャラクターの強化学習に適したアルゴリズムである．キャラクターはその設定パラメータに沿って意思決定しアクションを選択する．アクションはゲーム状態に変化をもたらし，その行動の結果からパラメータを変更する（Fig. 2.9）．

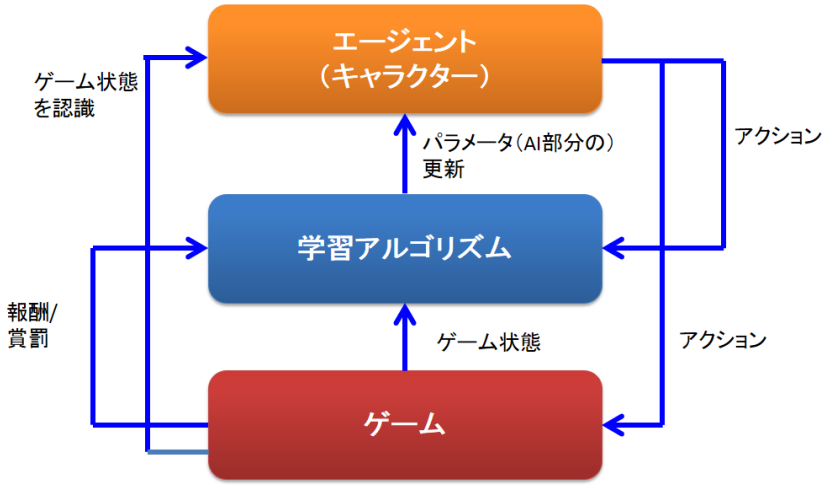


Fig. 2.9　ゲームキャラクターの強化学習の仕組み［36］

Microsoft Research（マイクロソフト）は 2004 年など極めて早い時期から，デジタルゲームにおける強化学習の研究に取り組んでおり，『Forza Motorsports』（Turn 10 Studios, 2005 年）では Drivatar と呼ばれるプレイヤーのドライビングの技術を学習してドライブを行うゴーストを生成する機能がある［36］．『Forza Motorsports』ではコースがセグメント（コースの一部分）に分割され，そのセグメントごとに理想のコースが設定されており，その理想のコースからの「ずれ」を計測する．Drivatar は，そのズレを再現するように運転モデルを学習する．さらに Microsoft Research は格闘ゲーム『Tao Feng」（Studio Gigante, 2003 年）の上で，Q 学習を持つ AI キャラクターと人間のプレイヤーの対戦から学習させる研究を行い，実際に強化される結果を得た［28］．敵と自分の位置，速度関係の状態に対してアクション（パンチ，キックなど）を選択し，敵キャラクターの HP の減り具合を報酬として学習を行う．一方で，相手の攻撃を至近距離で避けた時に報酬を与えることで，上手に避ける学習が実現された．

『逆転オセロニア』（DeNA，2016 年）は，オセロをベースとした戦略的な対戦ゲームアプリであり，各コマが独自のスキルを持っており 6×6 の盤の中で勝利を競う戦略的なゲー

ムである．新規のキャラクターをリリースする際には，対戦棋譜は存在せず，バランスを定量的に可視化することが難しくなっているため，AI の学習環境であるゲームシミュレータを構築することで，エージェントの自己対戦による「深層強化学習」によって検証を行う[37]．キャラクターのリリース前にこのような学習／可視化を行うことによって，意図していない強力な使い方がないか，特定のデッキに入った時に強くなりすぎないかといった確認ができ，高頻度でゲームに新要素が追加され，なおかつ検証する組み合わせの多いソーシャルゲームのタイトルで有用である

このように強化学習は，主にゲーム研究，ゲーム開発における調整，オプショナルな機能開発に応用されてきた．

2.6.3 進化アルゴリズム

遺伝的アルゴリズムを始めとする進化アルゴリズムは，「開発工程において優れた NPC を作り出す」「ゲームの中で NPC を進化させる」などの応用例が想定される．進化アルゴリズムは集団に対するアルゴリズムであり，そのためにメモリと計算量が母集団に比例して大きくなる傾向がある．そこで，特に後者においては消費したリソースに対するゲーム的効果が求められる．

開発中の使用方法として有効な手法の研究例を示す．Kenneth O. Stanley 氏によって開発された，遺伝的アルゴリズムとニューラルネットワークを組み合わせた方法，N.E.A.T.（Neuro Evolution of Augmenting Topologies）を適用したゲーム「NERO」が公開されている［38］．各兵士のニューラルネットワークの入力は環境情報で出力は身体のコントロールである．そして，壁で囲われた空間に数十体を入れて戦わせ，撃退数が多くかつ，長く生き延びたほど優秀な兵士としてスコアリングし，そのスコアを基にエリート戦略を取って次の世代を作るが，その際に「N.E.A.T」によってニューラルネットワークのトポロジー自体が変化する．N.E.A.T.はゲーム内のコンテンツ生成にも使用される［39］.

ゲームの中の応用例として，『アストロノーカ』（muumuu，1997 年）は，遺伝的アルゴリズムを用いてバブーという敵 NPC を進化させる（Fig. 2.10）［32］．プレイヤーは野菜を育てるが，これを食べにくる害虫キャラクター「バブー」を撃退するために，野原にトラップをしかける．トラップ空間はグリッド状にマス目に区切られた空間で，プレイヤーはそこに扇風機や案山子や落とし穴といったトラップをしかけていくが，トラップを乗り超えて奥まで行けたバブーは，行った場所までに応じてボーナス点を獲得する．これがバブーの持つ適応度に反映し，適応度が高ければ高いほど，子孫を残す確率が高くなる，バブーは身体特性（体力，耐久性，腕力，脚力）や各種トラップに対する耐性など 56 種からなるパラメータ群を染色体として決められており，これが遺伝子となり，交叉により新しい世代が生まれる，突然変異率は 3%に設定されている．

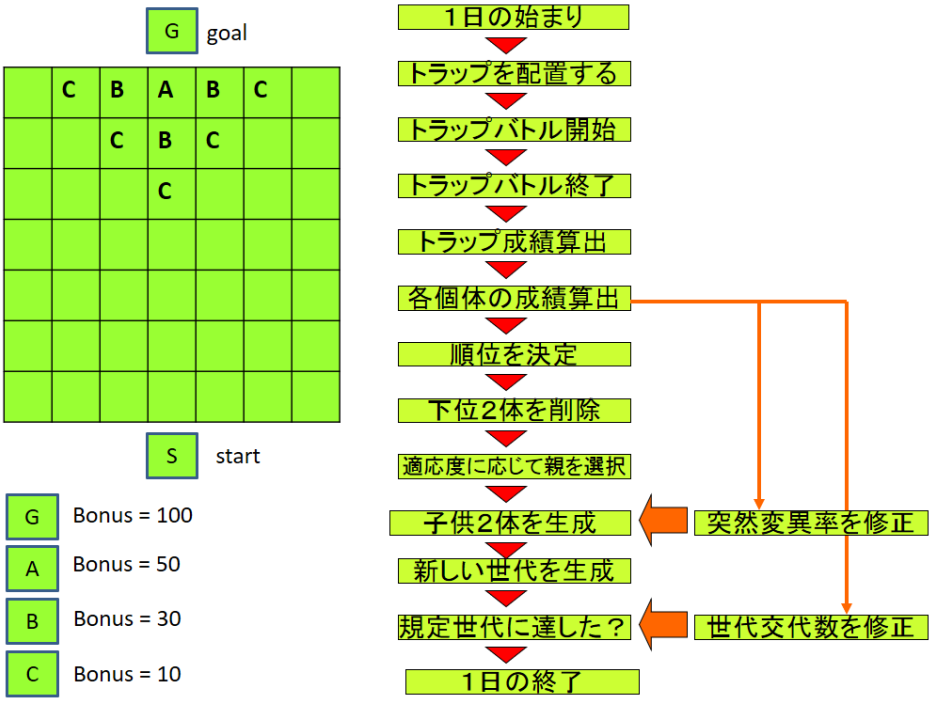


Fig. 2.10　『アストロノーカ』における遺伝的アルゴリズムのしくみ

バブーは 20 体存在し，ゲーム上では 1 体のバブーが 1 回のトラップに挑戦しているように見えるが，ユーザーが敵の進化を感じられるように，バックグラウンドでは 20 体のバブーを 5 世代分進化するようにシミュレーションさせている点が，ゲーム特有の工夫である．進化率が低い場合はさらに世代を重ね，逆に進化率が高い場合は世代交代数を減らし，常にユーザーが感じる敵の進化の割合を一定にする．

このように進化アルゴリズムは，ゲームの中，ゲームの外でも活用の可能性がある．しかし，シミュレーション・リソースと計算時間が必要なため，他のシステムとの協調がさせにくい問題があり，応用事例は少数である．

2.6.4　モンテカルロ木探索

モンテカルロ木探索（MCTS，Monte Carlo Tree Search）は 2006 年にコンピュータ囲碁で発明され，コンピュータ囲碁を飛躍的に強くした手法である［40］．この手法はシミュレーションを主とする方法で，評価関数が必要ないという特徴的な点がゲーム開発でも重宝され，デジタルゲームへの研究が多くなされている．モンテカルロ木探索は，候補の手が複数ある場合，それ以降のゲーム進行をすべて乱数シミュレーションで行い，その勝敗を評価値にして一手を選ぶアルゴリズムである．また，シミュレーションの途中で成績の良い手法に関してはシミュレーション回数を増やす UCB（Upper Confidence Bound）を指標とする手法が，アルゴリズムの要となっている．モンテカルロ法は乱数に基づくシミュレーション

であるが，モンテカルロ木探索は，ゲームツリー（ゲームの状態を記述）とモンテカルロ法を組み合わせた手法である．基本的には，今考えられる手に対してすべてランダムに行っていき，終局したら勝敗を返す．特に囲碁では大きな力を発揮し，「アルファ碁」(Deep Mind 社）にも基本アリゴリズムとして搭載されている．囲碁プログラムで多く採用されている [41] [42].

ゲーム状態を明示的に表現する必要のないモンテカルロ木探索は，デジタルゲームと相性が良い. たとえば, ストラテジーゲームでは『TOTAL WAR: ROME II』(Creative Assembly, 2013 年)，『Total War: Attila』（Creative Assembly，2015 年）で，キャンペーンモード(チュートリアルを含むシングルゲーム用モード)の調整に用いられている．ゲーム終局まで戦いを繰り返すことで，AI が取る戦術の効果を測定できる [43] [44].『TOTAL WAR』におけるゲームをプレイする人工知能の意思決定には以下の 3 つのレベルがある.「タスク生成」（複数の高度なレベルのタスクを設定する（これは MCTS を使わない))「リソース分配」（限られたリソースをタスクに分配する（MCTS を使う）「リソース配置」(MCTS ベースの意思決定プラナーがアクションシークエンスを決定する）を持つ AI がゲームをプレイする．このように,『TOTAL WAR』をプレイする人工知能がいったんできると，何千というコマンドを即座に発行しながら高速に大量にゲームをプレイさせることが可能となり，これによって，ゲームバランスの検証が人の手を借りず人がプレイするよりも圧倒的に高速なスピードで可能となる．

また『Fable Legends』（Lionhead Studios，未発売）では，複数のキャラクターの進行を決めるために，モンテカルロ木探索が用いられた．各位置をノードとしてそれぞれノードにおける行為を選択肢として，ツリー構造を考える．ここに，どのようにキャラクターたちを進めればよいかをシミュレーションしながら，自軍が最も優勢になり勝率が高くなる位置へキャラクターを移動する [45].

このようにモンテカルロ木探索は，アクションゲーム，ストラテジーゲームなどジャンルを問わずデジタルゲーム全般と相性が良く実用例も増加している．

2.7 ディープラーニングの応用

本節では，ディープラーニングのデジタルゲームへの応用について述べる．第 2.7.1 項では初期のディープラーニングのデジタルゲームへの応用について述べる．第 2.7.2 項では，産業応用事例についてまとめる．

「ゲームの中の AI」においてディープラーニングのゲームへの導入は，消費するメモリとリソースが他のプログラム要素と競合するため，学習自体をゲームに組み合み込むことは高度な調整が必要となる．一方で，制限の少ないゲーム開発工程やゲーム運営と言った「ゲームの外の AI」においてディープラーニングを使用することは現時点でも可能であり，そのような事例が増加している．特にゲームの外の AI はディープラーニングを試す絶好の

機会であり，さまざまな応用事例が増加している．

しかし，第 2.6.1 項で指摘したニューラルネットワーク，ディープラーニングのカスタマイズ性の弱さが解決されたわけではない．もしなんらかの不具合があった場合には，ほとんどの場合修正のためすべて学習をやり直す必要がある．そして学習の結果を思いどおりに扱うには熟練が必要とされ，部分的修正や高速なイテレーション（試行錯誤）が難しい現状にある．今後，ゲーム産業がゲームの中でディープラーニングを使いこなすためには，高速な学習によるイテレーション・サイクルの高速化と，カスタマイズ性の研究による強化が必要となる．以下で，初期のディープラーニングのゲームへの応用研究と，そこからゲーム産業における応用について述べる．

2.7.1 初期の応用研究

本項では，ディープラーニングのデジタルゲームへの導入の研究について述べ，次項 2.7.2 では，産業における応用について述べる．Deep Mind 社の Deep Q-Network（DQN）はディープラーニングと強化学習を組み合わせた方法である．1970 年代の 5 つの ATARI 社のゲームを，End-to-End（画像をインプット，コントローラの操作をアウトプット）として学習させ，ルールを教えることなく高得点を獲得する方向に強化学習を成功させた．さらに，DQN にモンテカルロ木探索（MCTS）を加えた方法によって，『アルファ碁』『アルファ碁ゼロ』（DeepMind 社）として，完全情報ゲームでプロ棋士に高い確率で勝利する囲碁 AI が生み出された [41]．

2018 年 8 月には，OpenAI(公開された AI 技術の集積を目指すプロジェクト)の「OpenAI Five」 [46]は，eSports で注目を集める MOBA（Multiplayer online battle arena）系ゲームの一つ『Dota 2』（Valve Corporation 年，2013 年）において，トッププロの人間チームに勝利した．その学習時間は，人間プレイヤーの 180 年分に換算される．

2018 年 12 月には不完全情報リアルタイムストラテジーゲーム『StarCraft2』（Blizzard Entertainment，2010 年）において，ディープラーニングで学習した AI「AlphaStar」が限定した条件ながらも（種族を固定，ゲーム情報を取得する API の提供），トッププレイヤーに勝利した [47]．

このようにディープラーニングのデジタルゲームへの初期の応用研究は，ディープラーニングの性能を，デジタルゲームを用いて試すことで，人間のプレイヤーとの差異を確認するために行われる傾向が強かった．

2.7.2 産業応用事例

本項では，ディープラーニングのゲーム開発，ゲームタイトルへ導入された産業応用事例についてまとめる．『Counter-Strike: Global Offensive』（Hidden Path Entertainment，Valve

Corporation，2012 年）は eSports 競技としての公平性のために，ユーザーのチートプレイを取り締まる必要がある．そこで，過去のチートプレイのログをディープラーニングによって学習し，チートプレイヤーの候補を探し出し，人間に報告する AI がゲームプレイを監視する［48］.

FPS（一人称視点シューティングゲーム，First Person Shooter）では，壁の向こうのキャラクターが透けて見える「ウォールハック」(Wallhack）というハッキングが存在する．『サドンアタック』(GameHi，2007 年）は，このハッキングを行っているプレイヤーを摘発するために，プレイヤーのプレイ画像を収集し，ディープラーニングによって学習させた［49］. この学習によって，AI は自動的に「ウォールハック」を判別し人間に報告する．チートプレイヤーはこれによって減少した

Ubisoft では，自動 3D アセットデータの分類にディープラーニングを用いた．3D アセットを回転させ画像を取り，その画像データをもとに，そのデータ／プロファイルを推定する［50］. そのために，さまざまなデータを用意し，画像とプロファイルの対応を学習させ，自動的に大量のアセットを分類できるディープラーニングをトレーニングした．

『サムライスピリッツ』の「道場モード」では，プレイヤーのプレイから学習を行う．ゲームステートをインプットとして，コントローラのキーをアウトプットとして，その間をニューラルネットワークで学習し，プレイヤーの行動パターンを習得した AI が生成される［51］.

このように産業におけるディープラーニングの応用は，かつては膨大な人月の工数のかかった工程を，ディープラーニングを用いた AI によって置き換えることが目指された．またプレイヤーログを学習する事例は，最初にディープラーニングがゲームの中で活用された事例となった．

2.8　ゲーム品質保証のための AI

本節ではデジタルゲームの品質保証プロセスへの AI の応用の全体像を述べる．品質保証プロセス（QA，　Quality Assurance）は，ゲーム開発の後半から完成へ向けて，「仕様通りにゲームが作られているか」，「バグがないか」，の二点を確認するためのプロセスである．ゲームの大規模化／複雑化に対して，ゲームテストを行う「テスター」と呼ばれる人員を増やすことで対応して来たが，一タイトルにつき最大数百人を管理するコストにも限界があり，2015 年ごろから，品質保証の工程を人工知能によって自動化する研究が，世界中のゲーム産業で推進されて来た．ゲーム品質保証に AI を利用する取り組みのうち代表的なものを **Table 2.4** に示す.

このような自動プレイボットはゲーム依存の部分がほとんどであるために，同じシリーズ以外に再利用は難しい．しかし，欧米の AAA タイトルの開発スタジオは同一のシリーズや，類似したタイトルを開発することが多く，再利用が十分に見込めるという見通しがある.

一方で日本のゲーム開発では同一のプロダクションで多種多様なゲームを作るケースのほうが多くある．

「人工知能によるデバッグ／品質保証の自動化」の分野が発展した背景には，ゲームの複雑化，巨大化，拡張性が，ゲームテストの難易度を上げてきた要因が大きい．この要因はますます大きくなっているため，膨大なテストを毎日人手で繰り返すのは不可能である．ゲーム開発が序盤を超えて，ステージ量産フェーズの中盤に入ると，とたんに人間の手には負えない領域まで，デバッグ・品質保証の難易度が上がる．この難易度の急激な変化が，品質保証プロセスへの人工知能の導入を困難にしている．つまり，あるところまで手の内にあると思っていたゲームの品質保証の作業が，手に負えない作業に突如として変貌する．この時点から，人工知能を開発しても間に合わない．そのために，開発の早い段階からの品質保証のための人工知能の開発を始める必要がある．人間の手に負えなくなった品質保証の作業は，人工知能に代替させる必要がある．

デバッグ，品質保証の自動化には 2 つの考え方が存在する．作業を完全に人工知能に任せるという方向と，人工知能が人間をサポートして作業効率を何倍にもするという方向である．自動化のためには簡単な問題から，難易度の高い問題もある．難易度の高い問題は人工知能のサポートに基づく人間の領域として残しておいて，徐々に簡単な問題を人工知能によって克服していくことが重要である．長い時間をかけて，徐々に品質保証の作業を，人間の手から人工知能に完全に委ねていくのが，この分野の技術発展の仕方である．そこで直面する問題は，どれだけ多くのプレイ可能な領域をカバーしているか，というカバレージが性能の指標の一つであり，多数のプレイを重ねて統計を上げる，自動プレイのバリエーションを増やす，など，様々な工夫が試されている．

『Candy Crash』（King，2012 年）では，1000 を超えるレベル（マップ）を作成する必要があり，それらのレベルがゲームデザインに合っているかを検証するために，AI にゲームプレイさせた [52]．遺伝的アルゴリズムによってニューラルネットのトポロジーを進化させる「ニューロエボリューション」を用いて，モンテカルロ木探索（MCTS）のプレイアウトを改良し，効率良く一つ一つのレベルをクリアしていく．

『Horizon Zero Down』(Guerrilla Games, 2017 年)では,「Apollo -Autonomous Automated Autobots」（自律型自動 bot）というシステムによって，キャラクターAI によってゲームを自動的にプレイし負荷を計測し，それがヒートマップの形でビジュアリゼーションされるシステムである．ヒートマップ上では負荷の高い点が赤く表示されるとともに，自動プレイ，あるいは人がプレイしたビデオが添付されており，すぐに再生できるような総合システムを構築し，包括的なソリューションを実行している [21]．

『ベヨネッタ 2』（プラチナゲームズ，2014 年）では，デバッグと品質保証のために「オートプレイ」が実装されている．ゲームステージに目印となるコーンを立てることができ，コーンおよびコーンの間のアクションや，コーンのあるポイントからポイントへの移動とその間の行動をスクリプトで定義する．これによって，人間の代わりにゲームを繰り返し自

動プレイさせる［53].

Table 2.4　ゲーム品質保証（QA，Quality Assurance）のための AI の実例年表

開発会社	年	システム	詳細
Cygames	2016	総合システム構築	OpenCV、Python、Appiumの組み合わせ
King	2016	『Candy Crash』における自動プレイによるマップ検証	モンテカルロ木探索とニューロエボリューションによるマップ検証
DeNA	2016	『FINAL FANTASY Record Keeper』における自動プレイ	ニューロエボリューションによるプレイヤーAIの作成
SQUARE ENIX	2017	『グリムノーツ』における自動ゲームバランス	遺伝的アルゴリズムを用いてプレイヤーAI群を進化させてゲームバランスを調査する
DeNA	2017	『逆転オセロニア』における自動ゲームプレイ	強化学習を用いたニューラルネットワークによるオートプレイを用いたバランス検証・調整
DELiGHT WORKS	2017	『Fate/Grand Order』における自動リプレイ	サーバーを経由したログの収集とコマンド関数列の再現
RARE	2017	『Thief』におけるUnreal Engine上のキャラクタービヘイビアの自動テスト	テストがクエリーの形でリスト化されて、毎晩テストされる
SEGA	2018	『龍が如く』～『北斗が如く』における自動プレイ	ログからの自動リプレイシステム
SEGA	2018	『D × 2真・女神転生 リベレーション』『コトダマン』開発・運営支援	ニューラルネットワークと遺伝的アルゴリズムを組み合わせ, シミュレーションによって『D × 2真・女神転生 リベレーション』のクエスト調整支援・マップ設計支援,『コトダマン』デッキ調整支援
コナミデジタルエンタテインメント	2018	カードゲームにおける画像認識	ディープラーニングによって8000種類のカードを認識する
EA	2018	『Battlefield 1』における模倣学習による自動プレイ	模倣学習によるキャラクターがゲーム内で戦い合う
Guerrilla	2018	『Horizon Zero Down』の自動プレイ	毎晩、自動的にAIキャラクターがゲームをプレイ
Ubi	2018	『Assassin's Creed Origins』のレベルアセット自動検証	スクリプトによるオブジェクトどうしの干渉テスト、キャラクターの生成ポイントと配置オブジェクトの干渉テスト／スクリプトによるテスト
Valve	2018	『Counter-Strike: Global Offensive (CS: GO)』のチート対策AI	ディープラーニングを用いて、ユーザーのログを解析し、チートを自動的に発見、排除
Nexon Korea	2019	『サドンアタック』のウォールハック対策へのディープラーニングの応用	壁の向こうのキャラクターが透けて見える「ウォールハック」をディープラーニングで学習させて自動検出する。
Ubi	2019	自動3Dアセットデータ分類	3Dアセットデータをディープラーニングで認識して管理
Ubi	2019	『The Division』におけるオートプレイ	既存のシステム、特にビヘイビアツリーを用いたBotによる自動作成
EA	2019	『Battlefield V』におけるオートプレイ	rostbite Schematics を用いたオートプレイによるカバレージの高い自動デバッグ

『The Division』（Ubisoft, 2019 年）は，もともとキャラクターAI 用に開発したビヘイビアツリーを使ってゲームを通してプレイできるようなボットを作成している［22].

『バトルフィールド V』（EA, 2019 年）は，マップ上に設置した大量のボットをポイントに沿って動かし, 戦闘し破壊を続けることで負荷をモニターしている. その AI は「Frostbite」ゲームエンジンが持つビジュアルスクリプティング環境 Frostbite Schematics を用いて記述

されている [23].

このように品質保証における AI は，学習・進化アルゴリズムを用いる方向と，ゲームの中の AI を再利用する場合がある．多方面のアプローチが試されているが，その成果はまだ部分的なものである．

2.9 考察

本章の前半（第 2.1 節-2.5 節）では，「MCS-AI 動的連携モデル」のデジタルゲーム AI 分野全体の中での定義を行った．「MCS-AI 動的連携モデル」はゲームの中の 3 つの AI「メタ AI」「キャラクターAI」「スパーシャル AI」を動的に連携させるモデルであり，AI 連携層に属する．そこで，様々なアーキテクチャ，モジュール，アルゴリズムを内包することになる．本章の後半（第 2.6 節-2.8 節）では特に学習・進化アルゴリズム，ディープラーニング，品質保証のための AI など，MCS-AI 動的連携モデルが内包するが，実際には未だ取り込めていない技術について述べた．これらの学習・進化・プロシージャル技術を取り込んだ MCS-AI 動的連携モデルの変化については，第 13 章で述べる．次章は，デジタルゲームの全体的構造について述べる．

3. デジタルゲームの構造

本章ではデジタルゲームの基本的原理と，本論文の前提となるデジタルゲームの人工知能の基本的知識を示す．第 3 節ではデジタルゲームの基本事項を述べる．よく使われる用語の中でも本論文で用いられる用語の解説，デジタルゲームの動作原理・内部構成などである．第 3.2 節ではデジタルゲームの人工知能の既存の知識を列挙する．第 3.3 節では，自律型キャラクターAI の内部について述べる．内部の構成は既存の人工知能の組み合わせであるが，デジタルゲーム特有のカスタマイズが為される．第 3.4 節では，キャラクターAI，メタ AI がゲーム世界を認識するための「知識表現」(Knowledge Representation, KR）について述べる．知識表現はデジタルゲームにおけるもっとも基本的な層となる．第 3.5 節はキャラクターAI とキャラクターの身体の関係について，キャラクターAI，スパーシャル AI の関係を軸に述べる．第 3.6 節は「ゲームの外の AI」について概説する．本章はすべて第 4 章以降，解説の前提となる知識ではあるが，既知の場合は飛ばして読んでも構わない．

3.1　デジタルゲームの基本事項

本節はデジタルゲームの基本事項を述べる．第 3.1.1 項では，デジタルゲームの動作原理を解説する．第 3.1.2 項では，デジタルゲームを構成する要素を列挙する．第 3.1.3 項では，ボードゲームとデジタルゲームの比較を行うことで，デジタルゲームの人工知能に課せられる諸条件を明確にする．

デジタルゲームは，スクリーンなど投影画面に画面を投影しつつコントローラやセンサーによってユーザーからのインプットを受け付け，インタラクティブにキャラクターやゲーム要素を操作するメディアである．デジタルゲームは，スクリーンの中に新しい現実を作り出す．これは「ゲーム世界」，或いは「ゲーム環境世界」と呼ばれる．ゲームユーザーは，コントローラのボタンを押し，レバーを倒したりしながら，ゲーム世界を変化させ，その変化がスクリーンに反映される．デジタルゲームはゲーム世界の中で新しい経験をユーザーに与えるメディアであり，ゲーム固有のユニークな体験をユーザーに与えることが必要である．物理シミュレーション，コンピュータ・グラフィクス，人工知能，など，さまざまな要素を融合させつつ，新しく深く記憶に残る体験とできるかどうかが，エンターテインメントとしてのゲームを成立させられるかどうかの分岐点となる．

デジタルゲームは，ユーザーの内面からさまざまな現実感（リアリティ）を引き出すかによって，ユーザーの体験を構成する．ゲームタイトルの狙いに沿ったリアリティをユーザーの内側から自発的に構成させることゲーム開発の中心的な課題である．デジタルゲームのユーザー体験の可能性を探求し，ユーザーが求める体験を提供するのがデジタルゲームの開発である．ゲーム開発技術は様々なユーザー体験を作り出す科学であり，グラフィクス，

サウンド，ＡＩなどを複数の技術要素を組み合わせて各ゲームタイトル固有のユーザー体験を産み出す．ユーザー体験には楽しい・怖い・感動する・興奮するなど単純な言葉を超えた様々なバリエーションと奥行きがある．デジタルゲームにおける人工知能技術はその技術要素の一つである．

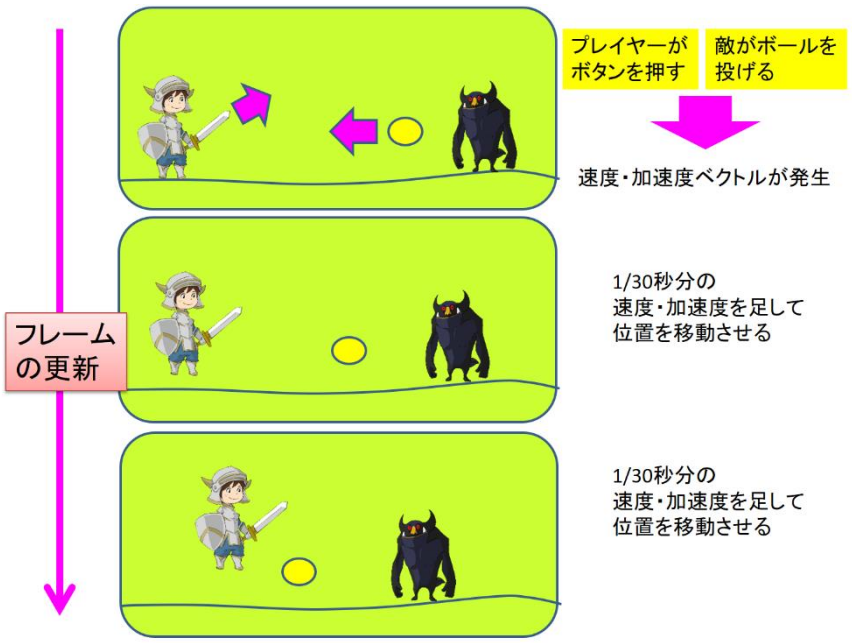


Fig. 3.1 デジタルゲームの動作原理

3.1.1 デジタルゲームの動作原理

デジタルゲームの動作原理について解説する．デジタルゲームは 1/30 或いは 1/60 秒を単位とする疑似的なリアルタイム・インタラクティブ・システムである．デジタルゲームは，画面を一秒間に 30 回，或いは 60 回更新する（Fig. 3.1）．一画面を「フレーム」と呼ぶ．つまりデジタルゲームは一秒間に 30 フレーム，60 フレームを更新する．ゲーム世界の更新や，コントローラからのインプットも，フレームを単位として行う．例えば，プレイヤーが走りながら A ボタン（ジャンプボタン）を押した時，右上のベクトルが発生するとする．これを一フレームごとに分割して，たとえば 30 分の 1 に分割して徐々に足し合わせて行く（図 1.1）．秒速 3m であれば，10cm を一フレームごとに足し合わせる．もちろん重力があれば，それも 30 分の一秒の間の加速度として足し合わせる．このようにして，キャラクターやマップ上のオブジェクト（物体）が移動する．プレイヤーの場合はプレイヤー操作が，NPC の場合はその人工知能が，オブジェクト（物）は外からの衝撃が，それぞれ運動を駆動し，物理シミュレーションの中で運動が展開される．

3.1.2 デジタルゲームの内部構成

本項ではデジタルゲームの内部構成を説明する.「レベル」(ゲームマップ) 上にある物(無生物) はオブジェクトと呼ばれる. 例えば, 武器や, 岩, アイテム, 家などである. レベルを構成する地面は地形 (テレイン, terrain) と呼ばれる. 地形には森や草, 壁や橋など移動できない対象も含まれる. キャラクター, オブジェクト, 地形など, ゲームを構成する要素はエンティティと呼ばれ, エンティティが集まってレベルが構成される. NPC の配置・動作, 地形, オブジェクト配置全体をデザインすることをレベルデザインと呼ぶ.

エンティティ間にはインタラクションがそれぞれ定義される. 衝突や回転, 重力や風や水の流れなどである. 例えば箱を川に置くと流される. 爆風で葉が揺れる, 衝突でドアが陥没する, などの現象がゲーム内でシミュレーションされる.

エンティティは, それを動かすプログラムと, 表現するデータからなる. このデータはアセットと呼ばれる. 3D モデル, テクスチャ, シェーダー, 衝突モデルなどである. アセットは主にアーティストによって作られ, プログラムは主にエンジニアによって記述される. プログラムの代わりに, 簡易的なスクリプト言語は, エンジニアかレベルデザイナーによって準備される.

3.1.3 ボードゲームとデジタルゲームの人工知能の比較

デジタルゲームにおける人工知能がどのような問題を内包するかを示すために, ボードゲームの人工知能との比較を行う. ここでボードゲームと言えば, 囲碁・将棋・チェス・チェッカーなどターンベースで区切られた盤上で駒を動かすゲームを指すこととする.

アクションゲームの人工知能は, 将棋や囲碁, チェス, チェッカーと言った古典的なボードゲームにおけるゲーム AI の技術の発展形であり, 多くが共通する. しかし大きな相違点もあり, それはデジタルゲームの本質と深い関わりを持っている (Table 3.1). そこでデジタルゲームの中でもアクションゲームとボードゲームを比較することが最もボードゲームとデジタルゲームの差異を明確にすると考え, ここではボードゲームとアクションゲームの人工知能の違いを明らかにする.

ボードゲームとアクションゲームの人工知能の導入場所が異なる. ボードゲームは,「ゲームをプレイする人工知能」であるため, ゲームの外に「プレイヤー」の人工知能として存在する. 将棋であれば将棋を指す AI, 囲碁であれば囲碁を打つ AI であるから, それらの AI は盤上, つまりゲーム内に現れない. しかし, アクションゲームの人工知能は, ゲームの構成要素の一つとして存在する. 敵であり, 味方であるキャラクターの人工知能であり, そのような人工知能は常にゲームの世界の中に存在する. 以下, それぞれの項目についての相違を示す.

Table 3.1　ボードゲームとアクションゲームの人工知能の違い

	ボードゲーム	アクションゲーム
ゲーム	既にある． （AI はゲームの外にある． ゲーム要素に AI は含まれない）	開発当初はゲームがない （ゲームと一緒に AI を作る ＝AI はゲームの一部）
ステージ(空間)	盤の目・離散的	3 次元地形・連続
時間	ターン	連続時間 （リアルタイム，1/30,1/60sec）
操作対象	駒	キャラクター/モンスターの身体
アクション	駒を動かす	身体を動かす
AI	プレイヤーとしての AI	キャラクターAI/メタ AI/ スパーシャル AI
ゲーム表現	ゲームツリー	ゲームの知識表現
状況	離散的変化	連続的変化
AI の目的	勝利	楽しませる．ゲームを成立させる．
ゲームのつながり	厳密にすべての手がつながっている（ツリー検索が有効）	一定時間，一定区間で区切れる． ランダムな要素も多い，
何を作るか？	賢い AI，　面白い AI	ユーザーの主観的体験（UX）のため．AI そのものが目的ではない．

時間と空間

AI が直面する時間と空間に着目する．AI が思考対象とする時間と空間の性質は，人工知能の特性を決定する．まずボードゲームの空間は離散空間である．通常，空間がグリッドやマスに分割されている．一方で，デジタルゲームでアクションゲームが展開される空間は連続空間である．平原や坂道やお城など，ゲームの中に連続的な世界が構築されている．

次に時間に着目する．ボードゲームはターン制であり，相手や自分が交互の手を打ち，考えている間ゲームは静止している．つまり時間は離散的である．一方で，アクションゲームは一秒間に 30 フレーム，或いは 60 フレームが標準であり（このフレーム数だとゲーム内のキャラクターやオブジェクト運動が滑らかに見えることが知られている），1/30 秒，1/60 秒を単位として微小な時間の間の運動を記述する．この短い更新時間によるゲーム更新は人間の目には連続と言ってよい時間である．映像では 1/24 秒が一コマになるが，ゲームはインタラクティブであるから，瞬時に反応した，と感じさせるためには 1/24 秒より短い更新が必要とされ，1/30 秒よりも短い更新が設定されている．

このように，ほとんどのボードゲームは「離散空間，離散時間」，デジタルゲーム（アク

ションゲーム）は「連続空間，連続時間」という違いがある．「離散空間，離散時間」の中の人工知能は「ゲーム状態の分岐ツリー」を作り，局面を追って行くことで最善手を見つける，というアプローチが一般的であり，様々な探索アルゴリズムが研究されている．

しかし，アクションゲームのように「連続時間，連続空間」になると，ゲームツリーや位置評価関数が膨大な点，膨大な時間の要素を含むようになり，そのままでは表現できない．つまり「ゲームの状態＝人工知能の考える状態」が不可能となる．そこで，アクションゲームでは，まず「ゲームをいかに表現するか」が問題となる．これを「ゲームの知識表現」，特に空間表現については「ゲームの世界表現」と言う．この「ゲームの知識表現・世界表現」がデジタルゲームの AI の基本であり，さまざまな表現形式が探求されてきた．これについては 3.4 章で後述する．

行為

アクションゲームのキャラクターの人工知能の最初の課題は，レベルデザインの中で空間をうまく用いた移動・運動を可能にすることである．たとえば狭い隙間を，体勢を変えつつ通り抜ける，助走をつけて崖をジャンプするなどのことを，デジタルゲームのキャラクターに行わせることは，とても複雑で難解な問題を含んでいることは，ロボットの行動生成と同様である．一方，ボードゲームにおけるアクションは，「駒を移動する」ことであり．とてもシンプルである．

身体

キャラクターのアクションとは身体を伴うアクションである．そこで「歩く」にせよ「弓を引く」にせよ，ゲーム内の環境と身体の空間的な組み合わせが必要となる．即ち足がどこに接地するか，また，弓の引く肘が壁に当たらないか，机の下にスライディングをするなら，そこにスライディングで通り抜けるのに十分なスペースがあるか，などである．このようにアウトプットする身体動作は常に環境の中でいかに可能か，ということを考慮しなければならない．一方，ボードゲームの場合，駒には身体的意味がなく，座標を示すマーカーであり，環境との直接的な物理的衝突を考慮する必要はなく，座標のみが必要である．

このようにアクションゲームの人工知能が思考する領域の性質と，ボードゲームの人工知能が思考する領域には明確な違いがある．これらの性質からボードゲームの AI は「最善手を求める論理的思考」の傾向が強く，アクションゲームの AI は「限られた時間と複雑な情報の中でそれなりの行動を見つける」傾向となる．アクションゲームの AI の場合は，瞬時瞬時の動作が連続的に積み重ねって行くことで形成される全体としての行動の意味が重要である．或いは，後述するように長時間に渡る戦略を決める場合，あるいは，キャラクター自身が自分の目的地（位置取り）を決める場合には，ボードゲームの AI と似た思考を行うこととなる．

3.2　ゲームの中の AI の全体像

本節では，デジタルゲームの中の AI の全体像を提示する．80 年代，90 年代前半ではゲームシステム部分と AI が，グラフィクス，メニュー画面処理，スコア処理，破壊処理など，各種ゲーム機能のコード群の中で混在していた．そこから 90 年から 2010 年にかけて 3 つの独立した人工知能「メタ AI」「キャラクターAI」「スパーシャル AI」へと独立した．それぞれの人工知能には役割があり，「メタ AI」はゲーム全体の状況・進行を監視し，形成・調整する機能を持ち，ゲームのあらゆる要素，キャラクター，地形，イベント，天候，建築に干渉してゲームを変化させる．「キャラクターAI」はキャラクターの頭脳であり，意思決定から身体運動を決定する．また「スパーシャル AI」は，主に目的地までの経路（パス）を計算するパス検索，キャラクターの目的と能力に合わせた目標地点を見出す．

Table 3.2　ゲーム AI リスト

AI の種類	目的
キャラクターAI	環境の中で賢いふるまいをする
メタ AI	ゲーム状況を調整・創造する
スパーシャル AI	キャラクターに必要な空間の認識情報を提供する
ナビゲーション AI	パス検索などキャラクターの移動に必要な情報を提供する
ボディ・レイヤー	身体の状態を調整する．ボディ・レイヤーとも呼ぶ．
モーション	キャラクターの身体をアニメーションさせる
フェイシャル	キャラクターの表情を変化させる
会話 AI	キャラクターの会話を制御する
プロシージャル AI	ゲームアセットを作り出す
スマートオブジェクト	オブジェクトからキャラクターを制御させる

ゲームの中の AI には他にもさまざまな AI がある（Table 3.2）．これらは相互に動的に結びつくが，大きく 3 つに分類することができる．「ボディ・レイヤー」「アニメーション」は「キャラクターAI」に属する AI であり，全体としてキャラクターを制御する．「フェイシャル AI」「会話 AI」も，ゲームが要求する場合には，キャラクターAI の一部として導入される．「プロシージャル AI」は，ゲームシステムに属する場合もあるが，メタ AI のエフェクターの一つとしてメタ AI に含まれる．

これらの AI 技術は大きくは「メタ AI」「キャラクターAI」「スパーシャル AI」に属して使用される（Fig. 3.2）．プロシージャル AI（自動生成 AI）は，メタ AI から使用されてゲームを自在に作り出す．これについては第 10 章で述べる．身体動作に関わる AI「ボディ・レイヤー」「モーション」「フェイシャル」はキャラクターAI に内包される．これについては

第 3.5 節で説明する．ナビゲーション AI はスパーシャル AI の一部として含まれる．

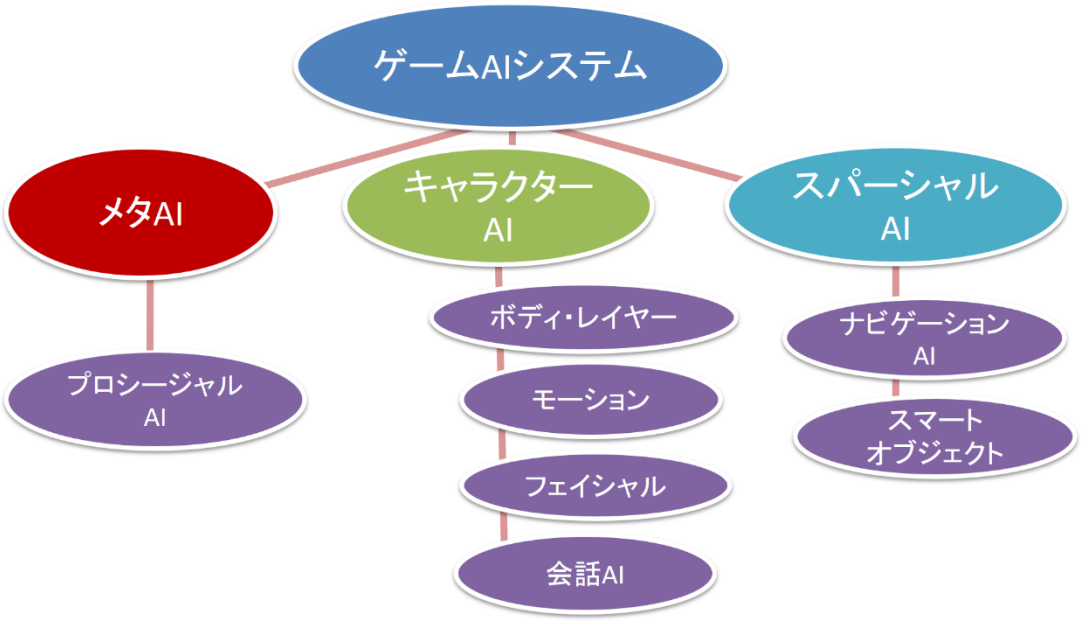


Fig. 3.2　ゲーム AI 組織図

以下，本論文では，「メタ AI」「キャラクターAI」「スパーシャル AI」に限定して考察して行く．この 3 つの AI については第 6，7，8 章で詳説するため，ここでは概説のみ示す．

3.2.1　　メタ AI

「メタ AI」はゲームの流れを作り出す役割を持つ．「メタ AI」は，ゲームの中と外をつなぐ要の役割を持つ．ゲームの外，というのは，現実世界と現実世界にいるユーザー（人間）を意味する．現実世界の状況や，ユーザーの身体的・心理的な状況を監視して，ゲームに反映する役割を持つ．メタ AI は，ゲームに干渉するためにさまざまなインターフェースを持つが，キャラクターAI に指令を送ることで，ゲームの状況を動的に変化させる．たとえば，キャラクターの目的を変更する，敵キャラクターの生成する場所・タイミングを動的に決定する，地形や天候を動的に変更するなど，ゲーム内のあらゆる要素をコントロールする能力を持つ．ユーザーの行動を予想できることがメタ AI の性能の指標の一つであり，その予想をもとにメタ AI はゲームに干渉する．たとえば，ユーザーの移動する予測経路に沿ってモンスターを待ち伏せさせる，などである．また，逆にユーザーをある一定の感情に導く機能を持つ．たとえばモンスターにユーザーを壁際まで追いつめさせることで緊張感を高める，などである．特にメタ AI のコントロール対象をキャラクターに制限した機能は，メタ AI がキャラクターAI の監督役になるため「AI ディレクター」とも呼ばれる [27] [54]．

3.2.2 キャラクターAI

「キャラクターAI」は，キャラクター全般のコントロールを担う AI である．主に意思決定，身体管理，モーションの決定の役割がある．ゲーム世界を認識し，抽象的な意思決定を行い，身体モーションを生成する．

モーションは，アニメーターが 3 D モデルをマニュアル（手動）で動かしてアニメーションを作ることもあるが，モーション・キャプチャー・スタジオ（MoCap Studio，と略される場合が多い）で人間や動物のアクターにマーカースーツを付けて演技させることで，実際のモーションをデータ化することも多い．しかし，ゲーム内ですべてのモーションを取ることはできない上に，モーションデータのサイズは大きく，メモリを圧迫するという問題がある．そこで膨大なモーションデータを用意する代わりに，一定のモーションデータから，空間に応じてモーションを変形する，或いはニューラルネットを用いて生成する手法がある [55]．それらはデータ圧縮という点でも，モーションのバリエーションを出す意味でも使用される．

3.2.3 スパーシャル AI

「ナビゲーション AI」の基本機能としては，始点と終点を指定し，その間のパス（経路）を返す「パス検索」（経路検索）である．「スパーシャル AI」（Spatial AI）は，ナビゲーション AI の自然な発展形であり，どのような複雑な地形であっても機能し，地形や空間を抽象化して捉える．地形・環境を専門とする知能であり，様々な地形の中で活動しなければならないキャラクターと地形・環境の間にあって，場所ごとの地形の複雑性を吸収する機能を持つ．キャラクターAI，メタ AI はゲームの地形データを直接解析することはない．ナビゲーション AI，スパーシャル AI に問い合わせることで，パスや目標点や地形の特徴を記したデータを受ける．

3.2.4 3つの AI の基本的関係性

ここで，これら 3 つの AI の成立に関わる基本的関係性について述べる．3 つの AI の成立の鍵となったのは「ナビゲーション AI」である．それまで，定められた領域（テリトリー）の中で決められた行動を取っていたキャラクターは「ナビゲーション AI」によって限定された局所から解放され，マップ全域を移動できるようになる．すると，キャラクターAI は長い時間の移動を考慮するため，短時間の外からの制御（「レベルスクリプト」或いは「スクリプティッド AI」）から長時間の自律的思考へと変化を促されることになる．また，ナビゲーション AI 自体もキャラクターの身体を複雑な地形に順応させねばならず，単なるパス以上の地形情報が必要となり，「スパーシャル AI」へと発展することになる．また，そのよ

うに大きな領域を自律的に移動するキャラクターが成立すると，今度はキャラクターの行動たちをゲームの提示する物語に沿わせることが必要となり，その役割を「メタ AI」が果たすこととなる．

3.2.5 メタ AI 以外の可能性についての考察

本項では，メタ AI，キャラクターAI，スパーシャル AI のモデル以外の AI システムの可能性について吟味する．第 1.7 節で述べたように，ゲームの中の AI に対しては知的な役割を持つことが AI の要素であるための条件である．そこで，ゲームのあらゆる要素に対して，それらが AI となる可能性を考える（Table 3.3）．

Table 3.3 メタ AI 以外の可能性

ゲーム要素	AI
レベル制御	メタ AI
敵・味方キャラクター	キャラクターAI
地形・自然環境（風，雲，川，など）	スパーシャル AI
メニュー	メニューAI
ＵＩ（ユーザーインターフェース）	UI-AI(ユーザーインターフェース AI)
描画	描画 AI
システム	システムＡＩ

このうちレベルデザイン，すなわちゲームの舞台装置に関しては「メタ AI」の領域である．メニュー，ユーザーインターフェース（コントローラ），描画，システム（各種設定など）に関しての AI は未だ定義されていない．たとえば，メニュー画面で良く使う項目が上方や前面に移動する，或いはメニューの側から特定の状況ではあるメニューが開かれることが学習されリアルタイムに推薦される，という「メニューAI」が考えられる．ユーザーインターフェースに関しては，ある状況下であるボタンの押し方を学習し，プレイヤーの操作の癖を学習する「ユーザーインターフェース AI」が考えられる．ユーザーの代わりにインターフェースを操作する，ボタンの押し方が足らない場合には補完する，などが考えられる．描画系に関しては，普通は AI と交わらないところであるが，ユーザーの好みに応じた描画に変化する，ユーザーの色覚などに応じて変化する「描画 AI」が考えられる．システムに関しては，ネットへの接続，セーブ/ロード，各種セッティングに関して，キャラクターインターフェースなどで，対話的に案内をする「システム AI」などが考えられる（Fig. 3.3）．

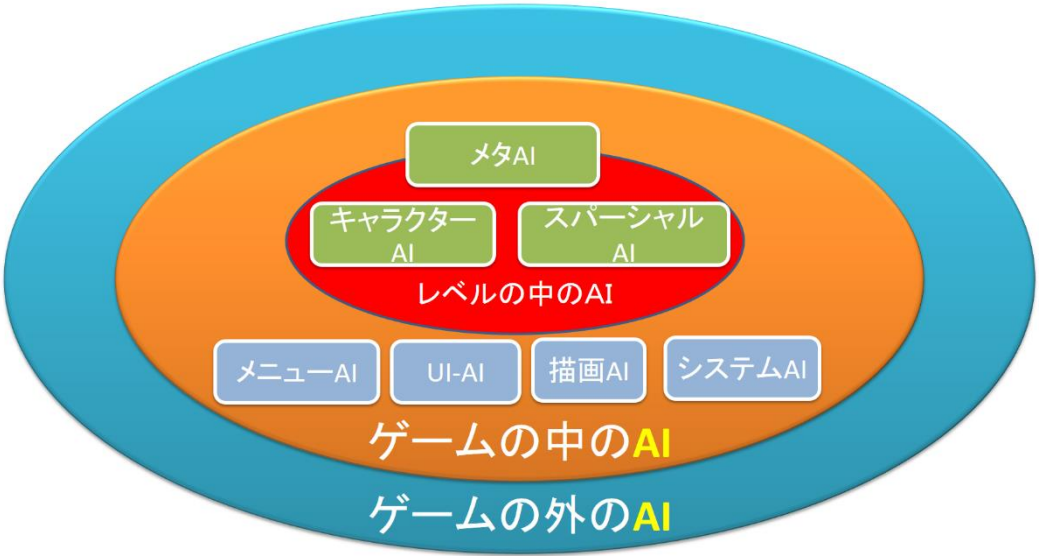


Fig. 3.3　ゲームの中のＡＩにおける階層

これらの AI に関しては, 現在は実現されていないが, これから実現される可能性はある. しかし, これらはリアルタイムに進行するレベルの中ではなく, メタ AI, キャラクターAI, スパーシャル AI とは直接関連しない. メタ AI, キャラクターAI, スパーシャル AI は「レベル内の AI」であるが,「メニューAI」「インターフェース AI」「描画 AI」「システム AI」は「レベル外の AI」と定義することができる.

3.3　自律型キャラクターAI の成立

本節では, キャラクターAI の自律モデル（autonomous character AI）について, それを支える技術と共に述べる. 自律型キャラクターAI とは, 外側から指示を与える操り人形のような制御ではなく,「自分自身で環境の情報を獲得して, 自分自身で意思決定をし, 自分で行動を形成していく」自律型ＡＩである. そのための基礎は二つあり, 一つは, キャラクターAI の内部と外部の環境世界をつなぐモデルである「エージェント・アーキテクチャ」（Agent Architecture）, そして, エージェント・アーキテクチャ内部を構造化する「ブラックボード・アーキテクチャ」（Blackboard Architecture）の 2 つである. 以下, 第 3.3.1 項では「エージェント・アーキテクチャ」を, 第 3.3.2 項では「ブラックボード・アーキテクチャ」について述べ, 第 3.3.3 項では考察を行う.

3.3.1　エージェント・アーキテクチャ

本項では, エージェント・アーキテクチャについて述べる. エージェント・アーキテクチャは, MCS-AI 動的連携モデルにおいて, キャラクターAI とメタ AI 双方に適用されるアー

キテクチャである．第 2.3 節で言及したアーキテクチャの層に属する技術である．

環境世界と AI 内部を結ぶ全体の仕組みは，エージェント・アーキテクチャと呼ばれる．AI は内部構造を持ち，一方で環境は自然構造を持つ．AI の内部と外部を区別し，外部から情報を取得する「センサー」（感覚器），内部から外部へ影響を与える「エフェクター」（効果器）によって結びつけるモデルである［56］（Fig. 3.4）．このアーキテクチャはもともとロボットの AI の基本アーキテクチャであるが，仮想空間のエージェントの人工知能研究を通して 2000 年頃からゲームの人工知能に導入された［11］［57］．ゲームの場合はセンサー部分も先の知識表現や，或いは言葉そのものも入力として扱うことで拡張されている．エフェクターの部分も多様であり，キャラクターの身体や魔法，武器などがこれに相当する．

エージェント・アーキテクチャには 6 つのモジュール「センサー」「エフェクター」「認識」「意思決定」「行動生成」「記憶」からなる（Fig. 3.4）．世界と知能を結ぶのが，世界から刺激・情報を受け取る「センサー」，そして世界に対して影響を及ぼす「エフェクター」である．「認識」モジュールは，諸感覚から集められた刺激・情報を統合し一つの世界全体の表象を形成する．また「行動生成」モジュールにおいては様々な知能内部の流れが統合され，世界へアウトプットする一つの行動を形成する．「意思決定」モジュールは「認識」モジュールと「行動」モジュールの間を多層構造によって結び，下層であるほど反射的な（最下層は物理レイヤーと呼ばれる），上層であるほど抽象化された意思決定を行う．同時に平行した意思決定がされ，それらが統合されて次の「行動生成」モジュールにその決定が渡される．

環境世界，センサー，知能，エフェクター，そして環境世界，という循環を巡る「情報の流れ」が，人工知能を駆動する流れとなっている．このように自分で情報を取得し，行動するというアウトプットで世界とつながり，結果の変化をさらに自分で取得するということが，キャラクターを自律知能とさせる基本原理である．エージェント・アーキテクチャを流れる刺激・情報の流れは「インフォメーション・フロー」（Information Flow）と呼ばれ，流れる情報が各モジュールを励起して行く「データドリブン・アーキテクチャ」（Data-Driven Architecture）となっている［58］（Fig. 3.5）．

インフォメーション・フローは一旦形成されると，世界と知能を動的に結びつけ続けるヒステリシス機能を持っている．インフォメーション・フローは細い一本の流れではなく，様々な特徴を持つループが束になったものである．また，このように外部と内部を結ぶインフォメーション・フロー以外に，知能内部をめぐるインフォメーション・フローも存在する．これは様々な想起や内部思考に関連する情報の流れであり，内部循環インフォメーション・フローと呼ばれる［59］［60］．

エージェント・アーキテクチャのゲームキャラクターへの導入は 90 年代後半の MIT Media Lab の Synthetic Character Group［61］のバーチャル空間のクリーチャーの知能構造「C4 アーキテクチャ」モデル（図 5）［11］［62］に端緒を持ち，これが 2001 年のゲーム産業の国際カンファレンス「GDC 2001」（Game Developers Conference 2011）で発表され，『Halo』（Bungie，2001 年）のキャラクターの内部モデルとして導入された［58］．そこか

ら,『F.E.A.R』(Monolith Production, 2005 年),『Killzone 2』など大型の FPS に導入され,ゲーム産業全体に広がって行った [19] [63] [64] [65].

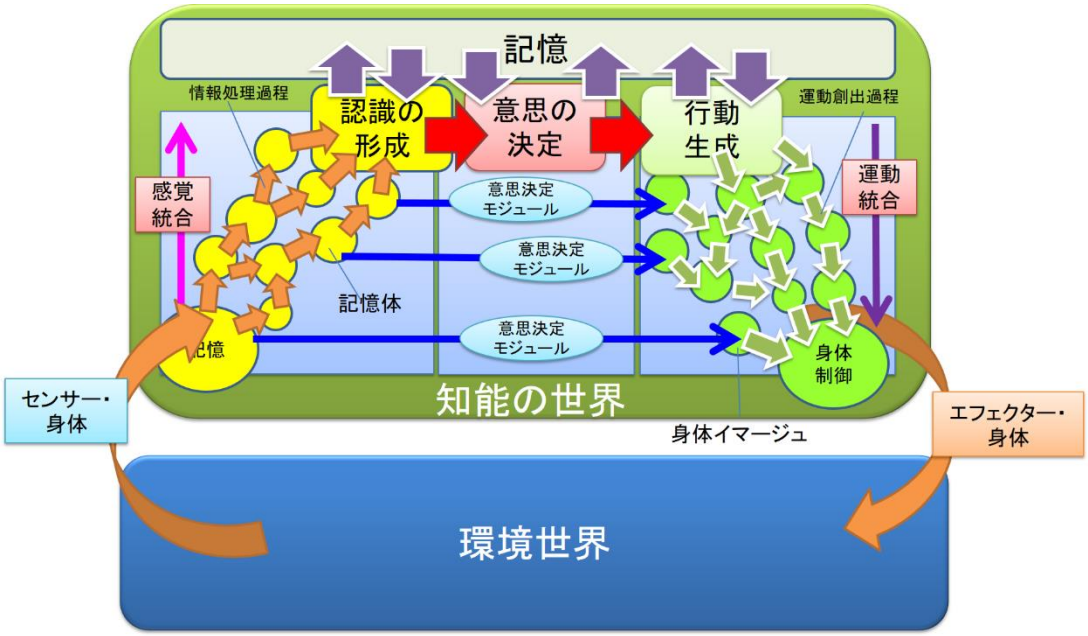


Fig. 3.4　エージェント・アーキテクチャ

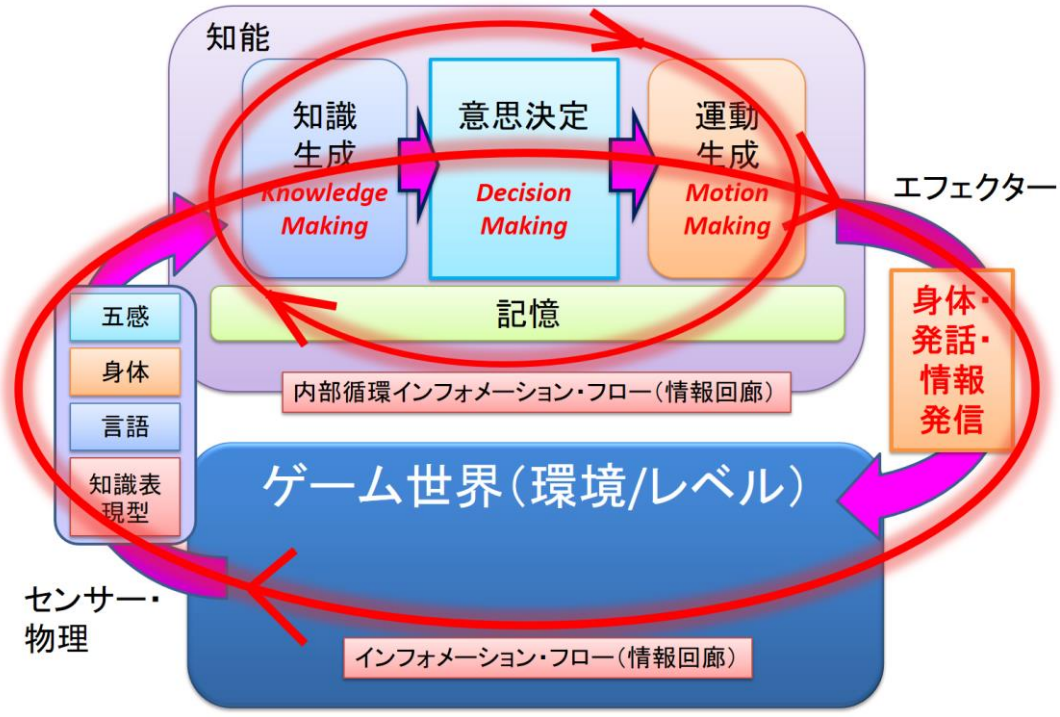


Fig. 3.5　エージェント・アーキテクチャとインフォメーション・フロー

またこのように全体を部品に分けて，組み上げて作る方式を「モジュール型設計」と言う．第 2.3 節で言及したモジュール・レイヤーに属する技術である．モジュール型設計はモジュール間のインターフェースさえ決められていれば，各モジュール内部を他のモジュールから独立して実装できるために，開発における柔軟性と設計の変更に伴う互換性に優れており，ゲーム開発ではモジュール型設計が基本となる．ゲームではキャラクターにそれぞれ役割が与えられている．プレイヤーの護衛であったり，敵であったり，ヒントを与える村人であったする．それぞれの役割を果たせるように，各モジュールが設計される．モジュール・レベルの設計モデルとしては前述した「C4 アーキテクチャ」やその発展形が用いられることが多い．以下の節で解説する．

3.3.2 ブラックボード・アーキテクチャの導入

「C4 アーキテクチャ」（Fig. 3.6） [11] [62]の特徴はブラックボード・アーキテクチャ（Blackboard Architecture）（Fig. 3.7）を基本構造としている．ブラックボード・アーキテクチャは 70-80 年代に発展し，分散した小型 AI を協調するための仕組みとして使用例も多かった [66] [67] [68]．90 年代では一端収束し，汎用的な技術となり，00 年代に至ってエージェント・アーキテクチャ内部の構造をモデル化するためにも用いられた．ブラックボード・アーキテクチャは 3 つの要素からなり，一つの機能に特化した人工知能 KS（ナレッジ・ソース，Knowledge Source）群，中央に KS たちが情報を読み書きするブラックボード，KS たちの動作を統御する調整モジュールであるアービター（Arbiter）である． KS 群は，それぞれ直接お互いにはコミュニケーションせず，ブラックボードの読み/書きを通じて間接的にコミュニケーションを行う（Fig. 3.7）．たとえば，あるデータを受け渡すには，一つのモジュールがブラックボードボード上のある領域に書き込み，別のモジュールがその領域を読み込む. また, 黒板側にも領域を区切って役割を持たせたものを領域黒板と呼ばれる [68]. 上位 AI であるアービターは KS たちの書き込みや読み込みの動作の順番を調整する. C4 アーキテクチャも意思決定に用いる専門的なモジュール群（Fig. 3.6，左列）を記憶領域（Fig. 3.6，右列）を介してつながる形式となっている．

C4 アーキテクチャは，ブラックボード・アーキテクチャをエージェント・アーキクチャに応用したものである．すなわち，思考モジュールを KS として実装し，記憶モジュールを黒板とする．アービターが消えているのは，ナレッジ・ソースの実行順番が固定されているからであり，思考モジュール同士は直接インタラクションすることはない．このようなアーキテクチャのメリットは各モジュールを独立に変更・革新できることである．即ち，柔軟性と拡張性に優れており，このような特性は，要求が拡大・変更されやすいゲーム開発では必須のものである．

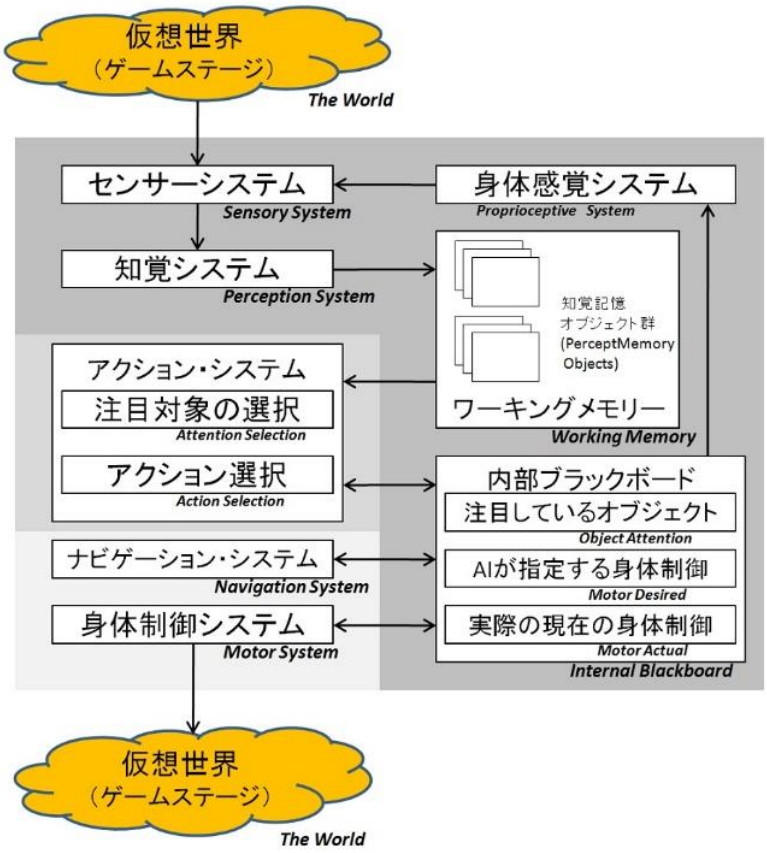


Fig. 3.6　C4 アーキテクチャ［62］

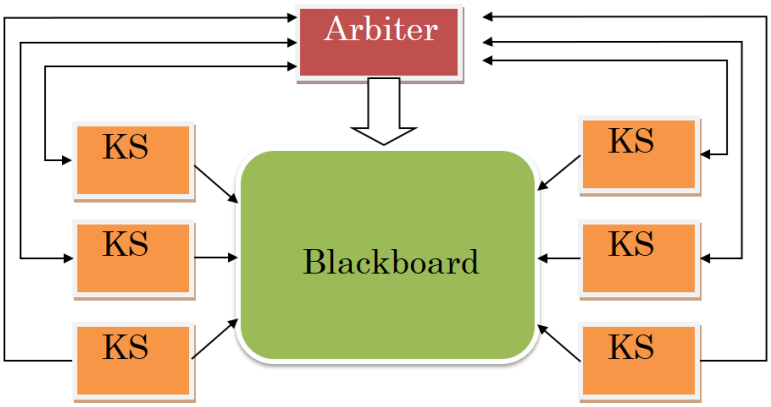


Fig. 3.7　ブラックボード・アーキテクチャ［69］

3.3.3　　考察

このようにエージェント・アーキテクチャとブラックボード・アーキテクチャが，自律型キャラクターAI の基礎アーキテクチャとして導入されて行った．特に後者は再利用可能なモジュール型設計を可能とし，ゲーム開発に合った構造を提供した．また自律型 AI の仕組

みはメタ AI の自律化にも用いられる．キャラクターAI の場合，キャラクターの身体周辺がセンサー領域，エフェクター領域であるが，メタ AI の場合，ゲーム世界全体をセンサー領域，エフェクター領域として，ゲーム世界の変化の方向を意思決定する．自律型 AI の導入によって，それぞれの AI が独立し，AI 同士の連携が可能となった（Fig. 2.4）．これは MCS-AI 動的連携モデルの基礎となるものである．

3.4　知識表現

本節ではデジタルゲームにおける知識表現について述べる．キャラクターAI やメタ AI とゲーム世界の間には，知識表現層が存在する．デジタルゲームのキャラクターAI，メタ AI は対象の持つ知識表現を通して対象を認識する（Fig. 3.8）．将棋や囲碁などボードゲームでは，盤面の表現がゲーム状態の知識表現となっているため，知識表現をしている，と強く意識することが少ない（Table 3.1）．しかし，通常のアクションゲームの場合には，連続的な地形，連続的な時間の中でキャラクターが活動することになる．そして，各キャラクターがゲーム環境世界を認識するためには，地形やオブジェクト自身がその知識表現データを持っておく必要がある．これは「ゲームの知識表現」（KR，Knowledge Representation）と呼ばれる．特に空間に関する知識表現は「世界表現」（WR，World Representation）と呼ばれる［70］［71］．知識表現はゲームでは幾つかの種類がある．世界表現，オブジェクト表現，アフォーダンス表現，身体の知識表現などである．世界表現は「スパーシャル AI」がゲーム世界の特徴を抽出する形式を与える．

「ゲームの知識表現」のさまざまな形式を以下に説明する．それぞれの知識表現を豊かで精緻にすればする程，AI のゲーム世界に対する理解を深めることができる．それゆえに，さまざまなタイトルの開発を通じて有用な知識表現，世界表現の形式が探求されて来た．それらは，タイトルを超えたゲーム AI 開発の汎用的な技術である．以下，それぞれの知識表現を解説する．第 3.4.1 項では「世界表現」について，第 3.4.2 項では「オブジェクト表現」について，第 3.4.3 項では「身体表現」について述べる．また，第 3.4.2 項では「オブジェクト表現」にはさらに複数の表現があり，「シンボル表現」「アフォーダンス表現」「敵の表現」を順に述べる．

Fig. 3.8　デジタルゲームのための知識表現

3.4.1　世界表現

本項では，世界表現について述べる．世界表現はゲーム地形・空間に関する知識表現である．世界表現はキャラクターがそのゲーム世界の空間的認識を得るための基礎であり，客観的な情報の上にそのキャラクター固有の主観的な認識情報が付与される．ナビゲーション・データで最もよく使われる形式はウェイポイントとナビゲーション・メッシュである [52]．ナビゲーション・データを基底として，ポイントやメッシュに地形や空間の情報を積み重ねて行く．これは位置依存情報（location-based information）とも呼ばれる [72]．例えば，敵基地の周りのナビゲーション・メッシュには「危険」のタグが，湖の側には「水の側」のタグが，森の近くには「森の側」のタグが付与される（Fig. 3.9）．これは実際の地形データと照らし合わせることで自動的にタグを付けることができる．そして，それぞれのタグに応じてコストを上げておくことで，キャラクターの空間認識に個性を持たせる．例えば「危険」なメッシュには高いコストを，水が不得意なモンスターであれば「水の側」はコストを少し上げ，「森の側」はもし緑色のモンスターでなければ目立つのでさらにコストを上げておくことで，パス検索を行った時に，自然に苦手な場所を避けるようにすることが可能となる．

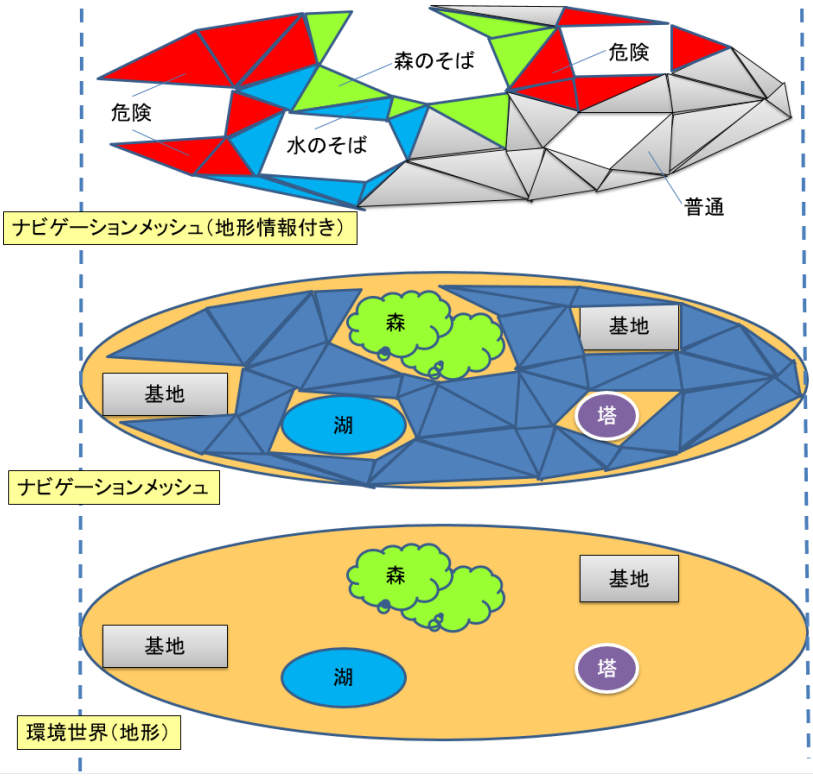


Fig. 3.9　地形（下図）とナビゲーション・メッシュ（中図）と地形情報を埋め込んだナビゲーション・メッシュ（上図）

90 年代初頭までほとんどのデジタルゲームはナビゲーション・データを持っていなかった．キャラクターがその上を移動するポイント・データか，固定領域（例えば部屋の中）が指定されているだけであった．或いはスクリプトの中で行ってはいけない場所に行かないよう制御していた．例えば「部屋の中心から 5m 以上離れると引き返す」などである．90 年代半ばの 3D ゲームの出現のタイミングから，序々にナビゲーション・データが導入され，その上の任意の 2 点に対して A*アルゴリズムによって動的にパス検索を行う手法 [6] [73] が急速に広まり，現在では標準的な方法となっている．パス検索によって，キャラクターはレベル上の任意の二点間を自由に移動する能力を得たのである．また『ARMORED CORE V』(FromSoftware, 2012 年)では三次元空間におけるパス検索が実装されている [74] [75].

ナビゲーション・メッシュは面であるからより地表の情報を反映しやすい.「雪」「水」「草」「コンクリート」「土」など地表の属性データを各メッシュに付属させ，この地表情報をパス検索のコストとして用いることで，各キャラクターに応じた個性的なパスを導くことができる [26]．例えば，泳げないキャラクターに対しては水属性のメッシュは通過コストを大きくしておけば水辺を通ることはなく，雪上移動が得意なモンスターには雪のコストを低くしておけば雪に沿って移動することになる．また「一方向にしか通れないメッシュ」属性を用いることで, キャラクターに落下, 回り込みを実現することができる [76] [77] [78] ．その他にもゲームデザインの要請に合わせて，隠れやすい場所，遠距離攻撃に適している場所などの情報を含ませて行くことで，より深い環境認識を AI に伝える程，キャラクターはより環境を巧みに利用した移動が可能となる． [79] [80]. また『Splinter Cell : Conviction』(Ubisoft, 2010 年）では，オブジェクトによって通れない場所が出来る，或いは逆に元あったオブジェクトが爆破されて通れるようになる，など動的なゲームの変化に対して，ナビゲーション・メッシュの形状を変化させて対応する機能も実装されている [81].

3.4.2 オブジェクト表現

ゲーム世界の中の各オブジェクトには，キャラクターがそのオブジェクトを認識する知識表現のデータを持たせておく必要がある．オブジェクトが持つべき情報は，キャラクターがそのオブジェクトに対して為すべき行動の種類だけある．最もよく使われるのが以下の表現である．キャラクターAI はこの表現に沿ったデータをオブジェクトから受け取ることでオブジェクトに対する認識を形成する．

シンボル表現

対象の分類のためのシンボルである．自然物か，人工物か，キャラクターか，敵か，味方か，敵であるとしたら，どんな種族か，などである．『Halo2』（Bungie，2004 年）では，敵全体の分類ツリーがあり，ツリー構造でキャラクター全般を管理している [82]．キャラクターAI はこれによって敵の種別と強さの認識を形成する．

アフォーダンス表現

アフォーダンスはそのオブジェクトに対してキャラクターが為し得る行動とそれを具体的に達成するための補助データである．例えば「岩」が軽ければ「特定の方向に動かすことができる」,「レバー」であれば「倒す」ことができる．さらに詳細なデータとして，オブジェクトのどの向きに押せば，その岩をどちらへ動かすことができるか,「車」に「乗り込む」のであれば，どの位置から乗ることができるか，など必要なデータと共に行動が指定されていることで, キャラクターはそのオブジェクトに対して正確な行動を取ることができる (Fig. 3.10)．『Halo2』では車など特殊なオブジェクトに対してアフォーダンス情報が準備されている [83].「モンハン日記 ぽかぽかアイルー村」(発売元：カプコン, 開発： FromSoftware, 2010 年）では，環境上のオブジェクトに行動ごとのアフォーダンス値が設定されている [84]．補助データとしては，例えばキャラクターを椅子に座らせる時に，「座る」アニメーションが自然に見えるためには，椅子の座る側の方向，椅子の高さなどアニメーション再生のヒントとなる情報が必要である．ゲームの世界には様々な椅子があり，そのすべてに対して固有のアニメーションを作ることはできないので，アニメーションを椅子に合わせて微調整するための情報がオブジェクト側に必要である．例えば樵が木を切るのであれば適切な位置を木が持ち，露店で列を作るなら適切な場所と手を伸ばして受け取る場所の情報を持つ．

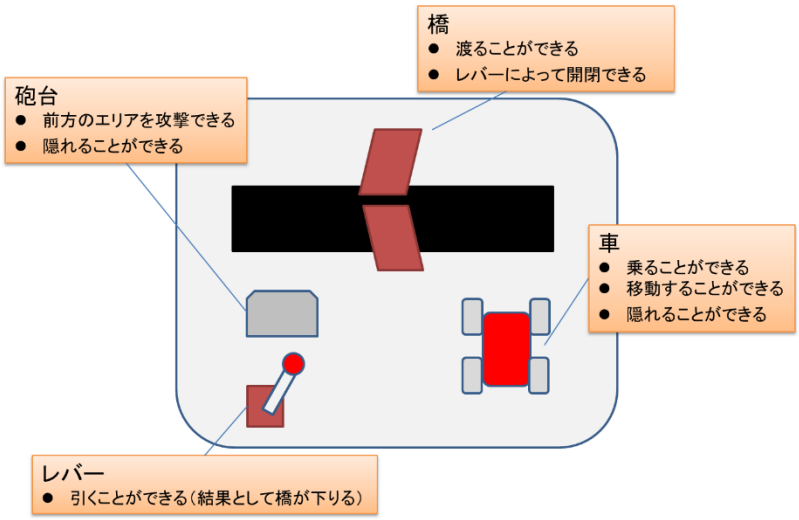


Fig. 3.10　オブジェクトのアフォーダンス表現の一例

アフォーダンス情報を持たせたオブジェクトを「スマートオブジェクト」(Smart object), 地形の一部の場合には「スマートテレイン」(Smart Terrain）と言う．これはアフォーダンス表現の最も良く取られる形である．例えば，「Bioshock Infinite」(Irrational Games，2K Games，2013 年）の各オブジェクトには，キャラクターを動作させる情報が埋め込まれて

いる［85］(Fig. 3.11)．ソファには「座ることができる」という情報と，「ソファの座ることができる場所」がオブジェクト表現として埋め込まれている．キャラクターは，この情報を頼りに「ソファに座る」という動作を行う．また高い場所にある窓には「見上げる」という「見上げるべきポイント」というオブジェクト情報が埋め込まれており，窓の近くに来ると窓の外を眺めるような動作を行う．つまり，一つの部屋の入ると，オブジェクトの側からその部屋における可能な行動の情報を受け取り，どのように動作を行うかを決めるのがキャラクターAI の意思決定のタスクとなる．

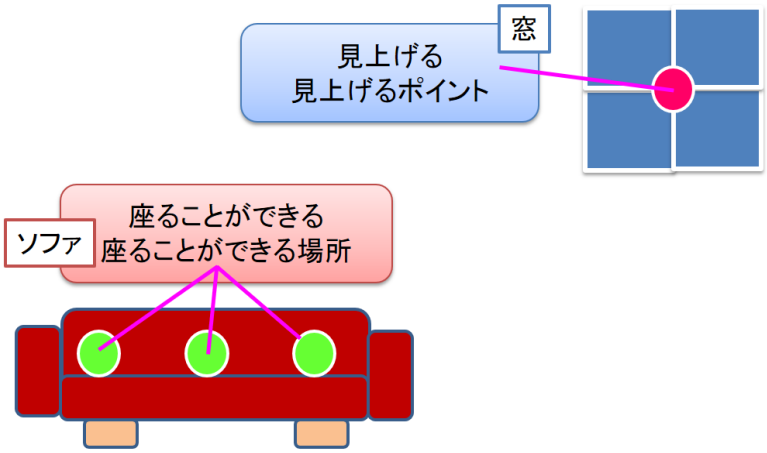


Fig. 3.11　オブジェクトの知識表現の例

敵の表現

敵の表現は，客観的な情報と，主観的な判断の二つの表現からなる．位置，種類，体力や魔力，性能（速度），種族など，客観的な情報に加え，キャラクターから主観的にその敵がどのように見えているか，という情報が意思決定のために必要である．具体的には，上記の客観的なパラメータを主観的パラメータに変更することで行われる．敵の性能や位置，速度から自分にとって敵がどれだけ危険であるかを示す「脅威度」は，評価式を作って算出する［86］．この脅威度の高さによって複数の敵からメインターゲットを選ぶことができる．また位置取り，速度・加速度ベクトル，履歴から敵の「意図」を推定する．「攻撃」「撤退」「待ち伏せ」などである．この推定された「意図」を意思決定で用いることで，相手を予測して行動する知能を作る．このような敵に対する認知はアウェアネス（Awareness）と呼ばれる［87］．これについては，具体的な例を第 9.2.3 項で示す．

3.4.3　身体の知識表現

身体の知識表現とは，「身体の静的構造の情報」と「身体の運動能力」の二つを抽象化することである（Fig. 3.12）．身体的構造は設定パラメータや，自己の身体の解析などから抽

出する．身体の運動能力は，設定パラメータ通りにアニメーションが作られていれば，設定をそのまま用いるだけでよいが，逆にアニメーション・データによって運動性能が決まる場合（アニメーション・ドリブンのゲーム設計）はキャラクターの運動シミュレーションから運動パラメータを抽出する．実際にテストステージで，キャラクターにアニメーションを再生させて，パンチの届く距離や，魔法の届き領域，ジャンプできる距離・高さ，加速曲線などの運動特性を抽出する．

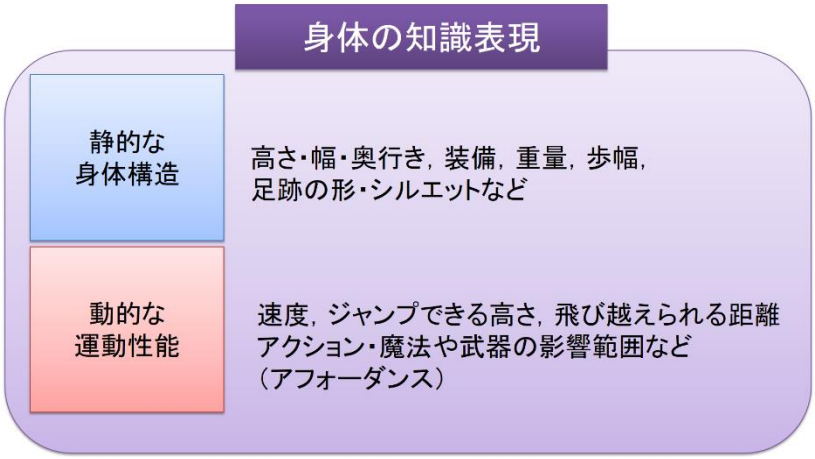


Fig. 3.12　身体の知識表現　上が静的な身体構造，下が運動性能

以上が，デジタルゲームの AI で使用される知識表現である．知識表現はゲーム開発中に蓄積され，ゲーム実行中に補完される．補完された知識表現を通じて，AI はゲーム世界を理解する．

3.5　キャラクターAI とスパーシャル AI の基本構造

本節では，キャラクターAI とスパーシャル AI の基本構造について述べる．キャラクター製作は，ゲーム製作の中でも，最も製作作業が集中する場であり，その為，複数の作業が巧みに組み合わされている．その中でも，キャラクターAI に含まれる「意思決定」と「キャラクター・アニメーション」の組み合わせは，最も複雑で高度な分野であり，また十分に研究が進んでいない分野でもある．その理由は，

(A) アニメーションと人工知能の意思決定が別々の分野として研究されてきた
(B) アニメーションと人工知能の意思決定の結び方の基礎理論がない

という主な 2 点である．(A)は，アカデミックな発展と方向によるものであり，(B)は工学的な問題であると同時に，「心と身体がいかにつながっているか」という心身問題と呼ばれ

る哲学的な問題でもある［88］．心身問題には数多の議論があるが，決定的な工学モデルは存在しない．知能と身体を結ぶ実装について，ゲーム開発は未だ明確なにモデルを持たない．論文などで提案された手法を参考にしながら，開発タイトルごとにアニメーションと人工知能の間のシステムを構築している現状である．

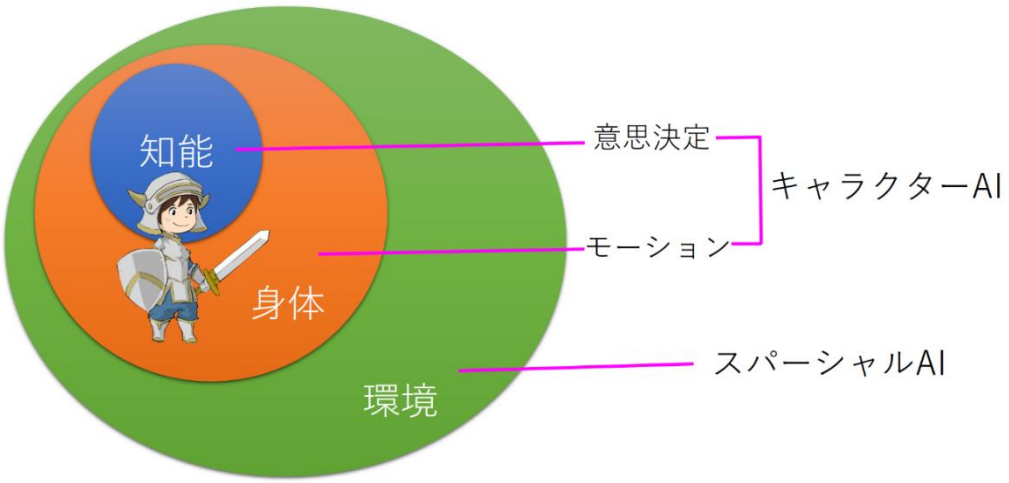


Fig. 3.13　キャラクターAI とスパーシャル AI の基本構造

ゲームキャラクター，特にユーザーが操作しないノン・プレイヤー・キャラクター（Non-Player Character，NPC）は，以下のような特徴を持つ(Fig. 3.13)．

（1） 構造化された身体を持つ
（2） 環境の中にある
（3） 人工知能を持つ

ここで構造化された身体，とは前章までに見て来たように，ポリゴンモデル，ボーン，リグ，さらにそのアニメーションモデルを持つ，ということである．環境とはゲームステージや地形のことである．環境と身体の空間的関係性を解決するのは「スパーシャル AI」の役割である．本節では「意思決定」と「アニメーション」と「スパーシャル AI」の関係性について述べる．

第 3.5.1 項では，階層構造の具体的な形について，第 3.5.2 項ではモーション・システムにおける多層構造について，第 3.5.3 項ではサブサンプション・アーキテクチャについて，第 3.5.4 項ではキャラクターAI におけるサブサンプション・アーキテクチャについて，第 3.5.5 項では，スマートオブジェクトを含む環境と人工知能と身体の全体像について，そして第 3.5.6 項で本節のまとめを述べる．

3.5.1 階層構造の具体的なシステム

デジタルゲームにおけるキャラクターシステム（キャラクターを動かす全体のシステム）は複雑な環境で，身体を動かし，目的を遂行するシステムである．キャラクターAI，モーションは，スパーシャル AI のサポートの元で実行される．スパーシャル AI は，身体のサイズや運動特性に応じた空間的特性・地形特性をそれぞれのキャラクターAI，モーションに提供する．各モジュールを汎用性の高い一般的なものとして作るために，各モジュール同士の関係性も汎用的な関係として構築する．特に，デジタルゲームは，それぞれ身体の違う数十から数百の NPC を製作する必要がある．それぞれのアニメーションに対するキャラクターAI を製作することを避けるために，二つの手法を導入する．

（X）キャラクターAI とモーション・システムの間に中間層を設置する（多階層化）
（Y）キャラクターの身体をできるだけ抽象化した情報を準備する（知識表現）

以下，この 2 点について説明する．

3.5.2 モーション・システムと AI のマルチレイヤー・アーキテクチャ

キャラクターAI と，モーション・システムを直接結ぶことを避け，中間のボディ・レイヤーを導入する（Fig. 3.14）．これによって多層構造（マルチレイヤー・アーキテクチャ，multi-layered architecture）を導入する．ここで，ボディ・レイヤーを導入する理由は以下である．

（1） キャラクターAI と，アニメーション・システムが密にリンクすると，それぞれの拡張性・独立性・再利用性が減少する．
（2） さまざまな身体ごとにキャラクターAI を作ることを避ける

このような中間層を作るアプローチは，大型タイトルのキャラクターシステムでは，良くとられるアプローチである．その理由は，数十〜数百の NPC の身体の違いを中間層で吸収することで，キャラクターAI が，それぞれの NPC の身体の特殊性に対応しなくて良いようにするためである（Fig. 3.14）．実際のところ，ボディ・レイヤーの抽象化は，２足歩行だけのキャラクターだけのゲームであれば，ある程度の抽象化が可能であるが，４本足，８本足，空中に浮遊するなど多種多様なモンスターの身体の形状を抽象化することは難しい．そういった場合には，ボディ・レイヤーは最小限の抽象化しかできず，モンスターの身体ごとにキャラクターAI を作る必要がある．

ボディ・レイヤーの役割は，アニメーション・システムから身体の状態を認識し，キャラ

クターAI に提供することである．はしごを登っている，剣を振る，走る，ジャンプしている，という身体の状態はボディ・レイヤーが維持している．アニメーション・システムが持つ個々のモーションの単位を，キャラクターAI が操作する実装は，キャラクターAI がアニメーション・システムに密接に結びついてしまう．ボディ・レイヤーはキャラクターAI から状態を指定され，アニメーション・システムからは，身体・アニメーションの特性情報（知識表現）を受け取る．はしごを登っている最中は剣が触れない，など，各状態の関係もボディ・レイヤーが調整する．ボディ・レイヤーは，今，どの状態を取れるか，という情報をキャラクターAI に提供する［89］.

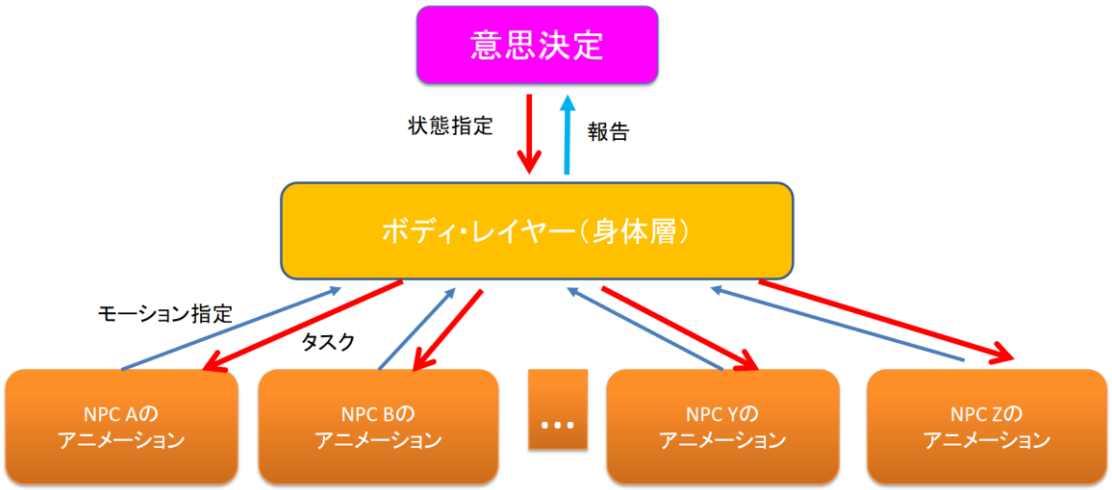


Fig. 3.14　理想的なアニメーション，ボディ・レイヤー，キャラクターAI の関係

3.5.3　　サブサンプション・アーキテクチャ

身体の反射から，高度な意思決定までを，並列的につなぐモデルを「サブサンプション・アーキテクチャ」（Subsumption Architecture）という（Fig. 3.15）．一つの場所にすべての刺激・情報を集めて意思決定を行う「中央集権構造」に対して，外界の変化に迅速に対応するモデルである．1985 年に Rodney Brooks によって発案された［90］．ここで説明するサブサンプション・アーキテクチャは，オリジナルの形ではなく，デジタルゲームに対応させたサブサンプション・アーキテクチャである［91］．以下に説明する．

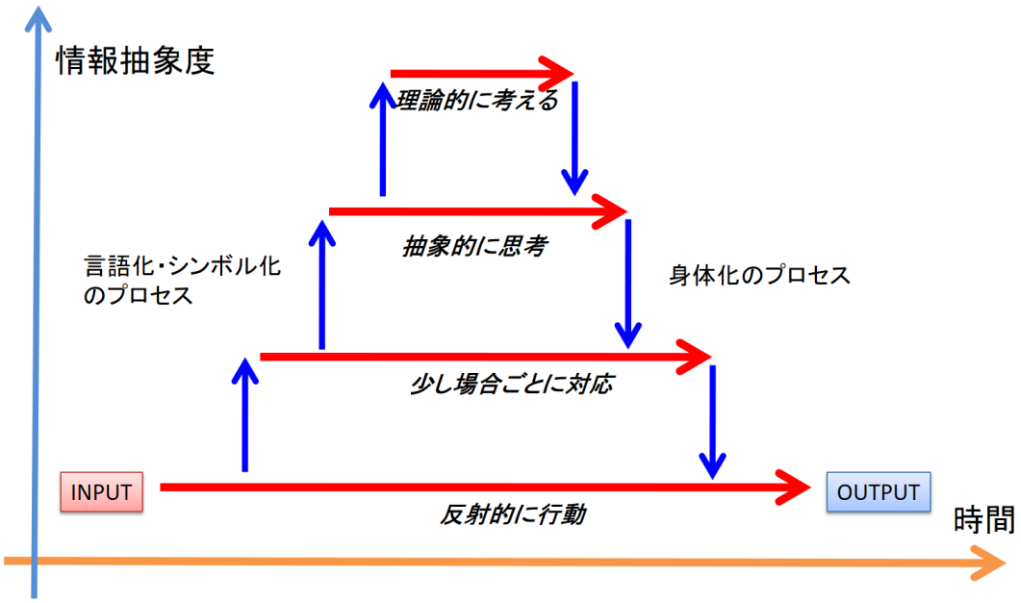


Fig. 3.15　デジタルゲームで用いられるサブサンプション・アーキテクチャ

デジタルゲームに対応させたサブサンプション・アーキテクチャは，各層がセンサーを持ち，独立に意思決定をそれぞれのレベルで下すが，常に上位の層が下位の層を抑制・実行する権限を持つ多層構造である．たとれば，ゲームキャラクターにおけるサブサンプション・アーキテクチャの作り方は以下のようである．まず，センサーから身体運動までの反射的なレイヤーを作る．たとえば，剣が振り下ろされそうになったらよける，などである．次に，その反射レイヤーを抑止・開放する権限を持つ上位のレイヤーを作る．たとえば，剣の速度が遅い場合には，剣を盾で跳ね返して（パリィして）攻撃する．この場合，最初のレイヤーは抑止されることになる．さらに，上位レイヤーとして，この戦闘の継続をやめて逃げる，ことを考える層を作る．このレイヤーは下二層のレイヤーを抑止して，その場から逃げ出す，という選択肢を取る．さらに次の最上位レイヤーは，このステージそのものにおいて，仲間を助ける，クリアをあきらめるなど，戦略的行動を考えるレイヤーを作る．それぞれのレイヤーは，それぞれにセンサーを持ち，独立に動作しつつ，レイヤー間の力関係が存在する．

3.5.4　キャラクターAI におけるサブサンプション・アーキテクチャ

キャラクターAI におけるサブサンプション・アーキテクチャは以下のようになる（Fig. 3.16）．まず，環境を解析したデータをスパーシャル AI が所持する．これは環境の事前解析とも呼ばれる．たとえば，地形を抽象化したスパーシャル・データなどが主である．スパーシャル AI はゲーム内で動的に地形を解析する場合もある．たとえば，位置検索技術と呼ばれるキャラクターの目的地を探索する手法などである．次に，意思決定，ボディ・レイヤー，アニメーション・システムの 3 つの層，それぞれにセンサーを付与する．それぞれの層が必

要とする情報を環境からセンサーを通じて取得する．

この 3 層同士の関係は，意思決定モジュールが行う意思決定アルゴリズム（第 7.3 節参照），ボディ・レイヤーが行う状態遷移，アニメーション・システムが行うモーション遷移がそれぞれ動作する．最下層となるアニメーション・レイヤーの動作がまず基本となるが，これをボディ・レイヤー，さらに意思決定が制御する．

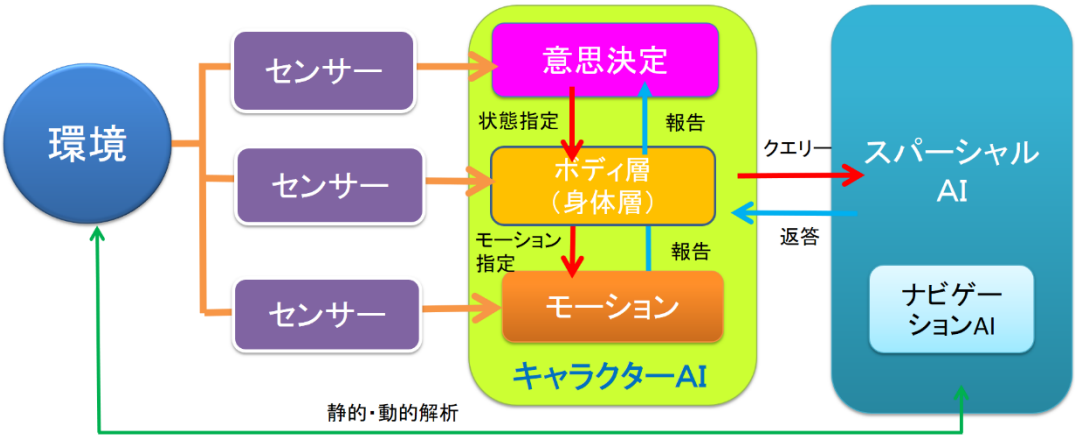


Fig. 3.16　ゲームキャラクターにおけるサブサンプション・アーキテクチャ

たとえば，最も単純な一つのアクションを選択することで済むタスクを考える．たとえば「敵に魔法を放つ」タスクを考える．キャラクターAI は，「敵位置」に「魔法を撃つ」行動を指定して，ボディ・レイヤーに渡す．ボディ・レイヤーがない簡単なキャラクター構造の場合は，直接，アニメーション・システムに渡す．キャラクターを敵方向に向かせて，「魔法を撃つモーション」を再生する．

3.5.5　スマートオブジェクトによる環境との相互作用

前述した通り「スマートオブジェクト」（Smart Object）はオブジェクトにキャラクターを制御させる方法であり，地形にキャラクターを制御させる手法を「スマートテレイン」と言う．たとえば「ジャンプをしてカブトムシを捕まえる」タスクを考える．この場合，まずキャラクターAI は，身体の知識表現として，「ジャンプできる高さ」を所持しておかねばならない．そして，「カブトムシの高さ」と「ジャンプできる高さ」を比較して，このタスクが実行可能かを判定する．判断が難しい場合は，とりあえずやってみる，ということでも良い．キャラクターAI は必ずしも完璧である必要はない．実行する場合は，ターゲット（カブトムシ）位置に対してジャンプ位置と軌道を調整して実行する．ジャンプ位置までのパスはスパーシャル AI が提供する．

また，たとえば「敵を攻撃する」タスクの場合，「敵キャラクターへ走って近づいて，1.5m

以内になったら剣を振る」という二つのアクションに分割することは，キャラクターAI の役割である．モーション・システムは，「敵キャラクターへ走って近づく」ために，ナビゲーション AI のパス検索のルートに沿ってキャラクターに「走るモーション」を実行する．そして，キャラクターAI は敵キャラクターとの距離を監視し 1.5m 以内になったら，「剣を振る」ように指令を出し，モーション・システムは「剣を振る」アニメーションを実行する．しかし，モーション・システムの側で「1.5m 以内になったら自動的に剣を振る」というルーティンを仕込んでおく場合もある．これによって，キャラクターAI の負荷を軽減すると同時に，反射的なアクションとして実装することができる．

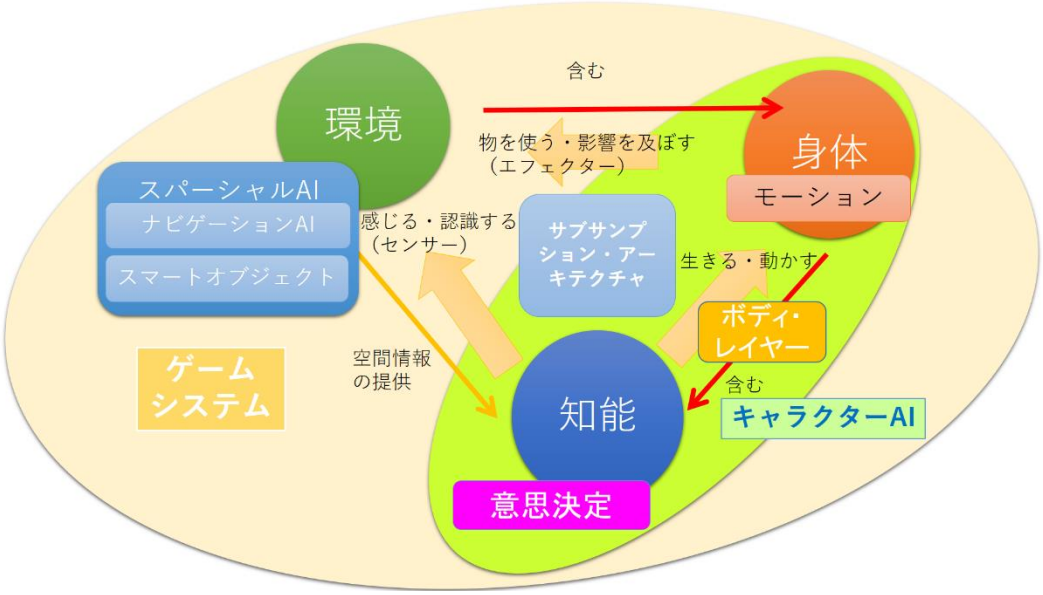


Fig. 3.17　環境，身体，知能の上の AI システム

3.5.6　まとめ

本節をまとめると以下のようになる．

(1)　ボディ・レイヤーを導入することでモーションと意思決定を分離する．
(2)　それぞれの環境と関係をするようにサブサンプション・アーキテクチャを導入する．
(3)　環境との関係はスパーシャル AI のサポートによって行う．

このように知能，身体，環境の関係は，サブサンプション・アーキテクチャを基本構造として，意思決定，ボディ・レイヤー，モーション，スパーシャル AI，の関係として構造化される（Fig. 3.17）．

3.6　ゲームの外の AI

本節では, ゲームの外の AI について述べる. ゲームはゲーム開発環境の中で製作される. そこで「ゲームの外の AI」と「ゲームの中の AI」の間には，さまざまな関係が形成される（Fig. 3.18）.

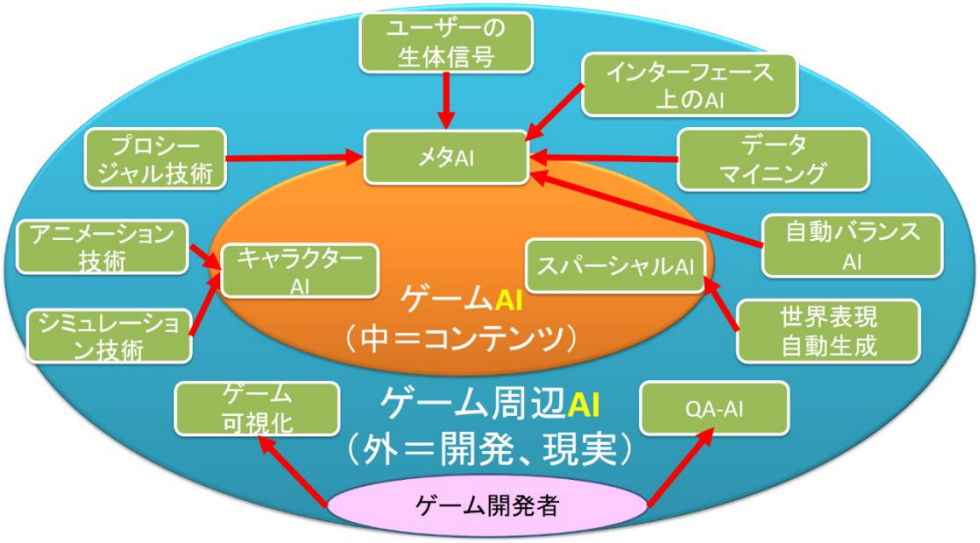


Fig. 3.18　「ゲームの外の AI」と「ゲームの中の AI」の関係

開発中はゲームの内部にある人工知能モジュールとゲームの外にあるモジュールが連携しながら，ゲームを組み立てて行く. 例えばパラメータをシミュレーションして決定する人工知能モジュールがある. またプレイヤーの代わりにゲームをプレイして負荷を計測する人工知能モジュール，品質保証の AI「QA-AI」（Quality Assurance AI）が存在する. また，ゲームの動作状態を記録し可視化（ビジュアリゼーション）する人工知能モジュールがある.

現在では携帯電話機のみならず，家庭用のゲーム専用機もインターネットにつながっており，ユーザーのプレイデータを収集することが可能となっている. この収集機能は「テレメトリー」と呼ばれる [92]. テレメトリーによって収集されたユーザーデータは「データマイニング」モジュールによって解析され，ゲームデザインへのフィードバックとして活用され，ゲーム内の世界の武器や防具の値段の変動などに使用される.

このような開発時に活用される人工知能モジュール群と，ゲームシステム内における人工知能モジュールが連携し，ゲーム開発環境が構築される. ゲーム開発は大規模化し，人の手で作り調整する段階を超え，人工知能による自動化によって効率化がなされている.

3.7　デジタルゲームの開発体制

本章では，AI 開発の全体の進め方について述べる. それぞれの開発者の役割，連携の仕

方，そこで形成されるデータフローなどである．第 3.7.1 項では，一般的なゲーム開発体制について，第 3.7.2 項ではゲーム AI 開発体制について，第 3.7.3 項では開発工程の内部構造について，第 3.7.4 項では開発コストの試算について述べる．

3.7.1　一般的なゲーム開発体制

本項では，ゲーム開発の一般的な体制について解説する．ゲーム開発は一人で行う場合もあるが，ほとんどの場合，チームを構成して行う．チーム開発の特徴は，以下の 3 点にある

(1) エンジニアとデザイナー，アーティストが共同で一つの作品を作り上げる．
(2) ゲーム開発手法はゲームタイトル毎に再構築する．
(3) ツールやデータパイプラインなど開発環境を開発する．

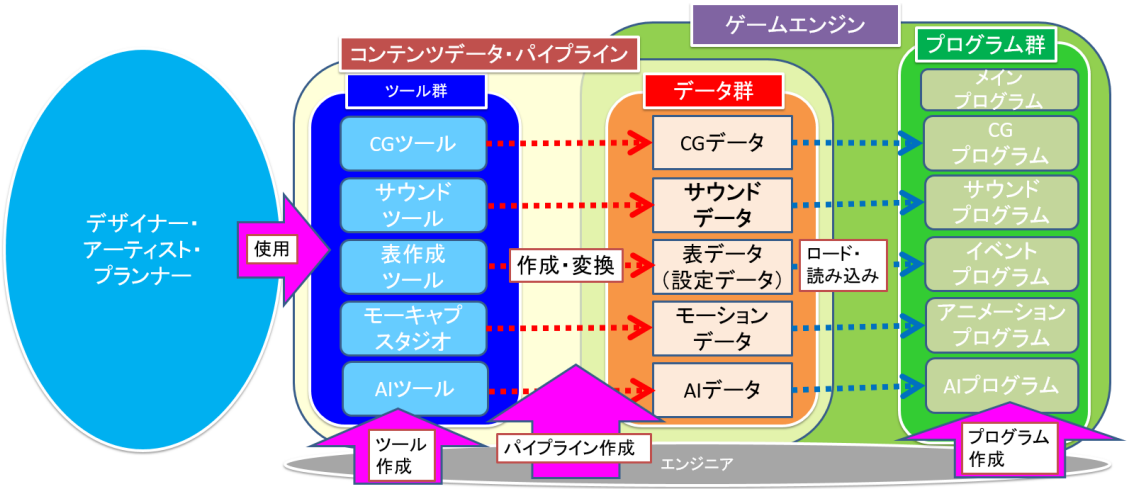


Fig. 3.19　一般的なゲーム開発体制

大きな枠組みとして，エンジニアは開発環境とゲームの仕組みを作りデザイナー，アーティストがその仕組みで必要とされるデータを作る．各分野に，エンジニアとデザイナー，アーティストが必要とされる．エンジニアはゲーム全体の構造を作るメインプログラマ，各分野の CG プログラマ，AI プログラマ，アニメーション・プログラマ，サウンドプログラマ，メニュー（ユーザーインターフェース）プログラマ，などから構成される．デザイナー，アーティストも，全体のゲームデザインを担当するメインゲームデザイナー，イベントを担当するイベントプランナー，戦闘を担当するバトルプランナー，3D グラフィックデザイナー，2D グラフィックデザイナー，サウンドデザイナー，ユーザーインターフェースデザイナー，モーションデザイナー，などから構成される（Fig. 3.19）．

開発環境はエンジニアとデザイナーが分業できる体制を作るのが目的の一つである．そ

のために「データ駆動型アーキテクチャ」が採用される場合が多い．即ち，マップデータ，キャラクターデータ，イベントデータなど，データを入れ替えることで，各ステージが成立する仕組みである．このアーキテクチャによって作業を独立に進めることで効率化できる．デザイナーは慣れた市販の CG ツールでキャラクターの 3DCG モデルを作る．そのデータをゲーム内のデータフォーマットに変換し描画できるように整備するのが CG プログラマの役割である．

3.7.2　ゲーム AI 開発体制

ゲーム AI 分野も同様に，AI の開発環境をゲームデザイナーが AI を作れるように整備する．特に AI 分野の場合はデファクトスタンダードになるような市販のツールがないため，AI エンジニアが AI ツールを用意し，ゲームプランナーがそのツールによってデータを製作する（Fig. 3.20，Fig. 3.21）．

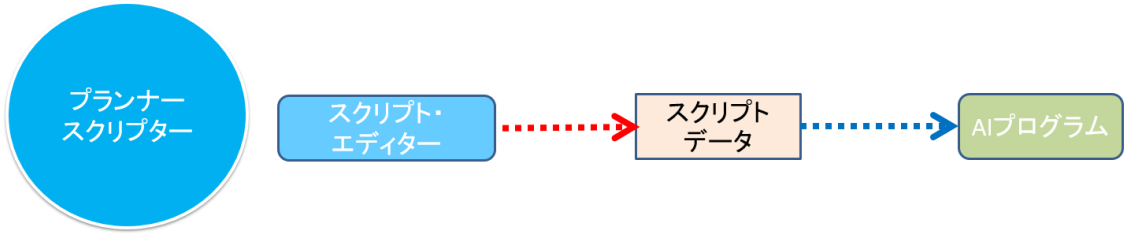


Fig. 3.20　スクリプトによるゲーム AI 開発体制

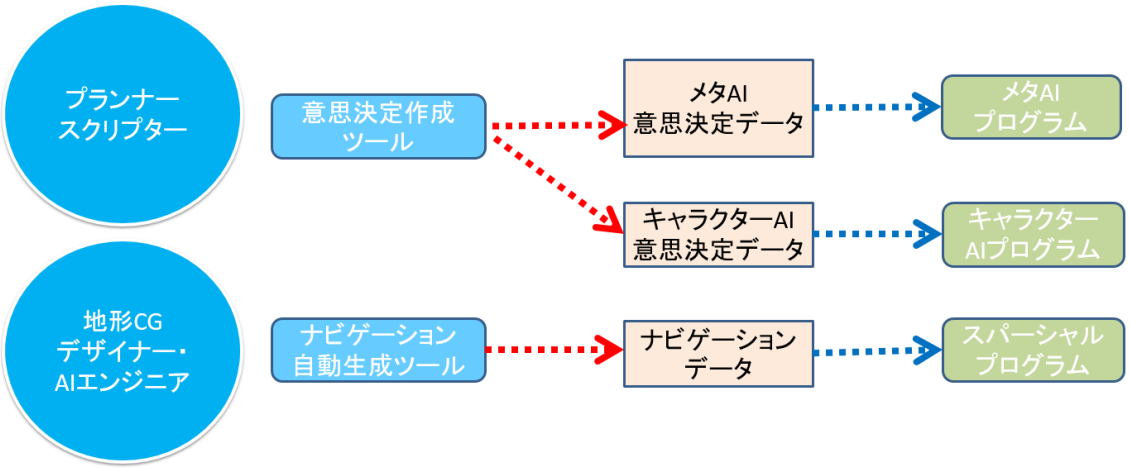


Fig. 3.21　MCS-AI 動的連携モデルにおける AI 開発体制

ゲーム AI 分野で必要となるデータはスクリプティッド AI の場合，スクリプトであり，メタ AI，キャラクターAI，スパーシャル AI に応じたデータである．このスクリプトの量はゲームスケールに依存するが，キャラクターAI のデータとの比較を 3.7.4 項で行う．

メタ AI，キャラクターAI に関してスケールは異なるが，どちらも意思決定システムであり，同一の意思決定ツールを用いることができる．この意思決定ツールを用いて AI を作成する．スパーシャル AI が必要とするデータは各世界表現に対応するナビゲーション・データや，位置検出技術ではゲーム内で実行するクエリー・データが作成される．前者は多くの場合には，現在では自動生成され，クエリー・データは専用ツールによって製作される場合が多い．

3.7.3 開発工程のレベル

ゲーム全体を進行する領域，キャラクターを動かす領域，ゲーム世界環境を抽象化する領域，３つの領域について考察する．それぞれの領域は「スクリプトレベル」「機能レベル」「AI レベル」からなる．「スクリプトレベル」は，逐一最小の単位となり制御を記述するレベルである．「モジュール・レベル」は，単一の機能のレベルである．これは「スクリプトレベル」で指定していたデータを生成する機能でもある．たとえば，「スクリプトレベル」ではキャラクターの目的地までの経路を点列によって指定するが，ナビゲーション AI ではそれを生成する．「AI レベル」はアーキテクチャの中でモジュールを働かせる．また，スクリプトレベルと AI レベルの違いは，抽象度の違いである．ゲーム世界をどれだけ抽象度高く理解し，抽象度の高いレベルで影響を及ぼせるか，に関して，スクリプト，モジュール，AI の次第に抽象度が上がって行く．

たとえば，ゲーム全体を進行する分野では，まずシークエンス的にキャラクターやオブジェクトを動かすスクリプトレベルの段階がある．これはゲームの状況を見ずに，一方的に制御するだけである．次に，ゲームの状況を取得し，影響を及ぼすモジュール・レベルがある．たとえば，プレイヤーが扉にたどり着くまで敵を生成し続ける，段階がある．第 9.1 節で述べるように，プレイヤーの進行度に応じて敵キャラクターを出現させる，などがある．次の段階として AI レベルのメタ AI があり，ゲーム全体を認識し，意思決定し，命令を出す，という関数から AI のレベルに上がった制御を行う．

工数から見ると，スクリプトレベルとモジュール・レベルは量的な差異である．たとえば，経路を点列で指定する工数に対して，ナビゲーション AI は経路をナビゲーション・データから計算で生成する．仮に，ナビゲーション AI が 5 万回パス検索を行うとすると，５万個の経路指定の工数を削減したことになる．しかし，実際，人間が生成・管理できる経路指定の数の上限は数百個であるから，それを超えたスケールではナビゲーション AI へ移行するか，ゲームデザインを工夫してパス検索が不要なレベルデザインにするようになる．一方，モジュール・レベルと AI レベルはクオリティの違いである．単一のモジュールはそのモジュールが持つ機能に特化した制御を行うが，AI レベルはモジュールを駆使して，知的な制御を実現する（Table 3.4）．

Table 3.4 スクリプトレベル，モジュール・レベル，AI レベルの関係

	スクリプトレベル	モジュール・レベル	AI レベル
ゲーム全体の進行	キャラクターやオブジェクトの変化のシークエンスを逐一記述（レベルスクリプト）	ゲーム状況のいくつかの指標をもとにゲーム状況を変化（レベルスクリプトを生成）	メタ AI
キャラクター	行為のシークエンスをスクリプトで直接記述（レベルスクリプト）	行為のパターンを関数で記述（行為のシークエンスを生成）	キャラクターAI
ゲーム環境世界	経路や移動可能領域を指定・目標位置を座標で指定	パス検索（経路を生成）位置検索（目的位置を生成）	スパーシャル AI（空間に関するあらゆる思考）

3.7.4 開発工数の比較

簡単な試算のために，ここではスクリプトの長さがすべて一定であると仮定する．つまりスクリプトに換算した時に何度スクリプトを呼び出すことに相当するか，という数を指標にする．

AI アーキテクチャが M 個のモジュールを持つとする．各モジュールが使用される一つのゲームの回数で使用される回数をAとすると，各 AI にM × A個のスクリプトに相当する制御を生み出すことになる．プレイヤーの仲間，あるいは定番の敵などゲームに登場する主要なキャラクターの数をN_{major}，サブキャラクターの数をN_{sub}，モンスターなど大勢で出てくるキャラクターの数をN_{minor} とする．それぞれに対応するモジュール数を，M_{major}，M_{sub}，M_{minor}各モジュールの使用回数をA_{major}，A_{sub}，A_{minor}，さらに，同一のスクリプトが現れる確率をP_{major}，P_{sub}，P_{minor}とすると，重複のいない異なる制御スクリプトに相当する制御数 S は，

$$S = N_{major} \times M_{major} \times A_{major} \times (1 - P_{major}) + N_{sub} \times M_{sub} \times A_{sub} \times (1 - P_{sub}) + N_{minor} \times M_{minor} \times A_{minor} \times (1 - P_{minor})$$

である．タイトル毎に各数値はことなるが，モジュールの数は 1~10 個の間である．ここで大型ゲームの試算として，以下のパラメータの想定とする．主要な敵，味方キャラクターが数体，モジュール数は最大でも 10 個，そこから呼び出されるスクリプトに相当する制御プ

ログラムの数をゲームの開始から最後まで 10 万とする．また，同じ処理が呼び出される確率はゲームの状況が多様なためにかなり低いと見積もり，0.01 とする．

$$N_{major} = 5, \quad M_{major} = 10, \quad A_{major} = 100000, \quad P_{major} = 0.01$$
$$N_{sub} = 20, \quad M_{sub} = 5, \quad A_{sub} = 10000, \quad P_{sub} = 0.1$$
$$N_{minor} = 100, \quad M_{minor} = 3, \quad A_{minor} = 1000, \quad P_{minor} = 0.2$$

この大型ゲームの場合，$S = 6090000$ となる．中型ゲームの試算として，以下のパラメータの想定とする．ここで中型は大型ゲームの 1/10 のスケールを想定している．すると，マップの状態の数は 1/100 になるので，各モジュールの使用回数を大型ゲームの 1/100 とした．またゲームの状態数が減少すると，似たシチュエーションが多くなるため，同一の制御が現れる確率を上げた．

$$N_{major} = 3, \quad M_{major} = 5, \quad A_{major} = 10000, \quad P_{major} = 0.1$$
$$N_{sub} = 10, \quad M_{sub} = 3, \quad A_{sub} = 100, \quad P_{sub} = 0.2$$
$$N_{minor} = 50, \quad M_{minor} = 1, \quad A_{minor} = 10, \quad P_{minor} = 0.3$$

この中型ゲームの場合，S=137750 となる．小型ゲームの試算として，以下のパラメータの想定とする．小型のゲームは中型よりさらに数分の 1 のスケールのゲームを想定している．味方・敵キャラクターもシンプルな動きちなり，呼び出し回数も減少し，類似確率を上げている．

$$N_{major} = 1, \quad M_{major} = 3, \quad A_{major} = 1000, \quad P_{major} = 0.3$$
$$N_{sub} = 1, \quad M_{sub} = 2, \quad A_{sub} = 100, \quad P_{sub} = 0.5$$
$$N_{minor} = 30, \quad M_{minor} = 1, \quad A_{minor} = 10, \quad P_{minor} = 0.8$$

この小型ゲームは，$S = 2260$ となる．また，さらに超小型のゲームの場合は，主要なキャラがいない，また，敵キャラクターは同じ動作を繰り返すため，呼び出し回数を 10，同じパターンの動作がくり返し行われることを想定して類似確率を 1.0 近くまで上げている．

$$N_{major} = 0,$$
$$N_{sub} = 1, \quad M_{sub} = 2, \quad A_{sub} = 100, \quad P_{sub} = 0.5$$
$$N_{minor} = 30, \quad M_{minor} = 1, \quad A_{minor} = 10, \quad P_{minor} = 0.8$$

この小型ゲームの場合は S=160 となる．

これらの試算を Table 3.5 にまとめる．

Table 3.5 ゲームのスケールと全工程をスクリプトの数に換算した時の数

数式の項	大型ゲーム	中型ゲーム	小型ゲーム	超小型ゲーム
N_{major}	5	3	1	0
M_{major}	10	5	3	-
A_{major}	100000	10000	1000	-
P_{major}	0.01	0.1	0.3	-
N_{sub}	20	10	1	1
M_{sub}	5	3	2	2
A_{sub}	10000	1000	100	100
P_{sub}	0.1	0.2	0.5	0.5
N_{minor}	100	50	30	30
M_{minor}	3	1	1	1
A_{minor}	1000	10	10	10
P_{minor}	0.2	0.3	0.8	0.8
S	6090000	137750	2260	160
S'	33000			
S'/S	0.00504	0.23	14.6	206.1

ここで想定した大型，中型ゲームは，スクリプトに換算すると数万を超える相当数であり，製作・管理を行うには困難な量である．スクリプトからモジュール型に移行することは，量の問題ではなく質の問題である．小型ゲーム，超小型ゲームの 2000 前後のスクリプト数は開発人数さえ整えば生成・管理可能な数であり，実際にこの規模の開発がスクリプトで行われることがある．中型・大型タイトルでは AI レベルの導入が不可欠であり，小型タイトルではかろうじてスクリプトのみで開発である．AI 技術の導入は，大型・中型タイトルでは数十万以上のスクリプト開発を削減し，小型タイトルでは数千個のスクリプト開発を削減する．

一方，AI レベルのシステムを構築するコストについては，それぞれのモジュールを作るコストと，各 AI を作るコストが必要である．この合計するコストが，スクリプトを準備するコストより大きくなれば，スクリプトを準備するという選択肢と競合することとなる．ここで各 AI の開発コストは，それが持つモジュールの開発コストとそれを組み合わせて AI として構築するコストである．

それぞれの AI が持つモジュールの数を

$$M_{metaAI},\ M_{charaAI},\ M_{spatialAI}$$

それぞれの AI の開発コストを，

$$C_{metaAI}，\ C_{charaAI}，\ C_{spatialAI}$$

とする．ここで，各モジュールのスクリプトに換算した時の最大数を K とすると，AI システムを構築するコストの最大値 S’ は以下の式で表される．

$$S' = K \times \left(M_{metaAI} + M_{charaAI} + M_{spatialAI}\right) + C_{metaAI} + C_{charaAI} + C_{spatialAI}$$

ここで S と上記 S’を比較した時，S’はキャラクター数にもステージ数にも比例しないので，キャラクター数，ステージ数が大きな場合に S’は S よりずっと小さくなる．たとえば，モジュールはたいていの場合 10 個より少ない程度であるから，モジュールの数をすべて 10，K を大きく見積もって 1000，C_{metaAI}，$C_{charaAI}$，$C_{spatialAI}$も大きくも積もって 1000 とすると，S’は 33000 となる．S と S’の比を取ると，大型ゲームの場合は 180 倍，中型ゲームの場合は 4 倍，小型ゲーム以下場合は 1 より小さくなる．つまり中型ゲームの場合はかろうじてメリットがあり，大型ゲームでは 100 倍以上のコストの削減につながる．

キャラクターを動かす分野では，まずキャラクターを操り人形のように外部から制御する段階がある．指定する座標へ行ってこの台詞を喋る，こういう経路で移動して指定した振る舞いを行う，などである．次にある特定の動作パターンを行う，という関数型制御の段階がある．たとえば，移動するプレイヤー・キャラクターに向かってくり返し直進する，などである．さらに自律型エージェントとなる段階がある．

3.8 考察

本章ではデジタルゲーム全体の構造を述べて．第 3.1 節ではデジタルゲームの基本的構造を紹介した．第 3.2 節では，デジタルゲームの中で実行される人工知能の全体像について述べた．第 3.3 節では自律型 AI の内部構造について述べた．第 3.4 節では，それぞれの AI がゲーム世界の認識を形成するために必要な知識表現について述べた．第 3.5 節では，キャラクターAI とキャラクターの身体の関係性についてまとめた．以降の章では，身体については，ほぼ言及しない．第 3.6 節では，「ゲームの外の AI」について概説した．第 3.7 節ではゲーム開発体制，ゲーム AI 開発体制，そしてコストについて述べた．本章を通して，デジタルゲームの発展が「ゲーム AI」システムを生み出し，そのシステムの内部は構造化され，３者の AI が連携することとなり，開発体制に変化をもたらし，開発コストの効率化が促されたことを述べた．この章で示した AI 連携の在り方は，まだ初段階のものであり，さらに大きなゲームデザインの要求に応えるために，より構造化されて行く．

4. 従来のゲーム AI 理論と問題点

本章では，MCS-AI 動的連携モデルを導入する動機となる，従来のゲーム AI 理論の問題点を示す．第 1 章の背景で述べたように，ゲーム AI システムは，自律型エージェントとしての人工知能，そしてナビゲーション AI による AI システム，として，物語的ゲームは「レベルスクリプト」「キャラクターAI」として発展してきた．しかし，2007 年以降，主に 3 つの変化によって，それまでのゲーム AI システムは限界を迎えるようになる．

(1) ゲームの大規模化と複雑化
(2) ゲームジャンルの融合
(3) ゲームのオープンワールド化

これら 3 つの要因をゲームＡＩシステム側から捉えると，以下のようになる．

(1) はキャラクターAI が解釈するべき物・事の急激な増加である．また身体構造も精緻になり，環境に合わせた身体運動が可能になったものの，実際に複雑な環境の中で身体運動を生成するシステムが必要となった．
(2) 世界的な規模でゲーム人口が増え，物語だけのゲーム，アクションだけのゲームよりも，物語もあるアクションゲームというジャンルが受け入れられるようになった．物語的ゲームにもアクションゲームにも対応できる総合的システムが必要となった．
(3) ゲームステージがオープンワールド化され，広大なマップにゲームコンテンツを生成する手段が必要となった．

以下では，第 1 章で提示した「アクションゲームのゲーム AI システム」と「物語ゲームのゲーム AI システム」の上記の変化に伴う限界について述べる．

4.1 アクションゲームの人工知能システムの限界

本節ではアクションゲームの人工知能システムの限界を示す．第 1 章で述べたように，アクションゲームは．レベルスクリプトによる制御から始まった．この仕組みは，3D ゲームへの変化において「キャラクターAI」「ナビゲーション AI」へと変化して行った．この進展は，レベルデザインの複雑化と大規模化によるものであった．地形の複雑さをナビゲーションが担い，ゲームの複雑さをキャラクターAI によって対応するシステムであった（Fig. 4.1）．

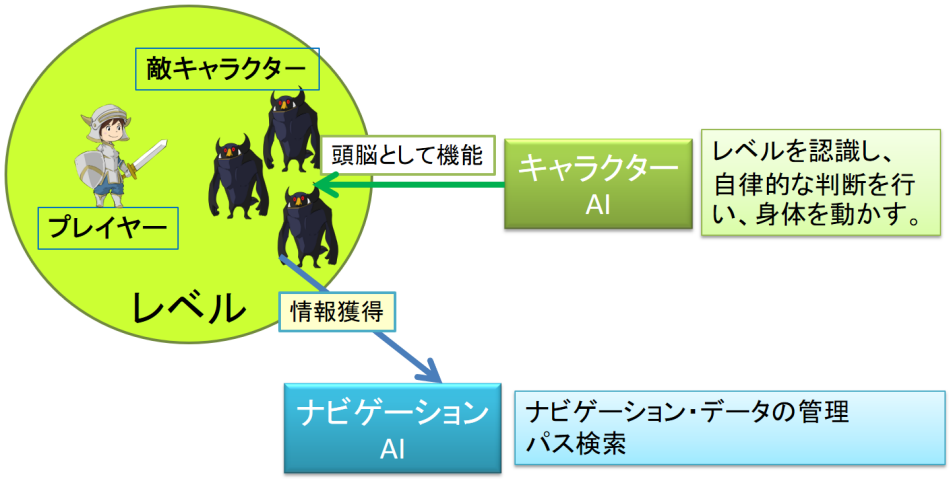


Fig. 4.1　アクションゲームにおけるゲーム AI システム（Fig. 1.4 と同じ）

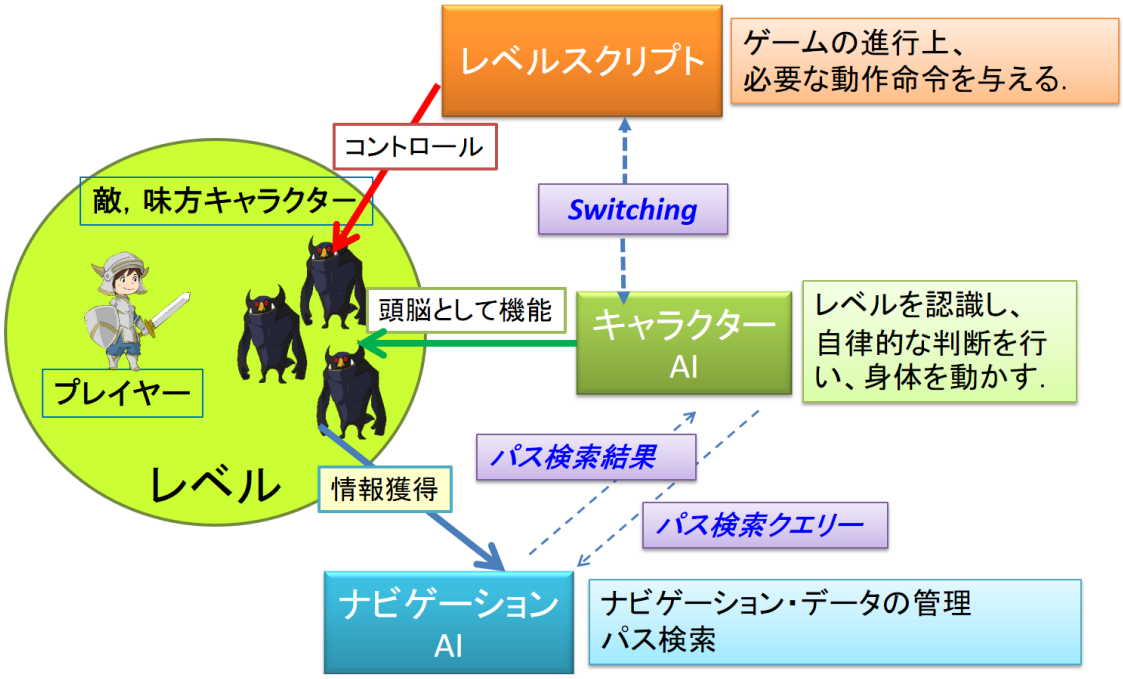


Fig. 4.2　LCN-AI 連携モデル

ところが，このシステムで物語的ゲームに対応しようとすると，自律的なキャラクターAI だけでは対応できなくなる．自律的でありながら，外からの制御を受け入れる「他律的」である必要がある．ゲームや物語の都合で行う必要がある動作を，いったん意思決定の自律的なシステムに置き換えて実現する必要がある．たとえば，「体力が半分以下になった仲間を助ける」，という仕様は，キャラクターが常に仲間の体力をモニターし，意思決定の中に「仲間を助ける」ことを判断し続ける必要がある．このように，キャラクターAI の自律的 AI の判断の中に，ゲーム的な都合を組み込むことは，ある程度までは可能であるが，要求が増えて来ると，意思決定の肥大化を引き起こす．また，ゲーム製作上，ゲームからの要求は変更

されることが多く，自律的 AI はその度に，大きな変更を強いられることになる．そこで，ゲームの要求をそのまま外部で実現する機構を独立しておくことが望ましい．そこで，外部スクリプトからの制御を行うために「レベルスクリプト」が導入される．「レベルスクリプト」，「キャラクターAI」，「ナビゲーション AI」からなる，このモデルを LCN-AI 連携モデル（Level-Character-Navigation-AI Cooperative Model）と呼ぶことにする（Fig. 4.2）．

LCN-AI 連携モデルは，「PlayStation3」(2006 年)，「Xbox360」(2005 年) 以降，つまり，おおよそ 2004 年以降の開発において，最もよく使われているモデルである．ただ，外部スクリプトで動かしている場合には，自律システムをオフにする．そうすると，かつてのシステム，つまり，他律的にキャラクターを動かすという手法に戻ってしまい，今度は 3D 空間の中で自律的にキャラクターを動かせなくなってしまう．「レベルスクリプト」とキャラクターAI の自律性に協調関係がなく，独立に動作させることで，お互いの干渉をなくし，必要に応じてキャラクターをレベルスクリプトから制御しやすくする点が長所であると同時に短所となっている．

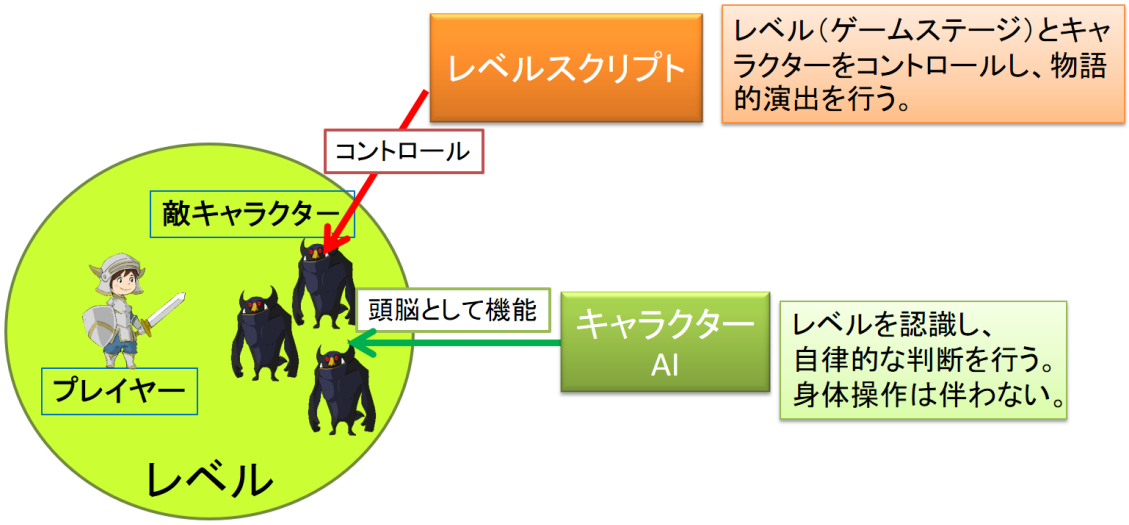


Fig. 4.3　物語的ゲームにおけるゲーム AI システム（Fig. 1.5 と同じ）

4.2　物語的ゲームにおける人工知能システムの限界

本節では，物語的ゲームにおける人工知能システムの限界を示す（Fig. 4.3）．物語的ゲームは，一貫してコンテンツの拡大と共にあった．つまり，物語のスケールの拡大，ゲームステージの拡大，設定の拡大である．そのような拡大に対して，物語的ゲームＡＩのシステムは強固であり，主にレベルスクリプトで対応し続けた．ゲーム開発の現場では多数のスクリプトター（スクリプトを書くプランナー）が養成され，大量のスクリプトが書かれた．しかし，このようなスクリプトシステムは，スクリプトの書き方がさまざまにあり，メンテナンス制が悪く，また，ゲームとのインタラクティブな動作は難しく，決められたタイミングで

一度実行されるだけである．たとえば，A という街につけば，M というキャラクターが X という台詞を話す，B という草原で L というキャラクターが出現し，座標（49，72）まで動き，そこで，Y という台詞を言い，戦闘が始まる，などである．

しかしこのような最初から決められた動作は，ゲームのストーリーや戦闘の流れが一つの方向に固定されているときに有効であるが，ストーリーやイベント，戦闘がダイナミックに生成するとなると，スクリプトでの対応が難しくなる．たとえば，オープンワールドになると，戦闘は固定された場所ではなく，フィールドのあらゆる場所で起こるようになる．フィールドそれぞれの固有の地形を考慮に入れるには，ナビゲーション AI の導入が必要である．また，キャラクター同士の会話も決まった場所ではなく，任意の場所で行われる．その時の立ち位置や台詞も，また状況によって変化する．個々のフィールドの違いに対応するために「ナビゲーション AI」も必要となった．そこで，物語的ゲームも，アクションゲームと同じ LCN-AI 連携モデルにたどり着くことになる（Fig. 4.2）．

4.3 LCN-AI 連携モデルの動作原理

本節では LCN-AI 連携モデルの動作について述べる．LCN-AI 連携モデルは，多くのタイトルで採用されている．その動作原理は以下である．

(1) キャラクターAI は自律的に動作する．
(2) レベルスクリプトは必要な場面で，キャラクターをスクリプトから操作する．
(3) レベルスクリプトが実行中は，キャラクターAI はオフ．両者はスイッチングの関係
(4) キャラクターAI が移動のタイミングで，ナビゲーション AI の目的地までのパス検索を要求（クエリー，Query）する．ナビゲーション AI は目的地までの経路を返す．

などである．「レベルスクリプト」，「キャラクターAI」，「ナビゲーション AI」のいずれか一つが排他的に実行されている．

このようなシステムは，レベルスクリプトによってキャラクターにゲームの文脈や物語に沿った演技をさせ，キャラクターの自律的知能はキャラクターAI で実装する．両者の担当が明確に分担されている．

4.4 LCN-AI 連携モデルとその限界

本節では LCN-AI 連携モデルの課題について述べる．

「LCN-AI 連携モデル」は，レベルスクリプトがゲームの流れを作り，キャラクターAI がキャラクターの意思決定を司り，ナビゲーション AI がパス検索を行う．このような人工知能モデルである程度のゲーム要件を対応できるが，オープンワールドを始めとする大規模

で複雑なゲームには対応できない．その理由は 3 点ある．

(1) レベルスクリプトは，すでに想定されている場面やタイミングに対して準備される．そのため，オープンワールドでは頻繁に起こる想定されてない動的に状況に対応できない．
(2) キャラクターAI は自律的に運動するが，それを調整するシステムがない．レベルスクリプトはキャラクターAI をオフにして，状況を作るが，キャラクターAI を動作させたまま動的に制御するシステムが必要である．
(3) ナビゲーション AI のパス検索の機能だけでは足りない．オープンワールドでは，地形の抽象的な情報を生かした運動を行う必要がある．

本章では従来のデジタルゲーム AI の問題点とその限界を示した．この問題を解決するための改善を「MCS-AI 動的連携モデル」の導入について，第 5 章で述べる．

4.5 考察

本章では，「アクションゲームの人工知能システム」と「物語的ゲームの人工知能システム」が共に発展しながらも，同じ LCN-AI 連携モデルにたどり着いた．LCN-AI 連携モデルにも上記のように，デジタルゲームの発展に対して足りない点があり，より動的なモデルへと変化する必要に迫られる．ゲーム開発においては，ゲームの内容を直接，開発者がコントロールする，という要求が先んじてしまう傾向があるが，ゲームの大規模化・複雑化はそれを許さない状況にある．ゲーム AI の歴史は，開発者が一つ一つ作り込んでいたキャラクター動作，ナビゲーションを，徐々に AI に任せて来た歴史でもある．メタ AI の導入は，レベルデザインの領域に AI を導入することであり，この変化はゲーム AI システム全体に大きな影響を及ぼす．第 5 章では，これを解決するために，各領域に固定されたレベルスクリプト部分をメタ AI へと変化させ，MCS-AI 動的連携モデルを提案する．

5. MCS-AI 動的連携モデル

本章では，MCS-AI 動的連携モデルの提案を行う．前章では，「アクションゲームの人工知能システム」と「物語的ゲームの人工知能システム」が共に発展しながらも，同じ LCN-AI 連携モデルにたどり着いた発展を述べた．そして，LCN-AI 連携モデルが持つ課題について述べた．LCN-AI 連携モデルが持つ課題を，MCS-AI 動的連携モデルはいかに解決するかを説き，このモデルの利点について述べる，LCN-AI 連携モデルでは，キャラクターはレベルスクリプトから他律される時間と，キャラクターAI 自体が自律的に動作する時間に明確にスイッチングされる．MCS-AI 動的連携モデルでは，スイッチングのような静的な切り替えでなく，メタ AI とキャラクターAI がコミュニケーションをしながら動的に連携することになる．つまり，キャラクターAI はメタ AI からの干渉を受けながらに，自律的に動作する．この両者の状況判断を「スパーシャル AI」が支える．以下，この MCS-AI 動的連携モデルを詳説する．

5.1 MCS-AI 動的連携モデル

本節では，第 4 章における問題点の解決案を考える．

(1) レベルスクリプトは，すでに想定されている場面やタイミングに対して準備される．そのため，オープンワールドでは頻繁に起こる想定されてない動的に状況に対応できない．

問題(1)に関しては，レベルスクリプト部分を「メタ AI」へ変換する．メタ AI はゲーム状況を認識し，ユーザーのスキル，ユーザーの心理状態などを推定し，ゲームを変化させる人工知能である．これによって，常に変化するゲーム状況を認識しながら，ゲームの流れを形成する．

(2) キャラクターAI は自律型であるが，それらをある程度自律的に運動してしまうが，それをまとめる AI がない．レベルスクリプトはキャラクターAI をオフにして，状況を作るだけで，キャラクターAI を生かしていない．

問題(2)に関して，キャラクターAI は自律的エージェントでありながら，必要に応じた演技ができる機能を持たねばならない．キャラクターの自律性を利用しつつ演技をさせる仕組みを作るために，メタ AI とキャラクターAI の間のコミュニケーションを作らねばならない．即ち，メタ AI からの命令に対して，それを実行し報告する，という形である．メタ AI

からキャラクターAI には，ゴールやタスクを与えることで，指示だけを与え，そのあとはキャラクターAI の自律性に任せる．これによってメタ AI とキャラクターAI は同時に実行されつつ，協調する．

(3)　　ナビゲーション AI のパス検索の機能だけでは足りない．オープンワールドでは，地形の抽象的な情報を生かした運動を行う必要がある．

問題(3)に関しては，パス検索を主とするナビゲーション AI から，ナビゲーション AI を内包し，空間に関する認識を補佐するスパーシャル AI への発展である．スパーシャル AI は，キャラクターの身体的特徴（身長，運動性能）に応じた空間的認識と，状況認識のサポートを行う．第 8 章で詳細を述べるが，ナビゲーション・メッシュやウェイポイントによって，ゲームマップを離散的に表現しパス検索を行う，ヒートマップを用いて戦況をリアルタイムに判断する，位置検索技術を用いてキャラクターの目的位置を決定する，など，位置に依存した思考を行う（Fig. 5.1）．

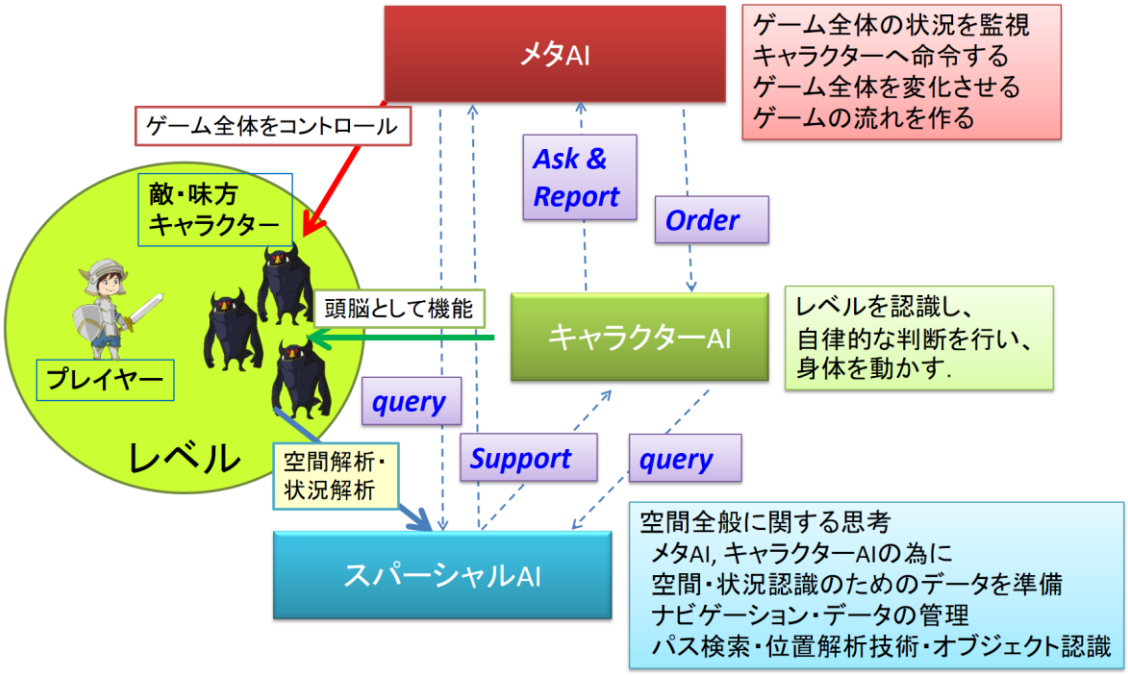


Fig. 5.1　MCS-AI 動的連携モデル（Fig.1.1 と同一）

この 3 点の改善を施したものが，「MCS-AI 動的連携モデル」である．「スパーシャル AI」は著者が提案する新しい AI であり，本モデルも新しく提案するモデルとなる．「メタ AI」「キャラクターAI」は「MCN-AI 連携モデル」と基本的な部分は同じであるが，メタ AI とスパーシャル AI が直接連携し，メタ AI はより広く詳細な空間・状況情報を用いてゲームを変化することができ，キャラクターAI はより空間・状況的特性を生かした行動ができる

ようになる．

Table 5.1　３つのＡＩとアニメーションの役割

AI・モジュールの種類	役割・機能	観察範囲	行動影響範囲
メタＡＩ	ゲームの流れを作る	ゲーム全体・ログ	ゲーム全体
ナビゲーションAI	ゲームの空間に関する思考	ゲームの環境世界	メタＡＩ・キャラクターＡＩをサポート
キャラクターＡＩ	キャラクターの意思決定	キャラクター周辺	キャラクター周辺
モーション・システム	キャラクターの身体制御	キャラクターの身体	キャラクターの身体

Table 5.2　LCN-AI 連携モデル，MCN-AI 連携モデルと MCS-AI 動的連携モデルの比較

LCN-AI 連携モデル	MCN-AI 連携モデル	MCS-AI 動的連携モデル
レベルスクリプト あらかじめ場面･領域ごとに作られたスクリプトを実行	**メタ AI** ゲーム状態をリアルタイムに認識しながら，その変化を意思決定する	**メタ AI** ゲーム状態をリアルタイムに認識しながら，その変化を意思決定する
自律型キャラクターAI レベルスクリプトとはスイッチング， レベルスクリプトによる演技，自律的な行動は自律型キャラクターAI で行う ナビゲーション AI にはクエリーを出す	**自律型キャラクターAI** メタ AI，スパーシャル AI と動的に連携 演技と自律的行動を同じ意思決定システムで行い，スムーズに行き来する．	**自律型キャラクターAI** メタ AI，スパーシャル AI と動的に連携 演技と自律的行動を同じ意思決定システムで行い，スムーズに行き来する．
ナビゲーション AI クエリーに応じてパス検索を行う	**ナビゲーション AI** クエリーに応じてパス検索を行う	**スパーシャル AI** 空間・状況解析を自律的に行い，メタ AI，キャラクターAI に提供する． クエリーに応じて空間・状況解析を行う

「メタ AI」「キャラクターAI」「スパーシャル AI」は，扱う問題の領域が異なる（Table 5.1）．メタ AI はゲーム全体の流れを作るために，キャラクターを始めとするゲームのあらゆる要素をコントロールする．キャラクターにはその状況に合った役割を指示する．キャラクターAI は環境を認識し，意思決定をし，自らの身体行動を生成する．この身体行動の生

成部分に密接にアニメーション技術が結びつく．スパーシャル AI は，環境世界の空間的特性を抽出し，キャラクターAI，メタ AI に提供する．これによって各ゲームステージは抽象化され，それぞれのステージごとに AI を製作する必要がなくなる．抽象化された空間情報の上にメタ AI，キャラクターAI が構築される．このようなそれぞれの AI の発展と新しい連携によって，LCN-AI 連携モデル，MCN-AI 連携モデルは，MCS-AI 動的連携モデルへと発展する（Table 5.2）．

本節では，LCN-AI 連携モデル，MCN-AI 連携モデルから，MCS-AI 動的連携モデルへの転換点を示した．

5.2 MCS-AI 動的連携モデルの動作原理

本節では MCS-AI 動的連携モデルの動作について述べる．その動作原理は以下である．

(1) メタ AI，キャラクターAI，スパーシャル AI は自律的にかつ独立に動作する．以下に記述することは並列的に動く．
(2) メタ AI は自律的に動作する．ゲーム状態を常に観察し，認識しつつ，キャラクターを含む NPC 全体に対して命令する．
(3) キャラクターAI は自律的に動作する．またメタ AI から時々に与えられる命令を目標として，自分自身で思考して実行する．また，キャラクターAI からメタ AI に対して行動の許可の申請や，情報の共有が行われることがある．
(4) スパーシャル AI は自律的に動作し，地形・空間を解析し，その情報をメタ AI，キャラクターAI に提供する．またメタ AI，キャラクターAI からの必要な情報を求めるクエリーに対して情報を提供する．

「メタ AI」,「キャラクターAI」,「ナビゲーション AI」が独立かつ自律的に動作する．MCS-AI 動的連携モデルは，それぞれの AI が自律的に動作しつつ，ダイナミックに連携する．メタ AI はユーザーとゲーム全体の変化に注目し，キャラクターAI は，その周囲の状況変化を注目し，スパーシャル AI はゲームの環境世界について観察する．MCS-AI 動的連携モデルによって，スクリプトによる決められた演技や，映像を使うことなく，インタラクティブなゲームの中で，自律型意思決定思考を持つ NPC 役を演じさせることが可能となる．

メタ AI からキャラクターAI へ渡される命令には 6 種類のタイプが存在する．これについて第 6.2 節で述べる．また第 7.3 節で述べるように，キャラクターAI には 7 種類の意思決定アルゴリズムがあり，それぞれの意思決定アルゴリズムに応じてメタ AI からの命令の伝え方が異なる．

5.3　MCS-AI 動的連携モデルの内部構造

本節では MCS-AI 動的連携モデルの内部構造について述べる．MCS-AI 動的連携モデルは，第 3 章で解説したキャラクターAI とナビゲーション AI のサブサンプション・アーキテクチャ（Fig. 3.4）の発展形として，階層型サブサンプション・アーキテクチャとなる（Fig. 5.2）．3 つの AI は，並列に動作しつつ，コミュニケーションする．キャラクターAI 内部も同様であり，「意思決定」「ボディ・レイヤー」「モーション・システム」は並列に動作しつつコミュニケーションする．

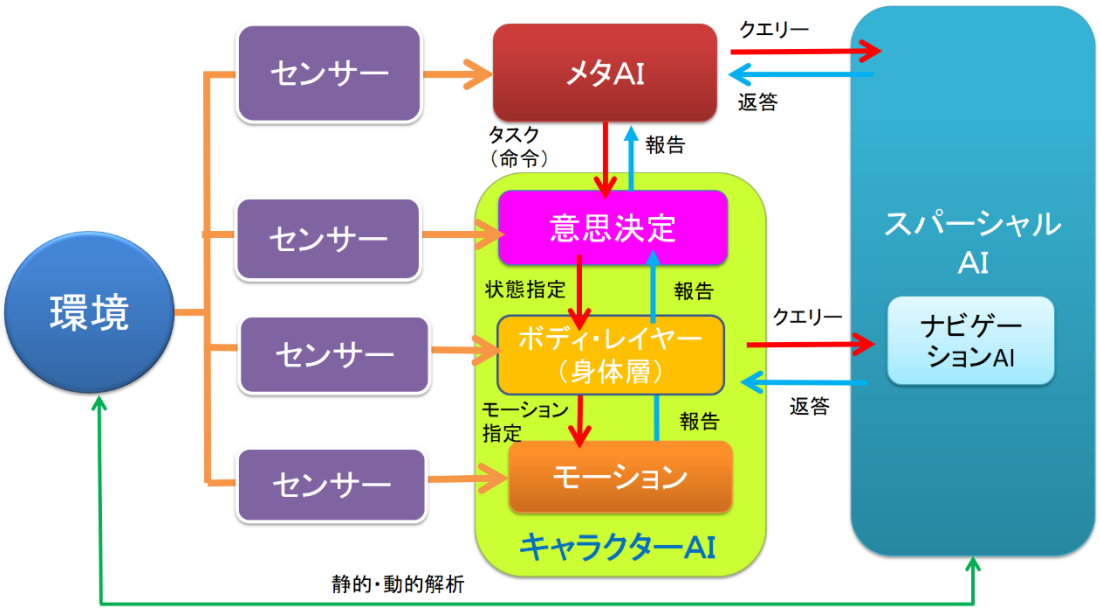


Fig. 5.2　MCS-AI 動的連携モデルの内部構造

キャラクターAI による意思決定は抽象的な意思決定である．知識表現を基礎として，環境からの情報をセンサーとスパーシャル AI によって取得するアニメーション・システムとボディ・レイヤーは，身体の自律性を持つ機構を表現している．キャラクターAI なしでも，反射的な行動を行うことができる．キャラクターAI，ボディ・レイヤー，アニメーション・システムは，サブサンプション構造によって階層化され，環境とスパーシャル AI と接続されている．

まずスパーシャル AI は，事前にゲーム環境世界を解析し世界表現を準備する．またゲーム内では，自律的に環境を解析してメタ AI，キャラクターAI にその環境内で可能な運動の情報を提供する場合と，逆に「メタ AI」「キャラクターAI」からは，情報提供のクエリーが渡される場合がある．たとえば，前者の場合には，動的に変化するゲーム状態（World State，ワールドステート，とも呼ばれる）に応じて，行動の可能性を検討し，新しく可能となった行動，或いは逆に不可能となった行動の情報を提示する．たとえば，第 3.4.2 項，第 3.5.5

節で述べたスマートオブジェクトは環境の側から可能な行動をメタ AI，キャラクターAI に伝える仕組みである．特定のオブジェクト，特定の座標において，複数の NPC を一定の期間に渡って制御することが可能である．後者の場合には，パス検索のクエリーは，パス検索のリクエストと共に，目的地が提示される．パス検索の中でも最小コストパス検索，最短距離パス検索（敵の近くを通る可能性がある），特定領域内パス検索などがある．与えられたクエリーに対して，「スパーシャル AI」は特定のデータを渡す．パス検索であればポイントの列と曲線とコスト，位置検索技術であれば候補ポイントなどである．

「メタ AI」から「キャラクターAI」に関しては「命令と報告」（Orders and Reports）形式である．まずメタ AI からキャラクターAI へ命令（Order）が下される．「プレイヤーを助けろ」や「プレイヤーの近くの敵を倒せ」「目的よりプレイヤーの前に走れ」などである．キャラクターAI は，それに命令に則した行動を行い，結果を報告する．

たとえば，メタ AI はゲーム全体の流れを見て，一体の NPC にプレイヤー・キャラクターの背後から攻撃するように指示をする．その一体を選ぶために，メタ AI は，プレイヤー・キャラクターの周囲 20m にいる数体の NPC からプレイヤーへのパス検索を行い，最小コストでプレイヤー・キャラクターにたどり着けるキャラクターを選択する．このパス検索はスパーシャル AI にクエリー（依頼）を出し，スパーシャル AI は，それぞれの NPC からプレイヤー・キャラクターへ，そもそも到達可能か，到達可能であるとしたら，そのコストを返す．メタＡＩに選ばれて「背後から攻撃せよ」の命令を受け取った NPC は，プレイヤーを攻撃するために必要な具体的な行動を，キャラクターAI を通じて形成する．たとえば，剣を振るキャラクターであれば「剣の攻撃範囲内に近づく」「剣を振る」という二つの動作を予定する．アニメーション・システムは，実際にこの二つをキャラクターの実際の運動として展開する．メタＡＩから指示のないＮＰＣは，自ら意思を決定し行動する．

本節では，MCS-AI 動的連携モデルの内部アーキテクチャを，メタ AI，キャラクターAI，ナビゲーション AI のサブサンプション・アーキテクチャからなることを示した．

5.4 MCS-AI 動的連携モデルとユーザー体験

MCS-AI 動的連携モデルはユーザー体験の形成を指向して構築・調整されるシステムである．本節では，本モデルとユーザー体験の関係について述べる．ここで言うユーザー体験は，開発者・ユーザーが主観的に体験するユーザー体験である．開発者自身がゲームプレイによって得られるユーザー体験，或いは，ユーザーから届くユーザー体験の報告，ユーザーの生体データを反映した指標などに対して，各 AI の修正・変化によってくり返し更新する．ユーザー体験という人間の体験の次元を全体的に捉える手法は未だ確立されていないが，相対的に LS-AI モデルで調整できるのは，「レベルスクリプト」「スクリプティッド AI」であり，LCN-AI 連携モデルで調整できるのは「レベルスクリプト」「キャラクターAI」「経路探索」であるが，MCS-AI 動的連携モデルは，各 AI の各機能を用いて，より大きな自由度

のもとでユーザー体験を形成できる自由度を有している．

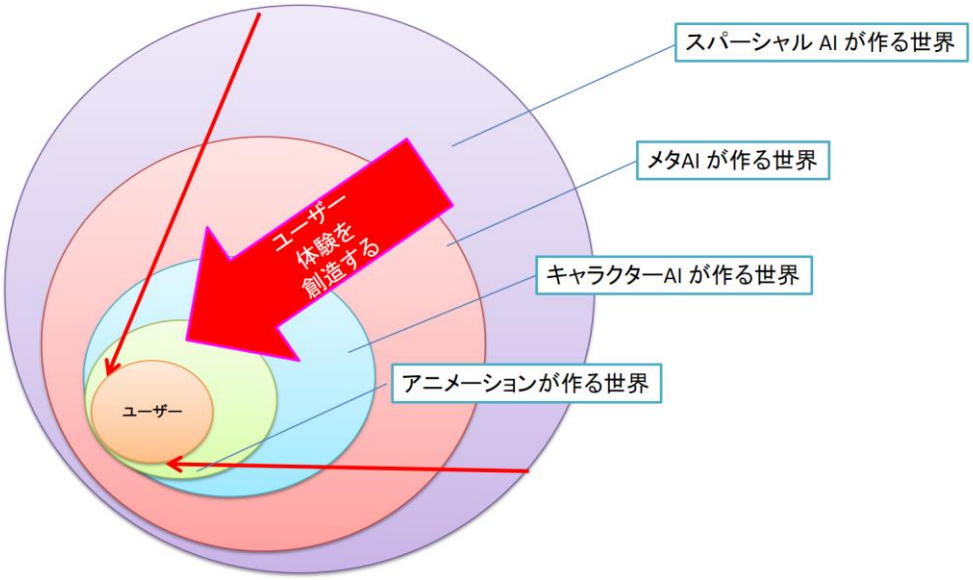


Fig. 5.3　MCS-AI 動的連携モデルによって形成されるユーザー体験

たとえば，「プレイヤーが敵に囲まれてピンチだと思う」ユーザー体験を作ることが目標であるとする．メタ AI は地形的に敵キャラクター側が有利な場所の選定をスパーシャル AI に依頼する．たとえば谷のある方向へ，メタ AI は，プレイヤーの周囲の敵数体に「谷の方向へ」指令を出す．キャラクターAI は，プレイヤーを谷の方向へ向かって攻撃する．最初のテストで十分に達成できていないと開発者が感じる場合は，各要素を調整しテストを重ねて行く．

MCS-AI 動的連携モデルは，ゲームコンテンツを作り出す AI システムである．前述のとおり，ゲームの複雑化と大規模化は，既存のように一つ一つのステージのレベルデザインを構築して行く手法ではコストと時間があまりにもかかり過ぎる上に，それだけの膨大なコンテンツを管理することも出来なくなった．そこで，メタ AI は，物語的コンテンツと状況を動的に構築する役割を持ち，スパーシャル AI は，場所ごとの特性を反映する役割を持ち，キャラクターAI はプレイヤー・キャラクターとインタラクティブに関わる役割を持ち，キャラクターの身体のアニメーションによって，プレイヤー・キャラクターと物理的インタラクションの形成を行う．この 3 つの人工知能とアニメーションによって，ユーザー体験を形成する（Fig. 5.3）．それぞれの AI の機能を変化させることで，どのようにユーザー体験を変化させることができるか，については，第 6，7，8 章で述べる．

MCS-AI 動的連携モデルは，デジタルゲームにおけるゲーム AI 開発を，AI 機能開発を，ユーザー体験の形成を橋渡しする．AI とゲームデザインを結ぶモデルでもある．空間と時間から見た場合，「メタ AI」は大局・長時間のゲーム全体の流れを形成し「キャラクターAI」はキャラクター周辺の局所・短時間のプレイヤーとのインタラクションを形成し，「スパー

シャル AI」は局所から大局に渡る時点の空間解析・状況解析を行う.「スパーシャル AI」が場所の特殊性を一般化し,「メタ AI」がゲーム全体の流れを作り, キャラクターAI がプレイヤーとのインタラクションをコントロールすることで, そのゲームのユーザー体験を形成する (Fig. 5.4).

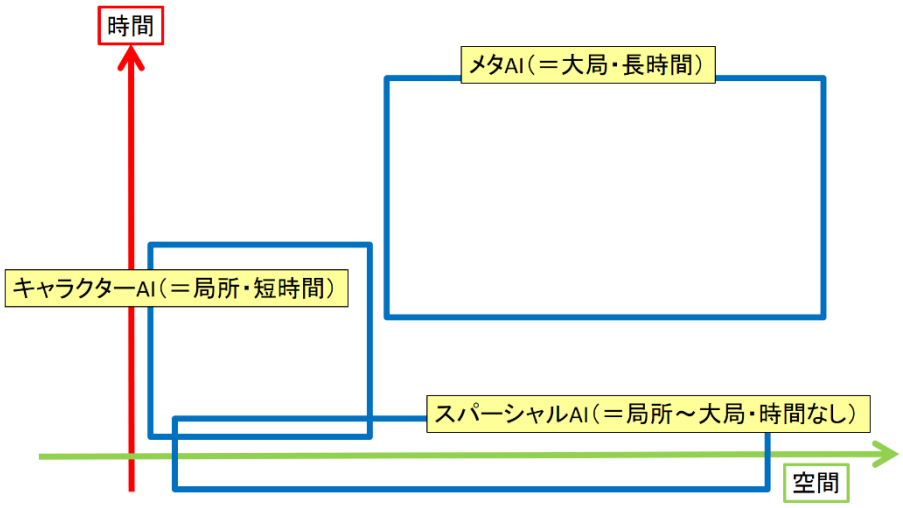


Fig. 5.4　MCS-AI 動的連携モデルの時間-空間スケール

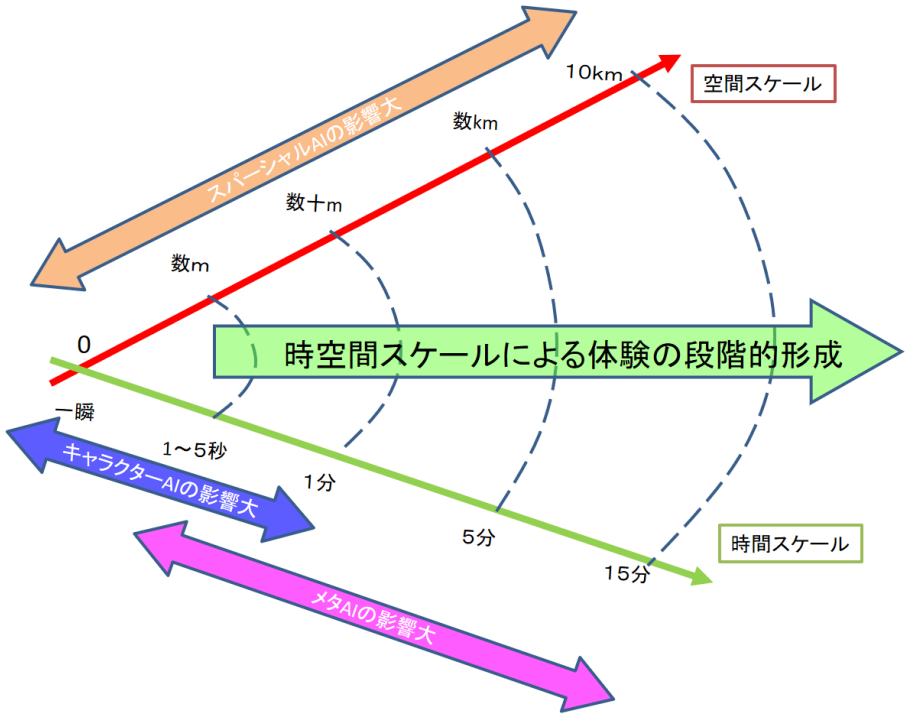


Fig. 5.5　MCS-AI 動的連携モデルとユーザー体験の段階的形成

ゲームデザインは通常, 多層的に形成される. 一瞬の楽しみ, たとえば, 敵を剣で攻撃する, などである. 次に数秒の楽しみがある.「敵を倒して経験値が入る」などである. さら

に「敵を複数倒してレベルアップする」という数分の楽しみがある．レベルが上がっていくと「新しい物語展開が広がる」という十数分の楽しみがある，と言ったようにである．このような階層的なユーザー体験に対して，キャラクターAI は局所・短時間の領域で，メタ AI は大局・長時間の領域で支配的である．空間的にはスパーシャル AI が常に局所～大局までのステージの空間・状況解析を担当する（Fig. 5.5）．

本節では，MCS-AI 動的連携モデルにおける各 AI が，それぞれ時間・空間スケールを持ち，その時間・空間階層構造によってユーザーの経験の形成を指向するモデルであることを述べた．

5.5　MCS-AI 動的連携モデルによるコンテンツの分解・蓄積と実現・生成

本節では，MCS-AI 動的連携モデルが，いかに技術を蓄積し，また逆にそれらを組み合わせてコンテンツを実現して行くのかを述べる．ここで言うコンテンツとは，そのゲームが実現したい戦闘・ドラマなどのことである．第 1.4.2 項で述べたように，MCS-AI 動的連携モデルの利点は，ゲームから要求される要件を，３つの AI の技術として分解して実装することで，それぞれの技術をゲーム内で再利用可能なものとして蓄積できるところにある．この時，分解によって得られる技術とは，Fig. 2.4 に見られるように，アーキテクチャ，モジュール，アルゴリズムなどである．そして，ある程度の各 AI の機能の蓄積が終わると，あとは，それらの機能の組み合わせるだけで，様々な AI 要件を実現できるようになる（Fig. 5.6）．

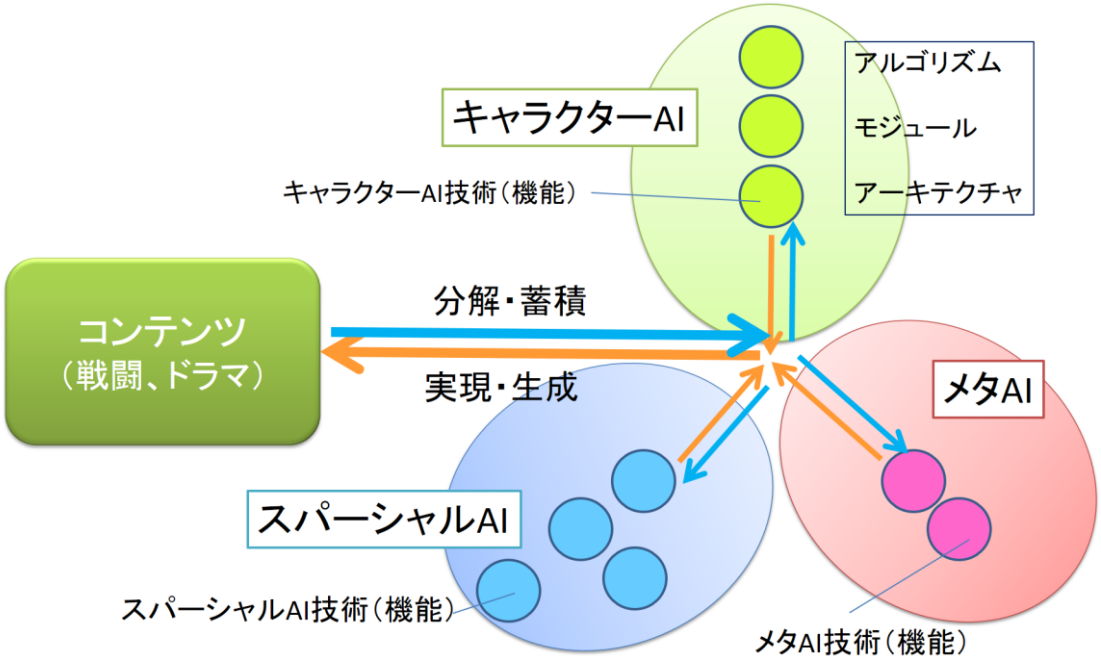


Fig. 5.6　MCS-AI 動的連携モデルによるコンテンツの分解と生成

第 1.4.2 項で述べたように，この再利用性と組み合わせによる開発によって，ゲーム AI 開発は効率化される．特に，大規模なゲームの開発の場合には，開発前半で蓄積された各 AI の機能によって，中盤〜後半以降，それらの組み合わせで，ゲームからの要求を実現できる．このような分解と再利用性は，一つのゲームで閉じるものではなく，すべてのゲームタイトルを通じて，メタ AI，キャラクターAI，スパーシャル AI の技術が蓄積され，再利用される．MCS-AI 動的連携モデルは，ゲーム開発全般の効率化を促進する．そこで，第 6〜8 章では，これまでのゲーム開発で蓄積されてきたメタ AI，キャラクターAI，スパーシャル AI の技術を整理する．さらに，第 9 章では，２つのゲームから実際に AI 技術を抽出する事例を示す．第 12 章では逆に，これまで蓄積してきた 3 つの AI の技術を用いて，具体的なゲームの製作を通じて，新しいユーザー体験を生成する事例を述べる．第 13 章では，これまで MCS-AI 動的連携モデルが取り込めてこなかった学習・進化・プロシージャル技術の取り込みについて述べる．

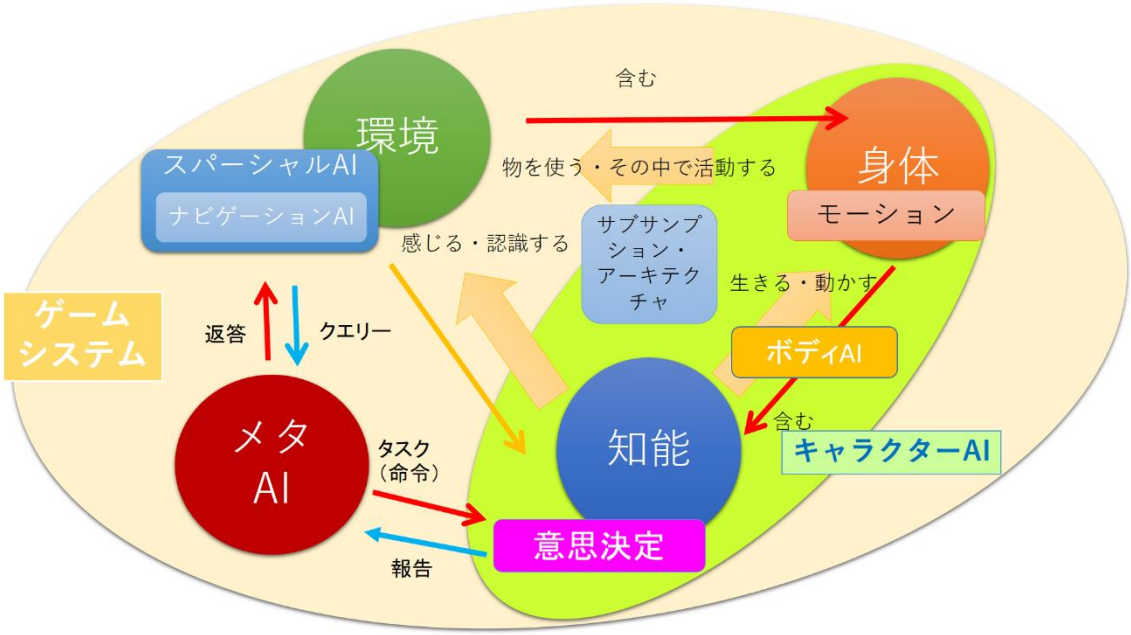


Fig. 5.7　ゲームキャラクターにおける「環境」「身体」「知能」の関係

5.6　考察

本節では，本章全体の考察を行う．MCS-AI 動的連携モデルは，ゲーム内における「環境，身体，知能」について，それぞれの場所に分散して AI を構築し，協調するシステムである（Fig. 5.7）．「環境」についてはスパーシャル AI が，知能・身体についてはキャラクターAI が，ゲーム全体についてはメタ AI があり，それぞれの AI が，自分の領域を認識し，他の AI と協調しつつ動作する．「メタ AI」によって，ゲームとして進めたい明確なゴールが提

示され，それを実現するために，キャラクターAI を始めゲーム全体が変化し，その全体の認識を「スパーシャル AI」がサポートする．

デジタルゲームはすでに全体が巨大なダイナミクスであるが，そのダイナミクスの上に，人工知能のネットワークが構築される．そして，逆に構築された人工知能ネットワークのダイナミクスを通して，デジタルゲームのダイナミクスを操作する．これが「MCS-AI 動的連携モデル」である．ゲームが精緻化・大規模化・複雑化が，それぞれの AI を生み出し，生み出された AI たちが協調するようになり，動的なネットワークを形成し，形成されたネットワークから逆にゲームを制御する．そして，ゲーム製作を通じて新しく加わる AI 機能は「メタ AI」「キャラクターAI」「スパーシャル AI」のいずれかに含まれることになる．

本章では，MCS-AI 動的連携モデルの提案を行った．以降の第 6 〜 8 章では，MCS-AI 動的連携モデルにおける「メタ AI」,「キャラクターAI」,「スパーシャル AI」，各 AI の具体的な技術と相互の連携について述べる．

6. メタ AI

本章では「MCS-AI 動的連携モデル」の中の「メタ AI」の役割と機能について示す．メタ AI はゲームシステムに宿る人工知能である．メタ AI はリアルタイムにゲーム状態を監視し，ゲームコンテンツに関する調整・変化を行う．MCS-AI 動的連携モデルにおけるメタ AI は，この全体のモデルが目指すユーザー体験を，キャラクターAI，スパーシャル AI を利用して実現する役割を持つ．メタ AI はゲームデザインの要であり，そこにゲーム開発者，特にゲームデザイナーの思考がソフトウェアとして構造化されることになる（Fig. 6.1）．

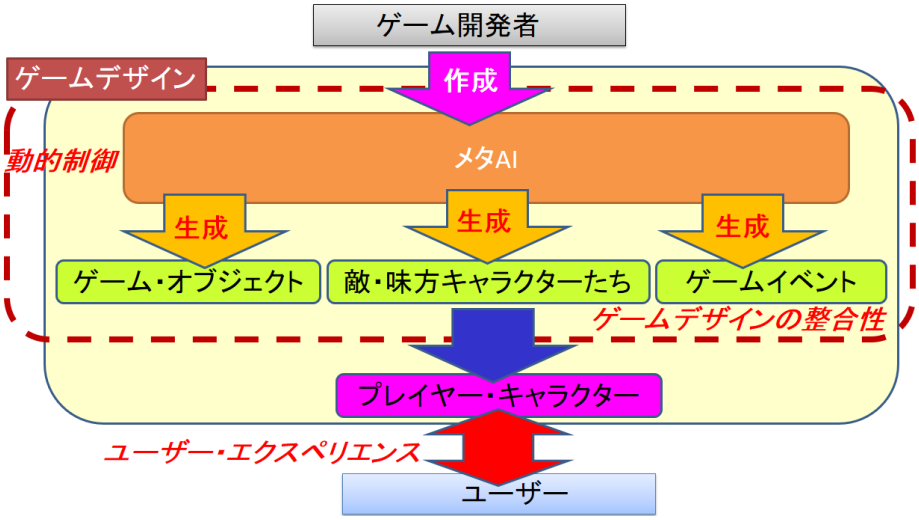


Fig. 6.1　メタ AI の概念図

Table 6.1　ゲームの静的から動的なシステムへの変化

ゲーム内分野	静的	動的
オブジェクトの運動	運動曲線など 静的データ	物理シミュレーション
地形・自然物・街など	固定モデル	プロシージャル技術による生成・変形
キャラクターの移動	固定パス	ナビゲーションAI
キャラクターの思考	スクリプティッドAI	自律型AI
ゲームイベント （戦闘、エンカウントなど）	開発中に作成・固定	AIディレクター （ユーザーを観察しつつ動的に生成・変化）
ゲーム	開発中に作成・固定	メタAI （ゲーム状況を観察しつつ動的に生成・変化）

デジタルゲームはその誕生以来，ゲーム開発者がゲームの様々な部分を作り込むことによって，固定化したコンテンツをユーザーに届けることで高い品質のコンテンツを提供し続けて来た．たとえばゲームがリリースされた時点でゲームの内容は，敵の配置，出現タイミング，難易度，動き，役割など乱数などランダムを用いる箇所を除いて定まっている．人工知能技術は，そのコンテンツを動的なシステムへと変化させる機能を持つ（Table 6.1）．さらに，メタ AI はゲームシステムレベルで，ゲームコンテンツへさらに動的により大きな変化をもたらす人工知能である．

6.1　メタ AI の内部構造

本節では，メタ AI の内部構造を示す．メタ AI の内部構造は，第 3.3.1 節で述べた，キャラクターAI と同じエージェント・アーキテクチャ（Fig. 6.2）を持つ．メタ AI は自ら身体は持たず，そのエージェント・アーキテクチャは，ゲーム全体の空間状態を俯瞰的な視座から大局的に認識し，短時間から比較的長時間（瞬時～数十秒）までの意思決定を行う（Table 6.2）．エフェクターはゲームのあらゆる要素である．ノン・プレイヤー・キャラクターへの命令や地形・天候の変化からサウンド，物語生成まで，ゲームのあらゆる要素を変化させることでユーザー体験を変化させる．

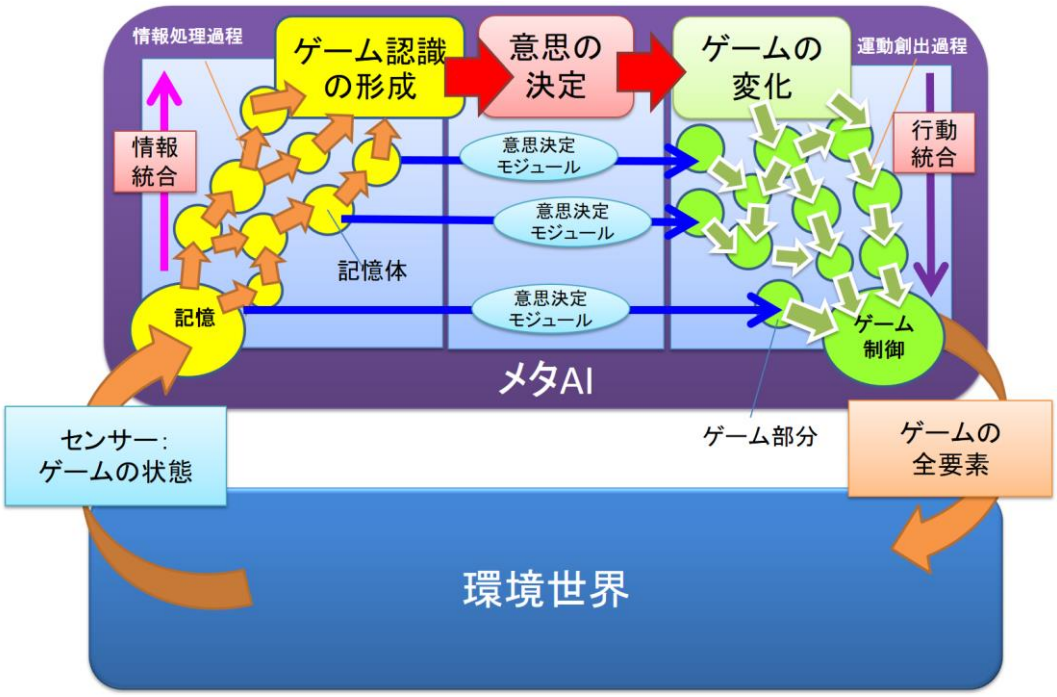


Fig. 6.2　メタ AI のエージェント・アーキテクチャ

メタ AI のセンサーは大局的かつ長時間に渡るゲーム世界の把握するために，通常のキャラクターAI のセンサーとは異なる．ゲーム全体を把握するために，たとえばナビゲーショ

ン・メッシュを単位として，メッシュ上の敵，味方キャラクターの状態を集める，或いは，影響マップを通してゲーム全体をグリッド状に分割した上で，敵．味方の勢力図を把握する(第 8.4.2 項)，などであり，空間全体を把握する知識表現を持つ．またゲーム全体の流れを把握するために，時間方向に対してはログを集積する．たとえば，プレイヤー・キャラクターの座標を取得し続けることで，プレイヤー・キャラクターの予測経路を割り出すことができる．

メタ AI の意思決定に関しては最終的にはゲームのどの要素をどのように動かすかを決定する必要がある．意思決定の手法はキャラクター同様，7 つの意思決定アルゴリズムのいずれかを用いることになる．短期的には一つの行動を NPC に指定するなど即自的変化を促すパターン，そして，長期的にゲームをあら方向に導いて行くものがある．前者はルールベース，ユーティリティ・ベース，ステートベース，ビヘイビアベースなど反射型意思決定が向いており，後者はゴールベース，タスクベース，シミュレーション・ベースが有効である．本節では，メタ AI の内部の基本構造と，キャラクターAI との対比について述べた．次節以降では，メタ AI がゲームの中で果たす役割と効果について述べる．

Table 6.2　キャラクターAI とメタ AI のエージェント・アーキテクチャの比較

エージェント・アーキテクチャの要素	キャラクターAI	メタAI
センサー	自分の身体を中心とした 局所的・短時間の変化を認識	ゲーム全体を大局的・ 長時間に渡る変化を認識
意思決定	短時間の自分の身体的行動を 決定する	大局的・短～長時間に渡るゲーム全体 の変化の方向を決定する
エフェクタ	キャラクターの身体	ゲーム全体・ ゲームのあらゆる要素

6.2　メタ AI の効果の分類

本節では，メタ AI がゲームに作用するパターンを分類する．メタ AI のゲームへの影響の仕方は 3 種に分類できる．

(I)　キャラクターをコントロールする場合
(II)　ゲームをコントロールする場合
(III)　コンテンツ生成する場合

である．(I) は戦闘などキャラクターの行動を変化させ，プレイヤーとのインタラクションに関わる変化である．(II) はイベント・クエスト生成やゲームダイナミクスそのものの変

化である．(III) はまったく新しいキャラクターを生成する，など，時間に捉われない新しいゲームの楽しみを提供する，場合である．以下，これを順番に説明して行く．

6.3 キャラクターへ命令する場合

本節ではメタ AI がキャラクターへ命令を与える場合について述べる．メタ AI はキャラクターへ命令することで，ユーザー体験上に狙った効果を実現しようとする．「ゲームの緩急」「難易度」「キャラクター表現」などである．これを順番に解説する (Table 6.3)．

Table 6.3 メタ AI の分類(I)「キャラクターをコントロールする場合」

ゲームタイトル分類名	名称	センサー範囲	意思決定（決定すること）	エフェクター（影響範囲）	効果
パックマン	波状攻撃	ゲーム状態	プレイヤーを包囲する･しない	敵キャラクター	緩急
LEFT 4 DEAD	AI Director	ユーザーの緊張度･位置 ユーザーのスキル	予測経路･配置キャラクター･ 配置タイミング･配置場所･	敵キャラクター	緩急
Warframe	AI Director	ユーザーの緊張度･位置	予測経路･配置タイミング･配置場所	敵キャラクター	緩急
LOST REAVERS	AI Director	ユーザーの感情値 プレイヤーの周囲の状況	敵キャラクターの出現位置･数	敵キャラクター	緩急
ゼビウス	AI	ユーザーの進行度	出現する敵テーブルの開始点	敵キャラクター	難易度
FINAL FANTASY XV	メタAI	プレイヤー･キャラクターと仲間、周囲の状況	仲間キャラクターの振る舞い	仲間キャラクター	表現

（注）『LEFT 4 DEAD』(Valve Software，2008 年）ではキャラクター全体を監督する AI を AI Director と呼んでいる [27] [54]．メタ AI の機能をキャラクターに限定した AI である [2]．この種別については第 15 章で述べる．

6.3.1 ゲームの緩急

ゲームの緩急は，ユーザーをゲームに引き込むために必要なリズムである．小さなゲームでは自然にできていたことも，ゲームの大規模化と複雑化のために，メタ AI による人工的なゲームプレイの緩急の生成が必要である．緩急には様々あり，難易度，緊張と緩和，タイムアタック的なゲームプレイとそうでないゲームプレイ，など様々であるが，メタ AI が介入する場合には，ユーザーの状態を推定してゲームの緩急をコントロールする．

たとえば第 9 章で詳細に説明する『パックマン』(株式会社バンダイナムコエンターテインメント，1980 年）は 4 匹のモンスターに一定時間でプレイヤーを包囲させた後，一定時間，4 隅へ離散されることを繰り返すことで，ゲームに緩急を作り出している [93]．

『LEFT 4 DEAD』(Valve Software，2008 年）は 4 人が一グループとしてプレイする襲い来るゾンビの集団と戦うマルチオンラインアクションゲームである．Valve Software の Turtle Rock Studios が前作『Counter Strike』の経験をもとに製作を行った．『Counter Strike』

が世界的に人気を博した理由を開発者自ら調査する過程で，偶然のバランスではあるが，攻撃と休止が交互に来ることを見出した．そこで，次回作ではそのようなゲームの緩急を人工的に作り出すために，敵キャラクターの出現の場所とタイミングをコントロールする「AI ディレクター」（AI Director）を導入した [27] [54].

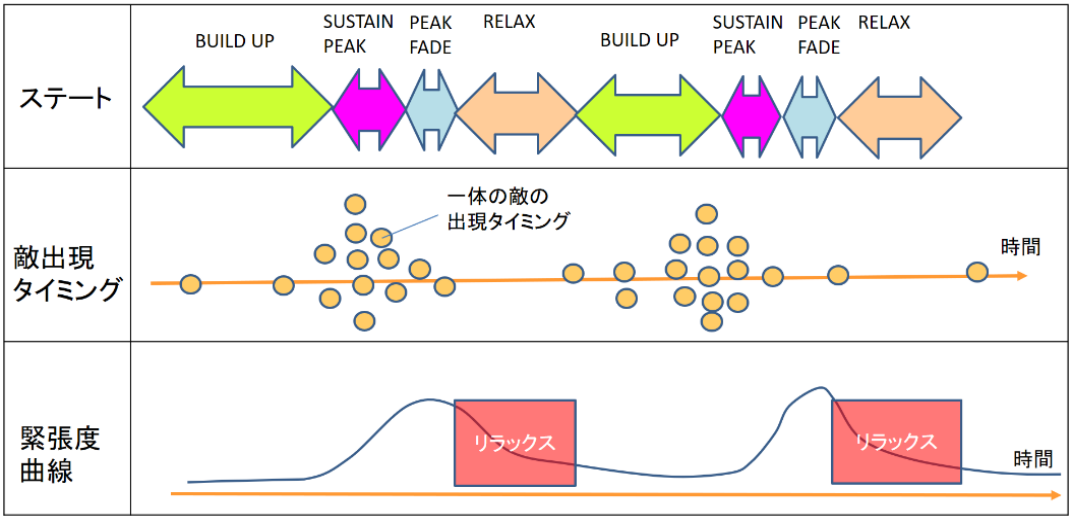


Fig. 6.3　『LEFT 4 DEAD』のユーザー緊張度曲線（中央）[27]から引用

AI ディレクター はゲームからユーザーの行動ログを集めて，ユーザーの緊張度を推定し，そこから NPC の出現タイミングと場所を決定し命令を出す．Fig. 6.3 においては，中法の曲線が，緊張度推定曲線である．一番上の点分布図は，メタ AI が指令を出した敵の出現タイミングと数である．一番下の点分布図は実際に出現した敵の出現タイミングと数である．ユーザーの緊張度の推定は，開発中は，手の発汗量（スキンコンダクタンス，SCL）や心拍数などから推定する．しかし，実際の市販のゲームではそのような生体デバイスをユーザーに着けることはできないので，ゲーム内の情報だけからユーザーの「緊張度」を計算する必要である．そこで，以下のルールを敷いて緊張度とみなす．

(a) 敵を撃破した時，敵への距離に反比例した値をプラスする．
(b) 戦闘ができない状況になったら値をプラスする
(c) ダメージを受けたら，ダメージに比例した値をプラスする
(d) 物理的に，押される/引かれる，というイベントが起こったらプラスする

これらのルールによる緊張度が一定間隔の時間ごとに計算される．このように計算された緊張度を使ってメタ AI は敵キャラクターを生み出し，ユーザーの緊張度を 4 つのプロセスを通じてコントロールする．ユーザーの緊張度を時系列データとして記憶し，ユーザーの緊張度に応じて状況を 4 つに分け，それぞれの状況にあわせてゲームを変化させる．

(a) **上昇状態**　プレイヤーの緊張度が目標値を超えるまで敵を出現させ続ける．
(b) **平衡状態**　緊張度のピークを 3-5 秒維持するために敵の数を維持する．
(c) **下降状態**　敵の数を最小限へ減少していく．
(d) **緩和状態**　プレイヤーたちが安全な領域へ行くまで，30-45 秒間，敵の出現を最小限に維持する．

このように『LEFT 4 DEAD』のメタ AI は上記の 4 つのステートを巡回する「ステートマシン」の構造を持っている．

『Warframe』（デジタル・エクストリームス，2013 年）は，第 12 章で詳説するが，ゲーム内では，プレイヤーを熱源としたヒートマップによる熱伝播シミュレーションが行われて各グリッドで温度が定義される [94] [95]．プレイヤーが近づき温度が上がって行く位置に敵キャラクターが生成される．逆に温度が下がって行く位置にある敵キャラクターは消滅される．

『LOST REAVERS』（株式会社バンダイナムコエンターテインメント，2015 年）はダンジョンに分け入って財宝を持ち返るアクションゲームである．ステージにあらかじめ配置されたキャラクターの他に，AI ディレクター が動的にキャラクターを配置し，敵の総数をコントロールする [96]．開発中には，ダメージを受けたタイミング，回復したタイミングのデータと，プレイ動画から苦戦した場所と余裕な場所を見極めて，感情値の数式が決定され，メタ AI のコントロールの指標とされる．

本項では，メタ AI がキャラクターへの命令を通じて，ゲームプレイの緩急を付ける手法を紹介した．

6.3.2　ゲームの難易度

本項では，ゲームの難易度をメタ AI によってコントロールする手法を述べる．最もシンプルな方法としては「EASY MODE」「HARD MODE」など静的に分ける場合もあるが，これは 2 通りの難易度しかない．メタ AI による難易度調整とは，動的にユーザーのスキルを認識しながら難易度を調整して行く手法である [97]．

たとえば『ゼビウス』（株式会社バンダイナムコエンターテインメント，1982 年）では，空中の敵は出現する順番が決まっており，プレイするとどんどん強い敵になって行くが，一度撃墜されると，出現テーブルが巻き戻り，弱い敵からやり直すことになる [98]．地上の敵の配置は変わらないので，ステージ自体はどんどん進んでいく．このようにゲームは一定のスピードで進行しながらも，レベル自動調整機能によって，難易度を自動的に調整することができる．

このようにメタ AI のキャラクターによる難易度調整は，敵キャラクターの出現をコント

ロールする方法があり，また，別の手法としては出現させて敵の賢さや性能を上下することでコントロールする．

6.3.3 キャラクター表現

本項では，メタ AI によるキャラクター表現について述べる．物語的ゲームにおいては，キャラクターは文脈に沿った演技をしなければならない．ムービーを挿入するようなノン・インタラクティブな決まった演技もあるが，たとえば戦闘中や移動中など，自律的に行動している中でも，メタ AI によって，所々に演技的な要素を入れることで，演出的効果を施すことが可能となる．

『FINAL FANTASY XV』（スクウェア・エニックス， 2016 年）は，プレイヤーと旅をする 3 人の仲間に対するメタ AI が実装されている [99] [100] [101]．それぞれのキャラクターは自律型キャラクターAI を持ち，自らのセンサー，意思決定，身体動作で行動する．しかし，戦闘や日常シーンの会話などで，いたるところで全体のコンテクストに合った行動・発言を仲間キャラクターに取らせるために，メタ AI が活用される．

メタ AI は調整役としての機能である．プレイヤーと各仲間の状態を監視して，プレイヤーがピンチの時には，一番近くて都合が良い（別の敵に攻撃している最中でないなど）仲間を選んで，最優先で駆けつけるように指令を出す．戦闘中メタ AI が仲間に与える指示としては，

(a) プレイヤーや仲間のピンチを助けよ
(b) プレイヤーが敵に拘束されているから助けよ
(c) プレイヤーが逃げているから追従せよ
(d) 作戦が発動したのでそれに合わせた行動をせよ

などである．このような指令を最適なキャラクターに送ることによって，ユーザーが仲間との友情を感じる演出と緩急のある戦闘を同時に実現することが目指された．

これまでは映像やレベルスクリプトによる完全なキャラクターのコントロールによる演劇的シーンなど，自律型キャラクターAI をオフにした形で演出が差し込まれるケースがほとんどであった．メタ AI によるキャラクター表現は，自律型キャラクターAI で動かしつつ，動的に演出を加える新しい手法を拓いた [100]．

6.3.4 考察

本節は，メタ AI がキャラクターをコントロールすることで，ユーザー体験上に現れる効果について述べてきた．NPC は最もダイレクトにプレイヤーとインタラクションする存在

であり，メタ AI によってキャラクターをコントロールすることは，もっとも直接的にユーザー体験を生み出す．メタ AI は，カットシーン映像を挟むことなく，インタラクティブなゲームプレイの中でキャラクターに演出的行動をさせることが可能にする．

6.4　ゲームをコントロールする場合

本節はメタ AI がゲーム全体をコントロールする場合について述べる（Table 6.4）．メタ AI は，キャラクター以外にも，ゲームを構成する要素を操作することで，ゲームを変化させることができる．メタ AI がユーザーの操作から，ゲーム全域に渡ってゲームダイナミクスをコントロールしゲームを変化させて行く場合がある．また，ゲーム全域ではなく，ゲームの一部をユーザーの行動履歴に応じて変化させる場合がある．本節では以下，この 2 つの場合について述べる．

Table 6.4　メタ AI の分類(II)「ゲームをコントロールする場合」

ゲームタイトル 分類名	名称	センサー範囲	意思決定 （決定すること）	エフェクター （影響範囲）	効果
SimCity	メタAI	ゲーム状態	街の発展	ゲーム世界全体(街)	ゲーム体験
The Sims	メタAI	ゲーム状態	キャラクター生成・街の発展	ゲーム世界全体（街・人）	ゲーム体験
Spore	メタAI	ゲーム状態	キャラクター、森、惑星の創造	ゲーム世界全体（宇宙）	ゲーム体験
Assassin's Creed Origins	メタAI	プレイヤー・キャラクターの位置	発生させるエリア・オブジェクト・キャラクター	ゲーム世界全体 （オブジェクト・キャラクター）	ゲーム世界
The Elder Scrolls IV: Oblivion	Radiant AI	プレイヤーの行動履歴	プレイヤーとのインタラクションの仕方	NPCのプレイヤーとのインタラクション	インタラクションの変化
FarCry4	AI Director	ユーザーの行動ログ・場所の情報	発生させるイベントの種類・場所・タイミング	プレイヤー以外の 全てのキャラクター	イベント 生成
The Elder Scrolls V: Skyrim	Radiant Stories	プレイヤーの行動履歴	新しいクエストの内容	クエスト	クエスト生成

6.4.1　ゲーム全域に渡ってゲームダイナミクスをコントロールする

本項では，メタ AI がゲーム全域に渡ってゲームダイナミクスをコントロールする場合を述べる．ゲーム全体を認識し，メタ AI の判断によって，ゲームを一つの方向にドライブする場合である．たとえば，ストラテジーゲームなどで，メタ AI が風向きをコントロールして，劣勢の軍勢が優勢になるようにコントロールする，というケースが考えられる．風上から野原に火をつけて敵軍にダメージを与える，などである．このマップ全体の情勢を認識するためには，「影響マップ」が多く用いられる．影響マップを用いたメタ AI によるゲーム全域に渡ってゲームダイナミクスをコントロールする例については，第 8.4.2 項で詳細に説明する．

これ以外に，メタ AI がユーザーとゲームダイナミクスを介在し，ユーザーの操作がメタ

AI によって解釈され，ゲーム全域に影響を及ぼすような場合がある．通常のユーザー操作によって引き起こされるゲームダイナミクスとの違いは，メタ AI によってユーザーの操作が何倍にも増幅され，或いは，自動的に修正され，ゲーム状態に影響を及ぼす，という点にある．僅かなユーザーの操作でも，それが何倍にも増幅した効果をもたらす場合は，たとえば，ユーザーに何らかのジェネレーションをゲーム内でさせたい場合に有効である．以下，実例を示す．

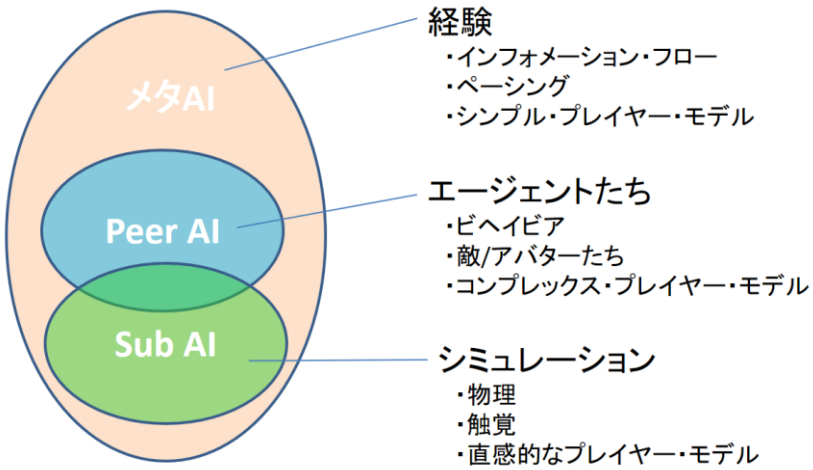


Fig. 6.4　「Meta AI」「Peer AI」「Sub AI」の分類図［1]から引用

「メタ AI」という言葉を最初に使ったのは，米のゲームデザイナーのウィル・ライト（Will Wright,　1960-）である．「AI : A Design Perspective」［1]と題して 2005 年に AAAI の分化会 AIIDE(AI and Digital Entertainment）2005 で講演した中に，ゲーム AI の 3 つの分類「Meta AI」「Peer AI」「Sub AI」が行われた（Fig. 6.4）．メタ AI は，ユーザーをモデル化し，ゲームのペースを始めとしてゲーム全体を操作することで，体験のコントロールを指向する AI である．

「メタ AI」はユーザーの体験を対象とする AI である．ユーザーをモデル化し，ゲームのペースを始めとしてゲーム全体を操作することで，体験のコントロールを指向する．これは「ゲームデザイナー」に相当する役割を持つ．すなわち，データを生成し配置しユーザーをそこでプレイさせる．「メタ AI」は時にそのような役割を「Peer AI」「Sub AI」の力をかりて実行する．「Peer AI」は「キャラクターAI」と同義である．「Sub AI」はゲーム世界の自律シミュレーション機能である．ゲーム特有のゲーム世界の発展の仕方を定義する．

たとえば，『シムシティ』（MAXIS，1989 年）では，ユーザーが配置した街のオブジェクトが影響マップを通して街全体に影響を及ぼして行く「Sub AI」の機能が主であり，キャラクターAI である「Peer AI]の機能はほぼなく，「メタ AI」は「Sub AI」に大きく依存しながらゲームを進行する［102]．『The Sims』（MAXIS，2000 年）では，キャラクターは自動生

成され，キャラクターはその内面のパーソナリティー・モデルに従って，ユーザーの配置したアイテムを使用して行動するため，「Peer AI」の機能が大きく，「Sub AI」の機能は街や人間関係の発展のみを担当する［103］.『The Sims』では「メタ AI」は「Peer AI」に大きく依存しながらゲームを進行する.

『Spore』（MAXIS，2008 年）ではユーザーのゲームプレイを補助し，データを自動生成する AI が実装されている［104］［105］. たとえばユーザーがキャラクターを作ると，その形状に合わせてアニメーションが自動生成される.「メタ AI」はユーザーのプレイを解釈し幾重にも拡大してゲーム全体を変化させる役割を持っている.「Peer AI」は「The Sims」で使用されたシステムがほぼ踏襲されているが，「Sub AI」はミクロな宇宙からマクロな宇宙までを通してシミュレーションする巨大なシミュレーション・システムとして機能している.『Spore』はこのように自律発展する宇宙にユーザーが干渉しながらプレイする仕組みとなっており，その宇宙に対するユーザーの干渉の仕方を「Meta AI」がコントロールしている.

本項では，メタ AI がゲーム全域に渡ってゲームダイナミクスをコントロールする場合について，メタ AI がゲーム内の全知全能の存在としてゲームを変化させる場合（詳しくは第 8.4.2 項参照）と，メタ AI がユーザーのインプットを増幅してゲームを変化させ，ユーザーを全能の存在として振舞わせる 2 つのパターンが存在することを述べた.

6.4.2　ゲームの一部をユーザーの行動履歴に応じて変化させる場合

本項では，メタ AI がゲームの一部をユーザーの行動履歴に応じて変化させる場合について述べる. プレイヤーの行動履歴に応じたイベント生成やクエスト生成，或いはＮＰＣとのやり取りの変化などである. その目的は，すでに作られたイベントやクエストではコンテンツが足りない場合に，生成機能によって補う，というものである. 或いは，プレイヤーの行動がゲームに反映することで，ユーザーを楽しませるためである.

『FarCry4』（Ubisoft Montreal，2014 年）ではメタ AI がプレイヤーを監視し続けて，イベントを自動生成する「動的エンカウンターシステム」が実装されている［106］.「エンカウンター」とはここではキャラクターとの遭遇，という意味であり，イベントと同義である.『FarCry4』は，広大なフィールドがシームレスにつながったオープンワールド型のゲームであり，オープンワールドにおけるメタ AI のあり方と効果を示した. まず 4 種類のキャラクター「原住民」「敵兵士」「野生動物」「仲間」を用意してイベントを作る. イベントは単体で出ることもあれば，これらをインタラクションさせて作成する場合もある.

(1)　ジープを奪う
(2)　野生動物を狩る（ゾウやトラ）
(3)　動物を飼育する

(4)　野生動物同士が争う

またより大きな一連のイベントも用意されている．たとえば，

(5)　見回りに行く
(6)　捕虜を捉える
(7)　基地に連れて行く
(8)　拷問をする

など一連のイベントである．これらは「塔」「幽閉所」「キャンプ場」「工場」「ミッションポイント」など舞台装置の近くで発生する．「エンカウンター」を特徴付けるのは地上，道路，水辺，野生区域などの地形タイプ，時刻，天気，頻度，優先度，などの条件設定，ランダムに生成される「エンカウンターリスト」である．AI ディレクターは，多すぎず，少なすぎず，最小限の距離を置いて，場所やエリア，時刻や天候，発生頻度，エンカウンターの優先度など考慮しつつエンカウンターを生成する．前作であれば，バグの 80%がミッションに起因するものであったが，動的エンカウンターシステムによってミッションを自動生成することで，大量のミッションを一つ一つデータとして生成するよりも大きく製作工程を効率化した［106].

『The Elder Scrolls IV: Oblivion』(Bethesda Game Studio，2006 年)，『The Elder Scrolls V: Skyrim』(Bethesda Game Studio，2011 年) では「Radiant AI」「Radiant Stories」が実装されている［107].「Radiant AI」は，プレイヤーの行動履歴から NPC の行動を変化させる．「Radiant AI」によって NPC たちは一日のスケジュールを持ちながらも，プレイヤーの行動履歴と特性に応じてプレイヤーとのインタラクションを変化させる．「Radiant Stories」はプレイヤーの行動履歴からクエストを自動生成し，NPC にロールを割り当てる．「Radiant AI」は AI ディレクターレベルの機能であり，「Radiant Stories」はメタ AI レベルの機能である．『Oblivion』から『Skyrim』へ向けて，「Radiant AI」から「Radiant Stories」が発展した．

本項では，メタ AI がゲームの一部をユーザーの行動履歴に応じて変化させる場合について述べた．プレイヤーの行動履歴をインプットとすることで，各ユーザーにとってユニークな展開を創出するのが目的である．このようにメタ AI はユーザーに固有の体験を与える役割を持つ．

6.4.3　考察

本節では，ゲーム全体をコントロールするメタ AI が生み出す，さまざまなユーザー体験を述べた．ゲームそのものに展開や，ダイナミクスを，メタ AI が変化させることで，ユー

ザーのプレイの波及を大きくし，イベントを生成する．これは，決められたコンテンツを実現するためにあったレベルスクリプトと対照的である．次節では，メタ AI の指示によってコンテンツそのものを生み出す場合について述べる．

6.5　コンテンツ生成する場合

本節ではメタ AI がコンテンツ生成する場合について述べる．メタ AI がプレイヤーの行動履歴から，新しいコンテンツを生成しユーザーに提供する，というパターンがある．特に，新しいキャラクター・コンテンツを生成する場合がある．以下，この事例を説明する（Table 6.5）．

Table 6.5　メタ AI の分類(III)「コンテンツ生成する場合」

ゲームタイトル 分類名	名称	センサー範囲	意思決定 （決定すること）	エフェクター （影響範囲）	効果
Shadow of Mordor Shadow of War	Nemesis System	プレイヤーとの戦闘の履歴	身体の変化のさせかた	敵キャラクターの外見	新しいキャラクターコンテンツの提供
Forzamotor Sports	Drivatar	プレイヤーの行動履歴	プレイヤーのプレイの特徴	キャラクターの学習・創造	新しいキャラクターコンテンツの提供
Killer Instinct	Shadow	プレイヤーの行動履歴	プレイヤーのプレイの特徴	キャラクターの学習・創造	新しいキャラクターコンテンツの提供

『Shadow of Mordor』（Monolith Productions, 2014 年），『Shadow of War』（Monolith Productions, 2017 年）は，一度倒した敵キャラクターが，プレイヤーとの戦闘の履歴を反映した経験と外見を持って復活し，プレイヤーへ復讐しに来る．プレイヤーの倒し方が外見に残され，さらにプレイヤーに突かれた弱点が強化される［108］．『Forza Motorsports』シリーズ（Turn 10 Studios, 2005-）は強化学習によって［36］，『Killer Instinct』（Rare, 2013 年）はケースベーストリーゾニング（case-based reasoning）によって，プレイヤーの行動履歴から，プレイヤーの行動の癖を学習したプレイヤーの分身を作り出し，プレイヤーと対戦させる機能を持っている［109］．これはゲームのログをゲーム進行とは異なるゲームの外側に逃がした上で，機械学習を行い，再びゲームへと干渉する．

今後，機械学習のデジタルゲームへの応用が増加する中で，「メタ AI」は，機械学習の一つの応用の方向を示し，導入の可能性を広げると予想される．「メタ AI」のこれからの可能性については，第 13 章で述べる．

本節ではメタ AI がコンテンツ生成する場合，それは単なるアルゴリズムによるコンテンツ生成ではなく，ゲームの状態や，ユーザーの行動履歴を認識した上でのコンテンツ生成であり，最も遅いユーザーの行動に対するゲームからの応用と言える．メタ AI は，このように，キャラクターAI がプレイヤーに対して瞬時の応答行動をすることに対して，大きく時間のかかる応答を築くことができる．このように，メタ AI はキャラクターAI と共に，ゲームからのスケールの異なる応答をユーザーに向けて積み重ねることで，新しいユーザー体

験を作り出すことができる.

6.6 考察

本章では, メタ AI の内部構造, メタ AI が生み出す効果の分類を述べてきた. メタ AI は, 局所から大局へ至るまでゲームの様々な要素を変化させて, コンテンツをコントロールする. 動的にゲームデザインするゲームデザイナーとしての AI がメタ AI である. 大きく分けると 3 つの手法があり,「敵・味方問わずキャラクター達をコントロールすること」「ゲーム全体をコントロールすること」「コンテンツを生成する」である.

メタ AI がキャラクターAI に干渉するときには, キャラクターAI の意思決定システムへ干渉する. この干渉の仕方については, 次章で分類し述べる. メタ AI がゲーム全体をコントロールする場合には, 天候にせよ, オブジェクトにせよ, エンティティにインターフェースを設定する. これは知識表現やアフォーダンス表現の延長となる. 最後に, コンテンツを生成する場合は, プロシージャル技術とメタ AI 技術が結び付くこととなる. これについては, 第 13 章で述べることとする. どのモードにせよ, スパーシャル AI のゲーム状況・空間の認識サポートを得つつ干渉する.

メタ AI はゲームデザインをリアルタイムで行う AI であり, ユーザーを理解し, ゲームを動的に変化させて行く. ゲームデザインの知見はこれまでは個人が主観的に所持するだけであったが, これからはメタ AI が人のゲームデザインを学習し, ある程度のゲーム生成, ゲーム自動調整を行うことになると考えられる. また将来的には, ゲームプレイログからゲームデザインの学習も部分的に可能になると予想される. メタ AI は, これまで属人性が強かったゲームデザインの知見を蓄積する人工知能となる.

7. キャラクターAI

本章では，MCS-AI 動的連携モデルにおけるキャラクターAI について述べる． 第 7.1 節では，MCS-AI 動的連携モデルにおけるキャラクターAI のアーキテクチャについて述べる．第 7.2 節では，メタ AI からキャラクターAI へ渡される命令の種類について述べる．第 7.3 節では，キャラクターAI で良く使われる代表的な 7 つの意思決定アルゴリズムを説明すると同時に，メタ AI がどのようにそれぞれのアルゴリズムに作用するかを述べる．

7.1 MCS-AI 動的連携モデルにおけるキャラクターAI

本節では第 7.1 節では，MCS-AI 動的連携モデルにおけるキャラクターAI のアーキテクチャについて述べる．MCS-AI 動的連携モデルにおけるキャラクターAI は，スパーシャル AI のサポートを受けながら，メタ AI とコミュニケーションしながら行動する．そこでキャラクターAI に求められるものは，環境で自律的に活動する自律型エージェントとしての思考と，同時にゲームシナリオ上で与えられた役割の通りに演技をする知的能力である．ここでは前者を「自律型意思決定」，後者を「演技的意思決定」と呼ぶ．「自律型意思決定」と「演技的意思決定」を分けて実装するのではなく，メタ AI からの命令によって，自律型意思決定が演技的意思決定にシームレスに移行する（Fig. 7.1）．

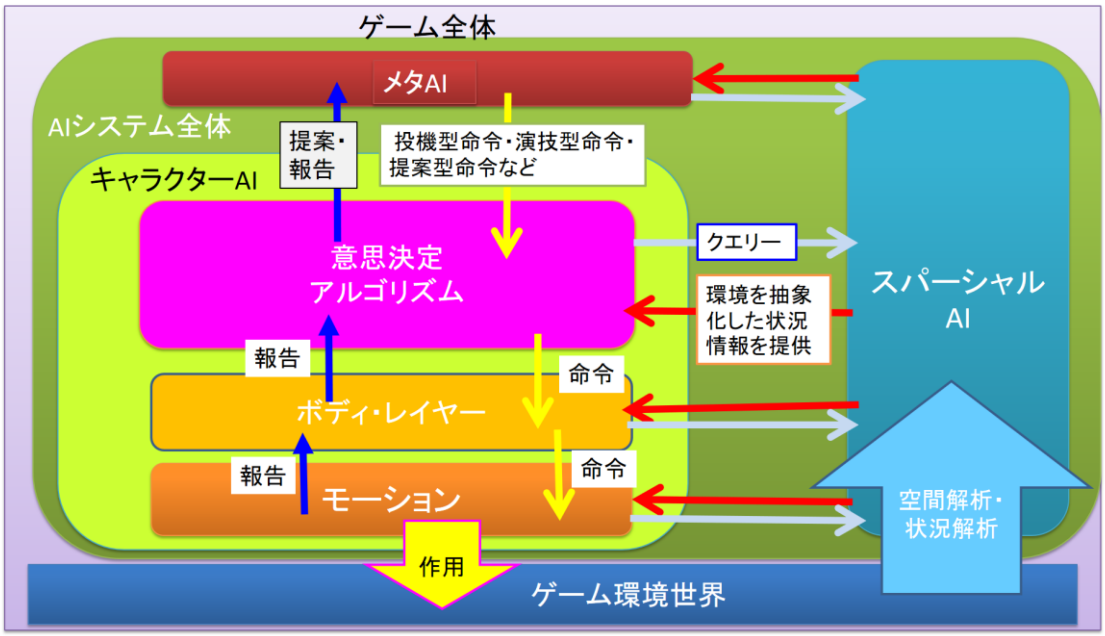


Fig. 7.1　MCS-AI 動的連携モデルにおけるキャラクターAI の 2 つの側面

メタ AI とキャラクターAI の意思決定の間の関係は，いくつかの種類に分類することができる．以下，MCS-AI 動的連携モデルにおけるキャラクターAI の意思決定とメタ AI の関係性について述べる．

7.2　キャラクターAI とメタ AI の関係性

本節では，キャラクターAI とメタ AI の関係性について述べる．メタ AI がキャラクターAI に命令を与える場合には，以下の 6 つの種類，「演技型命令」「投機型命令」「提案型命令」「目的型命令」「モード型命令」「連携命令」がある（Table 7.1）．これを順に説明する．

Table 7.1　メタ AI の命令のタイプの分類

メタ AI の命令のタイプ	メタ AI からキャラクターへ命令の内容
演技型命令	キャラクターを完全に想定した演技通りに動かす命令
投機型命令	成功・失敗にかかわらずゲームの状況を変化させるための命令
提案型命令	NPC 側から行動の提案に対して，メタ AI が許可を与える
目的型命令	目的を与えて，目的を解除するまでは遂行させ続ける
モード型命令	継続的に一つの行動スタイルを強いる
連携命令	複数の NPC に命令を与えることで連携的な実行を促す

(1) **「演技型命令」**　キャラクターを完全に想定通りに動かす命令である．たとえば，「戦闘が始まる前に，剣を振り上げて掛け声をかける」などをしてやる気があることを見せる，「街に到着した時にこれから行うミッションをプレイヤー・キャラクターの側へ来て身振りや言葉で説明する」するなど，その場に合った演技を行う．確実に行う必要があり，また失敗の可能性のない命令である．

(2) **「投機型命令」**　成功・失敗の可能性どちらが高かったとしても，実行させることでゲームの状況を変化させるための命令である．たとえば，「10 秒毎に敵基地に向けてキャラクターを数体攻撃に行かせる」，「戦闘が始まったら敵キャラクターにプレイヤーを噛みつきに行かせる」などである．その命令が成功するか，しないかではなく，プレイヤー側に圧力をかけることが目的となる．「投機型命令」は目的を与えるだけで，その過程についてはキャラクターAI 側に任される．ただ一度，実行しようとすれば，すぐに解除される．

(3) **「提案型命令」**　NPC 側から行動の提案に対して，メタ AI が許可を与える形で行われる．NPC が「プレイヤーを助けに行って良いか？」「プレイヤーを回復して良いか？」

「プレイヤーが戦っているモンスターを倒して良いか？」などに対して許可を与える．また，複数の NPC に同じ行動を同時に取らせないためにも用いられる．例えば，プレイヤーに対して複数の回復魔法がかかってしまう行為はたいへん愚かなシーンを作ってしまうので，まずメタ AI に行動を提案し，提案を受けたメタ AI は，提案した複数のキャラクターから最適なキャラクターを選択する．

(4) **「目的型命令」** メタ AI から目的を与えて，目的を解除するまでは遂行させ続ける．その実現方法についてはキャラクターAI 側に任される．

(5) **「モード型命令」** これは目的ではなく，キャラクターに継続的に一つの行動スタイルを強いる命令である．たとえば,「プレイヤーを追尾せよ」というモードは，プレイヤーがどのように移動しようと，定期的のパス検索を行い追尾し続ける．NPC へモードを設定・解除することで，ゲーム状況をコントロールする．

(6) **「連携命令」** 特定の複数の NPC に対して，一連の命令を与えることで連携的な実行を促す命令である．たとえば，プレイヤーを包囲する場合には，プレイヤーの最寄りの 5 体のキャラクターに同時にプレイヤーへ向かって歩かせる，などをさせる．また，敵・味方に同時に命令を与えることで戦闘を演出場合もある. たとえば, 敵NPC と味方NPC をプレイヤーの目の前で争わせる，プレイヤーを敵に包囲させておいて，味方 NPC に救出させる，などである．

このようにメタ AI は，さまざまな強さと方法でキャラクターに干渉する．その干渉の仕方によってキャラクターの挙動の変化の仕方が異なる．

7.3 意思決定アルゴリズム

本節ではキャラクターAI の意思決定における意思決定アルゴリズムについて述べる．自律型キャラクターAI の中心にある意思決定モジュールは, 意思決定アルゴリズムからなる．ここでは，キャラクターAI の意思決定でよく使われる 7 つの意思決定アルゴリズムについて，特にメタ AI との関わりにおいて述べる．これらは現在のゲームで最もよく使われる意思決定アルゴリズムであり, いずれか一つを用いるか，これらを複数組み合わせることで構築する [110]．意思決定モジュールは，メタ AI とコミュニケーションを行い，メタ AI と協調して，ゲーム状況を創造的に変化させていく．

意思決定アルゴリズムにおける「〜ベース」という言い方は,「〜」を意思決定の構成する要素として，選択する，という意味である．意思決定アルゴリズムは，大きく反射型と非反射型に分類される（Fig. 7.2，Table 7.2）．反射型とは，自分を含む世界の変化に対して行

動を生成することを言い，「ルールベース」「ステートベース」「ビヘイビアベース」がこれに当たる．

Table 7.2　キャラクターAI で用いられる意思決定アルゴリズム

アルゴリズム名	反射型/非反射型
ルールベース　（Rule-based）	反射型
ステートベース　（State-based ）　[111] [112]	反射型
ビヘイビアベース　（Behavior-based）　[82]	反射型
ユーティリティ・ベース（Utility-based）　[84]	非反射型
ゴールベース　（Goal-based）　[80] [26] [113]	非反射型
タスクベース　（Task-based）　[114] [115]	非反射型
シミュレーション・ベース（Simulation-based）　[74] [75]	非反射型

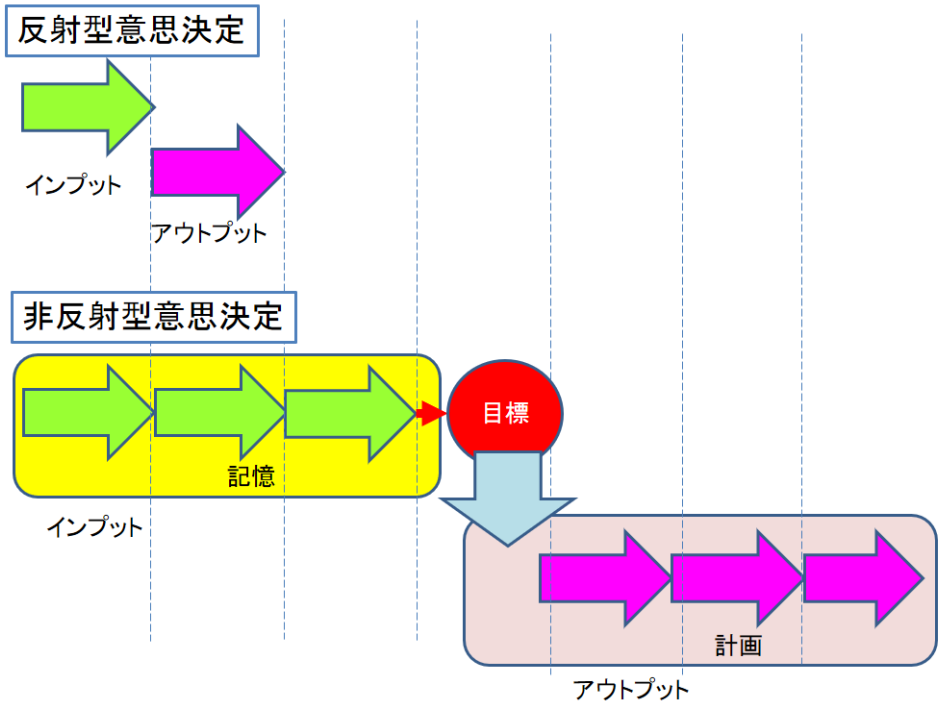


Fig. 7.2　反射型意思決定と非反射型意思決定

一方で，反射型でない意思決定法を非反射型という．過去の記憶を持ち，そこから未来に対して，目標や仕事，評価や想像を作って行くことで意思決定を行う手法である．「ゴールベース」「タスクベース」「シミュレーション・ベース」「ユーティリティ・ベース」がこれに当たる．世界が変化するのを待つのではなくて，たとえば目的を持ってそれに向かって行動するゴールベース，世界の変化を予想して行動するシミュレーション・ベースなどである．

反射型は何より環境の変化に順応しやすい，という特徴があり，非反射型はキャラクター側が主導権を持って行動を構築する始点を持っている．

メタ AI との関係は，すべての意思決定アルゴリズムで可能であるが，前述したように，MCS-AI 動的連携モデルの特徴の一つは，それぞれの AI が独立に運動しながら協調するところにある．そこで，「メタ AI からキャラクターAI に命令する」場合も，メタ AI からキャラクターAI に単純に命令が伝えられる，だけではなく，メタ AI がキャラクターAI の意思決定に関して動的に干渉する．それぞれの意思決定アルゴリズムに関するメタ AI からの命令の伝え方，介入の仕方はそれぞれ異なる（Table 7.3）．以下，それぞれの意思決定アルゴリズムを解説しながら，メタ AI との関係の仕方を述べる．

Table 7.3　メタ AI から意思決定への動的な干渉の仕方

意思決定アルゴリズム	メタ AI からの動的な干渉の仕方	メタ AI との関わり方
ルールベース	(1) ルールを制限する (2) ルールを追加する (3) 優先度を変える	投機型命令
ステートベース	(1) ステートを指定する (2) ステートマシンを入れ替える	投機型命令
ビヘイビアベース	(1) ビヘイビアを制限する (2) ビヘイビアツリーを変形する (3) 協調ゲートを用いる	モード型命令
ユーティリティ・ベース	(1) 選択を指定 (2) ユーティリティ関数を変化	モード型命令
ゴールベース	最上位のゴールを指定する	目的型命令
タスクベース	最上位のタスクを指定する	目的型命令
シミュレーション・ベース	シミュレーションの深さ（負荷）を指定する	投機型命令

7.3.1　ルールベース AI（Rule-based AI）

本項では，デジタルゲームにおけるルールベース AI の基本原理と，MCS-AI 動的連携モデルにおけるメタ AI との関係について述べる．

基本原理

「ルールベース」の意思決定の単位は「ルール」となる．通常はルールを複数持ち，その状況に最も適したルールを選択することで行動を行う．ここで言うルールとは

IF（前置宣言文） THEN（後置宣言文）

という形式を言う．ルールベースの本質は，自分を含む世界の状態の変化に対する行動を，ルールとして記述していくことにある．これは，世界の変化が多様であり，それに対する行動が一つ一つ違う場合に最も効果を発揮する．メタ AI からは使用するルール群を指定すること，或いは逆にいくつかのルールを禁止することで制御される．

一般的なルールベースのアーキテクチャは，上記はルールを選択する「ルールセレクタ」モジュールを用意して，どのルールを選択するかを選ぶ構造である．自由なルールの行使方法を「ルールセレクタ」に記述する［116］（Fig. 7.3）．優先度を付けて選択する場合もあれば，発火したルールから乱数で選ぶ場合もあり，またより複雑な思考を書く場合もある．たとえば，発火した複数のルールを順番に実行する，一度選択したルールは当分の間使わないことにするなどシークエンスにルールを実行する，などである．

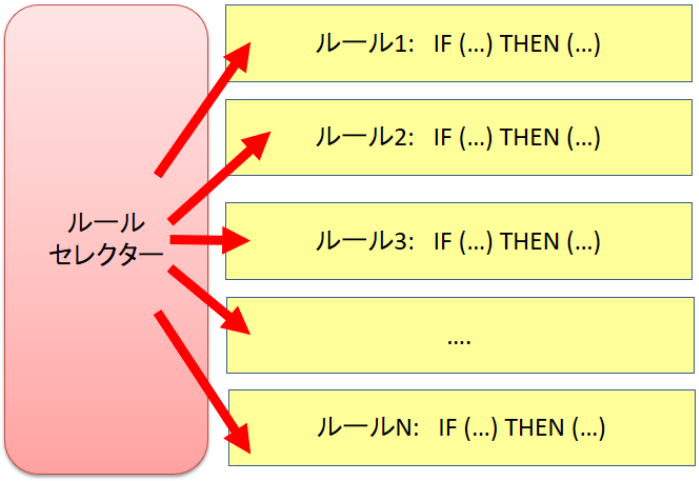


Fig. 7.3　ルールセレクタ［116］

RPG（Role Playing Game）においてはユーザー自身に仲間キャラクターの知能を「ルール群とその順番を付ける」ことで作成する機能が与えられることもある．『Dragon Age』（Bioware, 2009 年）などである．例えば「体力が半分をきったら回復魔法を唱える」「一番 HP の高い敵を攻撃する」などである．『The Sims 3』（Electronic Arts, 2009 年）では NPC の癖や特徴を出すために，各キャラクターにプロダクション・ルールを蓄積して行く．例えば「夕方になったらＴＶを見る」「朝起きたらジョギングをする」などである［117］.

MCS-AI 動的連携モデルにおける関係性

メタ AI は，ルールベース AI に対して，以下の方法で命令を行う．

(1) いくつかのルールをマスク（実行禁止）にする．

(2) いくつかのルールを優先度付きで加える．
(3) ルールセレクタの優先度を変化させる

以下，それぞれについて述べる．

(1) いくつかのルールを禁止にすることで，キャラクター，あるいはキャラクターの集団を特徴づけたい場合に用いる．たとえば，防御をするルールをすべて禁止にしてしまう，という具合である．

(2) 新しいルールを動的に加えることで，一般的には必要ないが，特殊な状況や場所でキャラクターの行動に特徴付けを行いたい場合に用いる．たとえば，足元を常に狙われるような場所では「何もない場合は，ジャンプする」などのルールを加えておく．或いは，「もし高い場所で立てる場所があれば，そこへ移動する」などである．

(3) 各ルールの優先度をつけることで，キャラクター，或いはキャラクターの集団の個性を変化させる．たとえば，攻撃的な集団に変化させる，或いは，個性をあえてばらつかせる，などの応用がある．

このように，ルールベース AI は，明確に表現されたルールを要素としているために，はっきりとした個性付けを，メタ AI からコントロールしやすい，という特徴がある．メタ AI からルールベースへの干渉は，短期的な反射的行動を指定するため投機型命令である．

7.3.2 ステートベース AI（State-based AI）

本項ではデジタルゲームにおけるステートベース AI の基本原理の説明と，MCS-AI 動的連携モデルにおけるメタ AI との関係について述べる．

基本原理

ステート（状態）はキャラクターの状態を指す．あるキャラクターが一つの状態にある場合には，キャラクターは状態で定義された行動を行う．「ステート」を指定することで，実行内容が決まる AI を「ステートベース AI」と言う．また状態から状態へ移る条件を「遷移条件」と言い，複数の状態を遷移状態で結んだシステムを「ステートマシン」と言う［118］．キャラクターは複数のステートを持ち，特定の条件で状態間を遷移する．例えば「歩く」という状態では歩き「攻撃」という状態では指定されたように攻撃する．

ステートマシンは，90 年代から現在までキャラクターAI で最もよく使われて来た技術である．『DOOM』（id Software，1993 年）［119］や『UNCHARTED 2』（Naughty Dog，2009

年）［Gregory 09b］など数百以上のタイトルで使用されている．また大規模なゲームでは一つのステートの中にステートマシンがあるような，階層型ステートマシンが使用されている（Fig. 7.4）．

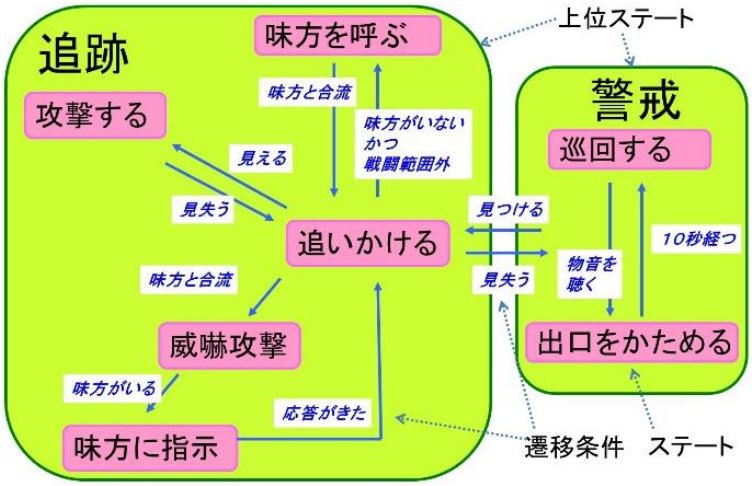


Fig. 7.4　階層型ステートマシン

階層型ステートマシンは，大きなデータとなるが，複雑なゲーム世界をモデル化するためには，ステートマシンの中にさらにステートマシンを含んだ 2 階層，3 階層の階層型ステートマシンが構築する必要がある．ステートマシンは堅実な制御法であり，現在でも最も汎用性のある方法の一つである．90 年代後半から 2010 年頃までは最もよく使われた方法であり，現在でも広く用いられている．

MCS-AI 動的連携モデルにおける関係性

メタ AI がキャラクターAI の意思決定のステートマシンに命令を与える方法は，以下の手法がある．

(1) 特定のステートを指定する．
(2) 複数の用意されたステートマシンのいずれかを指定する

などである．(1) は直接，実行させたいステートを指定する直接的な手法である．たとえば，その瞬間に攻撃をさせたい場合「攻撃」ステートを指定する．ステートマシンは遷移条件をたどって行くためには，順次，遷移条件を揃える必要があり，ステートの遷移を操作することは難しいため，そのような直接なアプローチが取られる．(2)は，ステートマシンを幾つか用意しておく，という方法である．たとえば，プレイヤーが弱ってきたときに使用するステートマシン，プレイヤーのスキルが高いときに使用するステートマシンを用意し，メタ AI が状況によって入れ替える．

ステートマシンはネットワークを形成するため，メタ AI によって一時的にステートを変化させることができても，その後は遷移条件によってステートが継続的に変化して行くため，継続的な影響を及ぼすのが難しい．ルールベースと同じく短期的な継続性のない命令であるため，投機型命令である．

7.3.3 ビヘイビアベース AI（Behavior-based AI）

本項ではデジタルゲームにおけるビヘビアベース AI の基本原理の説明と，MCS-AI 動的連携モデルにおけるメタ AI との関係について述べる．

基本原理

ビヘイビアとはキャラクターの振る舞いのことである．ビヘイビアベースとは，キャラクターの動作を基本に思考を組み立てることである．具体的な実装形式ではビヘイビアツリー（BT， Behavior Tree）と呼ばれる手法があり，90 年代，00 年代，そして現在でも使用されているステートマシンに代わって，2010 年代以降では最もよく使われる手法である．メタ AI からは直接，中間ノードを指定することで制御される．或いは逆に，いくつかのビヘイビアを実行不可とすることでコントロールされる．この制限によって，キャラクターの場所ごとの行動に特徴をつけることができる．或いは，ダイナミックに末端のノードにビヘイビアが一時的に加える場合もある．

ビヘイビアツリーは『Halo2』（Bungie，2004 年）において Damian Isla 氏によって発案された手法であり，ステートマシンから環構造を排して，ルートから末端のビヘイビアに向かって行動を選択して行く階層型ツリーの構造である（Fig. 7.5） [82]．末端のノードのみがビヘイビアであり，中間ノードはそれ以下のノードを含む中間階層ノードである．ルートから末端へ向かって選択がくり返され，末端へたどり着いて初めてビヘイビアが実行される．末端で実行が終わるとルートに戻り，再び末端へ向かった選択がくり返される [110]．

ビヘイビアツリー

ビヘイビアツリーの構造は，層状の階層構造からなる．ルートの方を上位，末端の方は下位という．上位から下位へ向かって処理の流れがある．一つの層には複数のノードが含まれており，それぞれのノードが下位の層を持つ．ただし，末端はそれ以上の層を持たない．末端のノードはビヘイビア・ノードと呼ばれ，キャラクターのビヘイビアを指定するためのノードである．この末端のノードからボディ・レイヤーに接続され，アニメーション・データを指定する．

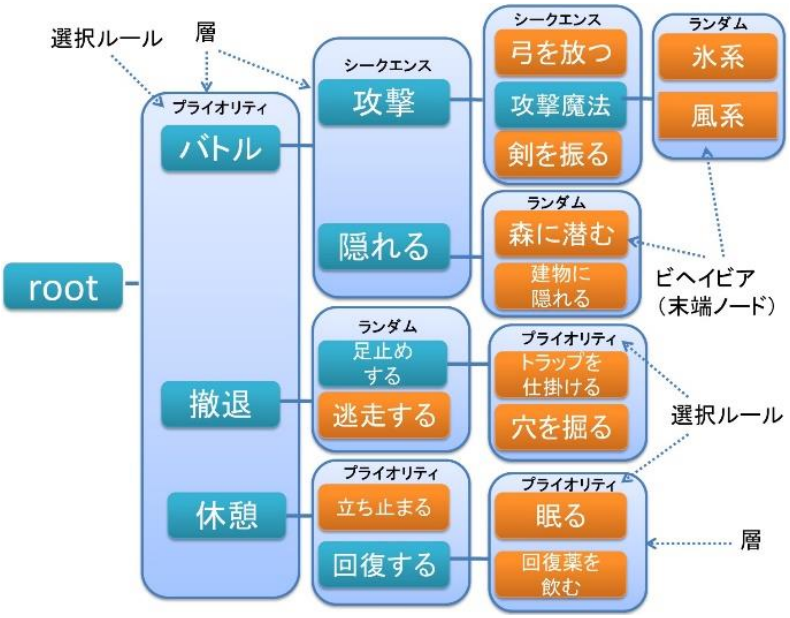


Fig. 7.5　ビヘイビアツリー

各層に一つの選択ルール（Selection Rule）が指定されている．この選択ルールに従って，層の中で実行するべき複数のノードとその順番が決定される．グラフ上で，それぞれの層の中では，優先度の高いノードから順番に並べて記述する，という決まりがある．この選択ルールが実行される前に，各ノードが現在，実行かどうかの判定が入り，実行可能でなければ候補から外れることになる．たとえば，「魔法」はそれを放つのに必要な「マジック・ポイント」が足りない場合は候補から外れることになる．

実行時は，ルートから末端へ向かって選択がくり返され，末端へたどり着いて初めてビヘイビアが実行される．末端で実行が終わるとルートに戻り，再び末端へ向かった選択がくり返される．各層（ノードの集合）は子ノード同士が競合するモデルであり，現在実行可能なノードの中で「シークエンス・ルール」はあらかじめ定められた順番で実行し，「プライオリティ・ルール」はあらかじめ定められた優先度に従って現在実行可能なノードの中で最も高い優先度を持つ実行可能なノードを実行し，「ランダムルール」はランダムにノードを実行する．

例えば，Fig. 7.5 のビヘイビアツリーでは，最初の層は「プライオリティ」が選択ルールであり，現在敵がいるならば「バトル」が選択される．もし敵がいなければ「バトル」「防御」は実行不能なので「休憩」が選択される．今「バトル」が選択されたとする．次のバトルが持つ層が実行される．この層は「シークエンス」が選択ルールであるから，「攻撃」「隠れる」が順番に実行される．次に「攻撃」ノードの層が選択ルール「シークエンス」に従って実行され，「弓を放つ」「攻撃魔法」「剣を振る」が順番に実行される．このうち「攻撃魔法」は層を持つので，「ランダム」ルールに従って使う魔法が選択される．次に「隠れる」ノードに処理が戻り，このノードが持つ層が「ランダムルール」に従って実行され，たとえ

ば「森に潜む」ノードが実行される．そしてルートへ処理が戻る．

以上のようにビヘイビアツリーは，状況に応じてどのようにビヘイビアを繋げていく意思決定アルゴリズムである．ビヘイビアツリーは制限の多い手法であるが，ツール上でゲームデザイナーだけで組み上げることができ，小規模から大規模にスケールすることが可能であり拡張性に優れている．またデバッグが比較的容易という利点があり，キャラクターにおけるデフォルトな方法になりつつある．『Spore』(Maxis，2008 年)，『CRYSIS』(Crytek，2012 年）など大型タイトルから小型タイトルまで，ゲームのスケールを問わず数百以上のタイトルで使用されている［105］[120]．

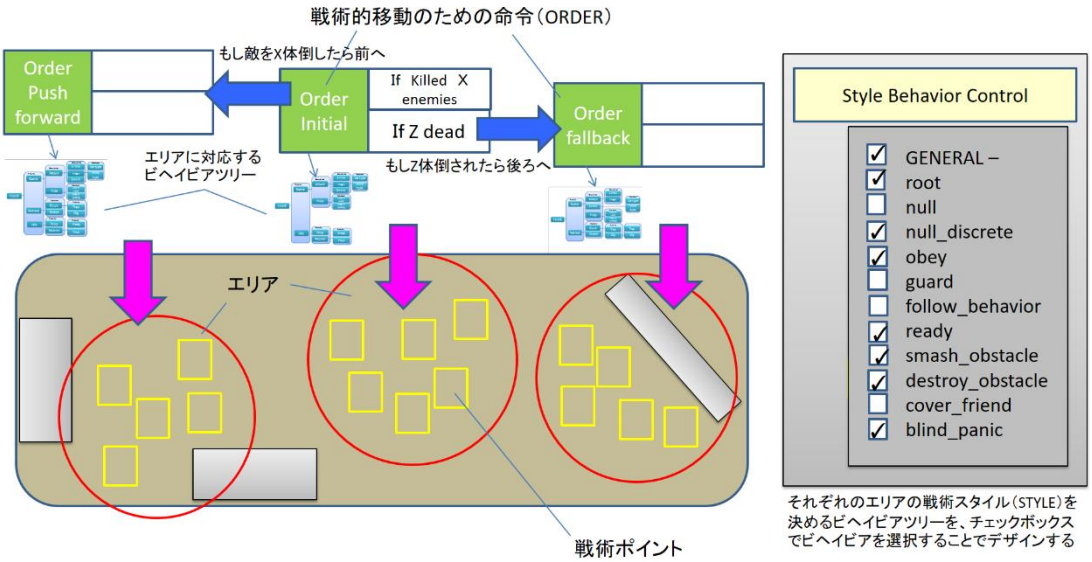


Fig. 7.6　『Halo2』におけるオーダー＆スタイル［121］

MCS-AI 動的連携モデルにおける関係性

メタ AI がキャラクターAI の意思決定のビヘイビアツリーに命令を与える方法は，以下の手法がある．

(1) いくつか要素をマスク（実行禁止）にする．
(2) ビヘイビアツリーの形を動的に変化させる
(3) カウンターゲートを使う．

(1) はマスターとなるビヘイビアツリーを作っておき，キャラクターや場所ごとに，いくつかの要素にマスクをかけることで，キャラクターごと，場所ごとのビヘイビアに特徴をつける．たとえば，『Halo2』では，マスターとなるビヘイビアツリーから，キャラクターごとのマスクをゲームデザイナーが付けて行くことでキャラクターの種別ごとの特徴をつける，という手法を取っている［82]．また，同ゲームでは，戦闘マップを「前衛」

「中間」「後衛」エリアに分けて，その場所ごとにビヘイビアツリーを設定する．前衛では，攻撃的になるようマスタービヘイビアツリーの防衛的行動にマスクをかけ，後衛では防御的な行動を取らせるようにマスタービヘイビアツリーの攻撃的行動にマスクをかける，などである．また前衛，中間，後衛のキャラクター群の移動はステートマシンによって行われる．たとえば，「味方が X 体以上戦闘不能にされると，後衛に下がる」などである．このステートマシンの機能を「オーダー」，各エリアのビヘイビアツリーを「スタイル」と呼び，全体を「オーダー＆スタイル」と呼んでいる（Fig. 7.6）．この場合，ステートマシンがメタ AI として機能している［121］.

(2) は動的にビヘイビアツリーを変える手法である．特定の場所や状況で，末端に動的にビヘイビアを加える，或いは，ビヘイビアの順番を変化させることで，その場所だけに特徴的な行動をキャラクターに取らせる．どのように変化させるかは，メタ AI がゲーム状況を認識し，それに適応するビヘイビアツリーへと変化させる．たとえば，『Driver San Francisco』（Ubisoft，2011 年）では，各領域の知識に基づいてビヘイビアツリーの末端の順番を動的に変化させる「Hinted-execution Behavior Trees」(HeBTs)システムが採用されている．この場合，ヒントを与える役割をメタ AI が果たしている［122］.

(3) カウンターゲートとは，ビヘイビアツリーにおいて，通過するたびに，カウンターが上がって行き，あらかじめ設定された上限を過ぎると，アクセスを禁止するゲートである，たとえば，「逃げる」というビヘイビアの前のカウンターゲートの上限を 2 としておくと，同じ構造のビヘイビアツリーを使う NPC の集団がいた時，２体が「逃げる」を選択した時点で，残りの個体は「逃げる」を選択できなくなりそれ以降は「逃げる」というビヘイビアを選択できなくなるので，の集団から 2 個の NPC のみが実際に「逃げる」ことになる．メタ AI は，カウンターゲートの上限値を変化させる，あるいは，リセットすることで，NPC の集団の挙動をコントロールする．この手法は『Crysis 2』（Crytek, 2011 年），『Spec Ops: The Line』（YAGER, 2012 年）において採用されている［120］［123］.

このようにビヘイビアベース AI に対するメタ AI の命令は，具体的にビヘイビアツリーの構造を巧みに用いたものが多い．ビヘイビアツリーは繰り返しルートから実行されるために，そのサイクルの中に巧みにメタ AI の意向を入り込ませておくため，変化はほぼ恒常的なものでありモード型命令に分類される．

7.3.4 ユーティリティ・ベース AI（Utility-based AI）

本項ではデジタルゲームにおけるユーティリティ・ベース AI の基本原理の説明と，MCS-

AI 動的連携モデルにおけるメタ AI との関係について述べる．

基本原理

ユーティリティとは効用のことである．ユーティリティ・ベースとは，自分が選択し得る行動が，どれぐらい効用があるかを推定し意思決定する手法である．ユーティリティ・ベースはキャラクターに用いる意思決定の中で最も古い手法であり，80 年代から現在に至るまで継続的に用いられている．行動選択のみならず，ゴール選択，武器選択する，魔法選択，経路選択などにも用いられる［124］．囲碁や将棋の AI では選択した手で局面がどれぐらい有利になるか，効用の計算を行う．

キャラクターの「攻撃の効用」とは，敵に対した与えることができたダメージ，味方に対してかけた「回復魔法の効用」とは回復した度合である．例えば「戦う」「防御する」「魔法を唱える」「回復剤を飲む」という 4 つの行為が可能である場合に，それぞれの行動の効用を評価式から評価し最高点を得た行動を選択する，これが効用による意思決定である．キャラクターによって評価式の形や式の定数を変化させることで意思決定に個性を持たせることができる［84］.

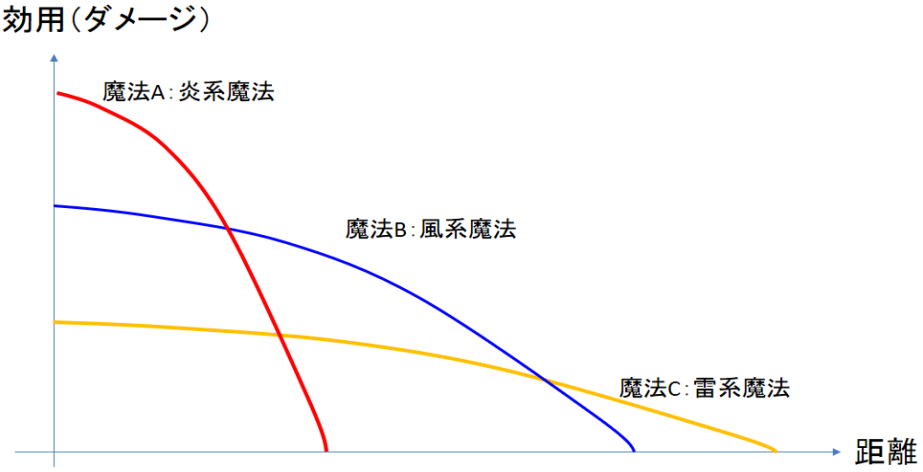


Fig. 7.7　魔法の距離-効用曲線

『The Sims』（Maxis，2000 年）は効用の概念を最大限に用いたタイトルである．キャラクターは 8 個の内部生理パラメータを持ち（「人と話したい」「眠りたい」など），各パラメータの効用曲線が定義されている．効用曲線とは，そのパラメータの数値に対する効用のグラフであり，これは直線ではなく，パラメータが上がるほど効用度は上がりにくくなる．これを限界効用逓減の法則と呼ばれる．『The Sims』のキャラクター達はこの法則に従う［103］.

ユーティリティ・ベースはまたゲーム開発が続く間に増えて行く要素に対する選択システムを構築する場合に，拡張性が高く有用である．例えば，あるキャラクターが魔法を 3 種類打てるとする．どの魔法を撃つかをユーティリティ・ベースで自動的に切り替えたい，と

する．魔法の威力は距離によって威力が異なり，それぞれの魔法の距離-効用曲線のデータを用いて自動的に切り替えることができる．今，魔法Aの効用曲線は近距離でピークがある，つまり近距離用の火炎系魔法であり，魔法Bは中距離であり効用曲線が中距離にピークがある風系魔法であり，魔法Cは遠距離で威力を発揮する雷系魔法とする．それぞれの効用曲線を使って，敵までの距離に応じた最も効用の高い魔法の選択を行う（Fig. 7.7）．新しい魔法Dを覚えるときには，距離-効用曲線を新しく定義するだけでプログラムを書き直す必要も，他の魔法との調整も必要ない．つまり拡張性の高い実装になっている．ルールベースであれば，すべてを書き直す必要がある．

MCS-AI 動的連携モデルにおける関係性

メタ AI がキャラクターAI のユーティリティ・ベースの意思決定に命令を与える方法は，ユーティリティ・ベースの意思決定が対象とする選択肢について，推薦する選択肢を指定することである．それによって，もう一度，指定した選択肢の評価を促すこと，或いは，選択肢のユーティリティ関数にプラス，あるいは，マイナスのバイアスを与えることで，特定の選択肢を選びやすく，あるいは，選びにくくすることである．たとえば，前半では魔法攻撃より剣攻撃を優先させたい場合は剣攻撃を指定し，後半では逆に魔法攻撃を指定することで，ユーティリティ関数にいくらかのプラスを与える．この手法は，第 9.2 節で言及する『クロムハウンズ』において用いられている．ユーティリティ関数による変化は解除しない限り恒常的なものであり，モード型命令に分類される．

7.3.5 ゴールベース AI（Goal-based AI）

本項ではデジタルゲームにおけるゴールベース AI の基本原理の説明と，MCS-AI 動的連携モデルにおけるメタ AI との関係について述べる．

基本原理

ゴールベースは，まず目標（ゴール）を決めて，その後にいかに実現するか，を考える非反射型意思決定アルゴリズムである．ゴールベース・プランニングと言えば，数分や数時間，数日と言った計画を立てるための意思決定の方法として使用されて来たが，デジタルゲームの場合にはこれをリアルタイム，1/30 秒，1/60 秒から 1 秒以内のその次の 1 秒〜数秒の行動プランを作ることが多く，「リアルタイムゴール指向プランニング」と呼ばれる [64]．メタ AI からは直接ゴールを指定することで制御でき，またメタ AI からの意図が明確に伝わるため，メタ AI とキャラクターの間のインターフェースとしては効率の良いアルゴリズムである．

デジタルゲームで使用されるゴールベースの手法は二つ存在する．一つは階層型ゴール指向型プランニング（Hierarchical Goal-Oriented Planning）であり，主に戦略から戦術，

行動に階層的に行動を構築する時に用いる（Fig. 7.8）．もう一つのゴール指向の方法は「ゴール指向型アクションプランニング」(Goal-Oriented Action Planning，GOAP）と呼ばれ，ゴールに向かってリアルタイムにキャラクター・アクションを生成する（Fig. 7.9）．

この 2 つは目標に向かって単位となる行動を組み合わせる異なる手法である．前者は複雑な長期的かつ抽象的なプランを前者のアルゴリズムはアクションレベル，つまり短期的かつ物理行動的なプランを作る場合に作る場合に用いられる．

ゴールを持つということは，未来という観念を持つことでもある．これによって反射的な行動から解放され，未来に向かって，未来のゴールから，自分の行動を未来に向かって構築する．この過程はプランニングと呼ばれ，一般に未来に向かって行動を展開して考えることをフォワードプランニング，逆にゴールから現在に向かって考えることをバックワードプランニングと呼ばれる［125］.

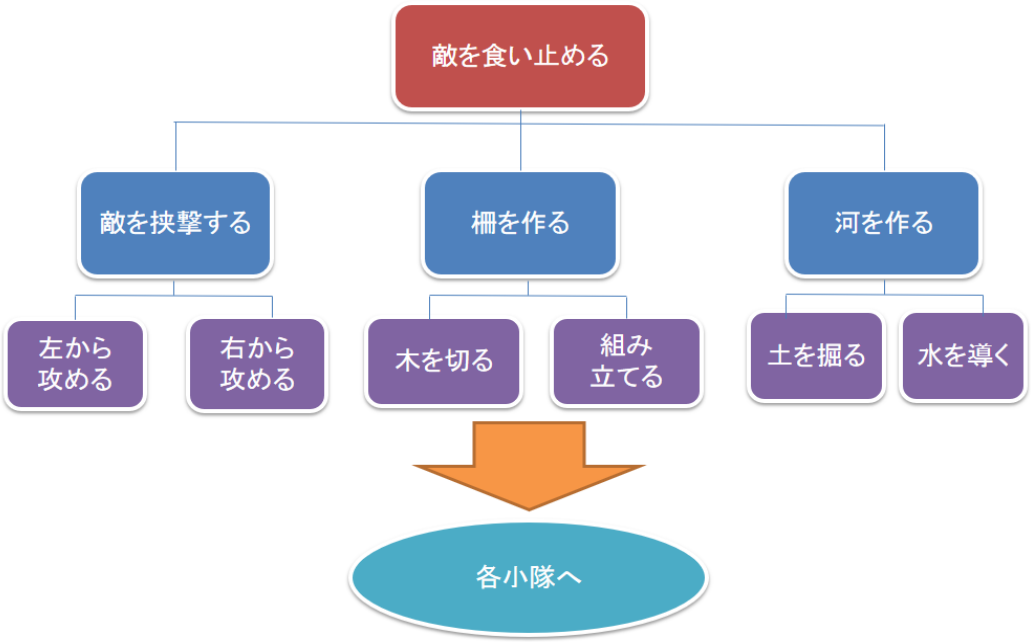


Fig. 7.8　階層型ゴール指向プランニング

階層型ゴール指向プランニングは，まず最終的な大きなゴールがあり，スクリプトなどによってより小さなゴールへと分解をくり返し，最終的に単純なコマンドまで分解する方法である[Buckland 04a].『クロムハウンズ』の NPC の意思決定で用いられ，NPC の性能・状態・ゲーム状況によってゴールの分解のされ方を変化せることで，戦況に応じた行動が生成された［19］[20] [75] [80] [126]．また「サカつく DS」(SEGA，2008 年）では，ゲーム全体の組み立てに用いられている［127] [128].

GOAP はゴールと初期条件の間をアクションの連鎖でつないで行く方法である．『F.E.A.R.』では，STRIPS（Stanford Research Institute Problem Solver）［125]を基本にした方法で，各アクションを「前提」「実際の行為」「効果」の 3 つを表現で記述する．「前

提」「効果」はシンボルで記述され，キャラクターのアクションをその形式で多数用意しておく（アクション・プール）.「ゴール」「前提」と同じ（或いはそれを論理的に含む）「効果」を持つアクションをシンボル・マッチングで探索し後ろ向きに連鎖させる．このようにして一連のアクションプランが作成される.「F.E.A.R.」（Monolith soft，2004 年）において Jeff Orkin によって開発された [63] [64] [113].

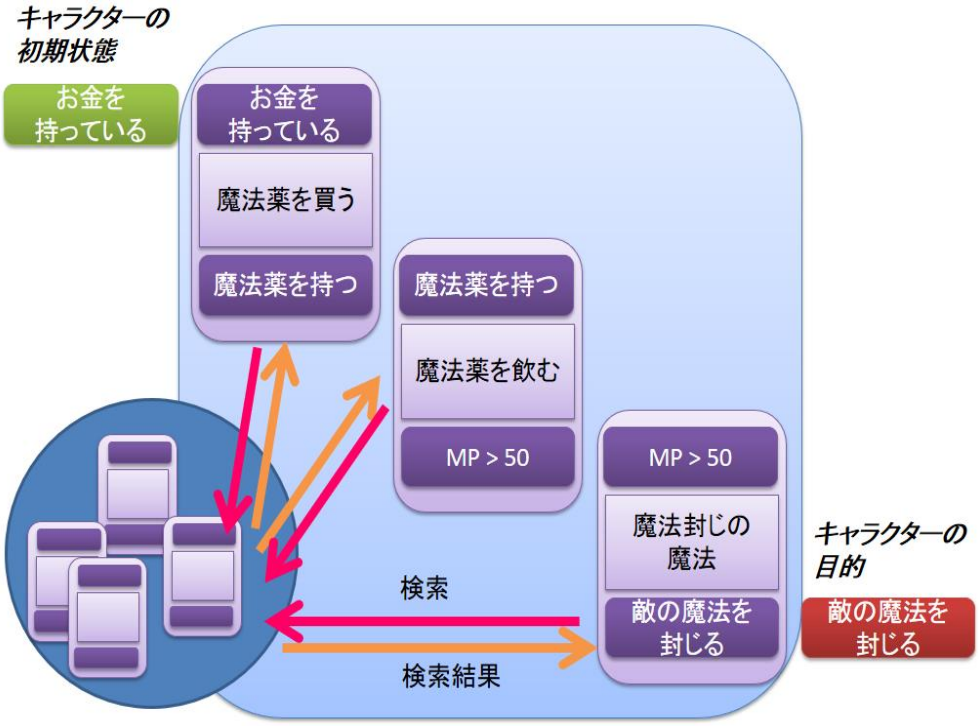


Fig. 7.9　キャラクターのためのゴール指向アクションプランニング

実際に「ゴール指向アクションプラン二ング」（Fig. 7.9）を行いながら解説する．まずゴールはとても強い魔法を放つボスモンスターがダンジョンの奥にいるので「敵の魔法を封じたい」．そこでプレイヤーであれば，魔法薬を買っておいて，戦闘の前に飲んで，戦闘に挑んで，最初に魔法を封じる魔法をかける，などと考える．ゴールは「敵の魔法を封じる」であり，多数のアクションが含まれる「アクション・プール」の中から「敵の魔法を封じる」を効果として持つアクションを検索すると「魔法封じの魔法」というアクションを見つける．今度は「敵の魔法を封印」の前提条件である「マジック・ポイントが５０以上ある」に注目する．もしキャラクターがマジック・ポイントを現時点で５０以上持っているのであれば，ここでプランニングは終了で魔法封じの魔法を発動して終了である．しかし，今，そうでなくマジック・ポイントが足りないとする．「マジック・ポイントが５０以上ある」を効果として持つアクションをアクション・プールから検索すると「魔法薬を飲む」というアクションが見つかる．前提条件は「魔法薬を持っている」である．さらに，この「魔法薬を持っている」を効果として持つアクションを検索すると「魔法薬を買う」と「モンスターB を倒す」

が見つかる.「もしお金を持っている」場合は「魔法薬を買う」,持っていなければ「モンスターB を倒す」を採用する.このようにプレイヤーの現在の状態とゴールとを前提条件と同じ効果を持つアクションを探してつないで行く方法を「チェイニング」(連鎖)と言い,チェイニングによって一連の行動を作り出す.「連鎖プランニング」とも呼ばれる.

MCS-AI 動的連携モデルにおける関係性

メタ AI がキャラクターAI のゴールベースの意思決定に命令を与える方法は明確であり,メタ AI からキャラクターAI へゴールを指定することである.目的型命令に分類される.これについては,詳細を,第 9.2 章で述べる『クロムハウンズ』では階層型ゴール指向プランニングが採用されている.

7.3.6 タスクベースAI(Task-based AI)

本項ではデジタルゲームにおけるタスクベース AI の基本原理の説明と,MCS-AI 動的連携モデルにおけるメタ AI との関係について述べる.

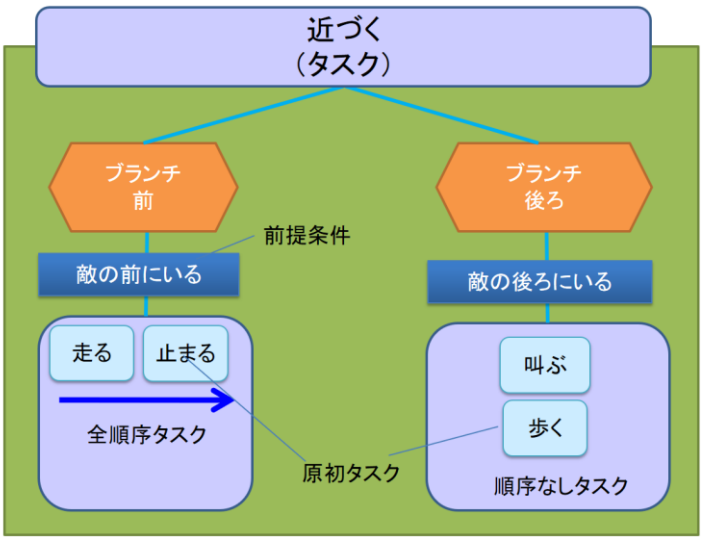


Fig. 7.10 階層型タスクネットワークにおけるメソッドの一例

基本原理

タスクベース AI は,行動の単位「タスク」を定めて,その単位によって行動を組み上げる意思決定法であり,タスクはそれ自身の定義とタスク間の結合ルールが定められている.タスクは,人工知能が持つある問題領域の中で発生する「実行するべきこと」を意味する.タスクは,問題領域内の対象に対する操作を組み合わせで表現される.一般に「タスク」をもとに意思決定を作るときに,背景となる問題領域のことを「ドメイン」(domain)と言う.メタ AI からは,最上位のタスクを指定することでコントロールできる.これもゴール指向

と並んで，最も効果的にコントロールできる意思決定アルゴリズムの一つである．

たとえば「敵をやっつけるための問題」「罠を作るための問題」「部隊を補給するための問題」などである．このよう「タスク」はより小さなタスクへ分解されて行く．それ以上分解できない最も単純なタスクのことを「原初タスク」（primitive task）と言う．たとえば「位置 A から B へ移動する」「魔法を撃つ」「回復薬を飲む」などである．このようにタスクベースは，あるドメイン内の問題をタスクとして表現し，最終的に「原初タスク」のシークエンスに分解される．

階層型タスクネットワーク（HTN，Hierarchical Task Network）［114］は，ある目標があり，目標を達成するタスクに分解される．分解の仕方をメソッドという（Fig. 7.10）．タスクはプリミティブな単純タスクになるまでメソッドによって階層的に分解される．タスク群には，タスクを実行する順序が定義されており，その制約を守れば順番を問わないため，結果として階層型タスクネットワークは一方向のタスク・ネットワークグラフを産み出す［110］（Fig. 7.11）．タスクの順序規則には 3 つの場合があり，すべてのタスクの順序が決まっている「全順序」（total order），部分的なタスクの順番が決まっている「半順序」（partial order task），「順序なし」（non-order task）がある［129］．

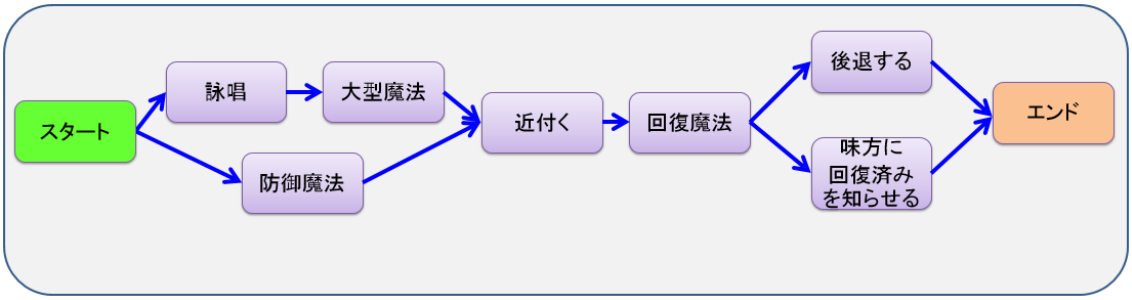


Fig. 7.11　生成されたタスクネットワーク

それぞれのタスクの分解の仕方であるメソッドは独立に定義しておき，メソッドをリアルタイムに適用として分解して行くことで，行動プランをゲーム内で生成する．これは，独立に分解の仕方を定義しておけば，開発の柔軟性を保てるからでもあり，どのメソッドを適用するかはゲーム内から指定することができるという動的に計画を立てられるメリットが存在する．この手法を初めてデジタルゲームで適用したのが『Killzone 2』（Guerrilla Games, 2009 年）である．メソッドを積み重ねることで（Fig. 7.12），数体のキャラクターに対して，わずか数フレームの内に 500 個に及ぶタスクネットワークを作り出す［65］．これはスクリプティッド AI と比較すると，数百倍の開発効率の良さである．

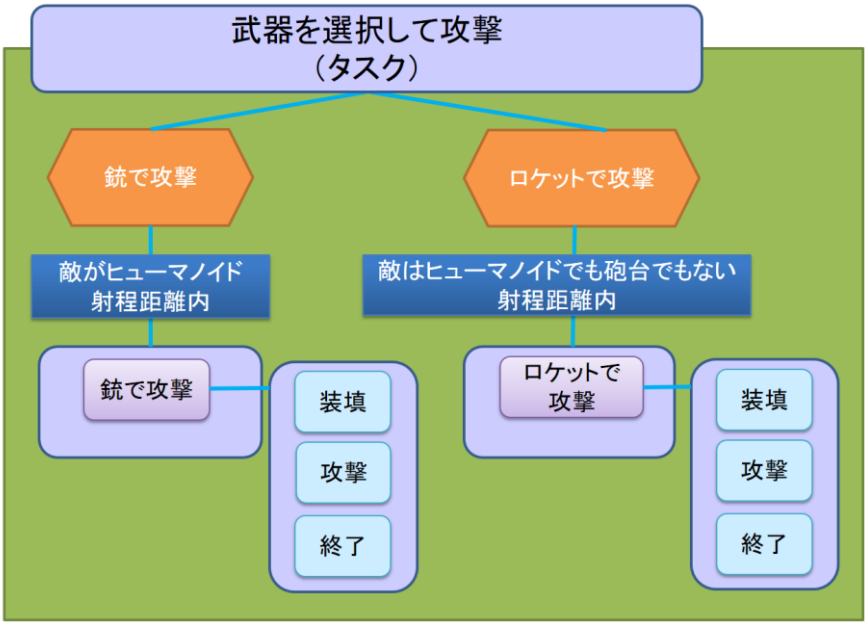


Fig. 7.12　『Killzone 2』におけるメンバーAI の階層型タスクネットワークのメソッド

MCS-AI 動的連携モデルにおける関係性

メタ AI がキャラクターAI のタスクベースの意思決定に命令を与える方法は，メタ AI からキャラクターAI へ，最上位のタスクを指定することである．目的型命令に分類される．タスクベース AI は，メタ AI からキャラクターAI へ明確に意図を伝えることができる．たとえば，上記の説明で用いた『Killzone 2』ではチーム AI から各メンバーに向かって，タスクを指定することでチームをコントロールしている［65］.

7.3.7　シミュレーション・ベース AI（Simulation-based AI）

本項ではデジタルゲームにおけるシミュレーション・ベース AI の基本原理の説明と，MCS-AI 動的連携モデルにおけるメタ AI との関係について述べる．

基本原理

意思決定を行おうとしている問題に対して，それぞれ手段を選んだ場合をシミュレーションし，最も良い結果を得たシミュレーションで選択した手段を選択する意思決定手法を「シミュレーション・ベース AI」と言う．たとえば，複雑な地形で自由度の高い運動性能を持つキャラクターを移動させる場合，そのモーションプランの作成が単純なロジックで解けない場合が多い．そこでキャラクターの運動を加速ベクトルの大きさとタイミングをある程度ランダムに組み合わせてシミュレーションし，その中で最良の軌道を見つける

シミュレーション・ベースには 2 つの手法がある．1 つは実際に現実のモデルを使って

シミュレーションを行う，つまり実際にゲームを動かしてみるという考え方である．もう 1 つは，AI が頭の中に 1 つのモデルを持ち簡易的なモデル上でシミュレーションする，という考え方である．前者はゲームとしての衝突モデルやキャラクター動作モデルを，後者はストラテジーゲームや RPG など戦略を組み立てる場合に用いる．実際に詳細な地形を使用する必要はなく，簡易的な当たりモデルの中，或いはナビゲーション・データの中のシミュレーションを行えば良い．或いは可能な軌道の組み合わせの中で動的計画法を用いて探索する場合も存在する．

パズルゲームで戦略を考える場合は，パズルゲームのモデル上でシミュレーションして最適な解答を考える．囲碁では，2006 年以降モンテカルロ木探索という方法が主流になっている．モンテカルロ木探索は，ある一手の評価を，その手以降をランダムに終局まで討つことでシミュレーションすることで評価を行う [40].

たとえば，ナビゲーション・メッシュによるパスが得られた場合に，この連続した長方形を渡る最短のパスをたどりたいとする（Fig. 7.13）．長方形と長方形の間の空間，これをポータル（通過する空間）と言うが，このポータルに対して数個の候補点を考えて，この候補点どうしをつなぐすべての経路を候補として考える．最短の組み合わせを得るためには，候補点を入れ替えては経路を計算し，動的計画法によって経路が短くなる方向へ向かって組み合わせを決定する．これは『ARMORED CORE V』（FromSoftware，2012 年）の 3D パス検索で取られている方法である [74] [75].

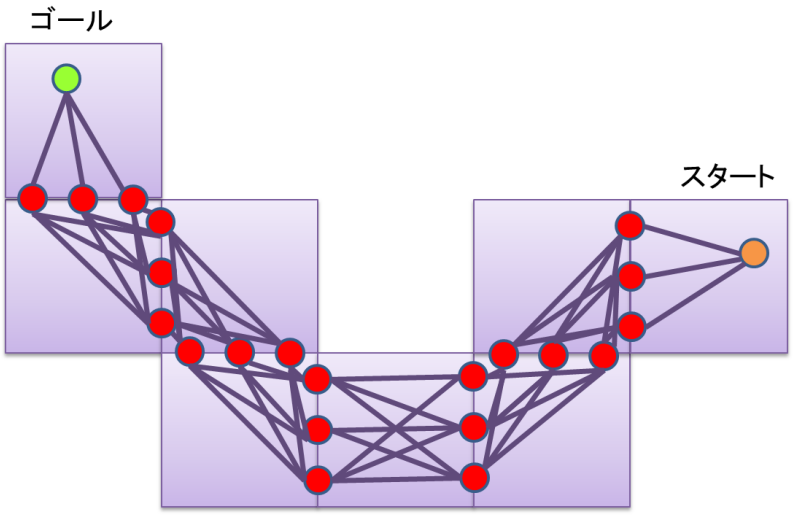


Fig. 7.13　シミュレーション・ベースの意思決定

MCS-AI 動的連携モデルにおける関係性

シミュレーション・ベースの意思決定は，スパーシャル AI から提供される抽象化された空間を用いて行われる．たとえばキャラクターの軌道計算も，実際のゲーム世界ではなく，ナビゲーション・メッシュ上で簡易的な軌道計算を用いることで負荷を軽くする．メタ AI

がコントロールするのは，シミュレーションの負荷である．シミュレーション・ベースの意思決定にどれだけの計算負荷を与えるかをコントロールすることで，シミュレーションの深さが決定される．そのシミュレーション結果に基づいて，キャラクターの行動が決定され，シミュレーションに沿って行動が展開されるため，これは投機型命令と言える．ゲームの難易度をシミュレーションの計算量によって調整する，などは，将棋や囲碁のソフトでは良く用いられる方法ではあるが，アクションゲームにおいても敵キャラクターの意思決定の計算量を増減することで難易度を調整することがある．

7.4 考察

本章では，MCS-AI 動的連携モデルにおけるキャラクターAI について解説した．第 7.1 節では，キャラクターAI が持つ自律的キャラクターAI がメタ AI からの命令の型によって，シームレスに演技的意思決定に移行することを述べた．第 7.2 節ではメタ AI からキャラクターAI に渡される命令の種類について分類した．第 7.3 節は，それぞれの意思決定アルゴリズムにおいて，メタ AI との関係の仕方を示した．メタ AI はキャラクターAI の意思決定部分とさまざまな形の関連を持ち，逆に言えば，意思決定の形式によって，メタ AI のコントロールの在り方はある程度制限される．メタ AI とキャラクターAI の関係性を考慮しつつ，キャラクターAI の意思決定の形式を決定する必要がある．

8. スパーシャル AI

本章では MCS-AI 動的連携モデルにおけるスパーシャル AI の役割について述べる．第 8.1 節では MCS-AI 動的連携モデルにおいてスパーシャル AI が持つ役割について述べる．第 8.2 節では，キャラクターAI，メタ AI の意思決定との関係について述べる．第 8.3 節ではスパーシャル AI の持つ空間解析技術について，第 8.4 節では状況解析技術について，それぞれの機能を列挙し，それぞれの機能が MCS-AI 動的連携モデルに提供する情報の役割について述べる．

8.1 MCS-AI 動的連携モデルにおけるスパーシャル AI

本節では，MCS-AI 動的連携モデルにおけるスパーシャル AI のメタ AI，キャラクターAI との関係について述べる．第 8.1.1 項では，スパーシャル AI とナビゲーション AI についてそれぞれの特徴を概説し，第 8.1.2 項ではスパーシャル AI の機能について述べ，第 8.1.3 項ではスパーシャル AI とナビゲーション AI の比較を行う（Table 8.1）．

8.1.1 スパーシャル AI とナビゲーション AI

キャラクターをよりゲーム世界とより密接に物理的インタラクションをさせるためには，空間の構造や地形の特徴を把握し，行動を組み立てるために必要な情報を抽出することが必要とされる．そこで，それぞれのキャラクターの身体特性・運動能力に応じた空間的情報の認識をサポートする「ナビゲーション AI」が形成された．

Table 8.1　ナビゲーション AI とスパーシャル AI の比較

	ナビゲーション AI	スパーシャル AI
機能	パス検索	パス検索を含む空間解析・状況解析全般
データ	ナビゲーション・データ	世界表現全般
アルゴリズム	パス検索	パス検索, 影響マップ, LOS マップ, 位置解析など全般
動作	クエリーに応答する	自律的に空間・状態解析を行う
サポート対象	主にキャラクターAI	メタ AI，キャラクターAI
スケール	キャラクタースケール	マルチスケール（キャラクタースケール～ゲーム全体のスケール）

「ナビゲーション AI」はゲーム開発において単一のキャラクターではなく，複数の種類のキャラクターの移動のパス検索をサポートする．そのためキャラクターのサイズごとに数種類のナビゲーション・データを生成・所持し，またキャラクターごとの身体的・運動的特徴を考慮したパス検索を行う必要がある．そのために，あらかじめゲーム開発中に地形解析を行った結果を保持し，ゲーム実行中にもメタ AI，キャラクターAI と独立にリアルタイムに地形解析を行う．たとえば，キャラクターがプレイヤーから見えない位置をプレイヤーが移動する度にリアルタイムで解析しメタ AI，キャラクターAI に提供する [130]．また，多種多様なキャラクターのキャラクターAI を製作する時に，ナビゲーションは最も身体のスケールと運動性能に依存する部分であり，ナビゲーション AI によってこの多様性が吸収されるおかげで，キャラクターAI は，全キャラクターを通じてある程度共通に記述することが可能になる．このように，ナビゲーション AI はメタ AI，キャラクターAI の要求に応じて経路を提供する．

さらにメタ AI の発展は，メタ AI が必要とするゲーム全体の空間・状況解析を必要とした．キャラクターの身体からゲーム全体まで，局所から大局までの空間解析・状況解析の情報を提供するために「ナビゲーション AI」は「スパーシャル AI」へと発展した．スパーシャル AI はナビゲーション AI が持つ機能に加えて，自律的にゲーム世界の解析する動作をする独立した AI システムである．本章では MCS-AI 動的連携モデルにおける「スパーシャル AI」について述べる．

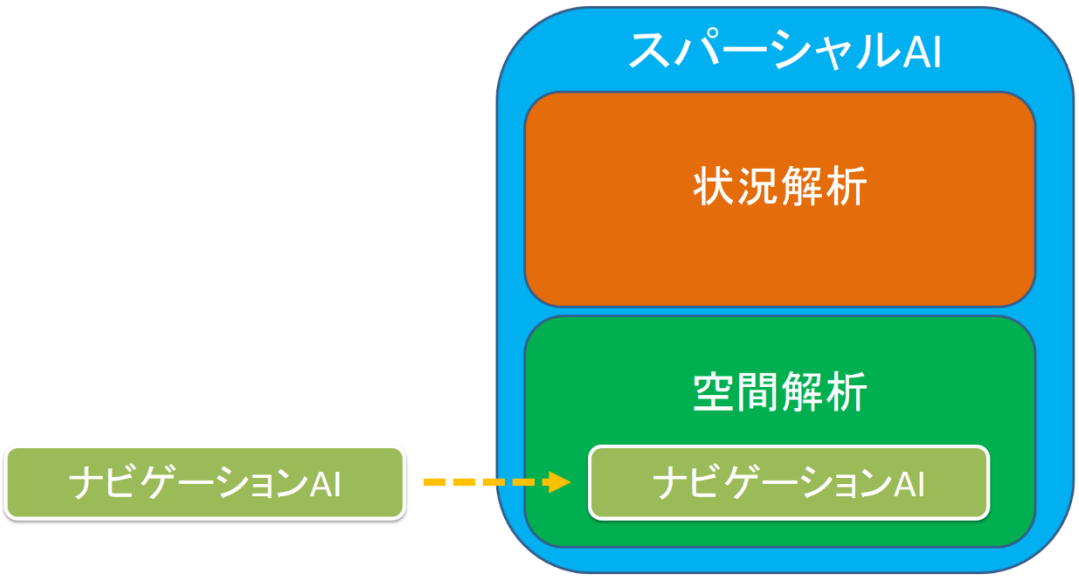


Fig. 8.1　ナビゲーション AI からスパーシャル AI への拡張

8.1.2　スパーシャル AI の機能

スパーシャル AI の機能は大きく二つあり，空間解析と状況解析である（Fig. 8.1）．空間

認識は空間の中で目的に応じた特徴を発見することである．目的に応じた経路，目的に応じた位置，などである．また状況認識は物理的なものに限らず，空間をベースとした抽象的な状況を認識することである．たとえば，敵・味方の勢力図などである．

スパーシャル AI は，複雑化するレベルデザインに対して，メタ AI，キャラクターAI の空間全般の認識をサポートする．地形や空間情報を抽象化する役割を持ち，その情報提供によって，メタ AI も，キャラクターAI も，抽象化された空間情報の上で汎用的で抽象度の高い思考を構築することが可能となる．たとえば，「ある地点に移動する」「移動するのに敵のいない安全な経路をたどる」などは，そのようなパス検索モードを選択すれば，どのような種類のキャラクターに対してもスパーシャル AI が必要なパスを提供する．キャラクターAI のみならず，メタ AI をサポートするスパーシャル AI は，キャラクターの周囲のスケールの局所的な地形・空間のみならず，ゲーム全体の大局的な地形・空間およびプレイヤー・ＮＰＣを含むゲーム状況を解析し，特徴を抽出する役割を持つ．

8.1.3　スパーシャル AI とナビゲーション AI の相違点

スパーシャル AI とナビゲーション AI の相違点は，ナビゲーション AI がメタ AI，キャラクターAI からのクエリーを受けて応答する AI であることに対して，スパーシャル AI がリアルタイムに自律的に活動する AI，という点にある．スパーシャル AI は常にメタ AI，キャラクターAI の空間，状況に対する行動の可能性を自律的に提示し続ける．メタ AI に対しては，メタ AI が動かすことができるステージ上のアセットの影響を解析し，その情報を提示する．たとえば，フィールドに山の上から岩を転がして状況を変化させることができるなら，その岩や動かす向きなどを提供する．またスマートオブジェクト，スマートテレインによって，オブジェクト側に人工知能を持たせて，そのオブジェクトを使用するタイミングや手法をメタ AI に自ら通知する．またキャラクターAI に対しては，ゲーム状況が変化し NPC が運動を行う中で，その瞬間に可能となる行動可能性を常に監視し，キャラクターに提示する．たとえば，A 地点から B 地点へ移動する途中で，目の前に木が倒れてきた場合には，どのような地点でどのアニメーションで飛び越えられるかを提示する．或いは，格闘ゲームで運動プランが実行されている時に，敵の動きが想定外の動きをした場合，現在の運動に連続する形でどのように身体を動かせばガードできるか，などを提示する．このようにスパーシャル AI は，メタ AI，キャラクターAI の意思決定ではカバーしきれない空間における行動可能性，さらに偶発的に起こる行動可能性をメタ AI，キャラクターAI にノティフィケーションすることで行動の柔軟性，行動の完成度を高める役割を持っている．

8.2　意思決定とスパーシャル AI

本節では，メタ AI,キャラクターAI とスパーシャル AI の関係について述べる．スパーシ

ャル AI は，メタ AI，キャラクターAI からのクエリーに応じて空間的情報を提供するが，逆にスパーシャル AI からメタ AI, キャラクターAI に自発的に提供する空間的情報がある．それは，第 3.4.2 節で述べたアフォーダンス情報である．メタ AI，キャラクターAI は，まずその空間において，どのような行動が可能かを認識し，その上で思考する，というパターンが多い．そこで，スパーシャル AI はメタ AI のコントロールする領域（アクティブエリアセット（Active Area Set，AAS）と呼ばれる）で実行可能な行動を提示し，キャラクターAI に関してもその周囲の空間で為し得る行動を提示する．メタ AI，キャラクターAI は，その情報を参考に意思決定を行う．反射型意思決定の場合，現在の状態をもとに行動を選択する．非反射型意思決定の場合，目的が達成できるように環境を変化させて行く．

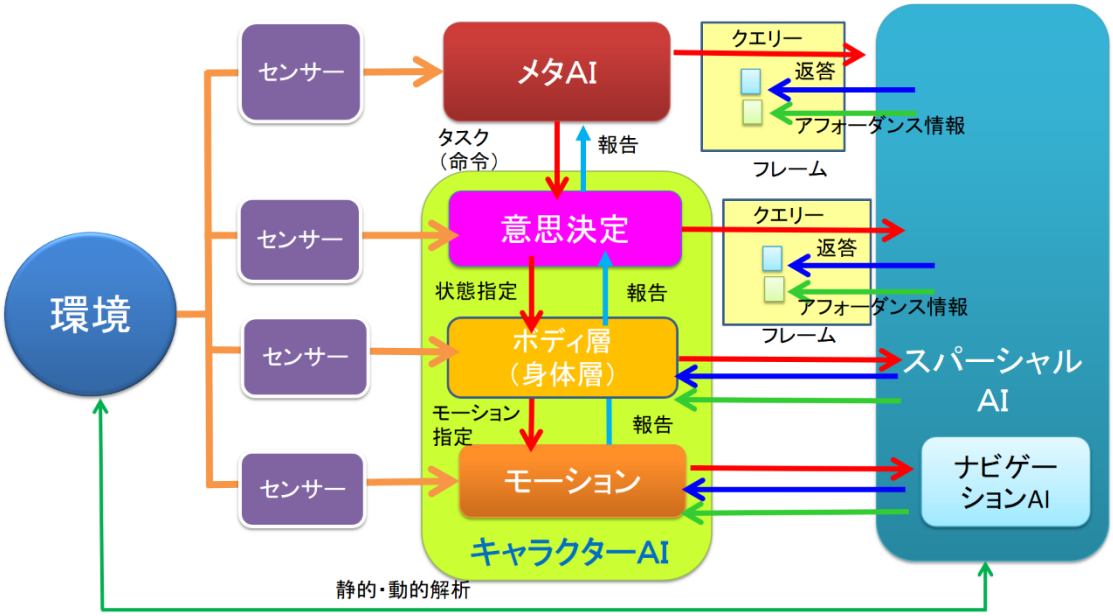


Fig. 8.2　スパーシャル AI とフレーム

どのような意思決定アルゴリズムであれ，行うべき行動が決定すれば，実際，その行動を実行するために必要な空間的情報が必要になる．たとえば，「敵を剣で攻撃する」のであれば，敵までのパスと剣を振り始める位置などである．つまり，スパーシャル AI には 2 つの役割があり，意思決定のために必要な空間的情報と，意思決定の後，行動の詳細をデザインして行くための空間的情報である．後者は前者より解像度の高い空間的情報を必要とする．

メタ AI，キャラクターAI には，意思決定を行う「フレーム」があり，そのフレームに必要な情報をスパーシャル AI は提供する．フレームへの情報提供は，意思決定前にスパーシャル AI から行われるが，意思決定後に必要な情報はクエリーの形でメタ AI，キャラクターAI から求められる（Fig. 8.2）．

また，スパーシャル AI はキャラクターAI のボディ層，モーション層に対しても情報の提

供を行う．ボディ層，モーション層が意思決定の決定を実行するために，より詳細な空間的情報を必要とする場合があるからである．たとえば，柵を飛び越える，という場合には，助走するパスと手をつく柵のポイント情報をスパーシャル AI から提供する．これは具体的な意思決定が行われたあとのボディ層，モーション層で必要となる情報である．

以下，スパーシャル AI の持つ空間解析を第 8.3 節において，状況解析について第 8.4 節について述べる．

8.3 空間解析

本節では，スパーシャル AI の空間解析の機能について述べる．スパーシャル AI の空間解析機能の基礎となるのは，空間の知識表現であり，第 3.4.1 項で紹介したように「世界表現」(World Representation）と呼ばれる．世界表現上にそれを解析して特徴を抽出するアルゴリズムを適用する．以下，デジタルゲームで良く使用される空間表現について，8.3.1「パス検索」，8.3.2「LOS/LOF マップ」，8.3.3「位置検索技術」の順番に述べる．それぞれ「基本原理」と「MCS-AI 動的連携モデル内における役割」に分けて記述する．

8.3.1 パス検索

本項ではパス検索の基本原理の説明と，MCS-AI 動的連携モデルにおけるスパーシャル AI の本機能の使用の仕方を述べる．

基本原理

連続空間を有限の空間要素で表現する仕組みがナビゲーション・データである．ゲーム開発で用いられるナビゲーション・データの代表的な世界表現は 2 種類あり，接続された凸多角形からなるナビゲーション・メッシュ（ナビメッシュ）(Navigation Mesh, NaviMesh）と点群からなるウェイポイント（Waypoint）である．ナビメッシュは，地形の中でキャラクターが歩ける場所を連結された三角形（凸多角形なら良いが，たいていは三角形を用いる）で敷き詰める．ナビゲーション・メッシュは地形の起伏やその表面の属性情報の表現に適しており，ウェイポイントはポイントを指定するため正確な位置関係の表現に適している．

パス検索のアルゴリズムは A*法やダイクストラ法, ルックアップテーブル法などが用いられるが，コストが最小になるパスを発見する [78]．コストは距離の場合もあれば，地形情報や敵の攻撃リスクを加味したコストになる．たとえば,「危険」な場所はコストを 2 倍にする，路面が雪であれば距離を 1.2 倍にしたものをコストにする，などである．たとえば，ロボットゲームでは，崖の近くのナビメッシュは落下の危険があり，また壁近くのナビメッシュは衝突しやすいので，崖の境界にあるナビメッシュを検出してコストを上げて

おくことで，できるだけ通らないようにする．逆にステルスゲームの場合は，物陰に隠れた方が良いので，物のある側のメッシュのコストを低くしてなるべく通るようにする．

本機能のキャラクターAI，メタ AI からの使用方法

スパーシャル AI が事前にキャラクターAI に提供するのは，キャラクターAI が持つフレームに必要な情報である．たとえば，該当するキャラクターからたどり着ける他のキャラクターやオブジェクトのリストである．メタ AI に対しても，メタ AI が持つフレームに必要な情報を提供する．ストラテジーゲームであれば提供する情報は，最も近い道のりにいる友軍の ID などである．メタ AI，キャラクターAI が目標を決めた後は，リクエスト・クエリーに応じて目標へのパスを提供する．

またスパーシャル AI は，メタ AI が地形やダンジョンを自動生成する場合，地形やダンジョンを動的に解析し，適切にスタート地点からゴール地点までがつながっているか，チェックを行う．これは「接続テスト」（コネクティビィ・テスト）と呼ばれ，パス検索を用いて行われる．たとえば「Age of Empires II」（Ensemble Studio，1999 年） は多数の兵士を操作する RTS（リアルタイムストラテジーゲーム）あるが，自陣と他陣がきちんと陸地でつながっていないとゲームにならないため，コネクティビティ・テストに問題があれば再自動生成を行う [131].

8.3.2 LOS/LOF マップ

本項では LOS/LOF マップの基本原理について説明し，MCS-AI 動的連携モデルにおけるスパーシャル AI の本機能の使用の仕方を述べる．

基本原理

「LOS マップ」（Line of Sight マップ，見晴らし距離マップ）は，各ウェイポイントから複数の方向に対する見晴らし距離が埋め込まれたマップである（Fig. 8.3）．この見晴らし距離を用いて，各ポイントからどの領域までが見えているかを判定することができる．或いは，そのポイントから攻撃が複数の方向のどの距離まで届くかを埋め込んだマップを「LOF ナップ」（Line of Fire マップ）と呼ぶ．たとえば，ある場所 a（敵）と b（自分）の間で視線が通っているか判断したい場合には，次の 2 つのチェックをする．

1. a から b に対する LOS 距離が実際の距離より長いか
2. b から a に対する LOS 距離が実際の距離より長いか

この 2 つが満たされていれば，a から b に対して射線が通っていると判定する．LOS 距離はあえて精度のない数値であるので，二重のチェックをかける．たとえば，a から b の実

際の距離が 7m とし，a から b 方向に対応する見晴らし距離が 8m であるとすると条件(1)が満たされている．次に条件(2)は，b から a 方向に対応する見晴らし距離が 10m なので満たされている．この 2 つから，a から b へ視線が通っていると判定する．このように LOS データを準備することで，どのような 2 点間であっても，簡単な計算だけで射線が判定される．『Killzone』は PlayStation2 の時代に，FPS における最も重い処理である射線判定を高速化することで, NPC に視線を考慮に入れた高度な動きをさせることを可能にした［79].

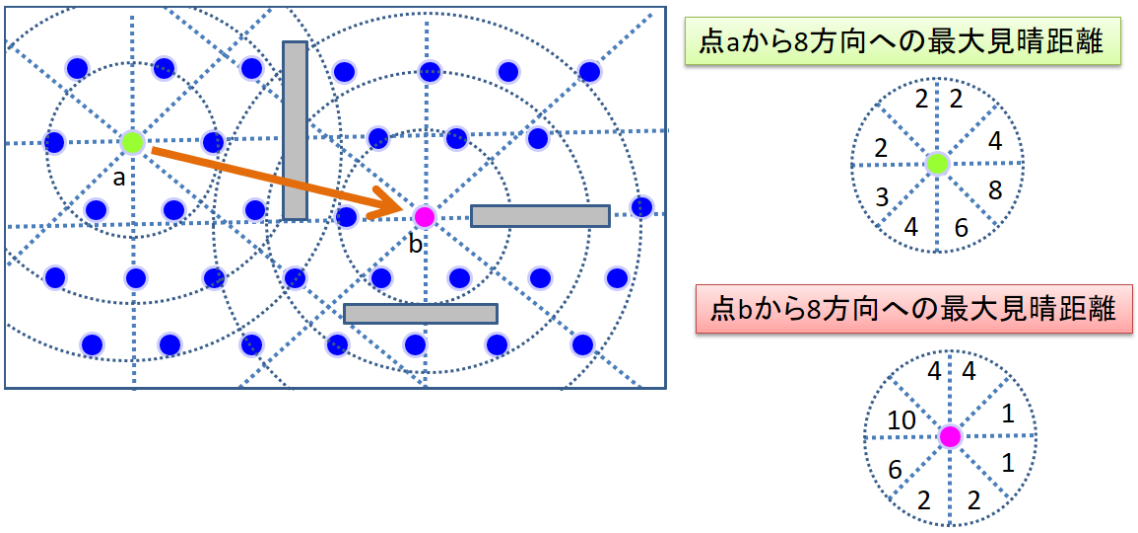


Fig. 8.3　『Killzone』における LOS マップによる世界表現［79］

本機能のキャラクターAI，メタ AI からの使用方法

スパーシャル AI は，本機能を用いて，プレイヤーに「隠れることができるポイント」（Hidden point）を提供する．たとえば，スパーシャル AI は，LOS マップを用いて，プレイヤーの周囲のポイントを精査し，敵の視線から逃れられるポイントをキャラクターAI に提供する．スパーシャル AI は，プレイヤーが「隠れる」行動が取れる可能性を伝えると共に，具体的な座標を提供する．メタ AI に対しては，スパーシャル AI は，プレイヤーを中心とした領域の中で，プレイヤーに斜線が通る戦術ポイントを検索し提供することで，メタ AI の戦術プランを補佐する．

8.3.3　位置検索技術

本項では，位置検索技術の基本原理について説明し，MCS-AI 動的連携モデルにおけるスパーシャル AI の本機能の使用の仕方を述べる［132].

基本原理

メタ AI にとっては，新しい NPC を動的にどこに配置するべきか，という課題がある．また，キャラクターAI にとっては，次の目的をどこにするべきか，という課題がある．「位置検索技術」は，目的に沿った最適な位置を動的に求めるためのアルゴリズムである．CRYENGINE（CRYTEK，2004-）上で開発され，2013 年以降使用されているが，これ以降，同様のシステムが複数のゲームエンジン上で実装されている［133］．本アルゴリズムは３つの段階からなる．

(1) 生成フェーズ
(2) フィルタリング・フェーズ
(3) 評価フェーズ

以下，順にそれぞれのフェーズを解説する．

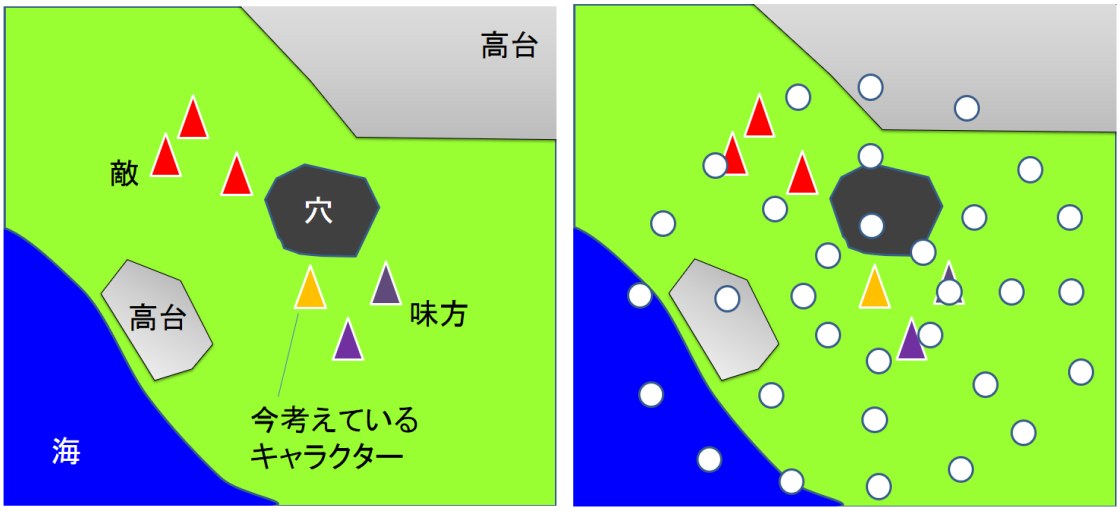


Fig. 8.4　位置検索技術の生成フェーズ

(1) 生成フェーズ

まず探索したい領域にポイントを動的に分布する（Fig. 8.4）．分布の形は普通グリッド状を取るが，同心円状でも，ランダムでも，地形や用途に応じて使い分ける．Fig. 8.4(a) では，弓兵キャラクターの次の位置取りを考える．まず，この弓兵キャラクター（図中央△キャラクタ 0）から同心円状に点列を生成する（Fig. 8.4 (b)）．

(2) フィルタリング・フェーズ

生成フェーズで生成した点から指定する条件でフィルタリングして，不要な点を除く．この時の条件の指定の仕方は，どの対象とどのような関係なのか，によって記述される．

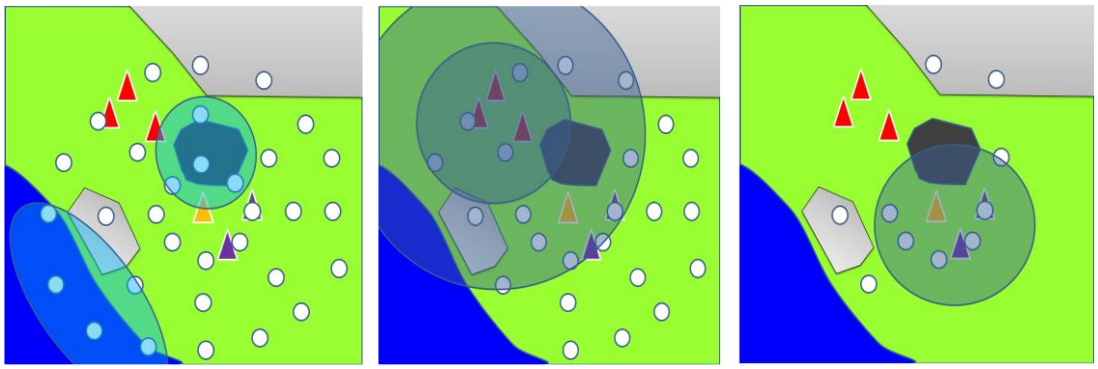
- 足場の悪い点を削除（Fig. 8.5 (a)）

- 敵中心から 5m 以内，10m 以上を削除(Fig. 8.5 (b))
- 味方中心から 5m 以内を削除（Fig. 8.5(c)）

などである．このように，一つ一つ条件によって重ねて行くことで，生成した点群から不要な点を除去して行く．

(3)評価フェーズ

一つの代表点を選択するために，スコアを付ける．この評価関数は，作り方に制限はないが，位置情報とは関係ない情報などを使うこともできる．たとえば，日当たりの良さ，高さ，見晴らしの良さ，などである．そして最も高いスコアのポイントを選択する．図の場合は，各ポイントに対して「敵からの遠さ×高さ」を評価関数として，最終的に一つのポイント（岩の上のポイント）を決定する（Fig. 8.6）．



(a)　(b)　(c)

Fig. 8.5　位置検索技術のフィルタリング・フェーズ

このように 3 つのプロセス「生成」「フィルタリング」「評価」によって，目的地を動的に決定する．

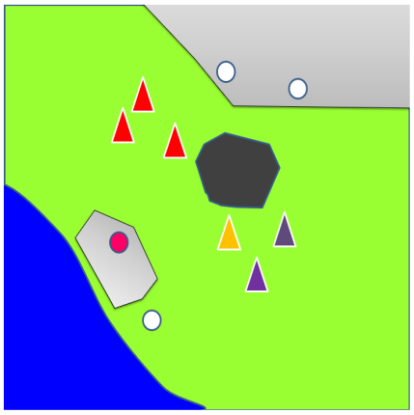


Fig. 8.6　位置検索技術の評価フェーズ

本機能のキャラクターAI，メタ AI からの使用方法

スパーシャル AI の位置検索技術によってキャラクターは「自分で目的地を状況に応じて決める」ことができるようになる．位置検索技術とパス検索技術によってキャラクターは「自分で目的地を決めて，目的地までの経路を決定する」ができるようになる．スパーシャル AI はキャラクターAI が環境の中で自律的に運動する知的能力の基礎を提供する．メタ AI に関して，スパーシャル AI は位置検索技術を用いて，動的にキャラクターやアイテムを配置する適切な場所を提供する．この例については，第 11 節で述べる．また戦闘のみならず，メタ AI は，NPC の街での立ち位置，プレイヤーに話しかける時の適切な立ち位置などを位置検索技術によって求め，NPC を向かわせる，ということが可能になり，違和感のない日常的な状況を作り出す基礎ともなる．

8.4 状況解析

本節では，スパーシャル AI の状況解析の機能について述べる．ゲームステージの状況は変化し続ける．この変化の中の，必要な変化を捉える役割がスパーシャル AI にある．キャラクターAI の場合にはセンサーからの情報によって自らの周囲の状況の認識を形成し，メタ AI の場合にもゲーム全体に対するセンサーから情報を得てゲーム全体の状況を形成するが，詳細な認識を形成するには十分ではなく，スパーシャル AI のサポートが必要となる．第 8.4.1 項「社会的空間解析」では街など日常的な空間において複数のキャラクターが自然な位置取りをするための空間解析について述べる．これを用いて自然な位置に NPC を配置する．第 8.4.2 項「影響マップ」は主に戦闘や陣取りゲームなど敵味方に分かれた場合の互いの勢力図を動的に形成するために用いる．

8.4.1 社会的空間解析

本項では，社会的空間解析の基本原理について説明し，MCS-AI 動的連携モデルにおけるスパーシャル AI の本機能の使用の仕方を述べる．

基本原理

社会的空間（Social Space，ソーシャル・スペース）は，人間同士の位置関係が作る空間のことである（Fig. 8.7）．たとえば，二人が話すときには対称な位置に距離を開けて話す．これは人にとって背中が最も無防備であるから，お互いの背中を監視することで無意識のうちに安全度を上げようとしているからだと考えられる．壁があれば，壁を背にして話す．三人であれば，真ん中に円形のスペースを開けて話す．人が集まったときに，人と人の間にできる空間のことを社会的空間と言う．ゲーム空間でもこの社会的空間を考慮することで，自然な群衆を作り出すことができる［134］．3 人のキャラクターが立ち話をする場合，何もない空間であれば真ん中に円を作るように等間隔に並ぶ．壁がある場合は，二人は壁に背を

向ける．これは，人間にとって背中から攻撃されることは弱点なので，お互いがお互いの背中を監視するような形になるからである．真ん中のスペースを O-Space，この外と三人を囲う円の間のスペースを P-Space，さらにその外とそれを囲う円の間のスペースを R-Space と言う［135］．この円の中を誰かが横切った時には，それを見ることができるキャラクターが視線で追う．また他のキャラクターもソーシャル・スペースの中に入らないようにパス検索を行う．

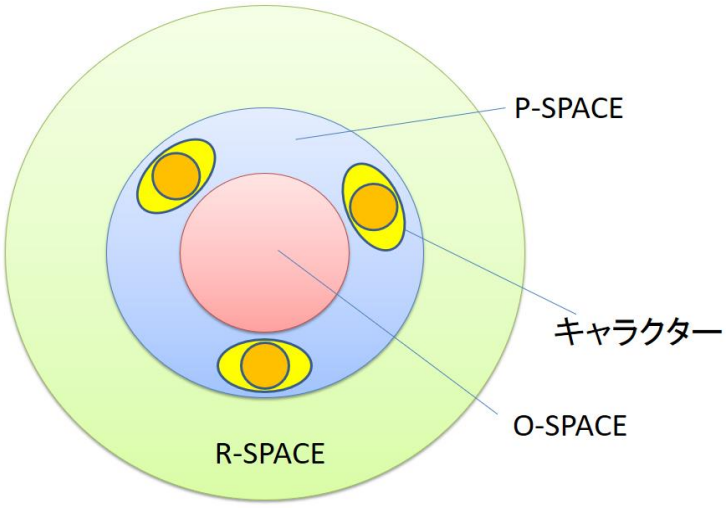


Fig. 8.7　キャラクターが作る社会的空間（上から見ている）［135］

本機能のキャラクターAI，メタ AI からの使用方法

社会的空間解析は，多数の NPC を出現させて位置取りや経路をより人間らしい配置にするときに必要である．スパーシャル AI の社会的空間解析の情報によって，メタ AI は動的に街など多数の NPC を自然に配置することができる，既にステージに出てしまっている NPC のキャラクターAI にとっては集団の中に自然に溶け込む位置取りをするために必要である．

8.4.2　影響マップ

本項では，影響マップの基本原理について説明し，MCS-AI 動的連携モデルにおけるスパーシャル AI の本機能の使用の仕方を述べる．

基本原理

影響マップは戦局を判断するために用いられる．マップを碁盤の目のように区切り（３次元の場合は立方体で区切る）．敵のいる場所を熱源，味方のいる場所を冷却源として周囲に熱を伝搬して行くモデルを考える．これをヒートマップ（Heat-map），或いは，影響マップ（インフルエンス・マップ，Influence map）と呼ぶ（Fig. 8.8）．

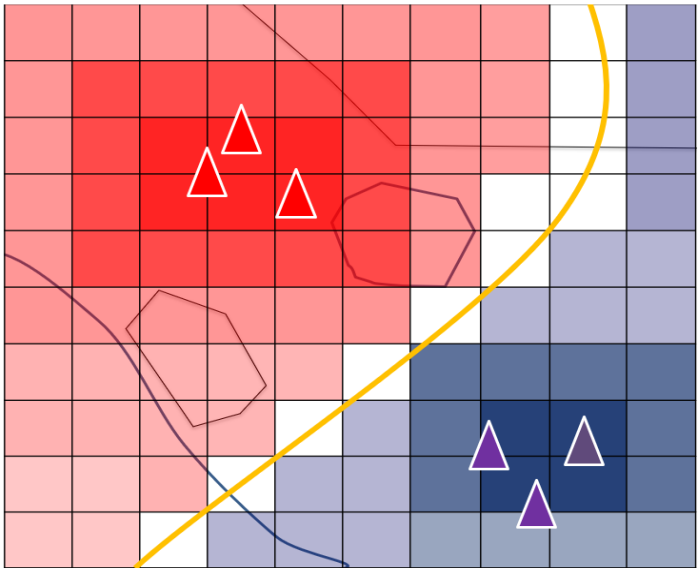


Fig. 8.8　影響マップ．中央のラインは勢力が均衡するフロントライン（前線）

熱源から周囲のマスに熱が伝播し，次にさらにその向こうのマスに伝搬する．熱源が移動すると熱の伝搬も変化する．熱源が移動しても熱が残るので，熱源の経路と進行方向には熱が残りやすい．また反対の冷却源も同様に経路と進行方向を冷却する．ある時点の熱の分布を取ると，敵・味方の勢力図が描かれる．可視化する場合は熱の高い場所を赤く，低い場所を青くすることで，天気予報の温度マップのように可視化される．このような熱の値をパス検索に載せることで，敵勢力を避けるようなパス検索や，逆に敵勢力をなるべく通過するパス検索を行うことができる．熱源と冷却源の中間地点は温度が相殺してゼロになり，温度がゼロになっている地点をつなげれば，フロントライン（前線，勢力が均衡するライン）を自動的に検出することができる（Fig. 8.8）［136］.

本機能のキャラクターAI，メタ AI からの使用方法

キャラクターAI はスパーシャル AI が提供する影響マップの情報を，パス検索のコストに反映することで，状況を考慮したパス検索を行うことが可能となる．たとえば，敵勢力の強さをコストに追加することで，迂回するようなパスを発見することができる．

ゲーム全体をコントロールするメタ AI には，スパーシャル AI が動的に生成する影響マップが必要である．以下，二つの勢力が森の中で争っているゲームがあり，メタ AI は情勢を見ながら，できるだけ勢力の均衡を保つことが目標であるとする（Fig. 8.9）.

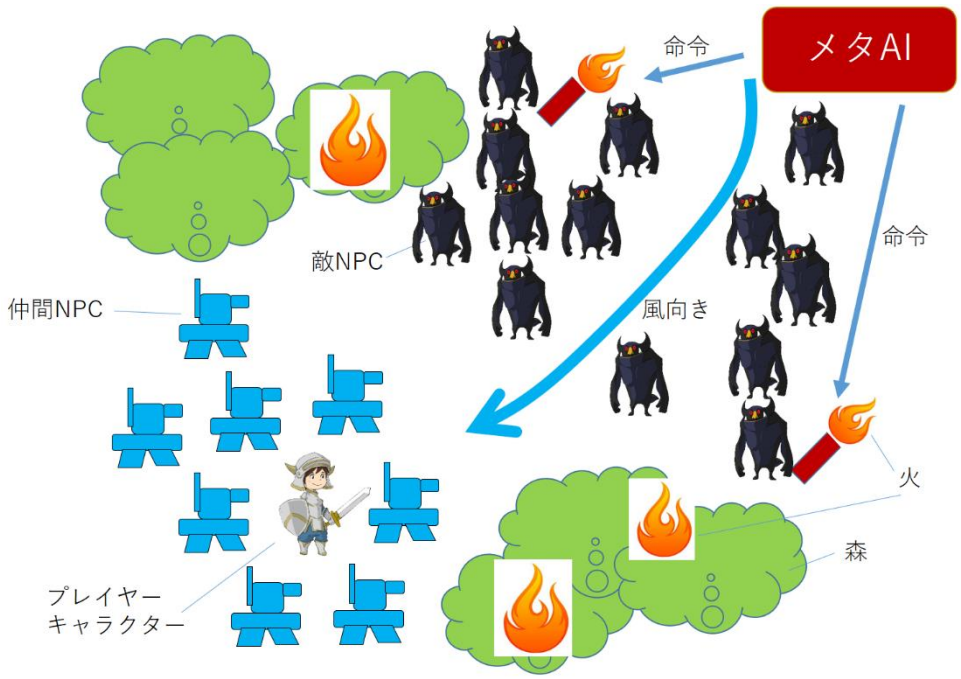


Fig. 8.9　天候を操ると同時に敵キャラクターに指示を出すメタ AI

影響マップから，プレイヤーが追い詰められているのか，それとも，それとも優勢なのかを知ることができる．プレイヤーの周囲が赤色，つまりプラスの値が多ければプレイヤーがピンチである，プレイヤーの周囲が青色，味方ばかりなら比較的安全ということになる（Fig. 8.10）．さらに影響マップを使えば敵，味方の最も密集している場所がわかる．そこで，メタ AI によってゲームをできるだけ両軍の勢力が均衡するように導く，たとえば，プレイヤー軍が有利な時は，敵軍が風上になるように風を吹かすと同時に，敵キャラクターに指示を出して，近くの森に火を付けさせるなどして，自然を利用して戦闘を動かす（Fig. 8.9）．

フロントラインが勢力動向の目安となり，これが動かないということは膠着状態にあることを示している（Fig. 8.11）．両軍を分断したい時は，ここに岩を転がす，風向きを変えて森に火を放つ，などをして変化を与える．もしプレイヤーがピンチならば，敵が最も密集している場所に岩を転がして，局面を打開する．逆にプレイヤーが優勢ならば，味方の最も密集している場所に岩を転がして，局面を引き締める．メタ AI は，この自然変化を，いつ，どのタイミングで，どのように行うかを，スパーシャル AI が提供する影響マップの助けをかりて指定する．

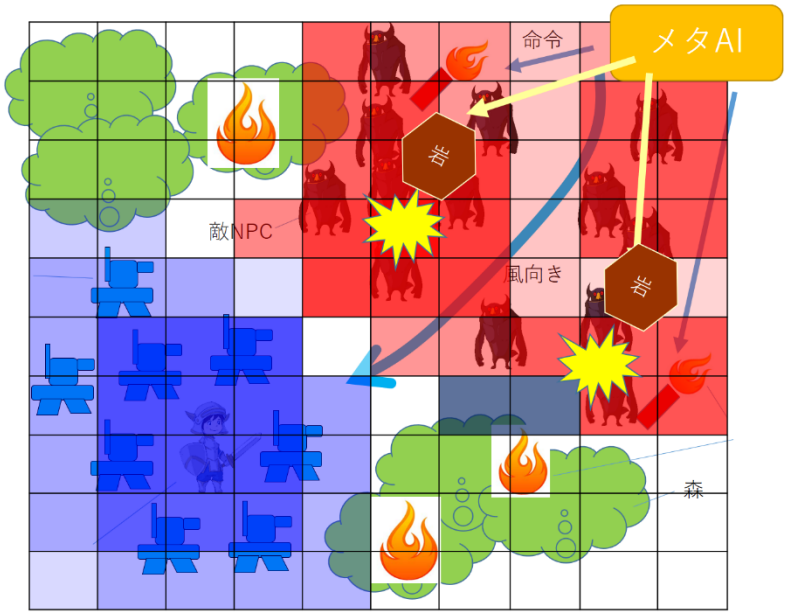


Fig. 8.10　岩を転がして局面を変化・調整させるメタ AI

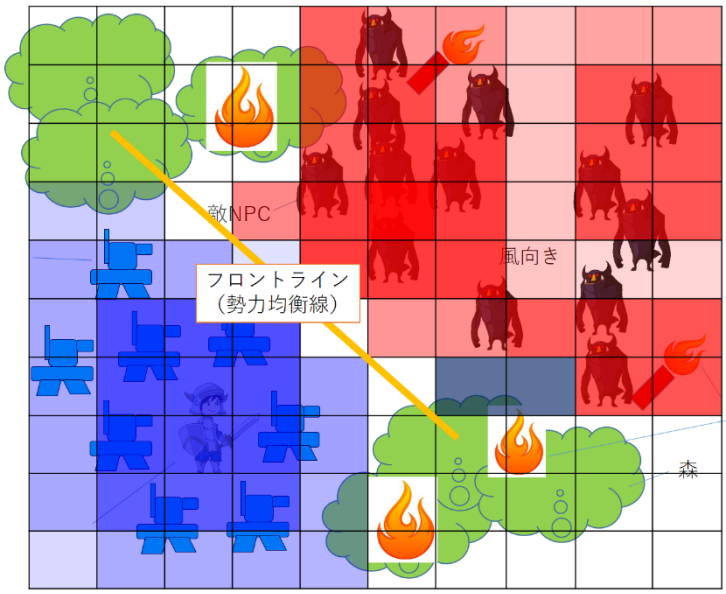


Fig. 8.11　ヒートマップから導かれるフロントライン（勢力均衡線）

8.5　考察

本章では, MCS-AI 動的連携モデルにおけるスパーシャル AI の役割について述べてきた. 第 1 節では MCS-AI 動的連携モデルにおいてスパーシャル AI が持つ役割について, 第 2 節では，キャラクターAI，メタ AI の意思決定との関係について，第 3 節では空間解析技術について，第 4 節では状況解析技術について，スパーシャル AI の持つ様々な機能を列挙し，

それぞれの機能が MCS-AI 動的連携モデルに提供する情報，その情報がメタ AI，プレイヤーAI についてどのように利用されそれぞれの運動に寄与しているかを述べた．

スパーシャル AI は，第 8.2 節で述べたように，常に提供する情報と，意思決定がなされた後に提供する情報がある．スパーシャル AI は，メタ AI，キャラクターAI が，その場で考えるべき問題のフレームの中身を具体的に準備する役割を持ち，このスパーシャル AI の果たす役割のおかげで，それぞれの AI はさまざまな地形や情報においても，同一のフレームの中で思考することが可能となり，思考を汎用的かつ抽象度の高い形で形成することが可能となる．

またユーザーから見た場合に，NPC がどれだけその場の地形・状況を考慮して活動しているかが，NPC の AI の賢さを判定する指針となる．すなわち，スパーシャル AI が，どれだけ地形情報，状況の特徴を提供し，その情報をメタ AI，キャラクターAI が活用できるかが，コンテンツの質に影響する．また，ゲームの中では戦闘が何百，何千回と行われるため，単調になりがちである．そのような戦闘の最大の変数が地形と状況である．それぞれの戦闘における地形・状況を反映することは，戦闘のバリエーションを豊かにし，ユーザーに飽きさせない，くり返しのない戦闘に近づけることができる．

MCS-AI 動的連携モデルにおける 3 つの AI について，第 6 章でメタ AI，第 7 章でキャラクターAI，第 8 章でスパーシャル AI について述べて来た．各 AI の持つそれぞれの機能を解説し，各 AI の具体的な連携の仕方を述べてきた．以下の第 9 章は，すでに完成したゲームを MCS-AI 動的連携モデルから捉え直すことで，そのタイトルの AI 構造を明らかにし，また各 AI の機能を抽出することで技術的蓄積ができることを示す．

9. 既存のゲームへの MCS-AI 動的連携モデルの適用例

本章では，MCS-AI 動的連携モデルの既存のゲームへの適用例を示す．既に出来上がったゲームではあるが，さかのぼって過去のゲームタイトルに本モデルを適用し，

- MCS-AI 動的連携モデルがどの程度あてはまるか
- MCS-AI 動的連携モデルが実装されれば，そのゲームはどのようになるか
- MCS-AI 動的連携モデルどのような技術を抽出できるか

について考察する．

取り上げるゲームは『パックマン』(株式会社バンダイナムコエンターテインメント, 1980 年）と『クロムハウンズ』((C)SEGA / FromNetworks,Inc./ FromSoftware,Inc., 2006 年）である．『パックマン』を取り上げるのは，このゲームがゲーム AI の原点として位置づけられる作品であるからである．『パックマン』の開発情報は，三宅がパックマンのゲームデザイナーである岩谷徹氏にインタビューをした内容と公開されたパックマンの設計書に基づいている [93] [137].『クロムハウンズ』は著者が 2006 年に AI 設計を担当したタイトルであるが，MCS-AI 動的連携モデルよりシンプルな連携モデルが実装されている．2005 年設計からの現代のゲーム AI への発展を明確に示す．

9.1 『パックマン』と MCS-AI 動的連携モデル

本節では『パックマン』に MCS-AI 動的連携モデルを導入した場合の可能性について論じる．第 9.1.1 項，第 9.1.2 項では現状の『パックマン』の AI について述べる．第 9.1.3 項では，MCS-AI 動的連携モデルを導入した場合の『パックマン』について考察する．パックマンは迷路の中を，モンスター 4 匹と接触しないように避けながら，ゲームステージ上に置かれたすべてのエサを食べ尽くすことでクリアするゲームであり，進めば進むほど難易度が上がって行く．

『パックマン』(株式会社バンダイナムコエンターテインメント，1980 年）はデジタルゲーム AI の起源として位置付けられる．それ以降のデジタルゲームの AI の典型として頻繁に引用される [138] [139] [140] [141]．各キャラクターが個性的な動きをして，その動き方のコンビネーションがゲーム性を成り立たせているからである．この，様々なキャラクターの運動の組み合わせによってゲームデザインを行う，という方向性は 1980 年以降のさまざまなゲームの指針となった．

MCS-AI 動的連携モデルから『パックマン』を捉え直すと，それぞれの AI に対応する機能は以下のようになる（Fig. 9.1）.

「メタ AI」に対応する機能 … 各ステージに，モンスターたちの挙動を以下の 3 つの要素の変化で特徴付ける.

(1) スピードレベル … 4 匹のモンスターとパックマンのスピード
(2) 出現タイミング … 4 匹のモンスターが出現するタイミング
(3) 波状攻撃タイミング … 4 匹のモンスターが同期してプレイヤーを包囲する攻撃

これらを変化させることで，ステージ毎の特徴と難易度をコントロールする.『パックマン』の波状攻撃とは，4 匹のモンスター（敵キャラクターたち）が連携してプレイヤーを追い詰め，また追い詰めたあとは，それぞれ決められたホームポイントへ四散する．追い詰める時間と緩和する時間を交互に設定することでゲームプレイの緩急をコントロールする．モンスターの「出現のタイミング」は，プレイヤーのパックマンがエサを食べた数を観測しつつ決定する．4 匹のモンスターは常に全員がステージに出ているわけではない．最初は 2 匹，時間を経てから 3 匹目，4 匹目と中央の巣（中央にある四角い囲い）からステージに出現する.

『パックマン』のメタ AI 的機能は，MCS-AI 動的連携モデルの定義するメタ AI に近いが，ゲームの認識が限定的であり（エサの数と時間），ある程度，設定パラメータで動作のタイミングが決定されるため，自律的思考が少ない.

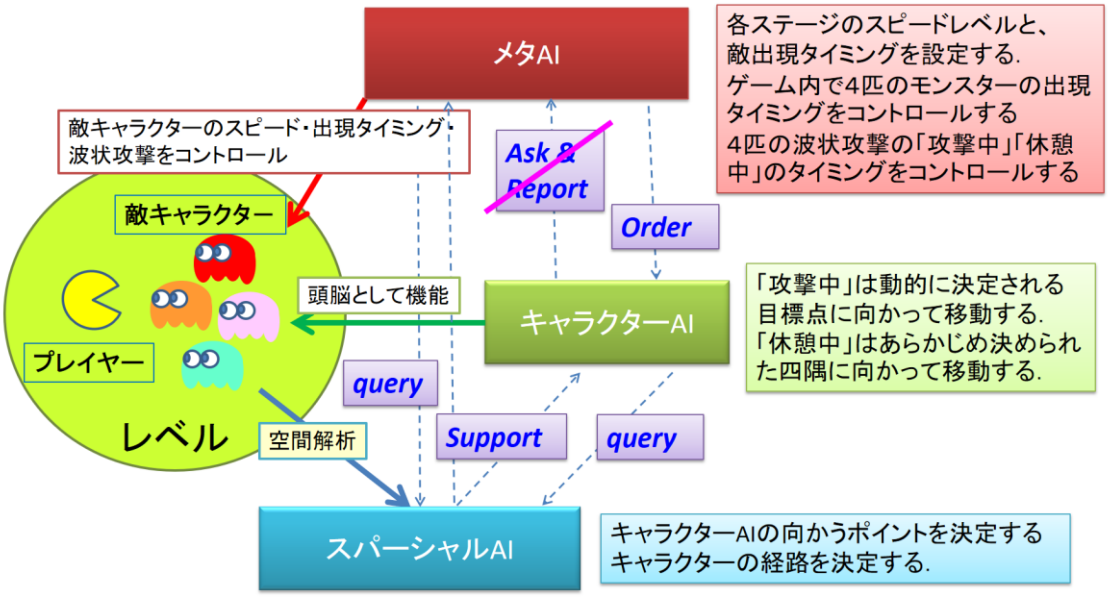


Fig. 9.1　『パックマン』における MCS-AI 動的連携モデル

「キャラクターAI」に対応する機能 … 4匹のモンスターのそれぞれの頭脳である．攻撃中は，それぞれのキャラクターがパックマンを包囲するように，個性的な追い詰め方をする．休憩中は，それぞれ割り当てられた四隅に向かうことで攻撃を緩和する．それぞれのモンスターはセンサーを持たず，周囲の状況を認識するわけではない．目的地を設定されて，そこに進むだけであるため，自律型キャラクターAI ではない．

「スパーシャルＡＩ」に対応する機能 … 攻撃中は，各，敵キャラクターが向かうポイントを決定する．また迷路の中を移動する経路を見つける．「位置検索技術」「パス検索技術」の最も簡単な段階である．ゲームステージがシンプルなため，単純な処理ではあるが，「スパーシャル AI」としての機能を有している．

このように，パックマンのゲームシステムを MCS-AI 動的連携モデルから捉え直すと，上記のように，３つの AI にそれぞれ機能を割り振ることができる．それぞれの AI の自律性が弱いが，この３つの AI の連携によって，パックマンのゲームデザインを表現することができる．以下，それぞれの AI について詳細を説明する．

9.1.1 『パックマン』のメタ AI 的機能

本項では『パックマン』におけるメタ AI に相当する機能を示す．『パックマン』における４匹のモンスターの連携攻撃の特徴は，前述したとおり「スピードレベル」「波状攻撃」（アタック・ウェーブ）「出現タイミング」の３つである．『パックマン』におけるメタ AI 的機能とは，ステージやプレイヤーの成績によって，この３つを変化させることである [142].

「スピードレベル」はプレイヤー（パックマン）の速度も決まるため，ゲーム全体のスピード感とバランスを決定する（Table 9.1）．あらかじめ各ステージの「スピードレベル」設定によって難易度を設定する．「波状攻撃タイミング」は「スピードレベル」に対応して定義される（Table 9.2）．ゲームプレイに緩急をつけるため，この Table 9.2 で定義される「攻撃」（Attack）と「休憩」（Rest）の交互のタイミングによって，モンスター４匹が波状攻撃（集団でいっせいに包囲する）をかける時間と，攻撃しない時間を交互に繰り返す．さらに，ゲームプレイ中は，どのタイミングでモンスターをステージに放つかを動的に決定する．その設定値は「出現タイミング」テーブルで定義する（Table 9.3）．このように『パックマン』のメタ AI 的機能は，「スピードレベル」によるゲームステージ設計，「出現タイミング」によるゲームの全体のダイナミクス，そして「波状攻撃」による中期的（十数秒）のダイナミクスなど，多重にコントロールされている．現代のメタ AI に先駆けた設計である．以下，このそれぞれ３つの要素について述べる．

スピードレベル

スピードレベルはパックマンとモンスターの速度を定義する．それぞれが状態に応じた速度が決められており，遅い方から速い方へ向かって，スピードレベル A，B，C，D がある．これは「スピードレベル」テーブルによって定義される（Table 9.1）．ステージ毎にスピードレベルが設定される．パックマンの状態は大きく二つあり，パワーエサを食べて無敵になっている時と，そうでない時である．また，何もない空間を進むとき，通路にあるエサを食べながら進むとき，パワーエサを食べているときである．モンスターは「通常時」「スパートした時」（残りエサが少なると加速する）「ワープロードを通過する時」（画面の左から右にワープできる通路をワープロードという）「パックマンがパワーエサを食べてイジケタ時」である．スピードレベルはそれぞれのステージの難易度を決める最も大きな要素である．

Table 9.1　『パックマン』におけるスピードレベル［93］

	スピードレベル			
スピード	A	B	C	D
22				
21			②	②
20			①	①
19		②		
18				
17	②	①		
16	①			
15				
14				
13				
12			イジケ	
11		イジケ		
10	イジケ		ワープ	ワープ
9		ワープ		イジケ
8	ワープ			
7				

※スピードは「1 フレームに 1 ドット進むスピードを 16，1 フレームに 2 ドット進むスピードを 32」として設定されている．

パックマンの状態定義［93］　　**モンスターの状態定義**［93］

パックマンの状態	
何も食べていない時	
エサを食べている時	
パワーエサを食べている時	
パワーエサを食べて逆転している時で何も食べていない時	
パワーエサを食べて逆転している時でエサを食べている時	
パワーエリを食て逆転している時パワーエサを食べている時	

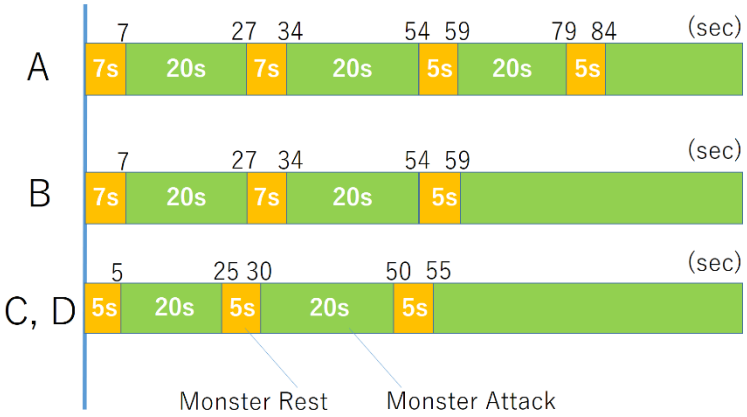
モンスターの状態	
通常時	
スパートした時	①②
パックマンがパワーエサを食べてイジケタ時	イジケ
ワープロード通過時	ワープ

波状攻撃

4 匹が連携してプレイヤーを一定時間包囲すると，そこから，それぞれに決められた四隅のホームポジションへ移動することで離散する．この包囲攻撃（Monster Attack）と離散（Rest，休息の意味であるが，実際は四隅に離散するのでこのように訳す）のリズムがあらかじめ決められた「波状攻撃タイミングテーブル」でコントロールされている（Table 9.2）．

Table 9.2　『パックマン』における波状攻撃タイミングーブル［93］

Speed Level

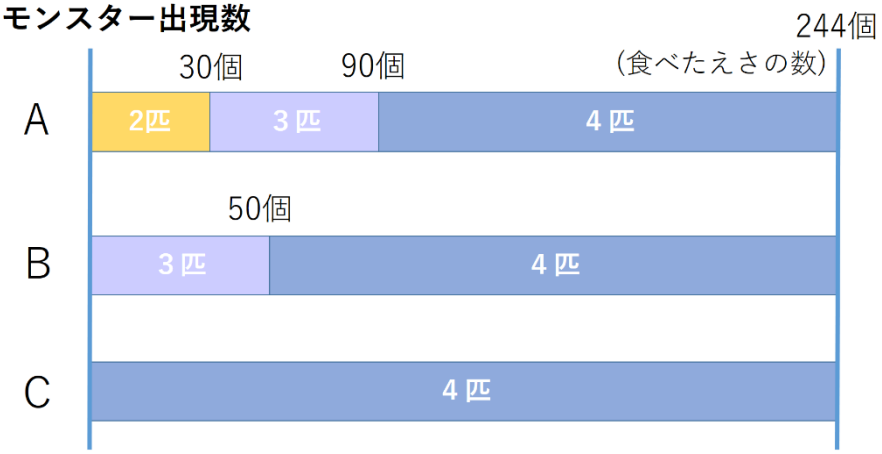


波状攻撃は 3 つのタイミングがあり，スピードレベル A，B，（C，D）に対して定義される．スピードレベル A に適用される波状攻撃タイミングテーブルは，7 秒の離散の後，20 秒のプレイヤーを包囲する動き，これが 2 回繰り返される．20 秒の攻撃のあと，今度は離散の時間を 5 秒に短縮し，再び包囲攻撃，そして 5 秒の離散，以降の包囲攻撃がなされる．レベル B に適用される波状攻撃タイミングテーブルは，7 秒の離散の後，20 秒のプレイヤーを包囲する動き，7 秒の離散，20 秒の攻撃，5 秒の離散，そして以降，包囲攻撃である．レベル C，D に適用される波状攻撃タイミングテーブルは，5 秒の離散的な周遊の後，20 秒のプレイヤーを包囲する動き，が 2 度繰り返される．その後，5 秒の離散，そして以降，包囲攻撃である．メタ AI からキャラクターAI へ渡される命令はモード型命令と言える．

出現タイミング

パックマンに登場する敵モンスターは常に 4 体であるが，常に 4 体がゲームステージ上に登場するわけではない．ステージ上に登場するモンスターの数によって，ゲームの難易度をコントロールする．プレイヤーの成績，ここではパックマン（プレイヤー）が食べたエサの数に応じて出現するタイミングをコントロールする．このパターンには 3 種類あり，難易度が低い方から高い方へ A，B，C であり，モンスターの出現タイミングテーブルによって定義される（Table 9.3）．これは A，B，C，… がスピードレベル A，B，C，…同じ記号を使っているが，お互い独立して各ステージにおいて定義されるものである．

Table 9.3　プレイヤーの成績によるモンスターの出現タイミング［93］



レベル A であれば，プレイヤーがえさを 30 個食べるまでは 2 匹，90 個食べるまでは 3 匹，それ以降は 4 匹がフルでゲームステージに出現する．レベル B であれば 5 0 個までが 3 匹，それ以降は 4 匹となる．レベル C であれば，最初から 4 匹すべてが登場している．

このようにパックマンは各ステージを「スピードレベル」，「モンスターの出現タイミング」，「波状攻撃タイミング」によって，各ステージの難易度と特徴を出しながら，プレイヤーによるゲームの進行に合わせて動的に変化させている．この変化は，あらかじめ決定された設定によって決定されているため，メタ AI の自律的判断ではない，という意味で，半自動的に決定される，と言える．この後，メタ AI は時代を経て，自律化されて行くが，パックマンはメタ AI の起源として捉えることができる．

9.1.2　『パックマン』のキャラクターAI・スパーシャル AI 的機能

本項では，『パックマン』のキャラクターAI・スパーシャル AI 的機能について述べる．敵キャラクターのモンスター 4 匹は，それぞれ動きに個性を持っている．キャラクターの動きの基本は波状攻撃であるが，「包囲攻撃」「離散」モードに分かれる．攻撃中は，それぞれの

モンスターはそれぞれ異なる動きをするように設定されている（Table 9.4）．アカは常にパックマンのいるマス（８ｘ８ドット）を追い，ピンクはパックマンの口先の３つ先のマスを目指し，シアンは赤モンスターのパックマンを中心とした点対称を目指し，オレンジはパックマンから半径約１３０ドットの外では赤モンスターの性格を持ち，半径内ではパックマンと無関係にランダムに動く．休憩中は，それぞれ決められた四隅のホームポイントに離散する．このようなキャラクターの目的地と目的地までの経路は，MCS-AI 動的連携モデルにおいては．『スパーシャル AI』に相当するが，『パックマン』はシンプルでコンパクトなステージ構成であるため，非常に軽い処理となっている．

Table 9.4　各キャラクターの目標地点と行動ポリシー［93］

モンスター	波状攻撃の状態	
	包囲攻撃	離散
 アカ	常にパックマンのいるマス（８ｘ８ドット）を追う	プレイフィールド上の右上付近を動き回る．
 ピンク	パックマンの口先の３つ先のマスを目指す	プレイフィールド上の左上付近を動き回る．
 シアン	赤モンスターのパックマンを中心とした点対称を目指す	プレイフィールド上の右下付近を動き回る．
 オレンジ	パックマンから半径約１３０ドットの外では赤モンスターの性格を持ち、半径内ではパックマンと無関係にランダムに動く	プレイフィールド上の左下付近を動き回る．

『パックマン』のキャラクターAI は外部から制御するスクリプティッド AI であり，またセンサーを有していない．その代わりとなるのが，プレイヤーの位置とプレイヤーの位置から相対的に定義される目標ポイントである．このポイントの計算とそこまでの経路の計算はスパーシャル AI に対応する．

9.1.3　考察：MCS-AI 動的連携モデルによる拡張案

本項では，MCS-AI 動的連携モデルを『パックマン』に適用した場合，いかなる変化をもたらすかを考察する．前節までに解説したように，『パックマン』のシステムは，各ステージにおける敵の挙動がまず設定され，ゲーム内においても時間とプレイヤーの成績（パックマンがエサを食べた数）によって展開がコントロールされている．この全体のシステムは「メタ AI」とみなすことができる．キャラクターAI，スパーシャル AI は，目標点への移動

に対してのみ機能している．またモンスター 4 匹は，4 匹同士の連携ではなく，メタ AI による統率によって連携しているように見せている．『パックマン』はメタ AI による精密なコントロールによってゲームデザインが支えられている（Fig. 9.1）．

『パックマン』は世界的ヒットをもたらしギネスブックに登録されたデジタルゲームの金字塔であり，ゲームデザインとして高度に完成された作品である．ステージの広さや形状がこのままであると，この実装で最適化されている．しかし，もし『パックマン』を拡張しようとすると，静的なパラメータに基づく半動的なメタ AI に比重を置いたモデルでは，すべてのテーブルを調整し直す必要がある．たとえば，ステージが自動生成される場合や，ステージが 3 D 迷路になる場合には，その都度，メタ AI のテーブルを作り直すだけでなく，キャラクターAI がステージの形状を認識し，動的に行動を形成し，モンスター同士が動的に協調して行く必要がある．MCS-AI 動的連携モデルを『パックマン』に適用することは，現在のパックマンに対応するだけではなく，ゲームデザインの拡張性を実現することでもある．

以下，MCS-AI 動的連携モデルを『パックマン』に導入すると仮定したときの，スパーシャル AI，メタ AI，キャラクターAI の拡張的機能を述べる．

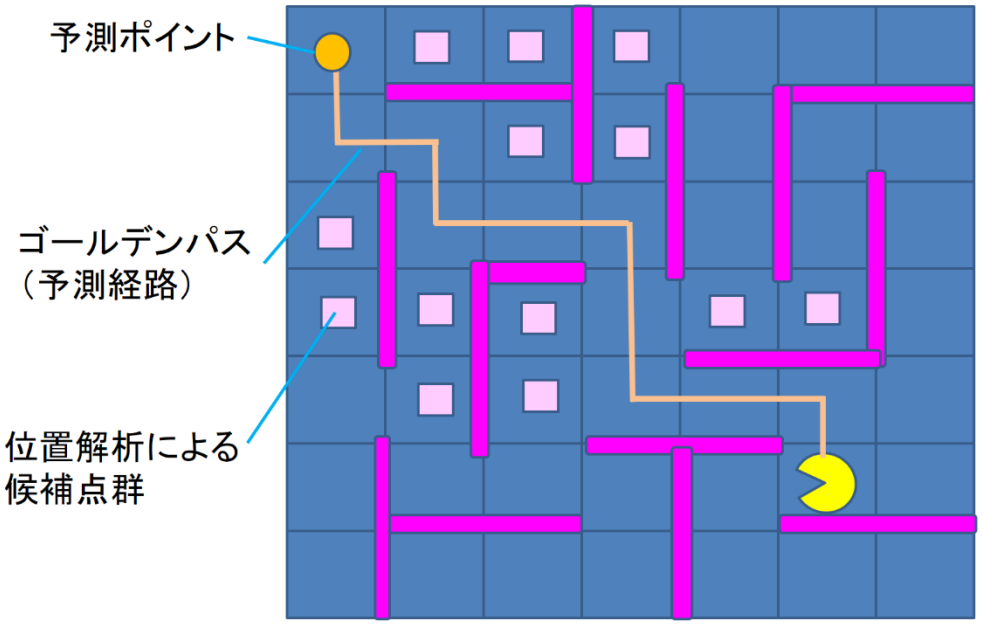


Fig. 9.2　『パックマン』の AI システムの拡張

(1) スパーシャル AI の拡張

スパーシャル AI は，各ブロックが連結された連結グラフとしてナビゲーション・データを持つ必要がある．第 8 章で述べたように，これによってステージ上のパス検索が可能となる．この準備によって，ステージがどのような形状であっても，汎用性のあるキャラクターAI，メタ AI を作ることが可能となる．つまり，『パックマン』のステージがどのように変

化しても，キャラクターAI，メタ AI を変化させる必要がなくなる．

またこのナビゲーション・データ上の影響マップは敵勢力の弱い場所を動的に探す方法を与え，プレイヤー（パックマン）の逃走先を推定に有効である．プレイヤー（パックマン）を認識した場所と時刻が複数あると，その場所を時間の順番をつなげると，次にどこへ向かうかが予測でき，予測に基づいた移動が取れるようになる．推定されるプレイヤーの目的ポイント（パワーえさ）などに対して，パス検索を行うことで得られるプレイヤーの予測経路は「ゴールデンパス」（Golden Path）と呼ばれる．この予測経路を囲うようにモンスターたちの目的地を「位置検索技術」によって決定する．また「影響マップ」によってモンスターが四散するときに向かうべき領域を動的に求め，メタ AI へ提供する．

(2) メタ AI の拡張

スパーシャル AI の拡張によってメタ AI はステージを動的に認識し，状況を予測することができるようになる．メタ AI はスパーシャル AI に，プレイヤーの目的予想ポイントと，それに伴う各モンスターの目的ポイントを求めるクエリーを発行し，その情報を受け取る．プレイヤーを攻撃するタイミングであれば，その情報に基づいて各モンスターに目的ポイントを伝え移動するように命令する．また逆にパックマンの逃げ道を完全に防ぐことがないように，包囲網に脱出可能な経路があるかどうかをスパーシャル AI にチェックを依頼する．脱出経路がない場合は，脱出経路ができるようなモンスターの配置の再計算をスパーシャル AI へ依頼する．ゲームの緩急に関しては，メタ AI がプレイヤーの緊張度を推定しつつ，モンスターに指示を与える．モンスターの退避先は，スパーシャル AI の提供する影響マップから求める．

(3) キャラクターAI の拡張

他律的キャラクターAI を自律型キャラクターAI へ拡張する．センサーによって自分の向いている方向の壁までにあるプレイヤーの位置とエサを認識させる．プレイヤーを最後に認識した位置と時間は「ラスト・スタンディング・ポイント」（Last Standing Point）と呼ばれ，キャラクターAI の意思決定で重要な役割を果たす情報であり，これを記憶させ，モンスター間で共有することで，プレイヤーの経路を予測し追跡に用いる．その記憶の内容の信頼度を時間と共に下げ，信頼度が一定値より下がると消去する．またメタ AI から指定された座標へ向かう命令を受けたときは，パス検索を行い指定の位置へ向かう．キャラクター同士の連携はいったんメタ AI を通して行う．また，敵キャラクターは完全にプレイヤーを追い詰めてはならない．そこで，メタ AI から「プレイヤーのために残しておくべき脱出経路」を受け取り，侵入しないように移動する．

『パックマン』は高度に完成されたゲームであるが，マップのスケールのコンパクトさゆえに，アルゴリズムレベルの工夫によってゲームデザインがコントロールされている．MCS-AI 動的連携モデルは，AI 同士の連携のレベルのモデルであり，アルゴリズムレベル

から AI 連携のレベルへ，ゲーム AI の品質を引き上げるものである（Table 9.5，Fig. 2.4）．このモデル MCS-AI 動的連携モデルの導入は，パックマンに拡張性を与え，ゲームデザインの拡張に寄与すると考えられる．

Table 9.5　MCS-AI 動的連携モデルを導入したときの変化

AIの種類	パックマン （オリジナル）	パックマン （MCS-AI動的連携モデル）
メタAI	「スピードレベル」、「モンスターの出現タイミング」、「波状攻撃による緩急」によって、各ステージの難易度と特徴を出す。各モンスターの行き先を指定する.	ステージの地形を認識し、パス検索と位置解析技術によって、各モンスターの行き先を指定する。また攻撃休止時に戻る領域に関しても、その都度、指定する。ユーザーの緊張度を推定しつつ、ゲームに緩急をつける.
キャラクターAI	メタAIから指示のあった位置に向かう.（非自律型）	パックマンを見た位置を記憶し、他のモンスターと共有し、自らパックマンを追い詰めつつ、最後までは追い詰めない意思決定を行う。メタAIから指示があった場合は、その位置にパス検索で移動する.（自律型）
スパーシャルAI	パックマンの位置を中心とした相対的位置を割り出し、目標地点としてモンスターのキャラクターAIに提供する.	パス検索機能が基本となる。さらに、プレイヤーの目標地点を推定し、そこから予測経路をパス検索で割り出し、位置検索技術で攻撃時の各モンスターの目標地点を決定する。「影響マップ」によってモンスターが四散するときの領域を動的に求める.
全体的な特徴	マップに最適化したアルゴリズム	ゲームの拡張に耐えうる総合的AIシステム

9.2　『クロムハウンズ』と MCS-AI 動的連携モデル

本節では『クロムハウンズ』に MCS-AI 動的連携モデルを導入した場合の可能性について論じる．第 9.2.1 項，第 9.2.2 項，第 9.2.3 項，第 9.2.4 項では現状の『クロムハウンズ』の AI について述べる [26] [80]．第 9.2.5 項では，MCS-AI 動的連携モデルを導入した場合の『クロムハウンズ』について考察する．

『クロムハウンズ』は，オンラインの３Ｄ空間内で，プレイヤー・チーム（最大６体）同士が戦うロボット・アクションゲームである．ロボットは地表を歩き，装備された武器（大砲，銃，爆弾など多数）によって相手や建造物を破壊することが出来る（Fig. 9.3）．ルールは 15 分以内に相手基地を爆破するか，相手チームを全滅させることが出来れば，勝利となる．敵基地は候補として３～４つの地点が指定されているが，ゲーム開始時にどれが本物かわからないため，探索して仲間と情報を共有する必要がある．基本は 6 対 6 のプレイヤー同士の対戦であるが，CPU 戦と呼ばれる AI チームと対戦するモードが以下の２つの目的で用意されている．

(i)　プレイヤーチームの対戦相手がいない場合に対戦相手になる．

(ii)　新人チームが研鑽を踏むための練習相手となる．

ここでは，このＣＰＵ戦に実装された人工知能を取り上げることとする．

Fig. 9.3　『クロムハウンズ』のゲーム画面 操作機体と通信塔

9.2.1　『クロムハウンズ』のゲーム AI モデル

本節では，クロムハウンズのオリジナルの AI システムについて述べる．この AI システムは「チーム AI」,「キャラクターAI」,「ナビゲーション AI」からなる．このシステムを TCN-AI 動的連携モデル（Team-Character-Navigation AI Dynamic Cooperative Model）と呼ぶ．以下，TCN-AI 動的連携モデルについて述べる [57] [26] [80].

クロムハウンズの NPC の 6 体のロボットは，最大 15 分，プレイヤーの 6 体のチームと 5 km 四方の 3D フィールドで戦わねばならない．キャラクターAI は自分で戦略ゴールを決定し，それを達成するためのプランを生成し実行する自律型エージェントとして設計された．戦略ゴールは各 NPC がユーティリティ・ベースで決定し，NPC はそれぞれの自律的に活動する．AI チームには，チーム全体に指示を出すチーム AI が存在する．チーム AI は，NPC に対して，戦略ゴールを命令することでコントロールする．チーム AI もまた，チームとしての戦略ゴールをユーティリティ・ベースで決定する．

ゴール指向の設計によって，チーム AI からキャラクターAI へ明確な意図を伝えることができる．チーム AI からキャラクターAI への干渉に強さは，ゲームの進行と共に強くなるように推移する仕組みが導入されている．ナビゲーション AI に関しては，敵，味方や本拠地．通信塔と言ったオブジェクトに対するパス検索のみが可能である．地形を戦術的に使う機

能を持たず，それが NPC の挙動を対オブジェクトに限定している．

そこで，TCN-AI 動的連携モデルのそれぞれの AI は以下のようになる（Fig. 9.4）．

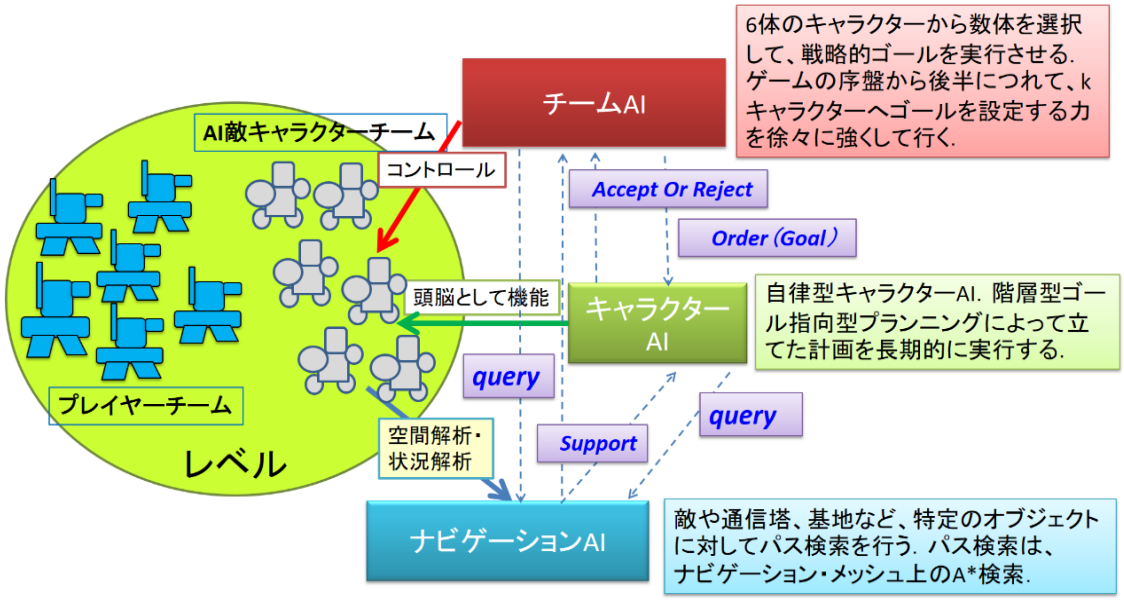


Fig. 9.4　『クロムハウンズ』における TCN-AI 動的連携モデル

チーム AI… 6 体のエージェントから数体を選択して，戦略的ゴールを実行させる．ゲームの序盤から後半につれて，各キャラクターへの干渉の力を徐々に強くして行く．

キャラクターAI … キャラクターはそれぞれ自律型キャラクターAI を持つ．それぞれ周囲をセンサーで把握し，自分の性能に応じたゴールを選択し，階層型ゴール指向型プランニングによって，自分で立てた計画を長期的（数十秒～２，３分）に実行する．

ナビゲーション AI… 敵や通信塔，基地など，オブジェクトに対してパス検索を行う「ナビゲーション AI」機能だけを持つ．しかし，それ以外の地形や空間を目標点とすることができない．ナビゲーションに関しては，地形に沿ってナビゲーション・メッシュを構築し A* アルゴリズムでパス検索を行う．地形に関しては崖や淵の境界のみナビゲーション・メッシュ上にマークの情報が載っている．

以下，それぞれの AI について詳細を述べる．

9.2.2　『クロムハウンズ』におけるチーム AI

本項ではクロムハウンズのメタ AI について述べる．開発初期に，対戦テストを通じて

個としてのＡＩの知能を高度化しても，プレイヤーチームに勝利することは難しい，という結果を得た，そこで，広いフィールドに戦力を分散するよりは，複数の NPC の戦力を一か所に結集してプレイヤーチームに攻撃することが必要となった．

連携した行動を取らせるために，NPC の戦略ゴールをセットにした「チーム戦略ゴール」を構成し，適切な NPC に戦略ゴールを割り当てるチームＡＩの仕組みを導入した（Fig. 9.5）．例えば，チーム戦略ゴール「敵基地攻撃」は，３つの「敵基地攻撃」戦略ゴールからなり，最も適切なメンバー（ここでは，敵基地に近く戦力のある NPC たち）を選択して割り当てる．ゴールを割り当てられた NPC は，現在実行中のゴールを再評価した値と，割り当てられたゴールの評価値を比較して，評価値の大きな方のゴールを選択する（つまり命令への拒否権を持つ）（Fig. 9.6）．これはチームの大局的な判断と NPC の局所的な判断を競合させるためである．「チーム戦略ゴール」は複数用意し，個の戦略ゴール決定と同様に，各ゴールに設定された評価式に従って最大の値を取るチーム戦略ゴールを採択する．

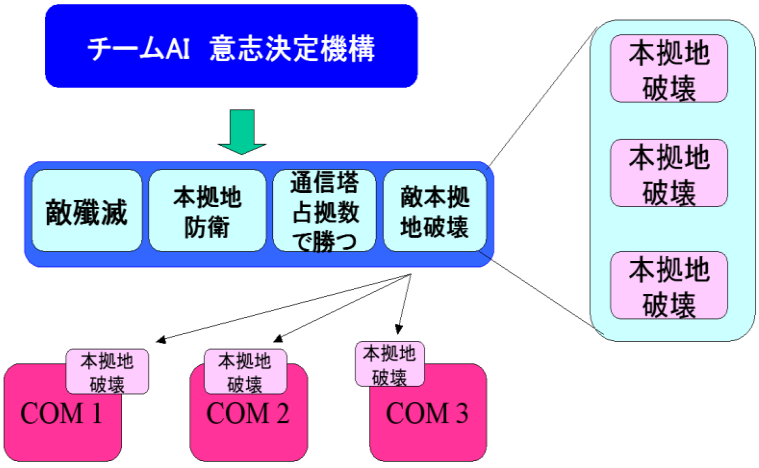


Fig. 9.5　チームＡＩによる戦略ゴールの割り当て［26］

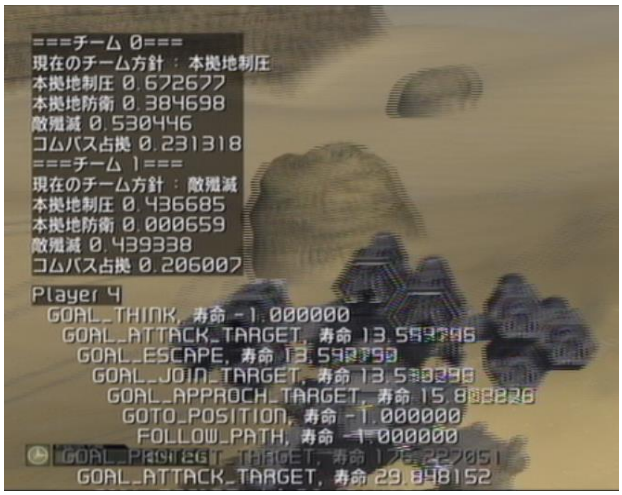


Fig. 9.6　チーム戦略に沿ったゴールを受け入れて行動する NPC

チーム AI はゲームの終盤で勝敗を決するタイミングで最も効果的であり，逆にゲーム序盤では各 NPC の自由な判断がゲームを面白くする．そこで，チーム AI が割り当てるゴールと NPC が実行中のゴールを比較する段階で，チーム側の評価値に変動ファクターをかけてチーム制御の強さを調整する．変動ファクターは序盤ではゼロ，中盤で序々に上昇，後半では 1.2 に固定し，チームＡＩの力が次第に強まるように調整した（Fig. 9.7）．この仕組みによって，これまで分散した戦力によって各個撃破されていた NPC は，終盤に戦力を結集することで「勝利へ向かう意志をプレイヤーに表現する」ことができるようになった．

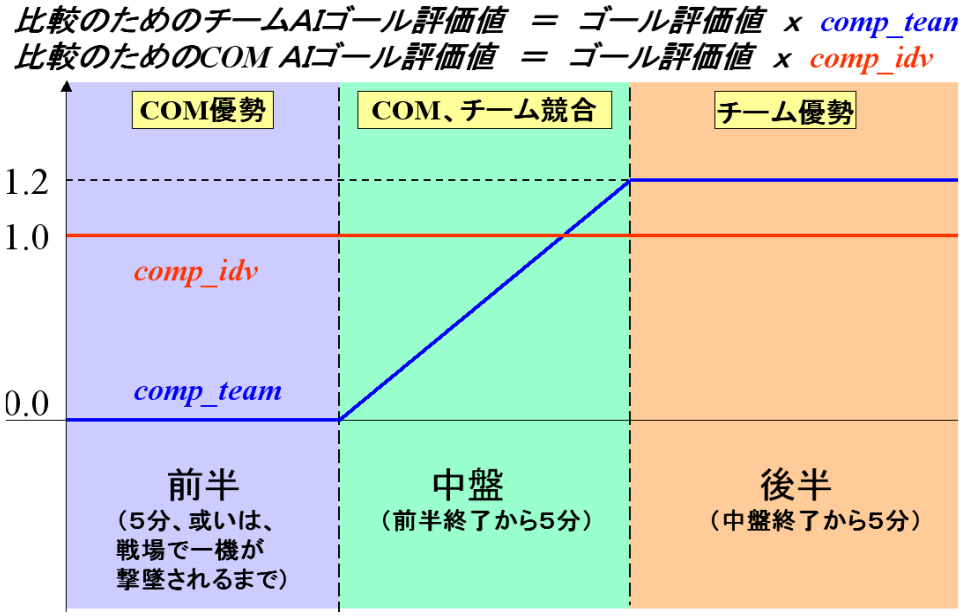


Fig. 9.7　ゲーム進行とチームＡＩの影響力の上昇［26］

9.2.3　『クロムハウンズ』のキャラクターAI

本項では『クロムハウンズ』におけるキャラクターAI の意思決定について述べる．『クロムハウンズ』では人間のプレイヤーチームと，NPC チームは対等な立場で戦闘を行う．そこで，プレイヤーから見て人間らしい知能に見えるように，自律型エージェントとして NPC を構築する，という指針を立て，全体のアーキテクチャとしてはエージェント・アーキテクチャを採用した．またプレイヤーの行動の観察から「通信塔を取る」「基地を確認する」「敵を攻撃する」など目的を一つ一つ実行していることが確認できたため，AI に対しても同様に大きな時間スケールで行動を形成できる階層型ゴール指向プランニングの技術を採用した．　階層型ゴール指向プランニングは，キャラクターにゴールを指定すると，その実行の仕方を思考して決定し実行するため，メタ AI からキャラクターへは戦略ゴールを指定するだけで良い．

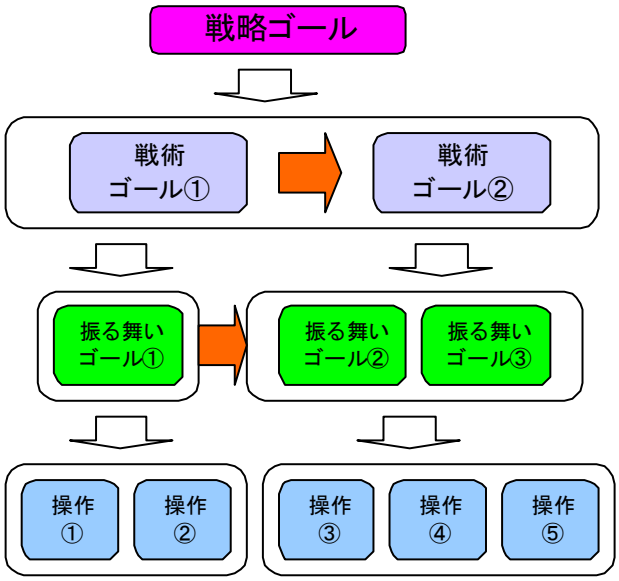


Fig. 9.8　階層的にゴールが分解されて行く模式図

階層型ゴール指向プランニングシステムとは，第 7.3.5 項で述べたように，まず達成するべき大目標（戦略ゴール）から出発して，その達成に必要なより小さなゴール（中間ゴール）に分解し，さらに，その中間ゴールを，最終的にはキャラクターの身体に対する単純なコマンドのシークエンスまで分解して行動を生成する方法である［26］（Fig. 9.8，Fig. 9.9）．ある程度想定される基本となるゴール（「移動」「戦闘」など）を実装した後は，プレイヤーチームと対戦を 100 回ほど継続し，対戦ごとに開発チーム内で討論を重ねながら漸近的にゴールを拡張した．最終的に，「戦略」「戦術」「振る舞い」「操作」4 階層からなる階層型ゴールを構築した．各ゴールはクラスとして実装し，スクリプトによって「ゲーム状態と内部状態」に応じた下位のゴールへの分解の仕方を定義する．各ゴールには達成時間を設定し，時間内に達成できなかった場合は「失敗」と見なす．戦略ゴールレベルの達成時間は 1 ～ 3 分であり，下位に行くほど，達成時間は短くなる．

最上層の戦略層のゴールの選択は，ユーティリティ・ベースの意思決定によって行う．それぞれの戦略ゴールには，リスクとリターンからなるユーティティ関数があり，その時点で最も高い効用を持つゴールを選択する．また，メタ AI がキャラクターに指令を与える時には，戦略ゴールを一つ指定する．キャラクターAI における階層型ゴール指向プランニングは，メタ AI の意図を明確に伝えることができるアーキテクチャである．

以下，それぞれの意思決定の詳細について述べる．

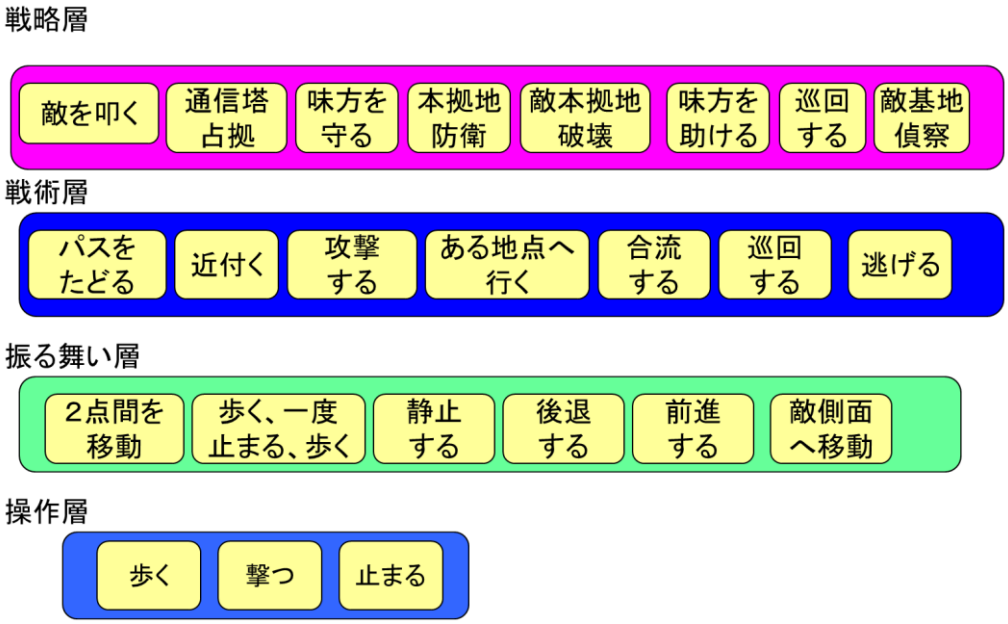


Fig. 9.9　クロムハウンズのゴール階層図（一部）［80］

戦略意思決定

ゴール指向の最上層である戦略ゴールには，それぞれ評価関数(ユーティリティ関数，第7.3.4 節参照)が設定されている．評価関数は，ゲーム状態とＡＩの内部状態（機体の攻撃力，防御力，残り体力）を変数とする関数であり，初等関数の組み合わせで一つ一つ作り込まれている．ゲームテストをくり返し，各戦略が戦況に応じて適切に競合するように微調整された．また機体のタイプによっては選択できない戦略を指定するフィルタをかけた（例えば「ディフェンダー」は「敵基地偵察」を行うことはない）．意思決定は，最大評価値を持つ戦略ゴールを選択することで行う．意思決定のタイミングは，

① 現在実行中の戦略ゴールを達成したとき
② 現在実行中の戦略ゴールを失敗したとき
③ チームＡＩから意思決定を促されたとき
④ 戦況が変わったとき

である．

短期意思決定

上記の戦略意思決定は，１～５分に渡る長い時間スケールに対応した意思決定であるが，例えば「曲がり角で敵と遭遇する」場合など，短期的に戦闘状態に入ることが多い．この場合，戦略ゴールを撤回して「敵を叩く」戦略を選択していては，実行中の戦略は頻繁にキャンセルされてしまう．そこで，戦略実行中でも「敵を叩く」ゴールを一時的に挿入し，一定時間（15 秒）の後クリアし，もとの戦略に沿った行動に復帰するようにした．この仕

組みによって，プレイヤーの攻撃に対してロバストな意思決定システムを実現した．さらに「射程距離に敵を捉えた場合は撃つ」「パイプ・砲台は射程距離に捉え次第撃つ」など，さらに短期的で反射的なアルゴリズムを実装した，このように，時間スケールに合わせた多段階の意思決定によって多様な時間スケールに対応するＡＩを構築した．

またＮＰＣ同士が相互通信可能領域内に入った場合，最新の敵情報が相互に交換される．このように個体のみならず，通信によってチーム全体の認識能力を拡大している．

また，「斥候」タイプの高速な機体は，敵基地の真偽を探索する役割を持ち，情報を味方の通信領域内に持ち帰って共有する．チーム全体として，簡単なナレッジ・マネージメント機能（「４つのうち３つが偽者なら残りが本物である」など）を持つ黒板モデルの共有メモリを持ち，収集した情報から推論して敵の真の本拠地を決定する．

センサーは各瞬間に敵の位置，速度ベクトル，敵の強さを収集し，認識は敵の脅威度，意図に変換する．脅威度とはその敵の自分に対する脅威度であり，遠ざかって行く強い敵と，近づいて来る弱い敵の脅威度は，後者の方が高くなるように計算する（Fig. 9.10）．

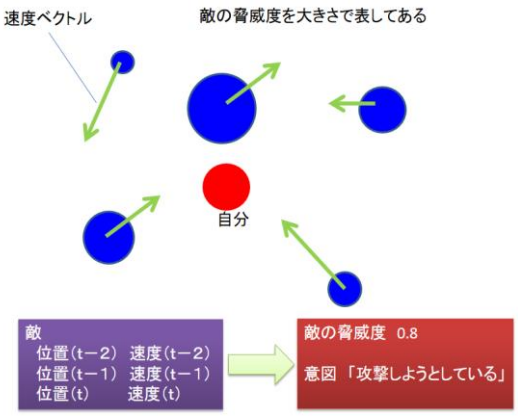


Fig. 9.10　敵の位置と速度の記憶から，敵の脅威度や意図を算出する．

9.2.4　『クロムハウンズ』におけるナビゲーション AI

本項では『クロムハウンズ』におけるナビゲーション AI について述べる．『クロムハウンズ』におけるナビゲーション AI は，パス検索のみのナビゲーション AI である．NPC の移動はナビゲーション・メッシュ上の A*アルゴリズムによるパス検索によって行う．ナビゲーション・メッシュデータは，当たりモデルから，建物の周囲など，細かい当たりモデルをメッシュのまま残したい部分にフィルタをかけ境界を保護しつつ，ポリゴンリダクション関数を用いてオリジナルのツール上で自動的に生成する．全 80 マップであり，各４～８万メッシュである．

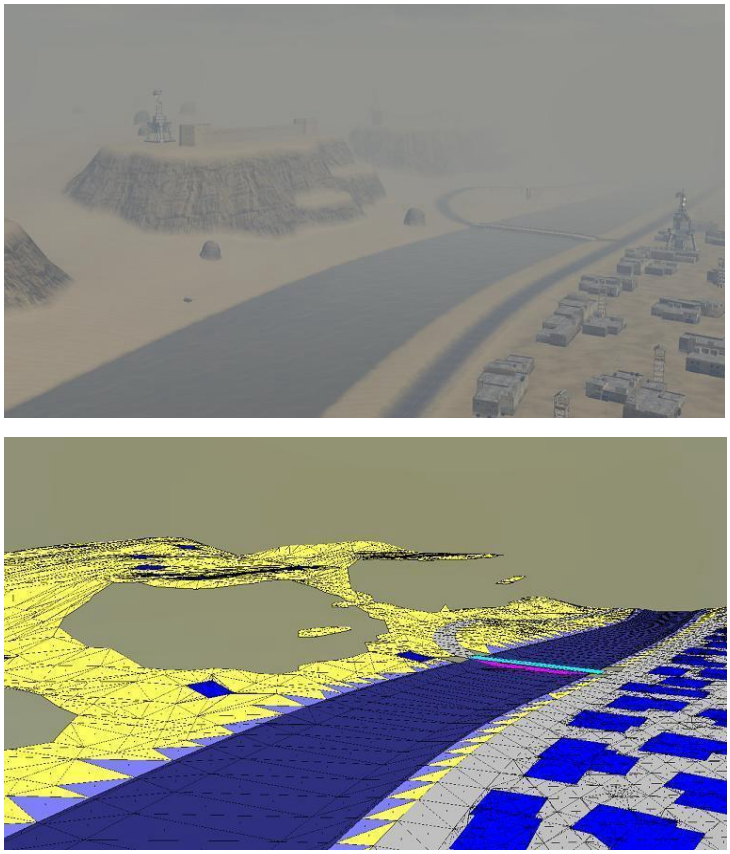


Fig. 9.11　描画マップ（上）と地表情報が埋め込まれたナビゲーション・メッシュ（下）（黄色は砂，紺色は水，青は建築，ピンクは橋の下，水色は橋）

各メッシュには生成の過程で，モデルデータから「地表状態」（砂，水，雪，コンクリート，土），地形解析から「トポロジー状態」（崖の淵，谷の淵）の情報を取得して埋め込み，A*検索においてコストとして換算する．例えば「水」「雪」「砂」などは機体の移動速度を減少させるため，同様に崖の淵は機体が引っ掛かりやすいためメッシュ上の移動コストを高く設定する．これによって A*パス検索は最短距離検索ではなく最短時間検索を計算することになる．例えば，最短パスでは「足を取られながら河を横切って戦闘」をするが，最短時間移動では「自然にコンクリートの橋を渡って移動する」ようになり，プレイヤーに対するＡＩの知性のアピールが大きい．また建築物についてもメッシュ生成過程においてオブジェクトとメッシュの衝突計算を実行し，建築物の下にあるメッシュを建築物の形状に沿って細分化する形で新しくメッシュを生成し，NPC が通行不能であるというフラグ情報を埋め込む（Fig. 9.11）．ゲーム内では「建築が破壊されるとその下にあるメッシュのフラグが解除される仕組み」になっており，ＡＩは動的なオブジェクト破壊に対応して新しくパスを見出す能力を持っている．

パス検索は単に移動能力を獲得するだけでなく，様々な活用法がある．例えば「味方（敵）

と合流する」ゴールはその中で定期的にパス検索をくり返すことで，味方へ漸近的に近付くアルゴリズムを実現する．また，デバッグにおいてナビゲーション・メッシュの接続状態をチェックするために利用することが出来る．

デバッグと多重アルゴリズム

リアルタイムパス検索のデバッグは，使用するメッシュ全てのメッシュの組み合わせをテストする必要があるが，「４～８万メッシュ×８０マップ」を全テストすることは不可能である．そのため，基地間の経路，通信塔間の経路など，頻繁に使用される経路のデバッグを集中的にくり返した（半自動チェックで 3000 回以上）．また，それ以外の経路ついては，万が一，パスを辿れなかった場合の補償策として以下のアルゴリズムで用意した．

(1) オブジェクトに引っ掛かった場合には，少し方向を変えて（後ろ，左，右）進行し直す．
(2) 谷に落ちた場合は現在行っている戦略ゴールをリスタートとする．
(3) 現在選択している戦略ゴールの達成時間をオーバーすると「ゴール失敗」となり新しい戦略ゴールが意思決定される．

ゲーム内ではプレイヤーにパスを阻害される（谷へ落とされるなど）ことも頻繁であり，上記の補償策はそこでも大きくＡＩの移動をロバストなものにする効果を持つ．

9.2.5 考察：MCS-AI 動的連携モデルによる拡張案

本節では，MCS-AI 動的連携モデルを『クロムハウンズ』に適用した場合，いかなる変化をもたらすかを考察する．以下，スパーシャル AI，メタ AI，キャラクターAI の拡張を述べる．

(1) スパーシャル AI の拡張

『クロムハウンズ』のゲーム AI システムにおけるナビゲーション AI は「あるターゲットへの移動」がすべての行動の基本となっている．敵座標，通信塔・敵基地など目標点への移動に限定されていた．この制限が，そのままキャラクターAI の制限となり，キャラクターAI は，どのターゲットへ移動するか，がユーティリティ・ベースの評価関数の基本となっている．

ナビゲーション AI からスパーシャル AI へ拡張を行うためには．まずマップ全域の地形的特徴を用いるために，位置検索技術の導入が必要である．さらに「通信塔・敵基地だけの世界表現」「敵位置だけの世界表現」など，多様な世界表現により，より高度な戦術思考がキャラクターAI にはできる．そこで具体的には，世界表現として以下の表現を加える．

基地・通信塔基地マップ表現

キャラクターAI が次の戦略を決めるときには，主に基地や通信塔の近さと重要度を評価していた．ただその近さはユークリッド距離であるため，実際に道のりとして近いかどうかにかかわらず決定されていた．そこで，「基地・通信塔の道のり距離が表現されたマップ」（Fig. 9.12）があれば，戦略決定の時に，一連の通信塔をめぐるパスや，基地をめぐる経路戦略に組みことができ，より高度な戦略を決定できる．

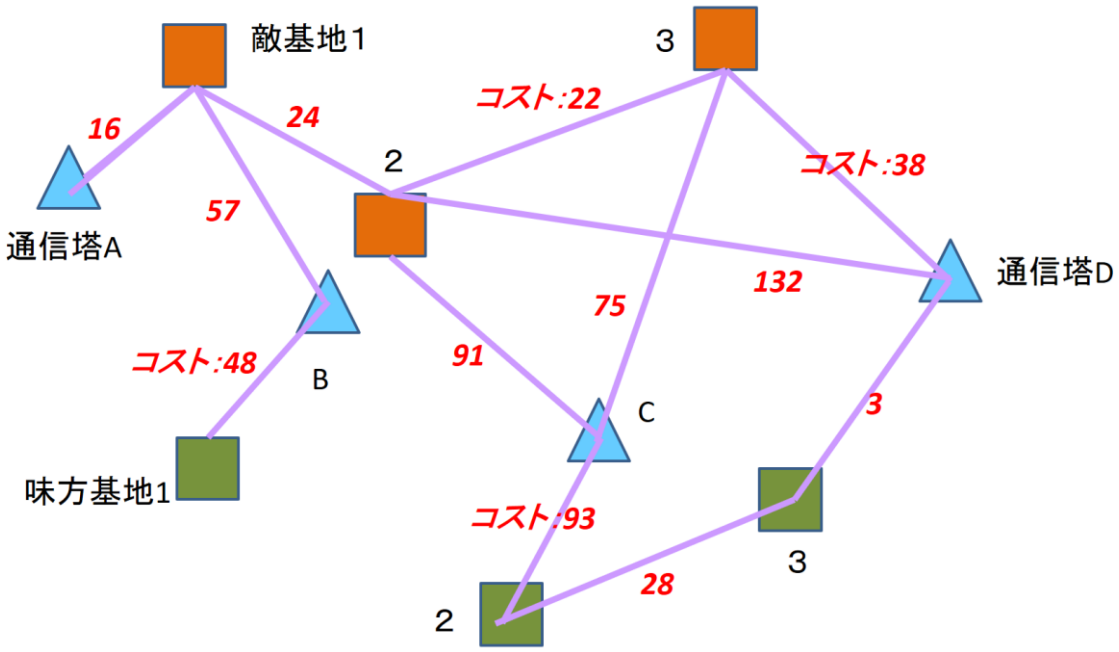


Fig. 9.12　基地・通信塔の道のり距離が表現されたマップ

敵基地推測機能

『クロムハウンズ』では敵基地の候補がいくつかあるが，どの基地が本物であるか，ゲーム開始時は不明である．そこで敵の出現ポイントから敵基地を推測せねばならない．上記の「基地・通信塔の道のり距離が表現されたマップ」があれば，敵の出現位置の統計情報から，敵基地を推測することが可能である．

逆空間推定

敵へのパス検索をして敵に近づき攻撃する，以外に，たとえば崖の上から敵を攻撃する，という戦術がある．そのためには専用の空間表現が必要である．敵の目的地を推定し予想パスを推測すること，その経路上を攻撃できるポイントへ移動すること，である．前者は「基地・通信塔の道のり距離が表現されたマップ」から推定することができる．後者は，ある空間と，そのポイントを攻撃できる領域にリンクさせた「攻撃領域-被攻撃領域」世界表現が必要である（Fig. 9.13）．

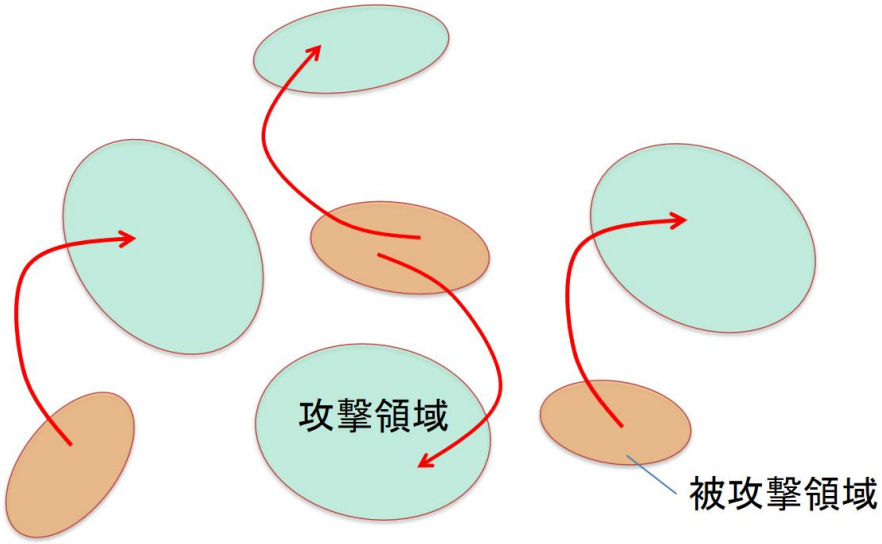


Fig. 9.13 「攻撃領域-被攻撃領域」世界表現

『パックマン』は均質なステージであるが,『クロムハウンズ』のステージは山や平原,湖や丘などバリエーションに富む．このような目的地を広大な空間から自由に選択し活用することができるようになれば，その場のその地形の特徴を使った多様性のある戦闘を作ることができる．

地形に沿った影響マップ

影響マップによって戦局全体を把握する．しかしグリッド状の影響マップは地形を反映することができない．そこで必要になるにが，地形に沿った影響マップである．あるグリッドが侵入できない山脈などで二つに分裂する場合など，その場に関係のない情報がグリッドに反映されてしまう．そこで地形に沿ったグリッドの構築が必要である．

スパーシャル AI による世界表現の拡大は，メタ AI，キャラクターAI の思考の土台を広げるものである．キャラクターAI の思考の構築の可否は世界表現の準備にかかっている．上記のような世界表現を準備することは，チーム AI とキャラクターAI の戦略層のみならず，戦術層，振る舞い層，操作層（Fig. 9.9）における各要素をより増加させ，戦略はより高度な戦略を，アクションはより精緻なアクションを取ることができるようになる．

(2) メタ AI の拡張

『クロムハウンズ』のチーム AI は広いフィールドで分散する戦力を，時間や機会によって離散・集合するメタ AI でもあった．しかし，メタ AI としての機能としては，勝利よりも，面白い戦局を演出することである．そこで，単に戦力を敵にぶつけるのではなく，より面白い戦局になるように，場を演出する力がメタ AI には必要である．そのため上記のよう

に，ナビゲーション AI はスパーシャル AI に発展させる必要がある．

たとえば，メタ AI は，敵基地を推定し，そこから自軍の基地へ至るルート（ゴールデンパス）をパス検索で算出する．算出されたパス検索から位置検索技術を用いて，待ち構える場所を決定し，そこに自キャラクターへ向かうように指示を出す．「おとり」となる自キャラクターを敵基地周辺から自基地へゴールデンパスに沿って帰還させ，プレイヤー機達を誘導し迎え撃つ，と同時に背後から奇襲させる，などが可能となる（Fig. 9.14）．

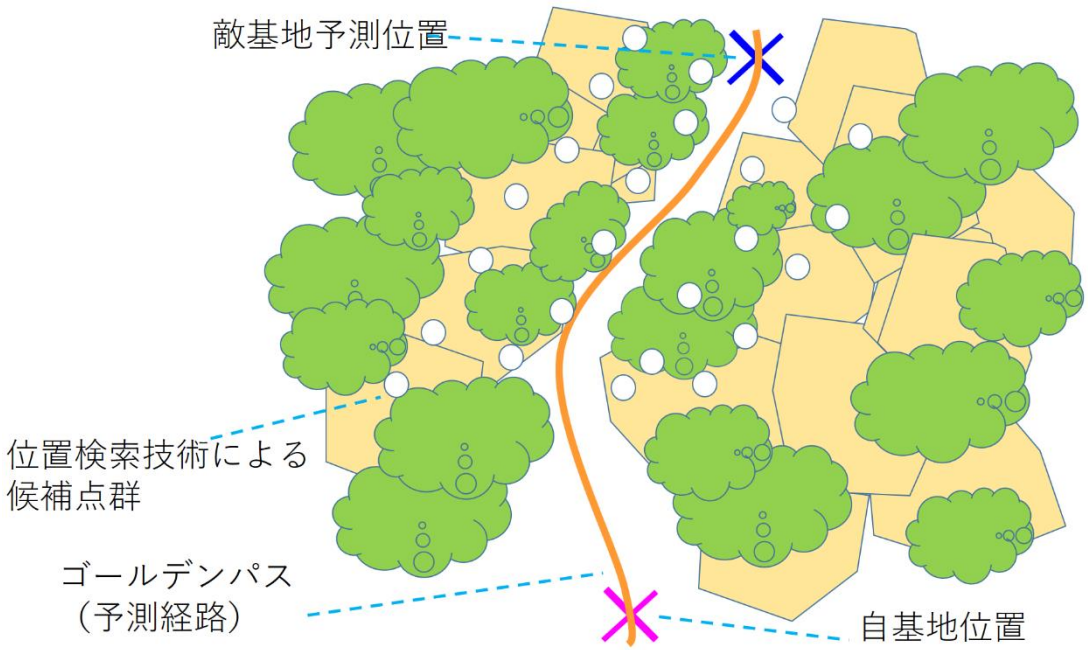


Fig. 9.14　『クロムハウンズ』の AI システムの拡張

このように，メタ AI は，スパーシャル AI の準備する世界表現の上に，地形情報を使った思考によって，その場所の特徴を活かした展開を創ることができる．

(3) キャラクターAI の拡張

キャラクターAI のゴール指向プランニングの長所と短所は，最上層の戦略ゴールを一つ選び，そのゴールに向かって行動プランを形成することにある．階層型ゴール指向の最上層の戦略を決定する思考は，常にその時点の戦局に応じて行われた．この手法で欠けているのは，相手（プレイヤーチーム）の動きの予測である．敵の予想と共に，行動を組み立てることができれば，予測経路「ゴールデンパス」のように，メタ AI と共に面白い戦局を作り出すことができる．

敵チームの動きを予測するには，敵キャラクターの知能モデルが必要である．敵キャラクターの知能モデルは，NPC の知能モデルを使うという手法がある [79]．つまり，敵を AI として，NPC をプレイヤー・キャラクターと同じ位置，同じ状況に置いた時に，

どのような行動をするかを，その時点の状態からシミュレーションを動かすことで予測を行う．シミュレーションが開発過程の中でより正確にプレイヤーチームの動きに一致するようになれば，より正確な予測を行うことが可能となる．敵を正確に予測できるようになれば，NPC 側の思考をより正確に作ることができる．

『クロムハウンズ』は，チーム AI，キャラクターAI，ナビゲーション AI の組み合わせを方針に設計された TCN-AI 動的連携モデルだった．MCS-AI 動的連携モデルは，その後の発展によって成立したモデルであり，『クロムハウンズ』の AI システムと比較することで，はっきりとその差分を知ることができる．MCS-AI 動的連携モデルは，ゲームデザインをコントロールする効果を持つモデルであり，それは「戦闘の創出」から「面白い戦局の創出」へと本ゲームを発展させる．物理的レベルの開発から，ユーザー体験・レベルの開発へと，開発の質を変化させる（Table 9.6）．

Table 9.6　MCS-AI 動的連携モデルを導入したときの変化

AIの種類	クロムハウンズ	クロムハウンズ （MCS-AI動的連携モデル）
メタAI	「チームAI」として、各NPCに戦略ゴールを与えることで、チームの戦力の集中・連携を行う。ゲームの序盤から終盤にかけて、チームを制御する影響力を上げて行く.	左記に加えて、地形と戦況に対する認識を、スパーシャルAIからの情報によって形成することで、「面白い戦局」を作り出すために、NPCに戦略ゴールを割り振る.
キャラクターAI	階層型ゴール指向型プランニングによる自律型AI	シミュレーション・ベースによる敵予測を加えた階層型ゴール指向プランニング
スパーシャルAI	パス検索	経路予測・位置解析技術・影響マップ 基地・通信塔基地マップ表現
全体的な特徴	戦場の創出	面白い戦局の創出

9.3　考察

本章では，MCS-AI 動的連携モデルを，『パックマン』と『クロムハウンズ』へと適用することで，『パックマン』の開発（1979-1980 年）の時代には AI とは言われていなかった技術を，MCS-AI 動的連携モデルのもとで，メタ AI，キャラクターAI，スパーシャル AI として抽出した．また，そこから逆に，メタ AI，キャラクターAI，スパーシャル AI を拡張することで，どのように『パックマン』を拡大しても対応できる，パックマンにおける汎用的な AI システムを得た．

続いて『クロムハウンズ』は，TCN-AI 動的連携モデルであるが，MCS-AI 動的連携モデルへとそれぞれの AI を拡張することで，「敵とエンカウントして戦闘する」ためのシステ

ムから,「地形を理解し, 敵の動きを予想し, プレイヤーを楽しませるための戦略を立てる」システムへと発展できる可能性を見いだすことができた. そのために必要な機能は, 空間的にはスパーシャル AI の世界表現の拡張, そして時間方向にはプレイヤーチームの運動の予想である.

このように, MCS-AI 動的連携モデルを過去のタイトルに当てはめることで, 以下の 3 点を検証することができた.

(1) MCS-AI 動的連携モデルの中で, そのゲームの構造をあらためて捉え直すことできる.
(2) MCS-AI 動的連携モデルとして AI を構築することで, ゲームデザインを拡大できる
(3) MCS-AI 動的連携モデルとして AI の機能として技術を抽出できる.

今後は, この成果を用いて, 他のさまざまなゲームタイトルを対象に MCS-AI 動的連携モデルを当てはめることで, そのゲームの構造を捉え直し, 3 つの AI の機能として技術を取り出し, さらにそのゲームデザインを拡張する可能性を見出して行く (Fig. 5.6). この試みは, 技術とゲームデザインの研究を並行して進めると同時に, 技術とゲームデザインの要素を MCS-AI 動的連携モデルの枠組みの中で蓄積して行くことができ, そのストックは今後のゲームデザインとそのゲーム AI システムに役立つことになる.

このようにさまざまな要件に出会うことで, それらは「メタ AI」「キャラクターAI」「スパーシャル AI」へと分解される. その分解の仕方は一つではない. また, 蓄積されて行く各技術は, 各 AI の中でも膨大なものとなり, やがてそれぞれの AI の中で体系を持つことになる. 特にメタ AI の定義については, ゲーム開発の歴史の中で発展・変化してきた歴史があり, 今もなお揺らぎつつ発展している [2].

10. MCS-AI 動的連携モデルがもたらすゲームデザインにおける影響の比較検証実験

MCS-AI 動的連携モデルがゲームデザインに関してどのような変化もたらすかを検証するために，ゲーム開発者に対して自由回答文のアンケートによる比較検証実験を行った．本章はこの実験の手法と結果と考察を述べる．第 10.2 節では実験方法について示す．第 10.3 節では実験で得られたアンケートの解析結果を示す．第 10.4 節では，本実験の結論と考察を示す．

10.1 比較検証実験の目的

本比較検証実験の目的は，MCS-AI 動的連携モデルがゲームデザインに対して，どのような変化をもたらすか，を調査することにある．MCS-AI 動的連携モデルを知った被験者のゲームデザインのアイデアを，知らせなかった被験者と比較することで，MCS-AI 動的連携モデルによるゲームデザインのアイデアへの影響を調査する．

10.2 比較検証実験の手法

本節では，検証手法について述べる．被験者は解説文を読んだあと，自由記述式アンケートに回答を記述する．回答用紙に空欄は 5 つ用意されているが，ファイル内で自由に拡張して回答を増やしても良いという説明がなされている．解説文は，MCN-AI 連携モデルの解説文（第 16.1.1 項に記載）と MCS-AI 動的連携モデル（第 16.1.2 項に記載）が用意されている．被験者に対する，2 つのアンケートパターン（MCN-MCN パターン，MCN-MCS パターン）が用意されている．アンケート記入はアイデアを日本語で記述する．異なるアイデアは番号をつけて分けて記述する．被験者は回答シート A（第 16.1.3 項に記載），回答シート B（第 16.1.4 に記載）に記述する．

アンケートパターンは以下の 2 種類である（Fig. 10.1）．

MCN-MCN パターン

まず MCN-AI 連携モデルを 5 分読み，回答シート A に『パックマン』についての新しいゲームデザインのアイデアを 20 分記述する．その後，再度 MCN-AI 連携モデルの解説文を 5 分読み，回答シート B に『パックマン』についての新しいゲームデザインのアイデアを 20 分記述する．

MCN-MCS パターン

まず MCN-AI 連携モデルを 5 分読み，回答シート A に『パックマン』についての新しいゲームデザインのアイデアを 20 分記述する．その後，MCS-AI 動的連携モデルの解説文を 5 分読み，回答シート B に『パックマン』についての新しいゲームデザインのアイデアを 20 分記述する．

両者のパターンで被験者が記述したゲームデザインのアイデアを比較する．

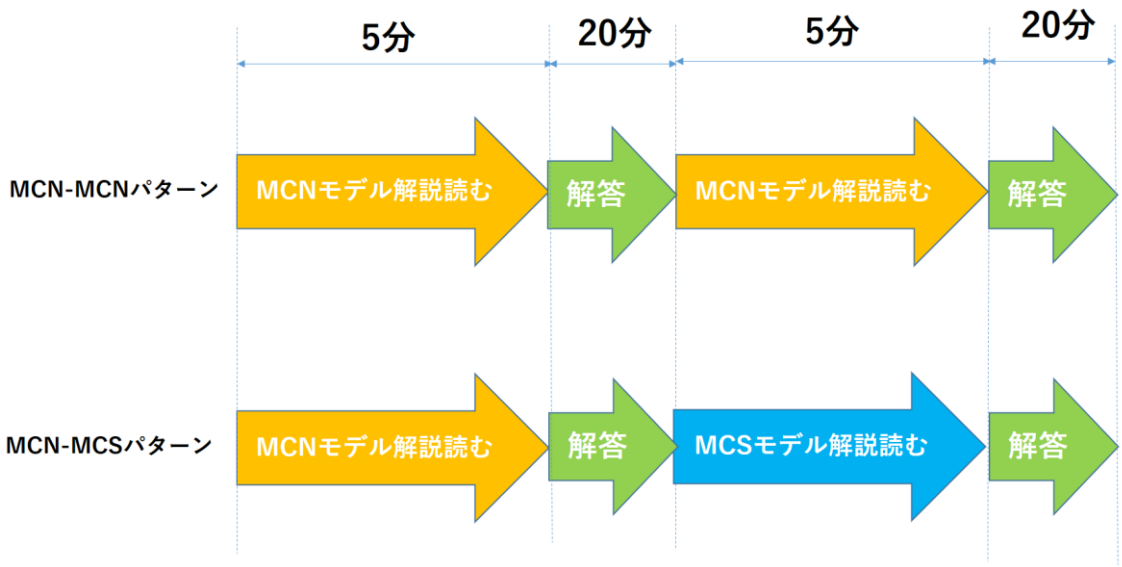


Fig. 10.1　2 つのアンケートパターン

予備実験の期間は 2020 年 5 月から同年 7 月で被験者は 5 名である．最初の 2 名はそれぞれ，MCN-AI 連携モデル，MCS-AI 動的連携モデルのみの説明を行い，アンケートを取った．残りの 3 名について MCN-MCN パターンが 1 名，MCN-MCS パターンが 2 名のアンケートを取った．本実験の期間は 2020 年 7 月から同年 10 月，被験者は MCN-MCN パターンが 13 名，MCN-MCS パターンが 13 名である．すべてインターネットを通じた映像インタビューを通じて行った．被験者はゲーム開発の経験を持つことを条件としたが，開発経験と職種については Table 10.1 の分布となった．第 10.3.11 項では職種ごとのアンケート結果の定性的解析を示す．

10.3　アンケート結果解析

本章では，アンケートによって得られたデータを以下の方針に従って解析する．まず定量的な 3 種類の解析を 3 つ行う．第 10.3.1 項では，2 つのアンケートで回答数を比較する．MCN-MCS パターンの後半の回答数が多ければ，MCS-AI 動的連携モデルがより多様なアイデアを生み出す原因となっていることを示すことができる．第 10.3.2 項では前半，後半

のキーワード頻度数の比較を行った．第 10.3.3 項では前半の後半のキーワードの共起ネットワークの比較を行った．どのようなキーワードが連環しているかを比較することで，アイデアの発想の傾向を調べる．第 10.3.4 項では MCN-MCN，MCN-MCS パターンの前半のキーワード頻度数の比較を行った．第 10.3.5 項では MCN-MCN，MCN-MCS パターンの前半の共起ネットワークの比較を行った．第 10.3.6 項では MCN-MCN，MCN-MCS パターンの後半のキーワード頻度数の比較を行った．第 10.3.7 項では MCN-MCN，MCN-MCS パターンの後半の共起ネットワークの比較を行った．第 10.3.8 項では，アンケートで記述された文章の中で 4 つのキーワード「メタ AI」「キャラクターAI」「ナビゲーション AI」「スパーシャル AI」の使用頻度を調査することで，記述された文章がどれぐらい提示した AI モデルと関連を持っているかを検証する．第 10.3.9 項ではナビゲーション AI を含んだアイデアとスパーシャル AI を含んだアイデアの数を比較する．第 10.3.10 項では，「スパーシャル AI」と関連したアンケート記述からキーセンテンスを抜き出し，多様なアイデアを生み出しているかを判定する．アンケートの自然言語文解析には KH Coder [143] [144]を用いた．第 10.3.11 項では職種ごとのアイデアの定性的解析を行った．

10.3.1 アンケート回答数の比較

MCN-MCN，MCN-MCS パターン両者の回答数を比較する．まず全体の傾向として，前半の平均が 3.9 例，後半の平均が 4.2 例であった．MCN-MCN パターンにおける前半回答数の平均が 3.7 例，後半回答数の平均が 4.7 例であった．MCN-MCS パターンにおける前半回答数の平均が 3.6 例，後半の平均回答数が 3.5 例であった．したがって前半に比べて後半の回答数の比は 1.1，MCN-MCN パターンが 1.27，MCN-MCS パターンが 0.97 であり，後半の回答数が MCN-MCS パターンで大きく増加しているとは言えない結果となった（Table 10.1）．またいずれの場合も中央値は 4.0 例であった（Fig. 10.2）．

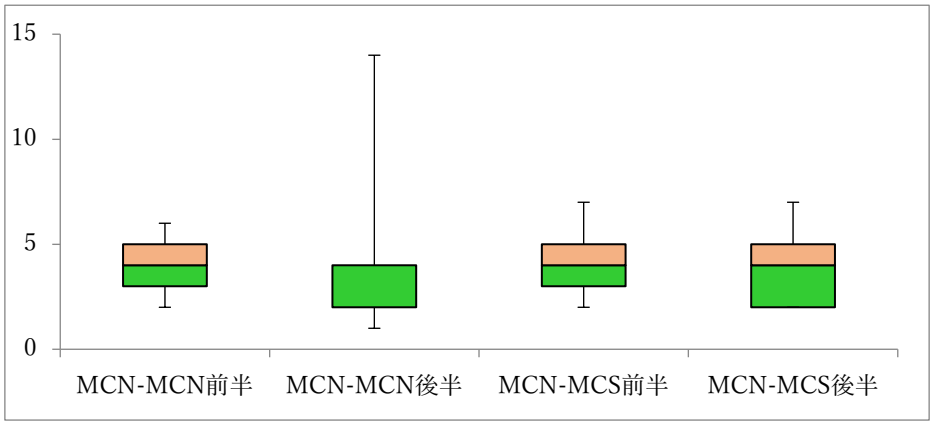


Fig. 10.2 アンケートにおける一名当たりのアイデア数の比較

Table 10.1 アンケートパターンと前半，後半のアイデア数，プロファイル

No.		解答数（前半）	解答数（後半）	ゲーム開発経験	職種
1	MCN-MCS	5	5	3年以上〜	アーティスト
2	MCN-MCS	4	3	3年以上〜	プランナー
3	MCN-MCN	2	1	3年以上〜	エンジニア
4	MCN-MCN	5	5	3年以上〜	プランナー
5	MCN-MCN	2	2	3年以上〜	エンジニア
6	MCN-MCN	3	2	3年以上〜	エンジニア
7	MCN-MCS	3	2	3年以上〜	エンジニア
8	MCN-MCS	3	2	1年以内	エンジニア
9	MCN-MCS	3	2	3年以上〜	その他
10	MCN-MCS	3	4	3年以上〜	エンジニア
11	MCN-MCS	5	6	3年以上〜	エンジニア
12	MCN-MCS	7	7	3年以上〜	その他
13	MCN-MCN	5	4	3年以上〜	その他
14	MCN-MCS	4	5	3年以上〜	エンジニア
15	MCN-MCN	2	2	3年以上〜	プランナー
16	MCN-MCN	4	4	1年以内	アーティスト
17	MCN-MCN	4	4	1年以上〜3年以内	エンジニア
18	MCN-MCS	4	3	1年以内	エンジニア
19	MCN-MCS	4	4	1年以内	その他
20	MCN-MCN	4	4	1年以内	エンジニア
21	MCN-MCN	5	3	1年以上〜3年以内	プランナー
22	MCN-MCN	4	14	1年以内	プランナー
23	MCN-MCN	6	13	1年以内	エンジニア
24	MCN-MCN	5	3	1年以内	その他
25	MCN-MCS	5	5	3年以上〜	エンジニア
26	MCN-MCS	2	2	3年以上〜	エンジニア

10.3.2 前半と後半のキーワード頻度数比較

アンケート記述文からキーワードを抽出し出現数の比較を行った．まず前半と後半におけるキーワードの出現数を比較した（Fig. 10.3）．被験者 26 人の前後半の回答から形態素解析の上，抽出を行った．前半後半とも上位 12 キーワードに関しては，順位の差異はあるが，ほぼ同様のキーワードのセットが見られた．上位 20 キーワードまで比較すると，前半では「ナビゲーション AI」「持つ」というキーワードに対して，後半では「スパーシャル AI」「ステージ」というキーワードが現れた．前半より後半の方が「キャラクターAI」「ナビゲーション AI」より「キャラクターAI」「スパーシャル AI」の頻度と順位が相対的に上がっている．特に後半では「スパーシャル AI」がアイデアに多く含まれていることがわかる．

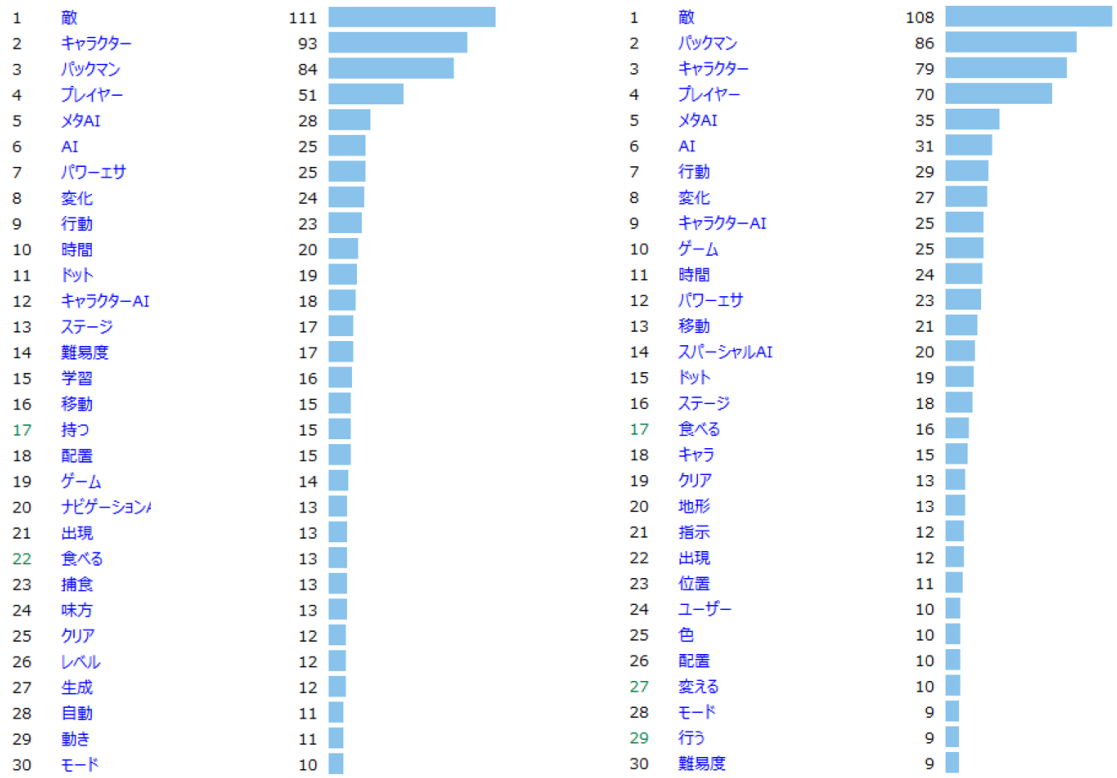


Fig. 10.3　前半のキーワードリスト(左)と後半のキーワードリスト（右）(上位 30 位)

10.3.3 前半と後半のキーワードの共起ネットワークの比較

次に，前半と後半のキーワード同士の共起関係について解析を行った（Fig. 10.4，Fig. 10.5）．前半では「制御」を中心とする連結数 17 の共起ネットワークが形成されている．また頻度数が高いキーワード「パックマン」「キャラクター」「敵」「プレイヤー」は連結数 4 の共起ネットワークを形成しているが，他のキーワードと関連は弱い．「メタ AI」「キャラクターAI」「ナビゲーション AI」の連結数 3 の共起ネットワークが存在するが，他のキーワードからは孤立している．一方，後半では「パックマン」を中心とした連結数 9 の共起ネットワークに「メタ AI」「キャラクターAI」「スパーシャル AI」が含まれている．また「キャラ」を中心とした連結数 15 の共起ネットワークが存在する．「ナビゲーション AI」はどこにも含まれていない．ここから前半は「メタ AI」「キャラクターAI」「ナビゲーション AI」を含むアイデアがあるものの，この 3 つと関連のないアイデアがより多く提出されていることがわかる．後半では「メタ AI」「キャラクターAI」「スパーシャル AI」を中心に形成がなされているアイデアの一群があると解釈される．

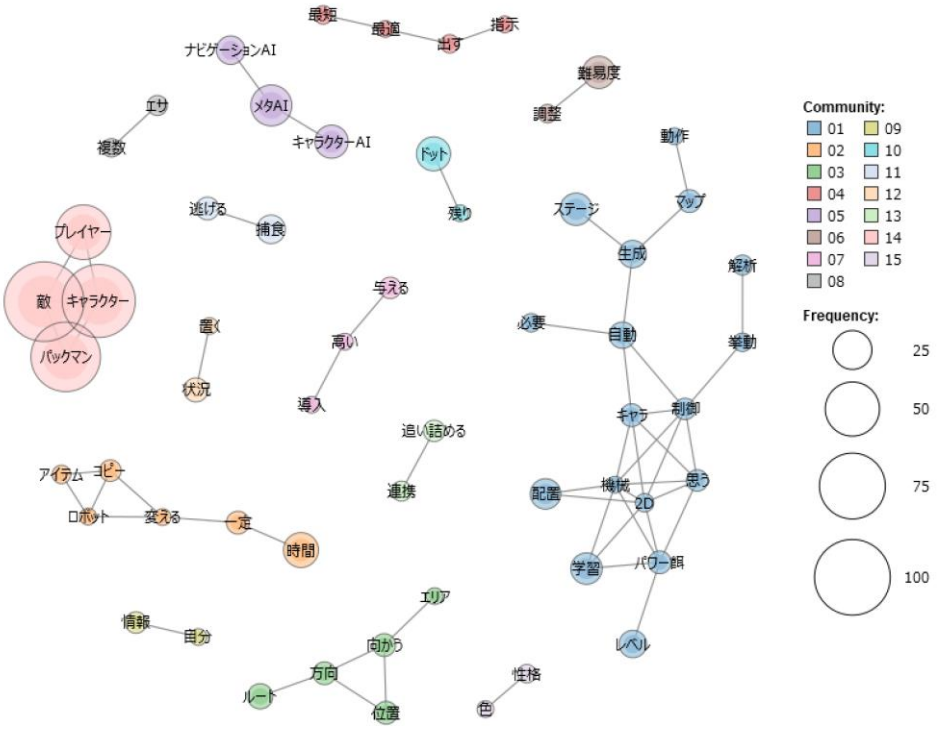


Fig. 10.4　前半の上位 60 個のキーワードの共起ネットワーク

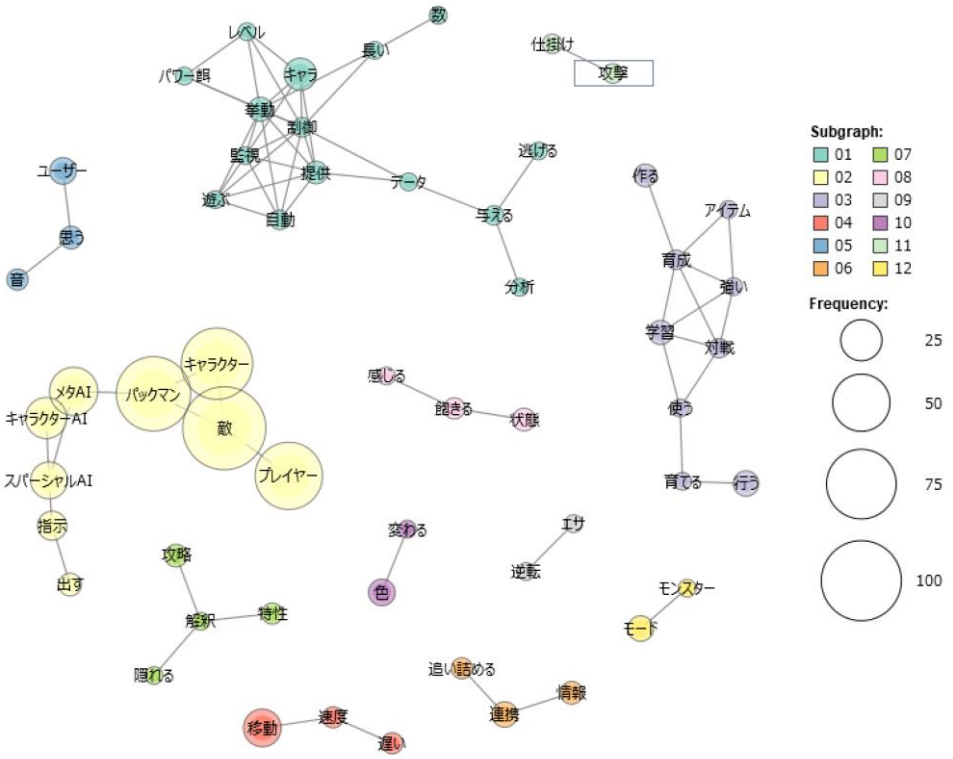


Fig. 10.5　後半の上位 60 個のキーワードの共起ネットワーク

10.3.4 MCN-MCN，MCN-MCS パターンの前半のキーワード頻度数比較

前半の回答に対して，MCN-MCN パターンと MCN-MCS パターンのキーワードの頻度数比較を行った．MCN-MCN パターンの被験者 13 人，MCN-MCS パターンの被験者 13 人の前半の回答から形態素解析の上，抽出を行った（Fig. 10.6）．上位 4 位までは同様であるが，MCN-MCN パターンでは 12 位「メタ AI」，21 位「キャラクターAI」となった．一方，MCN-MCS パターンでは 5 位「メタ AI」，10 位「キャラクターAI」，11 位「ナビゲーション AI」となっている．「メタ AI」はいずれも 12 位以内にあるが，「キャラクターAI」「ナビゲーション AI」は 10 位以下である．どちらのパターンも前半では「メタ AI」「キャラクターAI」「ナビゲーション AI」は中心的なキーワードではない．

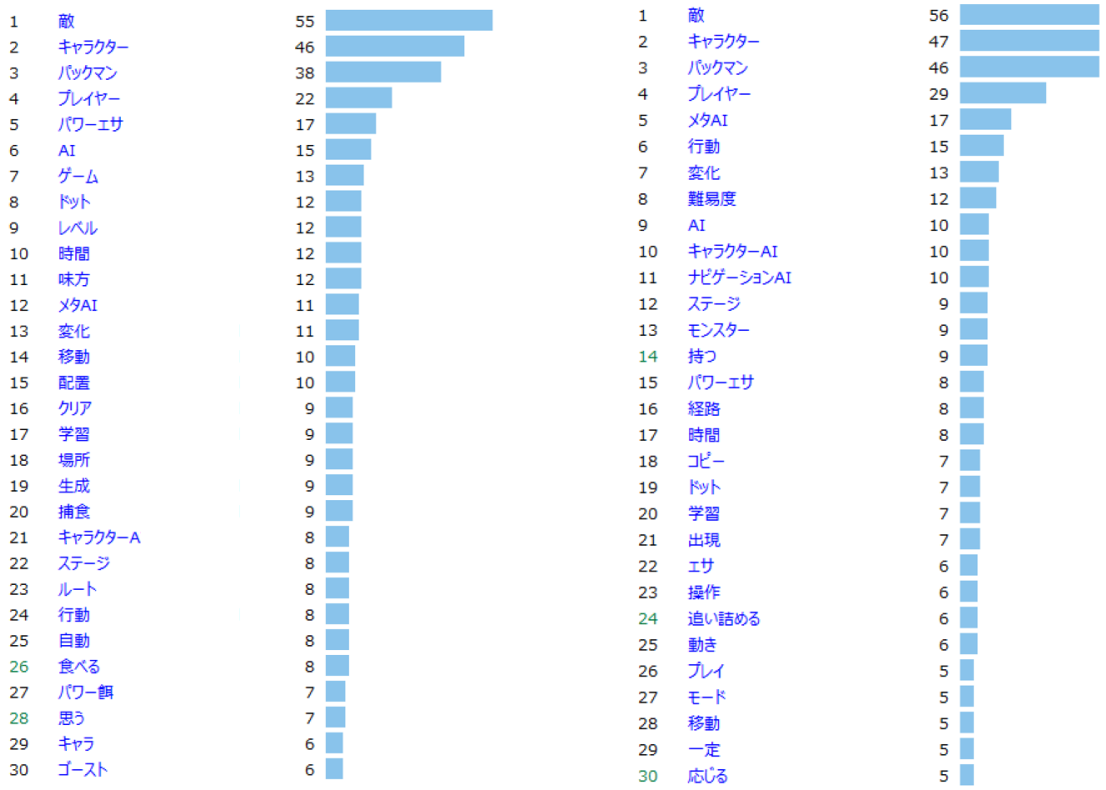


Fig. 10.6　前半における MCN-MCN パターン（左）と MCN-MCS パターン（右）のキーワードリスト(上位 30 位)

10.3.5 MCN-MCN，MCN-MCS パターン前半の共起ネットワークの比較

前半のキーワードに対して，MCN-MCN，MCN-MCS パターンの共起分析を行った（Fig. 10.7，Fig. 10.8）．　MCN-MCN パターンでは 4 つのクラスターが弱く結びついた共起ネットワークが形成されている．その中で「メタ AI」「キャラクターAI」はクラスターの端に位

置し，他のキーワードとは弱い関連しか持たない．

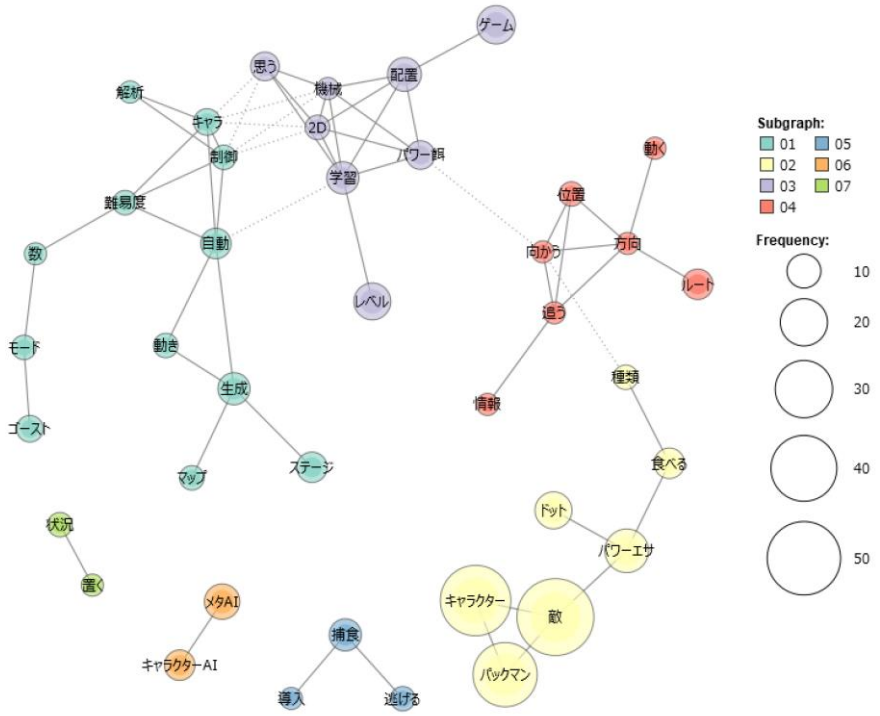


Fig. 10.7 前半 MCN-MCN パターンの上位 60 個のキーワードによる共起ネットワーク

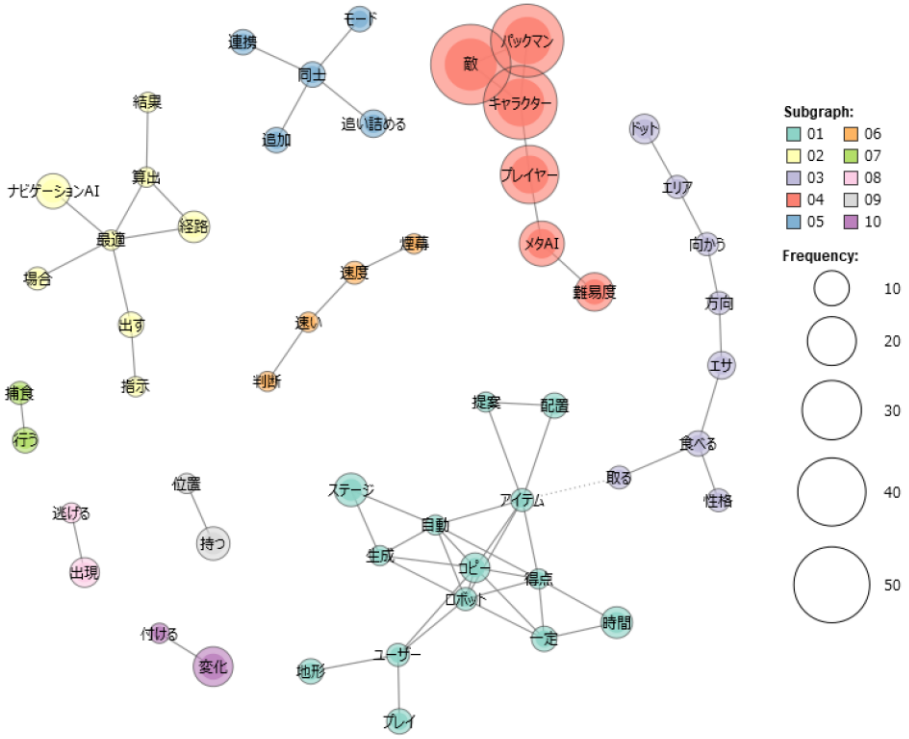


Fig. 10.8 前半 MCN-MCS パターンの上位 60 個のキーワードによる共起ネットワーク

MCN-MCN パターン，MCN-MCS パターン双方で，「メタ AI」「キャラクターAI」「ナビゲーション AI」は中心的なキーワードとなっていない．MCN-MCN パターンでは「メタ AI」「キャラクターAI」は結びつきながら他のクラスターと結びついていない．また，MCN-MCS パターンでは「メタ AI」「ナビゲーション AI」はそれぞれ連結数 6，8 のクラスターに属しているが，そのクラスターの中でも中心的な役割を果たしていない．MCN-MCN パターン，MCN-MCS パターンとも「メタ AI」「キャラクターAI」「ナビゲーション AI」が生成されるアイデアの中心的な位置にないことが確認できる．

10.3.6 MCN-MCN，MCN-MCS パターンの後半のキーワード頻度数比較

後半の回答に対して，MCN-MCN パターンと MCN-MCS パターンのキーワードの頻度数比較を行った．MCN-MCN パターンの被験者 13 人，MCN-MCS パターンの被験者 13 人の後半の回答から形態素解析の上，抽出を行った（Fig. 10.9）．上位 4 位までは同様であるが，MCN-MCN パターンでは 7 位「メタ AI」，12 位「キャラクターAI」，17 位「ナビゲーション AI」となった．一方，MCN-MCS パターンでは 5 位「スパーシャル AI」，9 位「キャラクターAI」，11 位「メタ AI」と順番が逆転している．また「地形」というキーワードが新しく入っている．

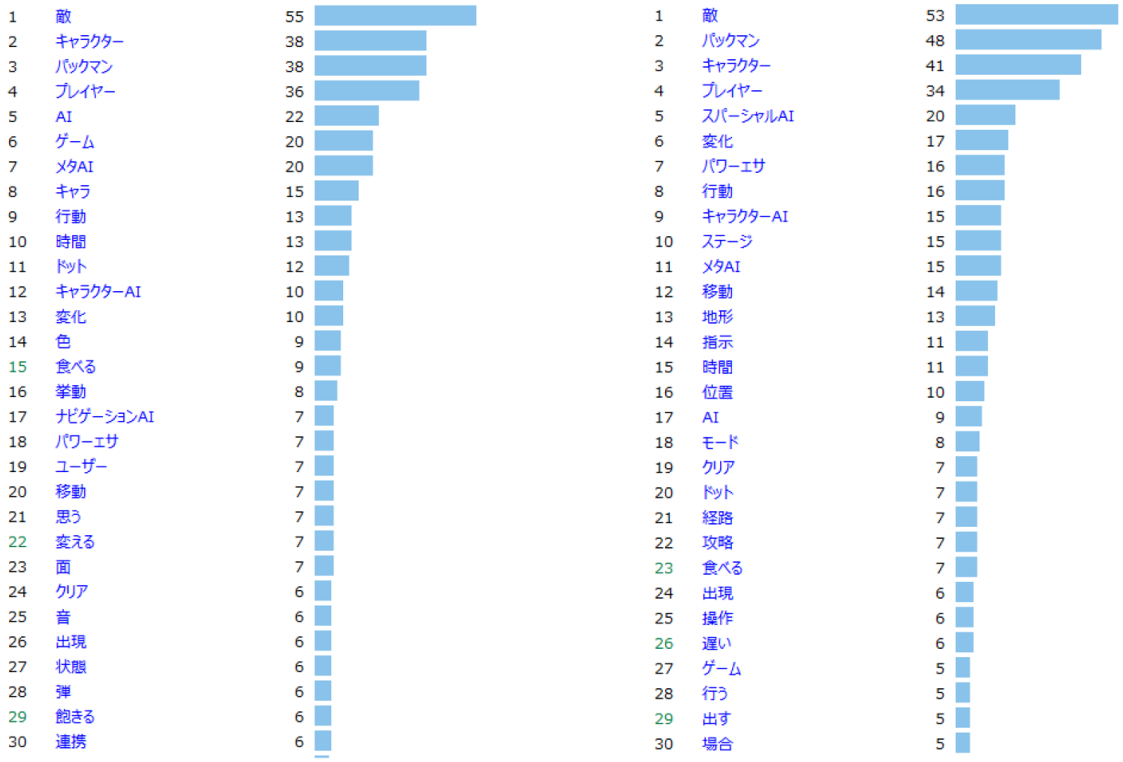


Fig. 10.9　後半における MCN-MCN パターン（左）と MCN-MCS パターン（右）のキーワードリスト（上位 30 位）

10.3.7 MCN-MCN, MCN-MCS パターン後半の共起ネットワークの比較

後半のキーワードに対して, MCN-MCN, MCN-MCS パターンの共起分析を行った (Fig. 10.10, Fig. 10.11)． MCN-MCN パターンでは数個のクラスターが弱く結びついた共起ネットワークが形成されている．その中で「メタ AI」は中心的な役割を果たしているが,「キャラクターAI」「ナビゲーション AI」はクラスターの端に位置し,「仕掛」「攻撃」を除いては他のキーワードとは弱い関連しか持たない.

MCN-MCS パターンでは 6 つのクラスターが弱く結びついた連結数 36 の共起ネットワークが形成されている．特に「パックマン」を中心とする連結数 7 の共起ネットワークには「メタ AI」「キャラクターAI」「スパーシャル AI」が相互に関連したネットワーク構造を形成している．またこの「メタ AI」「キャラクターAI」「スパーシャル AI」を含む共起ネットワークが周囲の連結数 4，7，9 の 3 つのクラスターと強いつながりを持って大きな共起ネットワークとなっている.

前半では，どちらのモデルも「メタ AI」「キャラクターAI」「ナビゲーション AI」は中心的なキーワードではなかった．しかし，後半では MCN-MCN パターンでは「メタ AI」「キャラクターAI」「ナビゲーション AI」は中心的なキーワードではないが，MCN-MCS パターンは「メタ AI」「キャラクターAI」「スパーシャル AI」が中心的なキーワードとなっている．この比較から，MCN-MCN パターンは，MCN-AI 連携モデルの影響がありながらも,「メタ AI」「キャラクターAI」「ナビゲーション AI」に関連しないアイデアがより多く含まれる．一方で MCN-MCS パターンは,「メタ AI」「キャラクターAI」「スパーシャル AI」と関連した MCS-AI 動的連携モデルの影響を持つアイデアを創出させる効果を持っていると言える.

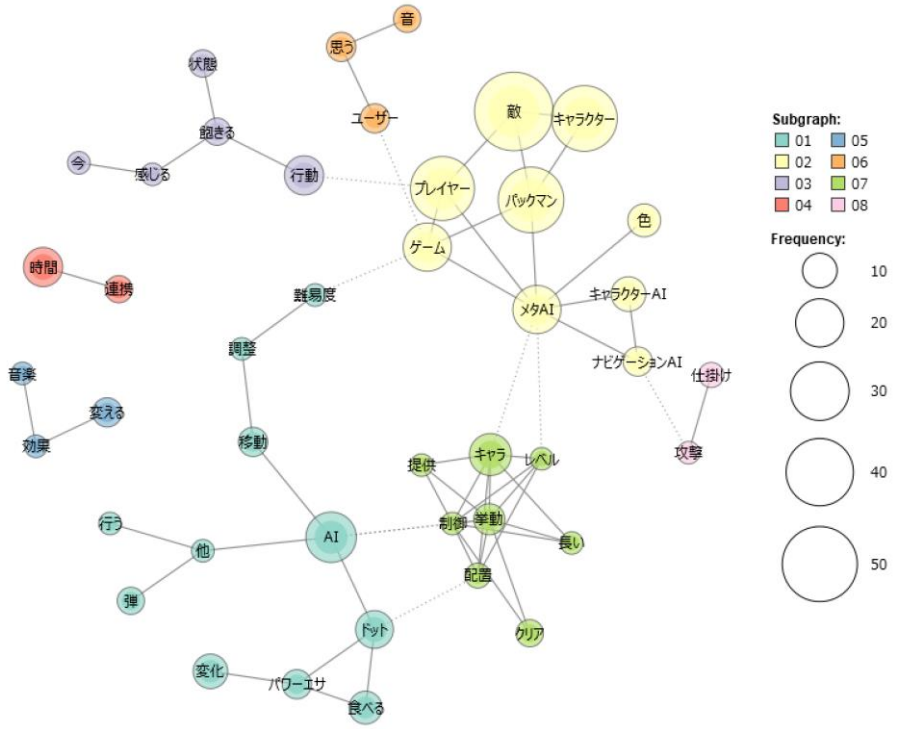


Fig. 10.10 後半 MCN-MCN パターンの上位 60 個のキーワードによる共起ネットワーク

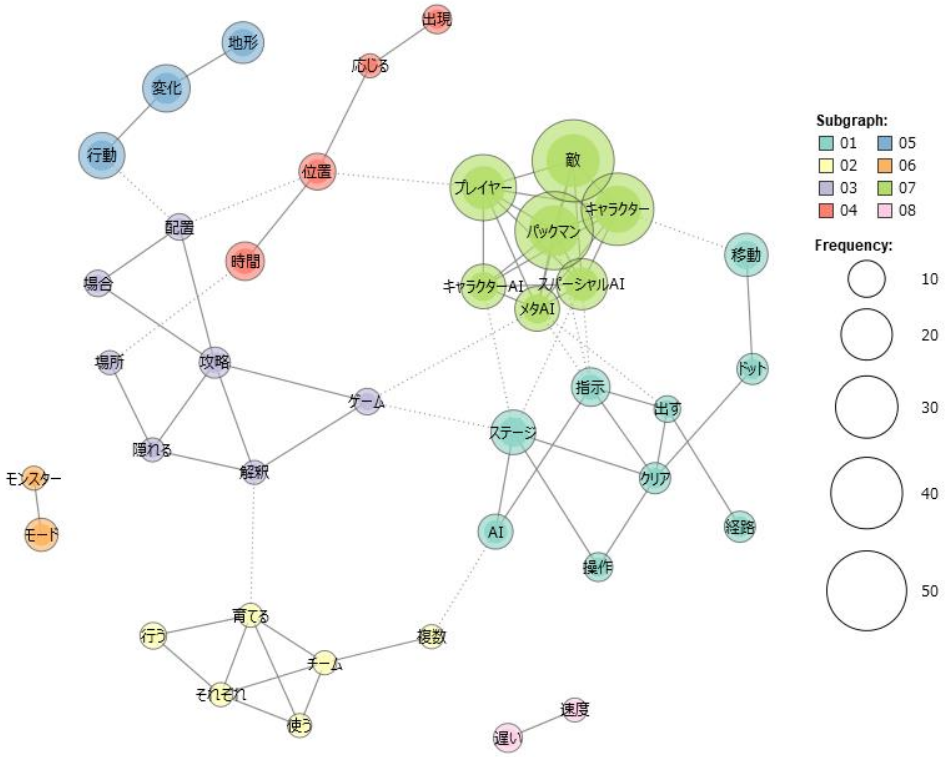


Fig. 10.11 後半 MCN-MCS パターンの上位 60 個のキーワードによる共起ネットワーク

10.3.8 「メタ AI」「キャラクターAI」「ナビゲーション AI」「スパーシャル AI」の出現数の調査

それぞれ個人の回答に含まれる「メタ AI」「キャラクターAI」「ナビゲーション AI」「スパーシャル AI」の出現数を調査した（Table 10.2）．前半のアンケートでは，この 4 つのキーワードが現れた回数は 55 回に対して，後半では 84 個と 1.52 倍に増加している．MCN-MCN パターンでは 23 個から 39 個と 1.69 倍，一人当たり平均 1.2 個増加し，MCN-MCS パターンでは 32 個から 45 個と 1.40 倍，一人当たり 1.0 個増加している．MCN-MCN パターンでは 5 名が増加，2 名が減少，6 名が変化なしである．MCN-MCS パターンでは 5 人が増加，3 名が減少，3 名が変化なしである．MCN-MCN パターン，MCN-MCS パターンとも「メタ AI」「キャラクターAI」「ナビゲーション AI」「スパーシャル AI」を含んだアイデアは増加しているが，MCN-MCS パターンの方が有意に増加しているとは言えない．

また，「メタ AI」「キャラクターAI」「ナビゲーション AI/スパーシャル AI」の 3 つすべてを含んだアイデアはアンケート前半では 1 例，一人当たり 0.03 例，後半では 10 例であり，一人当たり 0.3 例である．全般的に，後半のアンケートでは 3 者を連携したアイデアが出やすくなっている．

後半 10 例のうち，7 例が「メタ AI」「キャラクターAI」「スパーシャル AI」の 3 つすべてを含んだ MCN-MCS パターンの回答におけるアイデアであり，13 人の被験者の回答であるから一回答あたり 0.53 例である（ここで言う一回答とは，一人の被験者の一回の記入のことを指す．被験者は前後半と 2 回の回答の記入がある）．後半で 3 例が「メタ AI」「キャラクターAI」「ナビゲーション AI」の 3 つすべてを含んだパターン MCN-MCN パターンの回答におけるアイデアである．つまり MCN-AI 連携モデルを読んだ前半 25 回答，後半 13 回答の中で，「メタ AI」「キャラクターAI」「ナビゲーション AI」を 3 つ含んだアイデアは 4 例あり，一回答あたり 0.10 例である．MCS-AI 動的連携モデルが，より 3 つの AI を絡めたアイデアを生み出している．

Table 10.2 アイデアに含まれる 3 つの AI の名称の数の分布

検証番号	パターン	解答番号	前半に以下の語句が含まれる個数			後半に以下の語句が含まれる個数			前半合計	後半合計	後半-前半
			メタAI	キャラクターＡＩ	ナビゲーションAI	メタAI	キャラクターAI	ナビゲーションAI/スパーシャルAI			
1	MCN-MCS	1	0	0	0	0	0	0	0	0	0
2	MCN-MCS	1	1	0	0	1	1	1			
		2	1	0	0	1	1	2			
		3	0	1	0	2	1	1			
		4	0	0	1				4	11	7
3	MCN-MCN	1	0	0	0	0	0	0			
		2	0	0	0				0	0	0
4	MCN-MCN	1	0	0	0	0	0	0			
		2	0	0	0	0	0	0			
		3	0	0	0	0	0	0			
		4	0	0	0	0	0	0			
		5	0	0	0	0	0	0	0	0	0
5	MCN-MCN	1	1	1	1	2	2	1			
		2	1	3	0	1	2	0	7	8	1
6	MCN-MCN	1	1	1	0	0	1	0			
		2	0	1	0	1	1	0			
		3	0	0	0	0	0	0	3	3	0
7	MCN-MCS	1	0	1	2	0	0	0			
		2	0	0	0	0	0	0			
		3	0	0	0				3	0	-3
8	MCN-MCS	1	1	0	1	0	0	1			
		2	1	0	1	1	0	1			
		3	2	0	1				7	3	-4
9	MCN-MCS	1	1	2	0	1	1	1			
		2	0	0	0	0	2	0	3	5	2
10	MCN-MCS	1	1	1	0	0	0	0			
		2	1	1	0	0	0	0			
		3	0	0	1	0	0	0	5	0	-5
11	MCN-MCS	1	0	0	0	0	0	0			
		2	1	0	0	0	0	1			
		3	0	0	0	0	0	0			
		4	0	0	0	0	0	1			
		5	0	0	0	0	0	2			
		6				1	2	1	1	8	7
12	MCN-MCS	1	0	0	0	0	0	2			
		2	1	0	0	0	0	0			
		3	1	0	1	1	1	1			
		4	0	0	0	0	0	0			
		5	0	0	0	0	0	1			
		6	1	0	0	1	2	1			
		7	0	0	1	0	0	0	5	10	5
13	MCN-MCN	1	0	1	0	1	1	1			
		2	0	0	0	1	1	0			
		3	0	0	0	1	1	0			
		4	0	0	0	0	0	0			
		5	0	0	0				1	7	6
14	MCN-MCS	1	0	0	0	0	0	0			
		2	0	0	0	0	0	0			
		3	0	0	0	0	0	0			
		4	0	0	0	0	0	0			
		5				0	0	0	0	0	0
15	MCN-MCN	1	0	0	0	0	0	0			
		2	0	0	0	0	0	0			
		3				0	0	0			
		4				0	0	0	0	0	0
16	MCN-MCN	1	1	1	0	0	0	0			
		2	1	0	1	0	0	0			
		3				1	1	0			
		4				1	0	0	4	3	-1
17	MCN-MCN	1	0	0	0	0	0	0			
		2	0	0	1	0	0	0			
		3	0	0	0	0	0	0			
		4	1	1	0	0	0	0	3	0	-3

18	MCN-MCS	1	0	0	0	0	0	0			
		2	0	0	0	0	0	0			
		3	0	0	0	0	0	0			
		4	0	0	0				0	0	0
19	MCN-MCS	1	0	0	0	0	1	1			
		2	0	1	0	1	0	0			
		3	0	1	0	1	0	1			
		4	0	1	0	0	2	1	3	8	5
20	MCN-MCN	1	0	0	0	0	1	1			
		2	0	0	0	1	0	0			
		3	0	0	0	1	0	0			
		4	0	0	0	1	0	1	0	6	6
21	MCN-MCN	1	0	0	0	1	0	1			
		2	0	0	0	0	0	0			
		3	0	0	0	1	1	1			
		4	0	0	0						
		5	0	0	0				0	5	5
22	MCN-MCN	1	1	1	0	0	1	1			
		2	0	0	0	0	0	0			
		3	0	0	0	0	0	0			
		4	0	0	0	0	0	0			
		5				0	0	0			
		6				0	0	0			
		7				0	0	0			
		9				0	0	0			
		10				0	0	0			
		11				0	0	0			
		12				0	0	0			
		13				0	0	0			
		14				0	0	0	2	2	0
23	MCN-MCN	1	1	0	0	0	0	0			
		2	0	0	0	0	0	0			
		3	0	0	0	1	1	0			
		4	0	0	0	0	0	0			
		5	0	0	0	0	0	0			
		6	0	0	1	0	0	0			
		7				0	0	0			
		8				0	0	0			
		9				0	0	0			
		10				0	0	0			
		11				0	0	0			
		12				0	0	0	2	2	0
24	MCN-MCN	1	0	0	0	1	1	0			
		2	0	0	0	0	0	0			
		3	0	0	0	1	0	0			
		4	0	1	0	0	0	0	1	3	2
25	MCN-MCS	1	0	0	0	0	0	0			
		2	0	0	0	0	0	0			
		3	0	0	1	0	0	0			
		4	0	0	0	0	0	0			
		5	0	0	0	0	0	0	1	0	-1
26	MCN-MCS	1	0	0	0	0	0	0			
		2	0	0	0	0	0	0	0	0	0
合計			21	20	14	28	29	27	55	84	29

(注)赤字は MCN-AI 連携モデルを読んだあとのアンケートであり，青字は MCS-AI 動的連携モデルを読んだあとのアンケートである．

「メタ AI」「キャラクターAI」「ナビゲーション AI/スパーシャル AI」のうち，2 つのみを組み合わせた例は前半 13 例，後半 15 例で合計 28 例であるが，このうち 24 例（一回答

あたり 0.61 例，ここで一回答とは一回のアンケートという意味である．つまり前後半では 2 回ということである）が MCN-MCN パターンの回答，或いは，MCN-MCS パターンの前半に含まる．MCN-MCN パターンの回答，或いは，MCN-MCS パターンの前半に，メタ AI，キャラクターAI の組み合わせが 15 例，メタ AI，ナビゲーション AI の組み合わせが 6 例，キャラクターAI，ナビゲーション AI の組み合わせが 3 例であった． MCN-MCS パターンの後半の回答には 4 例（一回答あたり 0.30 個）が含まれ，メタ AI，キャラクターAI の組み合わせが 0 例，メタ AI，ナビゲーション AI の組み合わせが 2 例，キャラクターAI，ナビゲーション AI の組み合わせが 2 例であった．

「メタ AI」「キャラクターAI」「ナビゲーション AI/スパーシャル AI」の 3 つのうち 1 つを含む例は全部で前半 15 例，後半 10 例あるが，このうち，このうち 20 例（一回答あたり 0.51 例）が MCN-MCN パターンの回答，或いは，MCN-MCS パターンの前半に含まれる．メタ AI を用いるアイデアが 10 例，キャラクターAI を用いるアイデアが 5 例，ナビゲーション AI を用いるアイデアが 5 例である．MCN-MCS パターンの後半の回答に含まれるアイデアが 5 例（一回答あたり 0.38 個）である．キャラクターAI を用いるアイデアが 1 例，スパーシャル AI を用いるアイデアが 5 例である．

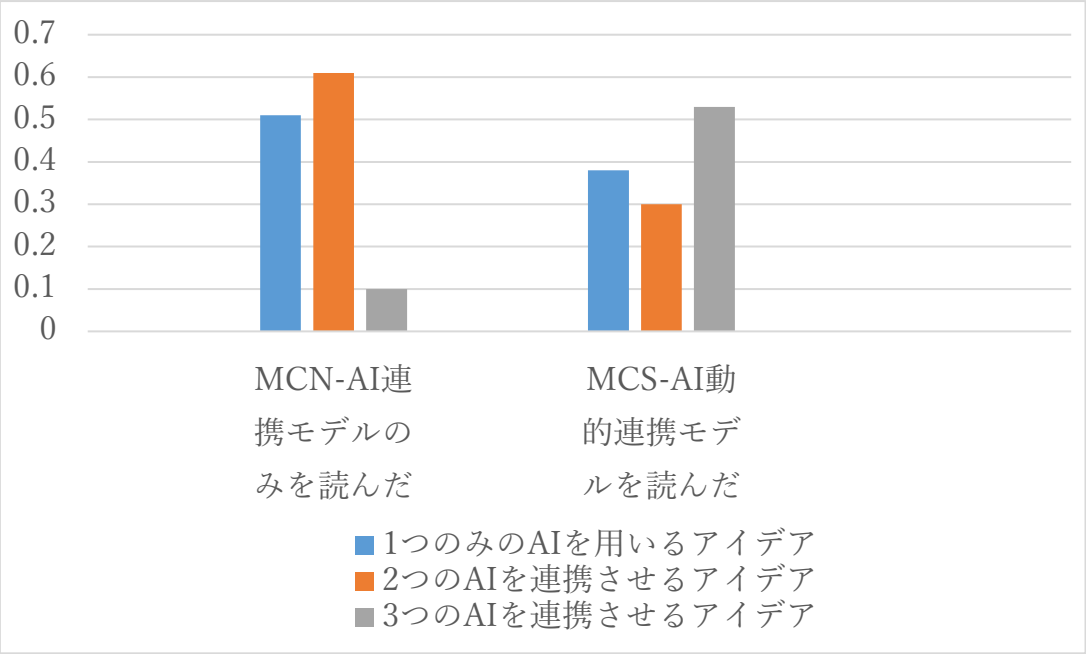


Fig. 10.12　AI を連携させたアイデアの個数（一回答当たり）の比較

以上をまとめると，Fig. 10.12 のように，MCN-AI 連携モデルのみを読んだ被験者は 1 つ，或いは 2 つの AI だけを連携するアイデアが 3 つの AI を連携するというアイデアがそれに比べて多い．MCS-AI 動的連携モデルを読んだ被験者は 1 つ，或いは 2 つの AI を用いたアイデアに比べて，3 つの AI が連携するアイデアが多い．MCN-AI 連携モデルのみを読んだ

被験者と MCS-AI 動的連携モデルを読んだ被験者を比べると，MCS-AI 動的連携モデルが 5 倍以上，3 つの AI を連携したアイデアを提出している．これは MCN-MCS パターンでは，3 つの AI を連携するという点が理解され発想に活かされていると解釈できる．

このような相違が現れた理由として，MCS-AI 動的連携モデルの場合には「メタ AI」「キャラクターAI」「スパーシャル AI」3 つの AI の連携が示されていたことに対して，MCN-AI 連携モデルはメタ AI とキャラクターAI，キャラクターAI とナビゲーション AI の連携は示されていたがメタ AI とナビゲーション AI の連携は示されていない．そのため，MCS-AI 動的連携モデルの場合には 3 つの AI がすべて連携した事例が多く，MCN-AI 連携モデルの場合には 2 つの AI の連携モデルが多く回答に現れたと考えられる．

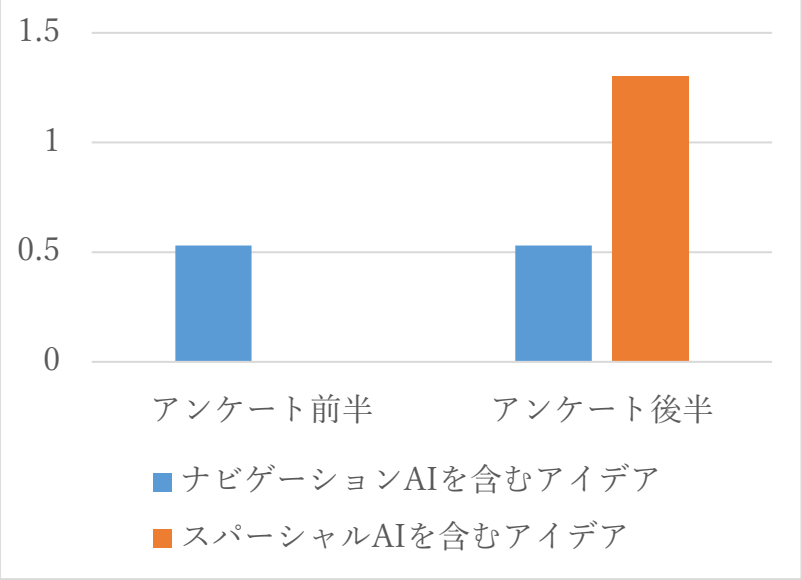


Fig. 10.13　ナビゲーション AI，スパーシャル AI を含む前後半のアイデア数(一人当たり平均)の比較

10.3.9 ナビゲーション AI とスパーシャル AI の比較

前半では 14 例の「ナビゲーション AI」を含んだアイデアがあった．これは MCN-AI 連携モデルの説明を読んで発想されたアイデアであり，被験者 26 人の回答に含まれる回答例の数であるため，一人当たり平均 0.53 例である．後半 26 例の「ナビゲーション AI/スパーシャル AI」を含んだアイデアのうち．「ナビゲーション AI」を含んだ MCN-MCN パターンの回答におけるアイデアは 7 例であり一人当たり平均 0.53 例，「スパーシャル AI」を含んだ MCN-MCS パターンの回答におけるアイデアは 17 例であり一人当たり平均 1.30 例である．この 17 例のうち前半で「ナビゲーション AI」を含んでいた被験者が出した例は 9 例，それ以外は前半「ナビゲーション AI」を使用せず，後半「スパーシャル AI」を新規に使用

したアイデアであった．前半，後半で「ナビゲーション AI」を用いたアイデアは，一人当たりほぼ 0.5 と同数提案されているが，「スパーシャル AI」の概念が，その 2 倍を超えるアイデアを喚起している（Fig. 10.13）．

このように「スパーシャル AI」がアイデアを喚起する背景には，「ナビゲーション AI」が「メタ AI」「キャラクターAI」の従属的な存在であったのに対して，「スパーシャル AI」が独立した自律的 AI として被験者に被験者が理解したことによると考えられる．

10.3.10 3つの AI を連携したアイデアの定性的な比較

ここまでの解析で，MCS-AI 動的連携モデルがアイデアの質を変容させることがわかった．そこで MCN-AI 連携モデルをもとに 3 つの AI を連携させたアイデアと，MCS-AI 動的連携モデルをもとに 3 つの AI を連携させたアイデアの定性的な比較を行う．以下，それぞれ 3 つの AI が連携するアイデアをアンケートより抜粋する．

MCN-MCN パターン前半（1 例）

(1)ゲームの難易度をよりシビアにする AI．ナビゲーション AI により，パックマンを端においつめるルートを割り出す．ナビゲーション AI と複数体の敵キャラクターのキャラクターAI により，パックマンの逃げ道を塞ぐように敵キャラクターを動かす．メタ AI により，パックマンを包囲するように敵キャラクターを出現させる．

MCN-MCN パターン後半（3 例）

(2)メタ AI：プレイヤーの挙動やクリア時間を監視し，敵キャラの移動範囲や速度をリアルタイム制御する．キャラクターAI：メタ AI からの命令に従いつつも，「このような挙動をしていいか」という提案をしながらより賢い AI を作っていく．ナビゲーション AI：キャラクターAI からの要求を受けて，敵キャラの挙動に必要は情報を提供しナビゲーションする

(3)敵キャラクターは弾を発射することができる．種類ごとに，弾速や飛距離が異なる．パックマンは動きを止めることで正面に向けて盾を構え，弾を防ぐことができる．また敵キャラクターの弾が他の敵キャラクターに当たると，弾が当たった敵キャラクターは消滅する．敵のキャラクター AI はナビゲーション AI の誘導でパックマンを追跡する．敵キャラクター AI はパックマンに近づくとメタ AI に弾を発射して良いかを問い合わせ，メタ AI は敵キャラクターの位置関係やパックマンの向き，盾を構えているか否かなどから発射を許可するか否かを判断する．

(4)マリオカートのパックマンとゴーストのみバージョン．メタ AI でコースを俯瞰し，アイ

テム出現数，障害物の設置をコントロールする．また，敵キャラの車は全てクッパのように当たってしまうとタイムロスに繋がるが，餌を沢山食べて一定数達すると立場が逆転する．ナビゲーション AI で敵キャラの強さはレベルに分けられる．原作と同じように，キャラクターAI によって色ごとにゴーストの運転性格が変わる．

MCN-MCS パターン後半（7 例）

(5)パックマンの平面迷路を立体迷路化する．より複雑になる敵キャラクターの制御をスパーシャル AI から収集するデータを各キャラクターAI に提供することで適切なゲームバランスを確保する．またメタ AI によってプレイヤーが最もパックマンの現状を理解しやすいアングルに自動でカメラを移動させてくれるようにする．

(6)パックマンを自分で動かすのではなく，プレイヤーが AI で動くパックマンに指示を与えて「より早くステージをクリアさせる」タイムアタック的なモードを追加する．スパーシャル AI が提供する情勢分析データをプレイヤーに提示しながら「いのちだいじに」「ガンガンいこうぜ」のような言葉で代替させた AI の動作ポリシーをプレイヤーがリアルタイムで切り替えできるようにする．いわば「メタ AI がスパーシャル AI から収集した情報をもとに敵キャラクターのキャラクターAI に指示を出す」一連の作業と同じものをプレイヤーに提供するものである．

(7)逆にプレイヤーが敵キャラクターの AI に指示を出すゲームにすることもできそう．メタ AI はパックマンを操作して，プレイヤーはスパーシャル AI が報告してくる各種データをビジュアライズされたものを元に 4 種類の敵キャラクターへ指示（キャラクターAI の行動指針を 4 択くらいにしたもの）を切り替えてパックマンを追い詰める．ステージごとに「パックマン（メタ AI）の勝ち」「敵キャラクター（プレイヤー）の勝ち」を記録して勝ち越しや最高勝率を争う．

(8)スパーシャル AI が現在のパックマンと敵キャラクター全員の経路を複数算出．メタ AI はこれらの経路を比較し，パックマンの逃げ道が少なくなる（＝難易度が高くなる）ような経路を採用するように敵キャラクターのキャラクターAI に指示を出す．

(9)一定時間毎に地形が変化する仕組みプレイヤーがうまくプレイできていない時に，メタ AI は敵キャラクターの追跡を鈍らせる（キャラクターAI の移動先に対して，パックマンの居場所ではない場所を指定する）とともに，パワー餌にたどり着きやすい地形を選択するスパーシャル AI は変化した地形に応じて位置情報をキャラクターAI に送る．

(10)スパーシャル AI とキャラクターAI の関係を密接にすることで環境に合わせたキャラ

クター生成を行う．例えば，メタ AI によって作り出されたステージが田舎の風景の場合はセリフに訛りを付与するなど．

(11)パックマンがステージ攻略をしていない間はペットとして飼えるようになるゲーム．日常生活で，たまごっち的な感じでパックマンを飼い，育てることでステージ攻略に有利な成長をさせたり，逆に憶病な性格に育ててしまうと，ステージ攻略が困難になったりする．キャラクターAI を育て，ステージ攻略では，メタ AI とスパーシャル AI が，キャラクターAI の変化を解釈し，それに合わせたステージ生成を行う．

まず定量的な解析を述べる．MCN-AI 連携モデルから発想された 4 つの解答において，3 例で「ナビゲーション AI で」という表現が用いられている．ナビゲーション AI を用いる，という発想が共通して見られる．MCS-AI 動的連携モデルから発想された 7 例において 4 例が「スパーシャル AI が」「スパーシャル AI は」という表現が用いられている．メタ AI，キャラクターAI と共に，スパーシャル AI が自律性を持つ主体的な AI として発想されている．また MCN-AI 連携モデルから発想された 4 つの解答はすべて「キャラクターAI」と「ナビゲーション AI」の関係から発想されている．MCS-AI 動的連携モデルから発想された 7 例において 3 例が「メタ AI」と「スパーシャル AI」の連携を含むアイデアとなっている．これは MCN-AI 連携モデルにはない特徴である．また「スパーシャル AI」がユーザーに情報を可視化する，というアイデアが 2 例存在する．ここでは「スパーシャル AI」と「プレイヤー」という新しい関係が発想されている．

次に定性的な解析を述べる．MCN-MCN パターン前半（1 例）の(1)に関して「ルート」という言葉が使われている．MCN-MCN パターン後半（3 例）のうち(2)では「ナビゲーション」，(3)では「追跡」，(4) では「コース」という言葉が使われている．ここで見られるのは，直線的な経路のイメージである．これは「ナビゲーション AI」が基本的なイメージとしてアイデアを支えており，その上で「メタ AI」によって変化を与えることが想定されている．また上記 4 例はいずれも「キャラクターAI」は必ず「ナビゲーション AI」と関連する形で記述されている．これは MCN-AI 連携モデルの連携を忠実に反映している．MCN-MCS パターン後半 (7 例) においては地形やステージを生成・変化させるアイデアが多い．たとえば(5)では「平面迷路を立体迷路化する」，(9)では「一定時間毎に地形が変化する仕組み」，(10) では「メタ AI によって作り出されたステージ」，(11) では「ステージ生成」という言葉が使われている．またそのような作り出された地形・ステージにキャラクターAI，ナビゲーション AI によって適応する，というデザインとなっている．また「スパーシャル AI とデータ」という観点が多く提出されている．(5)では「スパーシャル AI から収集するデータ」，(6)では「メタ AI がスパーシャル AI から収集した情報」，(7)では「スパーシャル AI が報告してくる各種データ」，(8)では「現在のパックマンと敵キャラクター全員の経路を複数算出」などの記述がある．また，「スパーシャル AI から他の AI への情報の受け渡し」

という仕様が提案されている．(5)(9)ではスパーシャル AI からキャラクターAI へ，(6)(7)(8)ではスパーシャル AI からメタ AI へ情報が受け渡されている．これは MCS-AI 動的連携モデルの連携の仕様がそのままアイデアに現れている．

10.3.11 職種ごとのアイデアの定性的解析と比較

本項では，被験者の職種ごとのアイデアの定性的比較を行う．アーティスト，プランナー，エンジニアの 3 職種である．以下，順番に職種ごとに述べる．

最初にアーティストは 2 つのアンケートパターンで各 1 名である．特色のあるアイデアを以下に列挙する．

MCN-MCN パターン（前半）

(1) パックマンの背景をプレイヤーの置かれている時間や場所を認知して美しく変わるのはどうだろう．前出から考えるとプレイヤーのリフレッシュを想起させられればと思う．背景色が変わるだけでも，プレイヤーの心理状況は変化すると考えられる．

MCN-MCN パターン（後半）

(2) 前述のより高度なパックマンになら，パックマンのキャラクターを成長させたい．よりかわいくなったり，よりつよくなったり，するのはどうだろう？色が変化させられたり，リボンのようなアイテムをつける事ができたりあまり華美でなくても少しの時間だけ変えられる事でユーザーは楽しめると思う．その少しだけを観たいためにゲームを持続させるだろう．パワーエサを食べた時だけ短い間出現するのでもいいかもしれない．その加減は，キャラクターAI からのフィードバックによりメタ AI によって出現させる．

(3) パックマンの楽しさはあの音にもあると思う．あの音をたくさん聞きたいと考えるなら，敷地は広い方がいい．遊んでいるトータル時間によってもメタ AI にその広さを加味させてしもいいと思う．

MCN-MCS パターン（後半）

(4) モンスターの考えを視覚化して，モンスターが 1 秒後どこに居るかを予測表示してくれる「ナビゲーションモード」

(5) 各モンスターの行動やパワーエサの出現などを解説・実況・アドバイス＆アシスト表示してくれる「パワフル実況モード」

MCN-MCN パターンでは「パックマンの外見，ゲームの音」といった視覚的・聴覚的アーティスティックな要素をメタ AI から変化させるアイデアが提案されている．MCN-MCS

パターンでは AI が「実況」を行うという新しいアイデアが提案されている．また(4)では「モンスターの考えを視覚化してユーザーに提示する」というアイデアが提案されている．これもまた視覚的な効果である．

次にプランナーのアイデアの定性的分析を行う．両パターンで各 1 名である．特色のあるアイデアを以下に列挙する．

MCN-MCS パターン（後半）

(6) より複雑になる敵キャラクターの制御をスパーシャル AI から収集するデータを各キャラクターAI に提供することで適切なゲームバランスを確保する．またメタ AI によってプレイヤーが最もパックマンの現状を理解しやすいアングルに自動でカメラを移動させてくれるようにする．

(7) パックマンを自分で動かすのではなく，プレイヤーが AI で動くパックマンに指示を与えて「より早くステージをクリアさせる」タイムアタック的なモードを追加する．スパーシャル AI が提供する情勢分析データをプレイヤーに提示しながら「いのちだいじに」「ガンガンいこうぜ」のような言葉で代替させた AI の動作ポリシーをプレイヤーがリアルタイムで切り替えできるようにする．いわば「メタ AI がスパーシャル AI から収集した情報をもとに敵キャラクターのキャラクターAI に指示を出す」一連の作業と同じものをプレイヤーに提供するものである．

(8) 逆にプレイヤーが敵キャラクターの AI に指示を出すゲームにすることもできそう．メタ AI はパックマンを操作して，プレイヤーはスパーシャル AI が報告してくる各種データをビジュアライズされたものを元に 4 種類の敵キャラクターへ指示（キャラクターAI の行動指針を 4 択くらいにしたもの）を切り替えてパックマンを追い詰める．ステージごとに「パックマン（メタ AI）の勝ち」「敵キャラクター（プレイヤー）の勝ち」を記録して勝ち越しや最高勝率を争う．

(6)-(8)は同一のプランナーのアイデアである．(7)パックマン AI に指示を与える，(8) 敵キャラクターに指示を出す，など，ゲームの根幹のアイデアを変化させるアイデアが提出されている．また (6)「スパーシャル AI から収集するデータを各キャラクターAI に提供する」(7)「スパーシャル AI が提供する情勢分析データをプレイヤーに提示」「メタ AI がスパーシャル AI から収集した情報をもとに敵キャラクターのキャラクターAI に指示を出す」(8)「プレイヤーはスパーシャル AI が報告してくる各種データをビジュアライズされたものを元に 4 種類の敵キャラクターへ指示」など，スパーシャル AI の作るデータをメタ AI，キャラクターAI，そしてプレイヤーに伝えることでゲーム性を変化させるアイデアが提出されている．全体として，大きな変更がゲームデザインに加えられている．

次にエンジニアのアイデアの定性的解析を行う．特色のあるアイデアを以下に列挙する．MCN-MCN パターンで 6 名，MCN-MCS パターンで 8 名である．特色のあるアイデアを以下に列挙する．

MCN-MCN パターン（前半）

(9) 味方キャラクターを登場させる．味方キャラクターは，特定の色の敵キャラクターを捕食できるが，それ以外の色の敵キャラクターには捕食される．メタ AI は味方のキャラクター AI に，マップ上のどこにどのキャラクターが存在しているか（プレイヤーが見ているものと同じ）を教える．味方のキャラクター AI はその情報を元に，自律的に，自分が捕食できる敵キャラクターを追いつつ，それ以外の敵キャラクターからは逃げる動作を行う．その際，捕食を優先するか，逃避を優先するかは味方のキャラクター AI の性格付けとなる．

(10)　パックマンと同じ要素を持つパックマン α を発生させる．メタ AI の管理下にあるキャラクターAI の亜種．　（ゲームの基礎が揺らぐ気がするのでもっと考える必要あり）

MCN-MCS パターン（前半）

(11)　逆パックマン パックマンの操作は行えず，プレイヤーは地形やパワーエサや敵キャラクターの配置を行う．地形を変更したり，パワーエサや敵キャラクターをどこに配置したらパックマンがどう動くかを予想しながら，パックマンを追い詰めて捕食すれば勝利．あくまでプレイヤーができるのは配置までで，その先はうまく AI の特性を利用して追い詰める必要がある．

(12)　暗闇モード（壁は見えいてる）パックマンの足音に反応する　パックマンの近くに敵キャラクターがくるとプレイヤーは音で認識できる．暗闇でゆっくり動くようになる敵，壁に向かって進み続ける（つまり止まってしまう）敵，ドルアーガのウィルオーウィスプみたいな動き．

MCN-MCN パターン（後半）

(13)　ドットから音を出るようにしてパックマンで提示された楽譜を満たす音階のドットを集めるゲームにする．メタ AI でパックマンが拾いたいドットに向けて敵キャラクターAI を向かわせたりする

MCN-MCS パターン（後半）

(14)　敵キャラクターもドットを食べる．食べたドットは捕食した時に遠くにばらまかれる

(9)(10)は新しいパックマンと類似したキャラクターを設定するというアイデアである．(11)はパックマンを操作しないアイデア，(12)は足音をゲームに取り込むアイデア，(13)はドットを取る音で音楽ゲームにするアイデア，(14)は敵もドットを食べて動的にドットの配置が変化するアイデアである．全体としてゲームのダイナミクスを変化させるアイデアである．

次に「その他」に属する中で，特色のアイデアを列挙する．両パターンで 2 名ずつであり，MCN-MCN パターンにテクニカルアーティスト 1 名，MCN-MCS パターンにシナリオライター1 名などである．

MCN-MCS パターン（前半）

（15） パックマン・敵キャラクター双方に AI を実装し，双方競わせて強化学習させることによって極限まで判断能力を高める．

MCN-MCS パターン（後半）

（16） パワーエサなどにも，地形による変化をつけたり，季節による変化（りんごが青かったり赤かったり）をつけたりなどする．

（17） 現実時間とゲーム内時間を連動させ，時間帯によるステージの変化や特性付けをスパーシャル AI に解釈させる．時間帯や季節によって攻略方法が変化するステージの中で，敵キャラクターにも行動の変更を細やかに支持する．例えば，夜は隠れられた場所が，朝になると見つかりやすい場所に変化する．あるいは春は青々としげっていた木に隠れることができたが，冬は木には隠れられず，枯れ葉の中に隠れる必要があるなど．

(15) は GAN（Generative Adversarial Network）のように競合学習させることで，お互いに高め合う仕組みを提案している．(16)(17)は現実世界の季節や時間と同期するアイデアである．ゲームの外と内側を結びつけている点が新しい点である．

以上のように，アイデアは多方面のアイデアが定性的に得られたが，必ずしも AI と結びついていないアイデアも多く存在する．

10.4 考察

本節では，被験者データの解析から考察を行う．以下，第 10.3 節の各項に従ってデータ解析結果をまとめる．

(1) 第 10.3.1 項では 2 つのアンケートパターンの回答数の比較を行った．2 つのアンケートパターンで，特に前半と後半の回答数の増大は見られなかった．

(2) 第 10.3.2 項では，前半，後半のキーワード頻度数比較を行った．前半より後半の方が「キャラクターAI」「ナビゲーション AI」より「キャラクターAI」「スパーシャル AI」の頻度と順位が相対的に上がっている．特に後半では「スパーシャル AI」がアイデアに多く含まれている．

(3) 第 10.3.3 項では前半の後半のキーワードの共起ネットワークの比較を行った．前半に比べて後半では「メタ AI」「キャラクターAI」「スパーシャル AI」を中心としたアイデアの形成がなされている．

(4) 第 10.3.4 項では，MCN-MCN，MCN-MCS パターンの前半のキーワード頻度数の比較を行った．MCN-MCS パターンで「メタ AI」がかろうじて 10 位に入っているが，それ以外の「キャラクターAI」「ナビゲーション AI」はいずれも 11 位以下であった．AI が用いられること自体が少ない．

(5) 第 10.3.5 項では，前半の回答に関して MCN-MCN，MCN-MCS パターンの共起ネットワークの比較を行った．どちらのパターンでも「メタ AI」「キャラクターAI」「ナビゲーション AI」がアイデアの中心にない．

(6) 第 10.3.6 項では，MCN-MCN，MCN-MCS パターンの後半のキーワード頻度数の比較を行った. MCN-MCN パターンでは「メタ AI」「キャラクターAI」「ナビゲーション AI」の順に広くランキングしているが, 一方, MCN-MCS パターンでは「スパーシャル AI」,「キャラクターAI」「メタ AI」と順番が逆転するが，いずれも上位に入っている．

(7) 第 10.3.7 項では後半の回答に関して MCN-MCN，MCN-MCS パターンの共起ネットワークの比較を行った．MCN-MCN パターンでは「メタ AI」「キャラクターAI」はクラスターの端に位置し，他のキーワードとは弱い関連しか持たない．MCN-MCS パターンでは「メタ AI」「キャラクターAI」「スパーシャル AI」が相互に関連したネットワーク構造を形成し他のキーワードと関連している．

(8) 第 10.3.8 項では,「メタ AI」「キャラクターAI」「ナビゲーション AI」「スパーシャル AI」の出現数の調査を行った．前半から後半にかけて両パターンで増加するが，大きな差は見られなかった．MCN-AI 連携モデルでは，2 つの AI を連携させるアイデアが，3 つの AI を連携するアイデアより 5 倍多い．また MCS-AI 動的連携モデルは 3 つの AI を連携するモデルが，2 つの AI を連携するアイデアより 3.5 倍多い．

(9) 第 10.3.10 項では,「メタ AI」「キャラクターAI」「ナビゲーション AI」をすべて含むアイデアと,「メタ AI」「キャラクターAI」「スパーシャル AI」をすべて含むアイデアの具体的なアイデアの比較を行った．前者は「ナビゲーション AI」を「キャラクターAI」から用いられる存在としてすべて発想されるが, 後者ではスパーシャル AI が自律的 AI として扱われる例が出現した．また前者では「メタ AI」「ナビゲーション AI」の協調例はないが，後者では「メタ AI」「スパーシャル AI」の協調例がみられた．これは，それぞ

れのモデルの解説をそのまま反映している．

(10)　第 10.3.11 項では，職種ごとのアイデアの定性的解析を行った．アーティストでは視覚・聴覚に訴えるアイデア，プランナーではゲームデザインの仕組み自体を変化するアイデア，エンジニアではゲームダイナミクスを工夫するアイデアが見られた．アイデアは AI によるものと，必ずしも AI に寄らないアイデアが見られた．

以上のことから，MCN-AI 連携モデルに対して，MCS-AI 動的連携モデルは次の効果を持っていることが言える．

（A）　両モデルは同じような数のアイデアを導く効果を持っているが，アイデアの質が異なる．

（B）　MCS-AI 動的連携モデルは，より AI に関連づいたアイデアを多く導く効果を持っている．MCN-AI 連携モデルでは AI と紐づけのないアイデアも多くみられる．

（C）　MCS-AI 動的連携モデルの中で，特にスパーシャル AI はアイデアを多く喚起する効果を持っている．

（D）　MCS-AI 動的連携モデルの中では，スパーシャル AI はメタ AI，キャラクターAI から独立した AI として存在感を持って受け止められる．

MCS-AI 動的連携モデルは，MCN-AI 連携モデルと比較して，アイデアの数を増加させるわけではないが，アイデアの質をより AI を多く含んだアイデアへと変化させる効果を持っている．

11. MCS-AI 動的連携モデルと AI Graph

本章では，MCS-AI 動的連携モデルと意思決定の仕組みである「AI Graph」について述べる．AI Graph は，キャラクターの意思決定システム作成ツールとしての基本設計と，MCS-AI 動的連携モデルの実現のための拡張された設計がある［145］.

AI Graph は MCS-AI 動的連携モデルとは独立に設計された意思決定システム作成ツールである．MCS-AI 動的連携モデルがトップダウンの設計であり，AI Graph は意思決定のレイヤーからボトムアップに構築された設計である．この AI Graph が MCS-AI 動的連携モデルが要求する仕様を満たすように拡張されたことによって，MCS-AI 動的連携モデルを実装し実現することができた．この融合事例については第 12 章で実例に沿って述べる．

第 11.1 節で AI Graph のキャラクターの意思決定ツールとしての基本設計を述べ，次に第 11.2 節で MCS-AI 動的連携モデルと AI Graph の関係と，MCS-AI 動的連携モデルが要求するそれぞれの性能に対する拡張された設計を述べる．第 11.3 節でまとめを述べる．

11.1 AI Graph

本節では「AI Graph」について述べる．第 11.1.1 項で AI Graph の原理について述べる．第 11.1.2 項では AI Graph の実際にツール化した「AI Graph Editor」について，第 11.1.3 項では，AI Graph の問題点について述べる．

AI Graph は，キャラクターAI における意思決定の設計原理であり，AI Graph Editor はその原理に従って実際にツール化したソフトウェアである．AI Graph は第 7.3 節で述べたような単一の意思決定アルゴリズムだけではなく，複数の意思決定アルゴリズムを組み合わせて用いる理論である．それぞれの意思決定アルゴリズムは長所と短所があり，単一のアルゴリズムでは対応しきれない問題がある．たとえば，第 7.3.3 項で説明したビヘイビアツリーはキャラクターの連続的なビヘイビアを記述するには向いており，ビヘイビアツリー全体がビヘイビアの展開へ向けて記述される一方で，ゲームの複雑な状況を捉えるには向いていない．第 7.3.2 項で述べたステートマシンはゲームの状況を分類して状態として表現するには向いているが，連続的な身体的行動を指定するには向いていない．そこでステートマシンとビヘイビアツリーを組み合わせる，という手法が考えられる．或いは，ゴール指向プランニングとルールベースを組み合わせる手法など，複数の意思決定アルゴリズムの組み合わせを考えることができる．このように AI Graph は複数の意思決定アルゴリズムの組み合わすシステムである．

意思決定アルゴリズムの組み合わせの中でも，ステートマシンとビヘイビアツリーの組み合わせは，キャラクターAI において有効である．ステートマシンはゲーム状態を捉え，ビヘイビアツリーはその状態におけるキャラクターの身体的行動を自由度高く設計するこ

とができる．そこで，AI Graph Editor では，ステートマシンとビヘイビアツリーの組み合わせを実装した．他の組み合わせについては，将来の仕事として残されている．本論文では，AI Graph はステートマシンとビヘイビアツリーの組み合わせの領域のみを扱う．第 12 章で述べる事例についても同様である．以下，AI Graph Editor について述べる．

11.1.1 AI Graph の原理

本節では，AI Graph の原理を述べる．ステートマシンは，状況認識が優れており，ビヘイビアツリーは連続した行動の小プラン（一連のビヘイビア）を定義できる点に長所がある（第 7.3.2 項，第 7.3.3 項参照）．この両者の長所を取り入れたステートマシンとビヘイビアツリーを階層的に入れ子構造で組み合わせる新しい手法を AI Graph と呼ぶ．

この設計以前に，以下のような状況があった．

(1) 第 7 章で説明した階層型ステートマシン（第 7.3.2 項）や階層型ゴール指向プランニングなどによって，意思決定の多くが階層構造となる．階層構造を取ることによって，階層がない場合に比べてコンパクトな記述となる．
(2) ステートマシン，ビヘイビアツリーは単独では限界がある．ステートマシンには動作を連続的に組合す手段がなく，ビヘイビアツリーは状態を表現することが難しい．ステートマシン，ビヘイビアツリーに限らず，それぞれの意思決定アルゴリズムには長所と短所があり，それらを組み合わせて使うことで，さまざまな意思決定を実現できる．

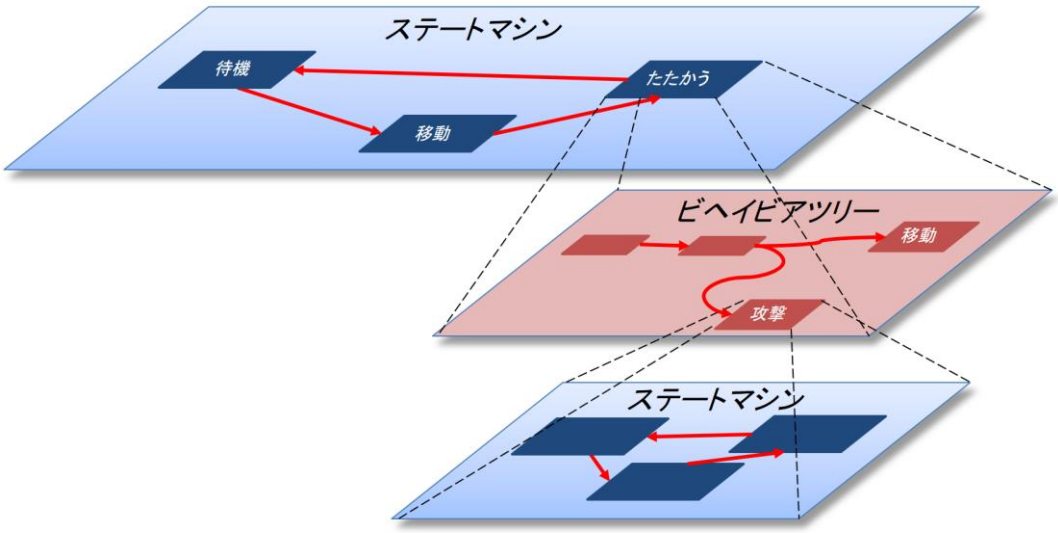


Fig. 11.1 ハイブリッド型ノードフォーマット [146]

AI Graph は第 7.3 節で述べた 7 つの意思決定アルゴリズムを階層的に組み合わせるシステムである．組み合わせ方としては，ある層における意思決定アルゴリズムのノードが，その下の層では，他のアルゴリズムのノードとして実装できる仕組みである．

たとえばステートマシンの 1 つのノードの中にビヘイビアツリーやステートマシンがあり，そのビヘイビアツリーのノードの中に更にステートマシンがある（Fig. 11.1）．この入れ子構造によってステートマシンの状態の分割による制御と，ビヘイビアツリーの滑らかな連続動作の記述，双方の長所を取り入れた意思決定表現を獲得することが可能となる．また上層から下層へ向かって問題のスケールを小さくすることができ，階層的に課題を解決するアーキテクチャにもなっている．この複合型のグラフが「AI Graph」であり，GUI（グラフィカルユーザーインターフェイス）によって構築するツール「AI Graph Editor」を製作した（Fig. 11.2）［146］.

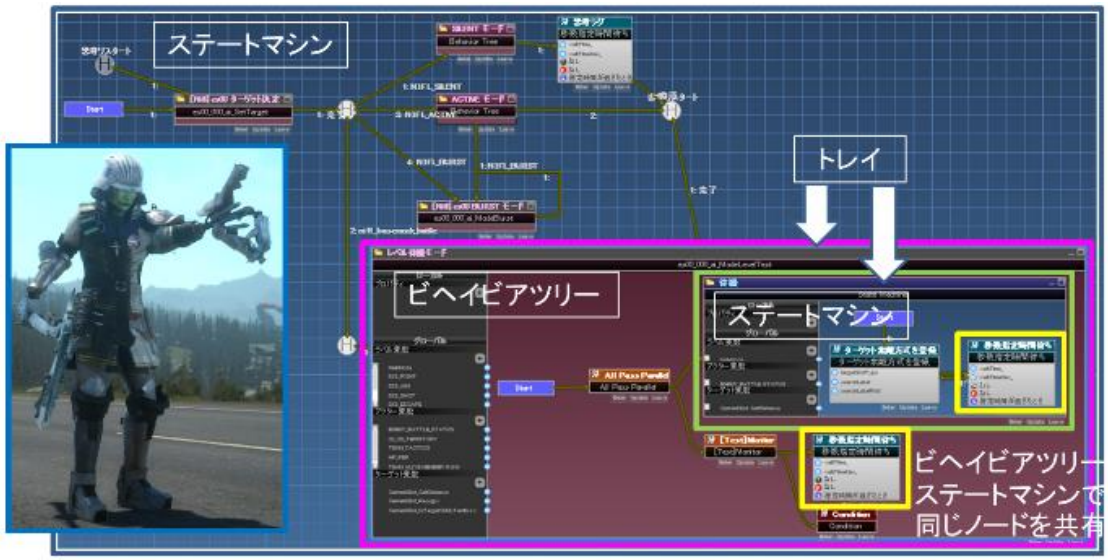


Fig. 11.2　AI Graph Editor （兵士の AI Graph）［146］

11.1.2　AI Graph Editor

本項では「AI Graph Editor」について述べる．AI Graph Editor は，AI Graph を実際にツール化したものであり，ステートマシンとビヘイビアツリーを製作することができる(Fig. 11.2).

「AI Graph Editor」の画面は, Fig. 11.3 のような形である. 中央の四角形の中が「AI Graph」であり，ノードとそれを結ぶコネクションの線からなる．ビヘイビア―ツリー，及びステートマシンが作成・表示される．ゲームからキャラクターAI が得た情報を記憶し，柔軟かつスムーズに下の階層に受け渡す仕組みとして，第 3.3.2 節で述べた「ブラックボード」がある．図の四角形の左端にある矩形の中は，上部がローカルブラックボード，下部がグローバ

ルブラックボードであり．ローカルブラックボードにはその層でのみ用いる記憶情報が記載され，グローバルブラックボードには，この AI Graph 全体で用いる記憶情報が記載される．

それぞれの階層がローカルブラックボードを持ち，センサーが収集した敵への距離などの情報を，グラフ上で線を結ぶことで高い階層から低い階層へ共有することができる．グローバルブラックボードに置いた情報は，すべての階層から参照することが可能になる．この仕組みによって，それぞれのキャラクターの持つ記憶を管理し，その情報を用いて各ノードのカスタマイズを行う．

キャラクターの持つ記憶としては，ブラックボード上には身体の状態，ゲームシステム情報（HP，アビリティ，装備，ゲームの進行情報など），仲間の情報（メタ AI からの指示や，バーティの情報），敵の情報（ターゲットの強さターゲットまでの距離や角度の情報），ナビ情報（位置検索結果）などが管理される．また逆に，そのトレイにおける思考の結果が書き込まれ，他のエージェントと共有されることもある．

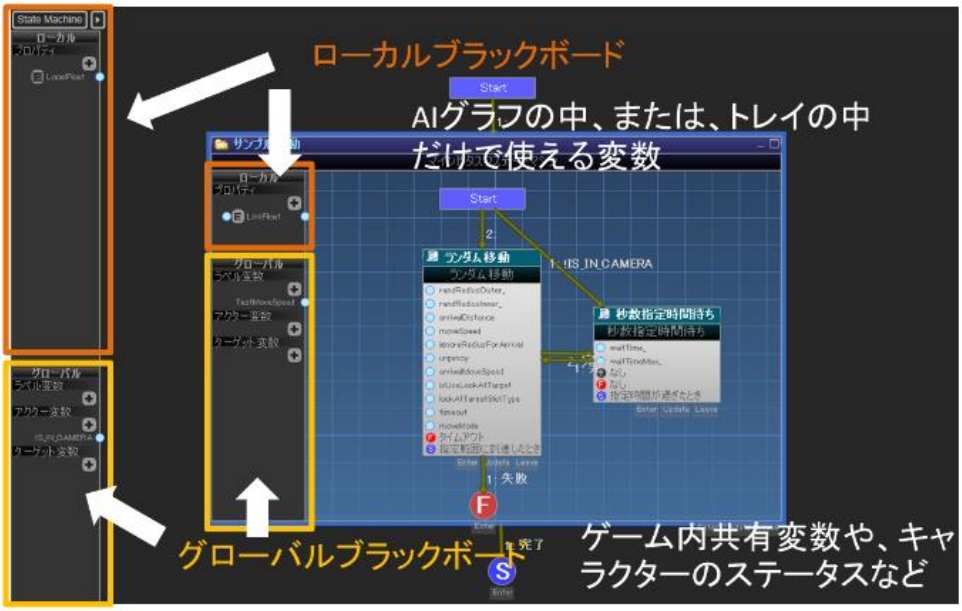


Fig. 11.3　AI Graph のブラックボード［146］

キャラクターの意思決定を作る仕事は，「AI Graph Editor」によって非エンジニア（レベルデザイナー）の仕事として確立することが可能となった．マウス操作でグラフを製作し，キーボードから数値を入力することでノードのカスタマイズを行う．これによって開発中りの仕様変更に伴う拡張に耐え，数値の調整をゲームデザイナーが独立して行えるようになり，FFXV において AI の量産が可能となった．

AI Graph Editor には階層型グラフシステムが導入され，データを適切な粒度で階層化できるようにすることで，AI データの拡張性（スケーラビリティ）を確保している．また，

トレイ（各階層のグラフ）を単位として階層化することで，データを適切な粒度に分割する．AI Graph Editor のデータはバイナリデータとしてシリアライズ（バイナリ形式によるデータパッケージング）され，ゲーム時にキャラクターの知能として実行される．AI Graph はメモリ上 20MB 程の使用量である．最上層から下位層に向かって実行するトレイが探索され，実行には下位階層がすべて終了して上位層に戻る．適切な粒度で階層化すると，不要な状態遷移の記述を減らし組み合わせ爆発を減らすことが可能となる［147］．多い場合は十数階層に及ぶグラフが作成された．

11.1.3　AI Graph の問題点

本項では AI Graph の問題点について述べる．AI Graph の問題点は，表形式のデータをコンパクトに表現することが難しい点にある．

ゲーム開発では，一般的にキャラクターごとの設定表がある．縦軸にモンスターの種類，横軸に各種の条件が書かれている．たとえば，敵の出現テーブルであれば，横軸には「草地」「砂浜」「高山」などの項目があり，出現させる場所にチェックを行う．ゲームプログラムはこのテーブルを読み込み，それぞれの場所にあったモンスターを出現させる．知能レイヤーにおいても，キャラクターごとに AI Graph やパラメータを変化させる表が必要となる．しかし，このような対応を AI Graph で記述しようとすると，膨大な AI Graph のノードを必要としてしまうため，末端のノードで表形式のデータを読み込み，場合分けを行う，という実装を行った．

モンスターであれば，中間レイヤーにおいてモンスターごとのパラメータ設定と AI Graph を指定する設定表，仲間キャラクターであれば，末端における状況における行動を指定する設定表が使用された．AI Graph だけでなく，この表形式のデータによって，キャラクターごとの知能のバリエーションが実現された．このような分類は AI Graph 上からは見えない処理であり，AI Graph のシステムがそのような分類を設計に取り込めていないことを示している．

Table 11.1　MCS-AI 動的連携モデルが必要とする機能と AI Graph の拡張機能の対応

MCS-AI 動的連携モデルの要件	AI Graph の拡張機能
多数のキャラクターAI を統一的に効率良く製作する	アセット化・オーバーライド機能
メタ AI とキャラクターAI のインターフェースを統一する	AI モード
一時的にキャラクターAI に演技的行動をさせる	トレイ割り込み実行

11.2 MCS-AI 動的連携モデルによる AI Graph の拡張

AI Graph の基本的特徴を前節までで述べてきた．しかし，AI Graph を MCS-AI 動的連携モデルに組み込むときには，AI Graph を拡張する必要がある（Table 11.1）．本節では，この拡張について述べる．具体的なその 3 つの拡張機能とは「アセット化・オーバーライド」「AI モード」「割り込み実行」である．第 11.2.1 項では MCS-AI 動的連携モデルが必要とする機能と AI Graph の拡張機能の対応について述べる．第 11.2.2 項では「アセット化・オーバーライド機能」について述べる．また第 11.2.3 項では，複数の AI Graph を使い分けるために「AI モード」について述べる．第 11.2.4 項では「トレイ割り込み実行」を説明する．

11.2.1 MCS-AI 動的連携モデルが必要とする拡張機能

MCS-AI 動的連携モデルでは，メタ AI，キャラクターAI，スパーシャル AI の相互の連携が必要となる．メタ AI からキャラクターAI に行動を指定する必要がある．しかし NPC は大量に製作される可能性が高く，それに伴い大量のキャラクターAI を制作する必要がある．そのために，一度作った AI Graph を再利用できるようにアセット化する機能，一度作った AI Graph の一部だけを元の AI Graph のデータアセットを壊すことなく変更する機能が必要である．これを実現する機能が「アセット化・オーバーライド機能」である（Fig. 11.4）．次に，そのように製作されたキャラクターAI に対して，メタ AI から制御を行うインターフェースが必要である．そこで第 7.2 節で述べた「モード型命令」の実装として，メタ AI からモードを指定するだけで NPC の行動を制御できる「AI モード」を開発した．これは「アセット化・オーバーライド機能」を用いて，すべてのキャラクターの基本となるステートマシンを最上位にアセットとして用意し，そのステートを NPC の種類に応じてオーバーライドして作り込んで行くことで，メタ AI から見たキャラクターAI のインターフェースを統一化する手法である．このインターフェースの単位が「AI モード」であり，メタ AI は NPC のキャラクターAI の AI モードを指定することで，演技や行動原理を切り替えることが可能となり，NPC にその場に適した振る舞いをさせることや，特定の演技をさせることできるようになる．AI モードを指定した後は，キャラクターAI の自律的行動が開始されるため，AI モードはメタ AI のゲーム全体の管理とキャラクターAI の自律性の双方を活かす仕組みとなっている．

さらに，キャラクターAI には．第 7.1 節で述べたように，メタ AI から AI モードによって大きく指定される行動スタイルよりも，一時的な行動を追加できる仕組みが必要となる．たとえば戦闘開始時にプレイヤーに飛びかかる，などである．任意の AI Graph を他の AI Graph に対して割り込み実行できる「トレイ割り込み実行」の機能を実装し，この機能によって AI Graph 上で短期的な演技の挿入が可能となった．

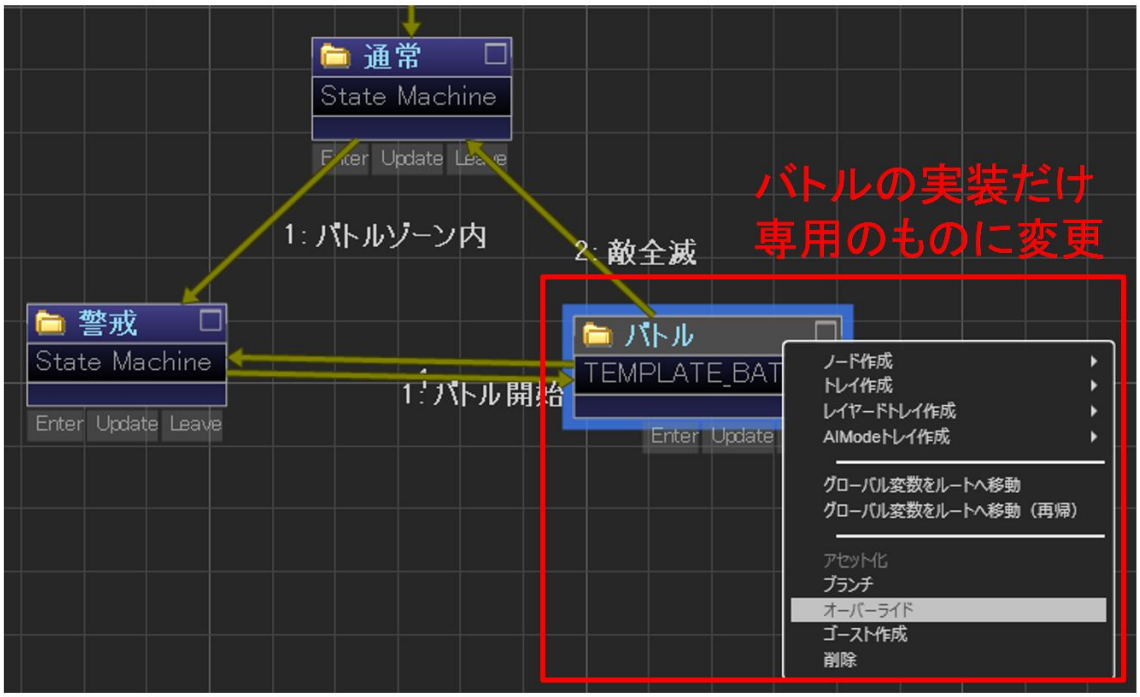


Fig. 11.4　アセット化されたステートマシンをオーバーライドする［146］

11.2.2　アセット化・オーバーライド

AI Graph Editor を用いてキャラクターAI の AI Graph が記述されるが，同じ種族や同じ分類のキャラクターたちの AI Graph の構造は類似しており，AI Graph 同士はパラメータの一部のみが異なる，あるいはグラフの一部分のみが異なる．そこで，キャラクターの行動を効率よく作成・編集するために，一度作った AI Graph をアセット化（固定した再利用可能なデータとすること）し，ライブラリとして再利用可能とし，さらにそれらを他の AI Graph から呼び出すことができると同時に，部分だけを書き換えることができるようにする「オーバーライド」機能が実装された（Fig. 11.4，Fig. 11.5）．たとえば，あるステートマシンをアセット化し，そのアセットを AI Graph から呼び出し，その中の一つのステートをオーバーライドする（書き換える）ことが可能である．

アセット化したトレイを部分的にオーバーライドすることにより，トレイにもともと定義されたメインの処理を変える必要がなく，各部分を書き換えては試すイテレーション・サイクルを行う環境を実現することができた．また，それぞれのキャラクターの特定部分のグラフだけを作れば全体のグラフを作れることになり，大幅な工程の効率化を実現した．

11.2.3　AI モード

すべてのキャラクターは,「AI モード」と呼ばれるステートが用意されている [148] (Fig. 11.5)．これは第 7.2 節で述べた「モード型命令」である．

「モード」はどのようなキャラクターでも持つ基本的な行動である．モードは複数あり「ウェイトモード」(待機)「Go To モード」(指定された地点へ向かう)「リード」(先導する)「フォロー」(ついて行く) などがある．モード間の遷移は共通に定義されている．このようなモードは，特に外部からキャラクターにゲームシーンに従った行動を行わせるために活用される．これらは第 7.1 節で述べた「演技用キャラクターAI」(Table 7.2) に対応する．

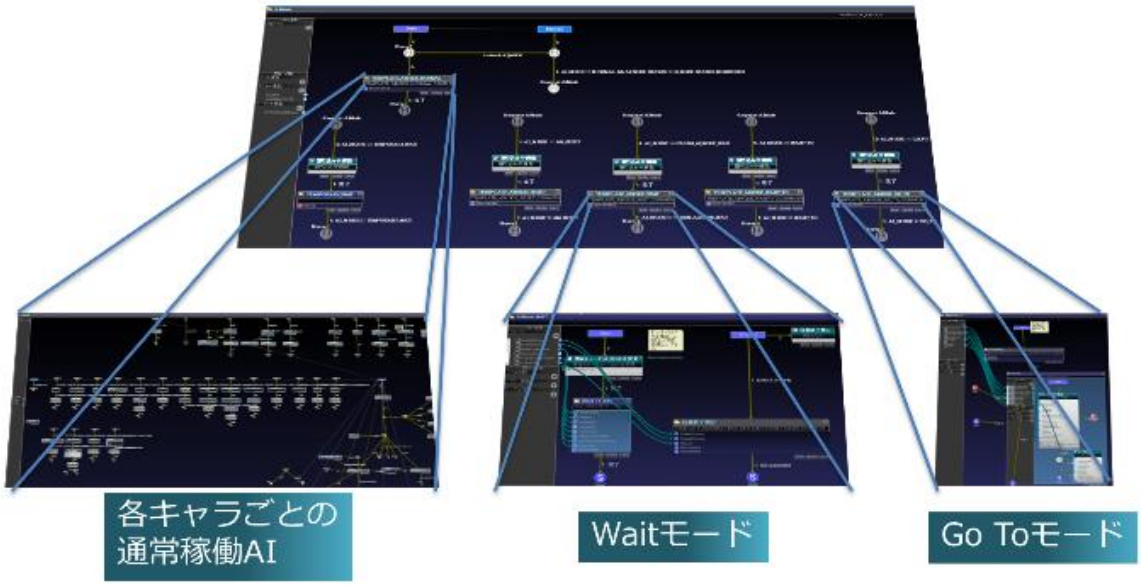


Fig. 11.5　AI Graph のオーバーライドとモード（仲間キャラクターの AI Graph）[146]

また「ノーマルモード」としてデフォルト状態を規定する「通常稼働 AI」があり，これは「自律型キャラクターAI」に対応する．ノンプレイヤー・キャラクターは自律型エージェントとしての側面と，プレイヤーを楽しませる役者として側面があり，この両者を AI モードによって実現している．

どのようなキャラクターも「モード」群を含めた基本となる「トレイ」が第 1 階層（最上層）にある（Fig. 11.5)．基本的な実装はモードとしてされているため，どのようなキャラクターもこの層だけで基本的な動作をすることが可能となる．各モードをオーバーライドすることによって，キャラクター毎の特徴を出すことができる．この仕組みによって，モード間の遷移は共通であるが，各モード内はキャラクターごとに異なるように作ることが可能となる．

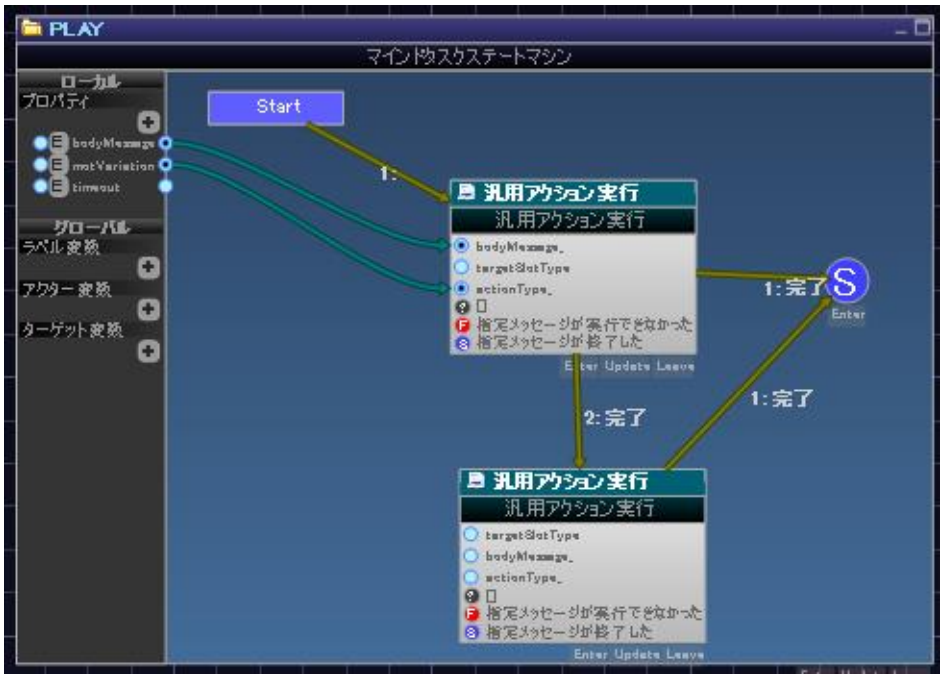


Fig. 11.6　「キャラクターに勢いよく近づく」AI Graph [146]

11.2.4　トレイ割り込み実行

AI Graph の中で一つのファイルや，一連の処理のシークエンスを「トレイ」と呼び，別の AI Graph から呼び出す仕組みを「トレイ割り込み実行」と言う．この機能によって，キャラクターに一時的な演技をさせることができる．つまり，自律型意思決定の中にシームレスに指定した演技を挿入することができる．呼ばされたトレイの処理が終わると，元の AI Graph に戻って戦闘を続ける．この時，ブラックボードを使って，呼び出す AI Graph から，呼び出される AI Graph へパラメータを渡すことができる．この仕組みによって，キャラクターに様々な一時的な演技を差し込むことができる．

たとえば，「モンスターが勢いよく走って来る」というトレイを作る（Fig. 11.6）．これを戦闘の AI Graph の冒頭に呼び出す．一定距離に近づいたら終了するように作っておく．終了すると戦闘の AI Graph に戻り，通常の戦闘を行う．このように，自律型意思決定と演出的意思決定をトレイ割り込み実行の仕組みを用いて，行き来することができる．

11.3　考察

本章では「AI Graph」について述べた．第 11.1 節で AI Graph は，キャラクターの複数の意思決定アルゴリズムを組み合わせて構築する仕組みとして設計されたことを述べた．第 11.1.1 では AI Graph の基本原理について述べた．第 11.1.2 項で AI Graph Editor につい

て述べた．AI Graph Editor を導入することで，非エンジニアでも AI Graph に含まれる技術を習得し用いることになり，開発水準を上げることができ，人工知能技術の質を高めることができる．第 11.1.3 節では AI Graph の問題点について述べた．これは大量の NPC の個々の特性を入力する設計が含まれていない問題点である．次に，第 11.2 節では．MCS-AI 動的連携モデルを実現するための AI Graph の拡張について述べた．そして，その具体的な３つの機能「アセット化・オーバーライド」「AI モード」「割り込み実行」について述べた．この拡張によって，AI Graph を MCS-AI 動的連携モデルを実現する技術として用いることができる．AI Graph と MCS-AI 動的連携モデルの応用事例を次の第 12 章で述べる．

12. FINAL FANTASY XV における MCS-AI 動的連携モデルの実装

本章では，『FINAL FANTASY XV』（スクウェア・エニックス，2016 年，以下 FFXV）（Fig. 12.1）における MCS-AI 動的連携モデルの実装を述べる［149］［150］．第 12.1 節では，本ゲームで AI に求められる要件をまとめ，FFXV における MCS-AI 動的連携モデルの導入の全体像について述べる．「キャラクターAI」に関しては第 12.2 節において，「メタ AI」に関しては第 12.3 節で述べる．「スパーシャル AI」に関しては第 12.4 節で述べる．

第 12.5 節，第 12.6 節では MCS-AI 動的連携モデルの上に構築された新しいシステムについて述べる．第 12.5 節ではメタ AI とキャラクターAI を応用した「プレイヤーAI」について述べる．第 12.6 では 3 つの AI の連携からなる「Face-to-Face 対話システム」について述べる．

第 12.7 節ではロギングとビジュアリゼーションについて述べる．第 12.8 節では，FFXV における MCS-AI 動的連携モデルの効果について述べる．

Fig. 12.1 『FINAL FANTASY XV』のゲーム画面

ゲーム産業における大型ゲームの開発では，その長期に渡る開発の中で，研究成果を実装し，より差別化され他社から抜きんでた特徴を持たせることが求められる．大型ゲームの開発は一般に 3〜7 年の期間を要する．その期間には研究を積み上げる期間と，それをゲーム

へ実装して行く期間がある．「AI Graph」「メタ AI」を含む MCS-AI 動的連携モデルの開発は基礎的な研究の上に開発に導入された技術であり，それ以外は開発を進めながら開発された．MCS-AI 動的連携モデルの上に，具体的な仲間 AI，モンスターAI，兵士 AI，プレイヤーAI，Face-to-Face 対話システムなどが実現された（Fig. 12.2）．

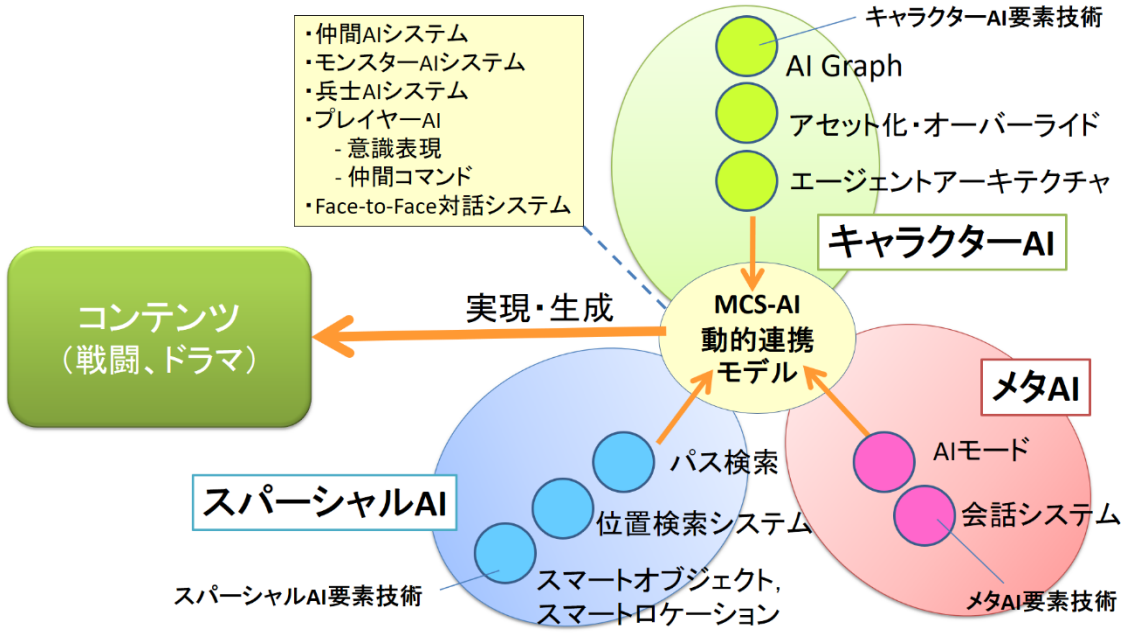


Fig. 12.2　FFXV における AI システム全体図

12.1　FFXV の MCS-AI 動的連携モデル導入の全体像

本節では FFXV において AI に求められる要件についてまとめ，FFXV における MCS-AI 動的連携モデルについて述べる．第 12.1.1 項では，FFXV における MCS-AI 動的連携モデルについて述べる．第 12.1.2 項では自律性意思決定と演技的意思決定の両者を含む全体的設計を示す．AI Graph を含む MCS-AI 動的連携モデルについて，第 12.1.3 項で説明する．この項では，すべてのキャラクターの AI Graph の最上層は，AI モードのステートマシンからなり，この AI モードをメタ AI から意思決定が指定することで，キャラクターがメタ AI からコントロールされることも説明する．第 12.1.4 項では，FFXV の実現すべきゲーム体験から MCS-AI 動的連携モデルの各 AI に必要とされる要素技術について考察を行う．第 12.1.5 節では，AI 開発チームの編成について説明する．

12.1.1　FFXV における MCS-AI 動的連携モデル

FFXV は広大なマップを仲間 3 人（3 人の NPC キャラクター）と共に，モンスター達と

戦いながら旅をする３Ｄアクション RPG である（Fig. 12.1）．戦闘は数十 km 四方のマップの至るところで行われる可能性があり，また草原，街，建物といったさまざまな地形があり，敵・味方の NPC はさまざまな場所で行動する必要がある．

仲間キャラクターはプレイヤー・キャラクターを守りながら戦闘する，或いは，会話し案内をする，という演技があり，自律型キャラクターAI の機能と，演技的機能が必要とされる．モンスター・キャラクターも，登場シーンなど飛び掛かる演出と戦闘をする自律型キャラクターAI の機能が必要である．また，モンスターはある程度広い領域を自由に移動できるため，戦闘はフィールドのどの地点でも起こり得る.そこで，その場を活用して戦うためにスパーシャル AI が活用され，全体の戦闘の流れを作るためにメタ AI が全体を制御する．

そこで，

（１） 戦闘全体の流れ，物語的場面を作るメタ AI
（２） 仲間キャラクターと敵キャラクターの自律型キャラクターAI
（３） 仲間キャラクター，メタ AI にその場所の特徴や状況を提供するスパーシャル AI

を組み合わせた MCS-AI 動的連携モデルによって AI システムを構成する（Fig. 12.3）．

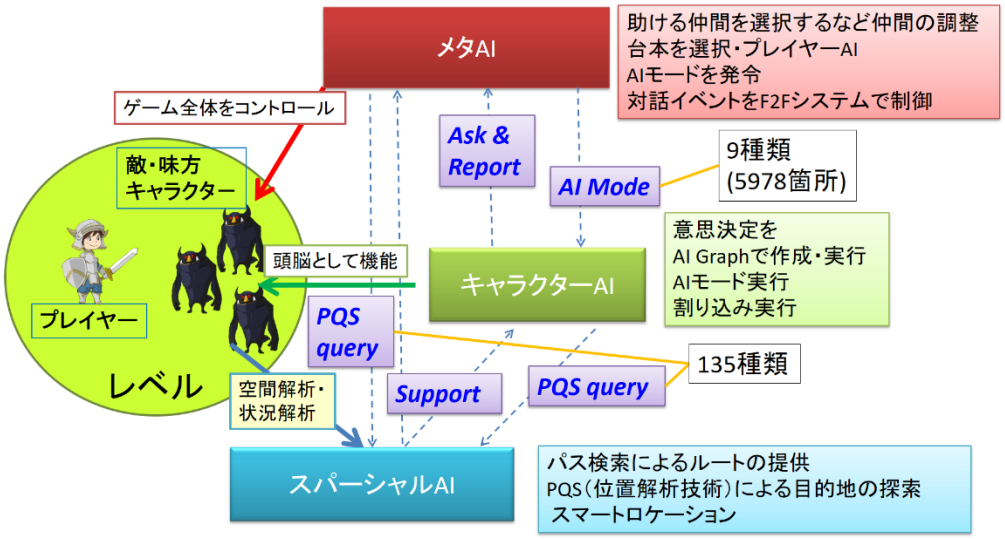


Fig. 12.3　『FINAL FANTASY XV』における MCS-AI 動的連携モデル

メタ AI からキャラクターAI へは，AI モードによって命令が伝えられる．メタ AI，キャラクターAI からスパーシャル AI への要求は PQS クエリーによって指定される．また，スパーシャル AI からはスマートオブジェクト，スマートロケーションの仕組みによってキャラクターAI を制御する．このように AI モードと PQS クエリーによって，メタ AI，キャラクターAI，スパーシャル AI 間のコミュニケーションの形式が統一され，またその統一され

たフォーマットの上に必要なモードとクエリーが構築される．

12.1.2 自律型意思決定と演技的意思決定

本節では，FFXV のキャラクターAI の意思決定が持つ二つの側面について述べる．FFXV は，物語的ゲームであり，アクションゲームであるため，第 7.1 節で述べたように，環境で自律的に活動する自律型エージェントとしての思考と，同時にゲームシナリオ上で与えられた役割の通りに演技をする能力が必要とされる．FFXV のキャラクターAI もまた，両者を兼ね備えることが必要とされる（Fig. 12.4）．たとえば，ノンプレイヤー・キャラクターは「戦闘が始まる前に，剣を振り上げて掛け声をかける」などをしてやる気があることを見せたり，街に到着した時にこれから行うミッションをプレイヤー・キャラクターの側へ来て身振りや言葉で説明したり，仲間が負傷した時に泣きながら側に駆け寄るなど，その場に合った演技をする必要がある．つまり，

(a) 自律性を持つ自律的意思決定
(b) ゲームを俯瞰的に見て文脈に沿った演技をさせる「メタ AI」
(c) 「メタ AI」からの命令を受けて，演技をする演技的意思決定

が必要である．

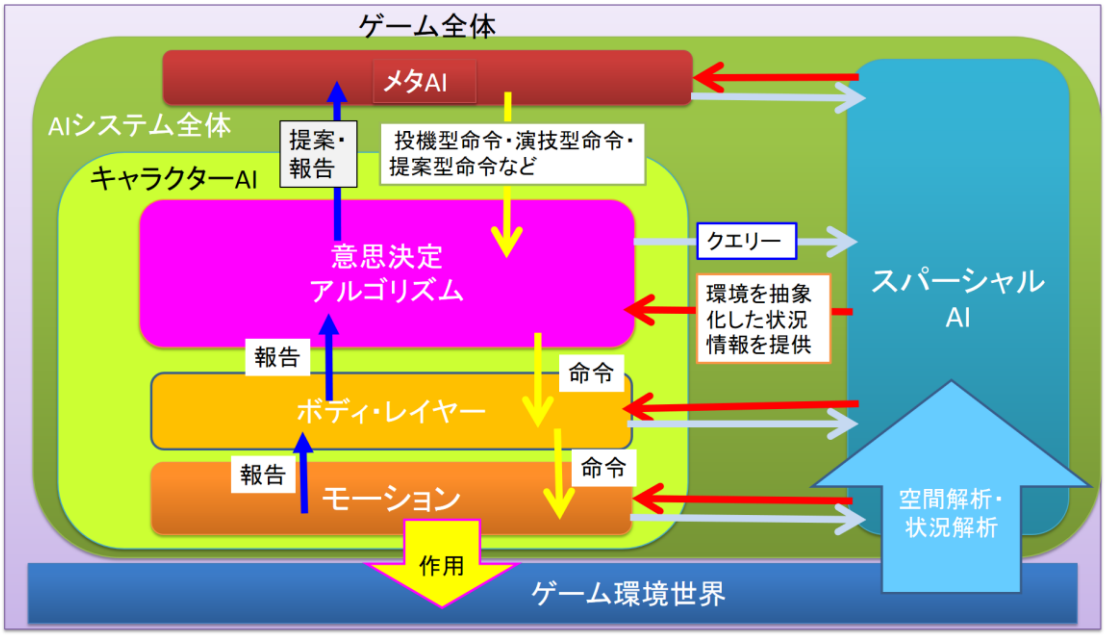


Fig. 12.4　MCS-AI 動的連携モデルにおけるキャラクターAI の 2 つの側面（Fig.7.1 と同一）

キャラクターAI には，ゲーム産業では一般に「多様性」「拡張性」「カスタマイズ性」が

必要とされる．「多様性」は統一的な仕組みでさまざまな AI を作ることができる仕組みであり，「拡張性」は必要に応じて知的機能を拡張して行ける機能であり，「カスタマイズ性」はパラメータやグラフ構造をエディタやツールから調整できることである．これらを統一的に実現する仕組みとして「AI Graph」が導入される．「自律型意思決定」「演技用意思決定」も，AI Graph を用いて構築される．

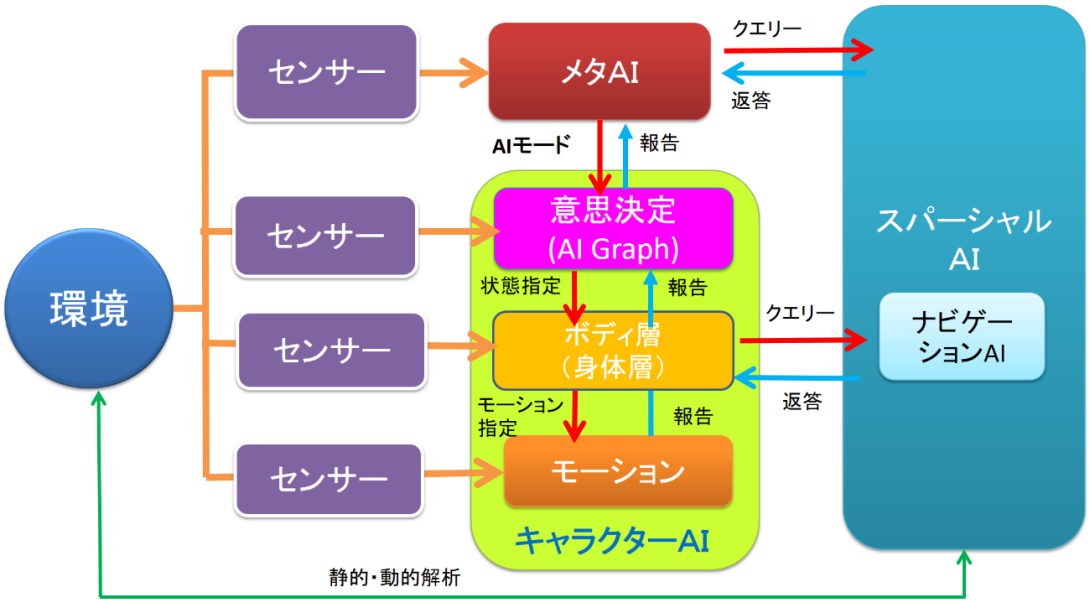


Fig. 12.5　FFXV における MCS-AI 動的連携モデル

12.1.3 MCS-AI 動的連携モデルの全体設計

本節では，MCS-AI 動的連携モデルのソフトウェア構造を示す．第 3.5.3 項で述べたような「サブサンプション・アーキテクチャ」によって構造化される（Fig. 12.5）．メタ AI から意思決定へは AI モードを指定することで命令を与える.また，意思決定は AI Graph で形成される．AI Graph Editor を用いて各チームの担当者が作成する．

FFXV では，意思決定とアニメーション層の間に「身体層」（ボディ・レイヤー）を導入した（Fig. 12.5）.「身体層」は身体の状態を大きく定義している [151].「走っている」「ジャンプしている」「梯子を登っている」状態（ステート）を管理する．身体層は状態ごとに行える行動を制限する．例えば「梯子を昇っている」時に「剣を振る」ことはできない，などの調整を行うことで，意思決定とアニメーションの独立性を高める．さらに反射的に行った行動（ダメージリアクションなど）による身体の状態変化を意思決定層に知らせる役割を持つ [152].

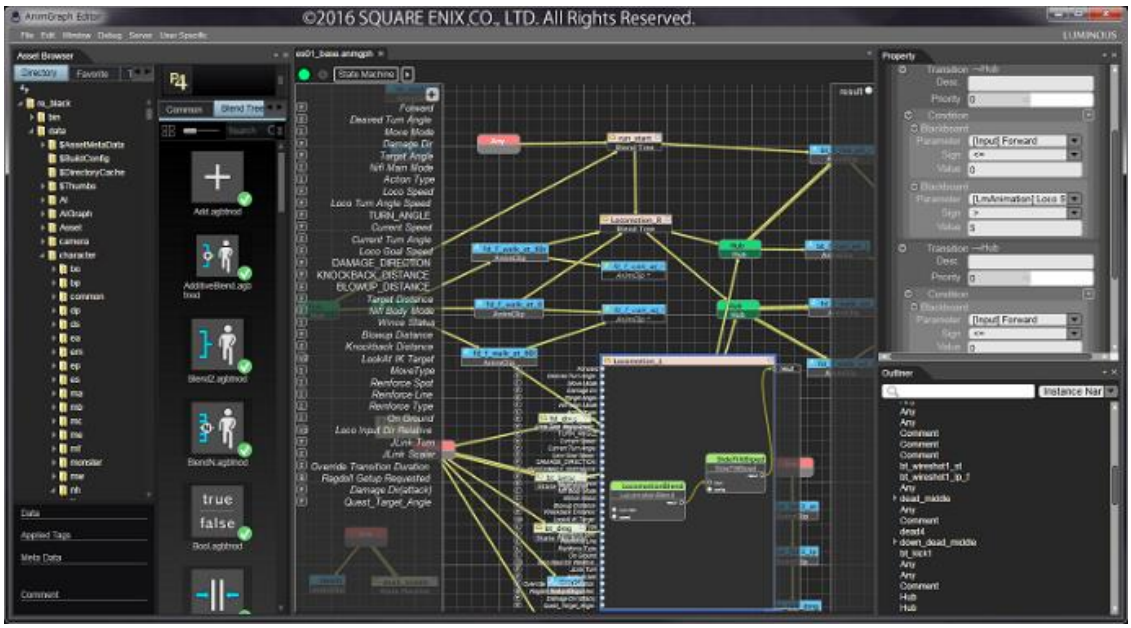


Fig. 12.6　FFXV の AnimGraph

最下層にあるのが「モーション層」である．その構造はモーションデータを持つノードからなるグラフである．モーション層ではアニメーション・ノード間の遷移が滑らかに行われるように制御される．キャラクターの身体を制御するには，知能レイヤーから身体レイヤーへ「走る」「剣を振る」「歩く」「座る」といった行動を指定するメッセージを出す必要がある．中間層を通して，これらの行動に対応するアニメーションを再生する．アニメーションの選択やタイミング制御をおこなうのが「AnimGraph」である．AnimGraph はステートマシン構造になっており，再生するアニメーション・データに対応するノード間を遷移する（Fig. 12.6）．

12.1.4 FFXV のユーザー体験と MCS-AI 動的連携モデルの技術要素

本節では，FFXV が実現するべきユーザー体験を示し，これを MCS-AI 動的連携モデルで実現する手法について述べる．まず FFXV について実現するべきユーザー体験の代表的な例を Table 12.1 に示す．このようなユーザー体験についてメタ AI，キャラクターAI，スパーシャル AI に必要な機能要件を分解し，必要な技術要素として抽出する．

このようなユーザー体験は，本タイトルの開発の最初に開発者全員に MCS-AI 動的連携モデルを解説し，ゲームデザイナーからヒアリングを行うことによって聴き取る．そこから AI 技術チームの中で 3 つの AI の機能へと分解していく．

たとえば「プレイヤーを仲間キャラクターがかばう」というユーザー体験を実現することを考える．これを実行する発端となるのは，まずメタ AI が「適切な仲間キャラクターを一体選び，プレイヤーを助ける」ように命令を出す．そしてスパーシャル AI が「かばうのに

適した位置を位置検索技術から決定」する. その座標を仲間キャラクターの AI が受け取り，「目的地へ向かう」をセットする．スパーシャル AI は「目的地までパス検索」し，パスを仲間キャラクターへ渡す.

「仲間 NPC がプレイヤーの意図を推定して先を走る」では，まずメタ AI が，プレイヤーがある目的に向かって走っている状況を把握し，仲間 NPC にプレイヤーと一緒に走るように命令を出す. その場合に, スパーシャル AI はプレイヤーと推定される目的地の方向に，仲間キャラクターが目標とする地点を検出し，またプレイヤーの前方に位置する場所を提供する．メタ AI から指令を受けたキャラクターは，スパーシャル AI の情報をもとにプレイヤーの前方で走る行動を行う.

このように 3 つの AI の持つ機能を活用することで，ユーザー体験を作り出すことができる．それぞれの AI の機能はたいへんシンプルに還元されており，逆にそれゆえに汎用性の高い機能となっている.

このように FFXV で実現するべきユーザー体験から，MCS-AI 動的連携モデルが内包するべき技術要素を抽出すると，Table 12.2 のような要素群となる．これらは MCS-AI 動的連携モデルの各 AI にそれぞれモジュールとして実装される．以下，第 12.1.5 節では FFXV の AI 開発のチーム編成について，第 12.2 節では，FFXV における MCS-AI 動的連携モデルにおける「キャラクターAI」と技術要素について，第 12.3 節では「メタ AI」とその技術について，第 12.4 節では「スパーシャル AI」とその技術要素について説明する.

Table 12.1　FFXV におけるユーザー体験の一例

ユーザーエクスペリエンス	メタ AI	キャラクターAI	スパーシャル AI
仲間キャラクターが近寄って体力を回復してくれる	適切な仲間キャラクターを選んで命令を出す	目的地へ向かい，ヒールをかける行動をセットする	プレイヤーまでのパスを検索する
プレイヤーの近くに来て話す	話すように命令を出す	目的地へ向かい，台詞を再生する	プレイヤーの近くの立ち位置を解析し，そこまでパス検索する
プレイヤーの意図を推定して先を走る	プレイヤーの先を走るように命令を出す	目的地へ向かって走る	目的地を推定して，仲間キャラクターの目指す位置を解析し決定し，パス検索を行う
モンスターがプレイヤーについて来る	メタ AI が「フォロー」モードをセットする	目的地に向かって走る	プレイヤーの位置に定期的にパス検索をかける

プレイヤーをかばう	メタ AI がプレイヤーを助けるように命令を出す	目的地でターゲットの方向を向く	プレイヤーと攻撃するモンスターの間で適切な位置を解析して決定する
戦闘中に戦況に応じた会話をする	メタ AI が適切な台本を選択する	戦いながら，決められた台詞を話す	戦闘の適切な位置をリアルタイムで解析する
モンスターが戦闘で適切な位置と軌道を取って戦う	戦闘モードを指定する	スパーシャル AI の指定した位置に移動しつつ戦闘する	プレイヤー達との相対的に適切な戦闘位置を提供する
仲間キャラクターが隠れながら進む	プレイヤーをフォローするモードを指定する	隠れるポイントからポイントへ移動する	建物のかげで隠れることができるポイントを見つける

Table 12.2　FFXV の基本技術リスト

人工知能技術	どんな問題を解決するか
MCS-AI動的連携モデル	AIシステム全体の構築モデル
メタAI	ゲームの流れを調整する
AIモード	キャラクターAIの行動方針を指定する
会話システム	場面に適した台本を選択する
Face-to-Face システム	キャラクターAI, スパーシャルAIと連携して対面でNPCを会話させる
スパーシャルAI	地形を認識する
ナビゲーション・メッシュ	大局的な経路を検索する
スマートウェイポイント	局所的な経路を設定する
位置検索技術（PQS, Point Query System）	戦術的な位置をリアルタイムで発見する
スマートロケーション	場がキャラクターを制御するシステム
キャラクターAI	キャラクターの頭脳を作る。
エージェントアーキテクチャ	キャラクターの知能全体の枠組み
ブラックボードアーキテクチャ	キャラクターの記憶
意思決定（ステートマシン）	ステートを基本とする意思決定の仕組み
意思決定（ビヘイビアツリー）	ビヘイビアを基本とする意思決定の仕組み
意思決定（AI Graph）	上記二つのハイブリッド型意思決定システム
AI Graph Editor	量産化を可能にするGUIツール
アセット化・オーバーライド	開発を効率化するための登録・ライブラリ機能
割り込み実行	演技用のAI Graphを割り込み実行する仕組み
アニメーション・グラフ	アニメーションの遷移を定義する
ボディ・レイヤー	身体の状態をステートで制御・管理する
サブサンプションアーキテクチャ	意思決定と身体の接続
プレイヤーAI	
ロギングとビジュアリゼーション	ゲーム全体の動作を可視化する

12.1.5 AI 開発チーム編成

本節では FFXV の AI 作成に関するチーム編成を述べる（Fig. 12.7）. MCS-AI 動的連携モデルは，複数の開発チームが分業して協調できる体制を作るのに適している.「AI 技術チーム」があり，このゲームタイトル開発前から，ゲームタイトルに予想される要件を考慮しつつ，キャラクターAI 技術としては「AI Graph」,「スパーシャル AI」としてはナビゲーション・メッシュ・システム，位置検索技術として PQS（Point Query System）のツールとランタイム（実行ソフトウエア）の設計・製作を行った. 実際のゲームタイトル開発が始まると，「仲間作成チーム」「モンスター作成チーム」「兵士作成チーム」それぞれ「AI Graph Editor」を使用した.「AI 技術チーム」は，個々のフィードバックの要求に応えながら，「AI Graph Editor」の改善を重ねて行くことで，一つのチームに対する改善が他のチームへと還元された.「仲間作成チーム」はメタ AI の実装も行った. スクリプティングの場合は，それぞれの場面に応じて作成していくが，MCS-AI 動的連携モデルはそれぞれの仕事がモジュール化されお互いの仕事に役立つ相乗的な仕組みとなっている.

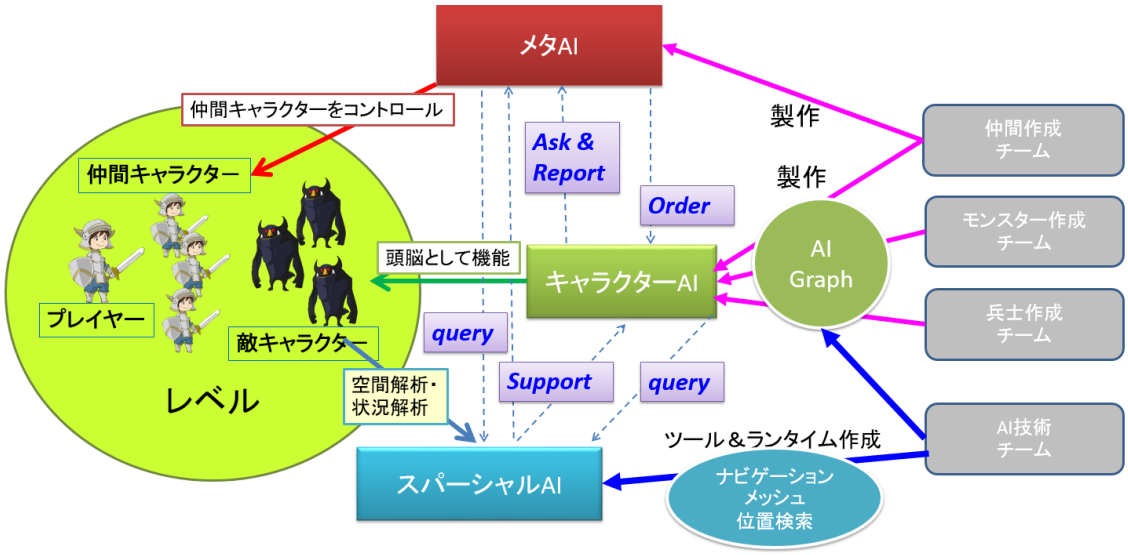


Fig. 12.7　FFXV AI システムと AI 製作チームの構成

12.2　キャラクターAI

本節では FFXV の MCS-AI 動的連携モデルにおけるキャラクターAI について述べる. キャラクターはエージェント・アーキテクチャ（3.3.1 参照）を持つ [153]. 意思決定モジュールは AI Graph によって形成される. 以下，第 12.2.1 節で「センサー」モジュール，第 12.2.2 節で「意思決定」モジュールについて述べる. 意思決定に関しては モンスター，仲間，兵士，それぞれの AI Graph について第 12.2.3 項，第 12.2.4 項，12.2.5 項にて説明す

る．

12.2.1　センサー

FFXV のモンスターでは二種類の扇状センサーが設定されている．一つは前方に展開した中心視野としての扇状センサー領域で，もう一つはやや後方にも広がった周辺視野としての扇状センサー領域である．それぞれの領域で取得できる情報の種類が異なり，対象発見までの時間が領域によって異なる．中心視野の方が認知するまでの時間が早い処理となっている（Fig. 12.8）．後方の視野は，ぼんやりとした認知であり，明確にターゲットとなるまでに中心視野よりも時間がかかるように差異化している［154］.

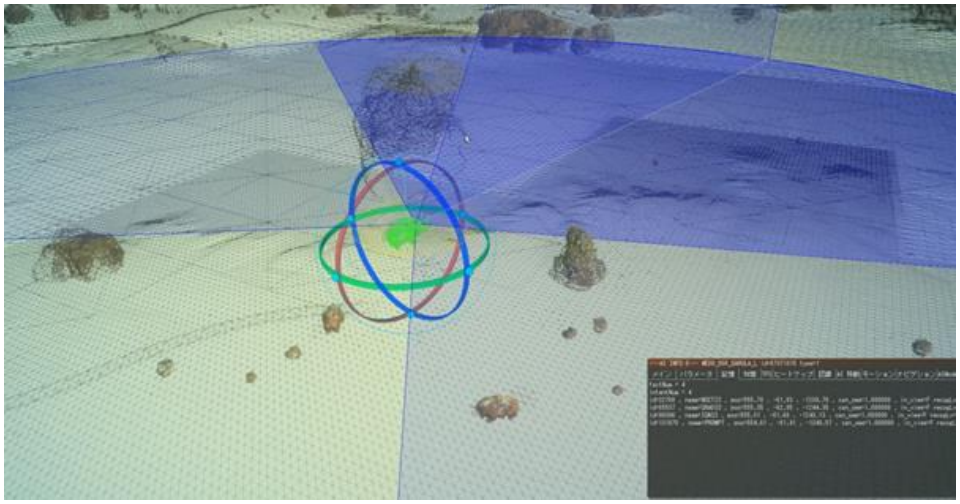


Fig. 12.8　キャラクターの扇状センサー［146］

センサーで認識した敵キャラクターには，それぞれのキャラクターごとに優先度の付け方が存在し，最終的に攻撃対象キャラクターが一体決定される［154］.

12.2.2　AI Graph による意思決定

本項では，FFXV における AI Graph の利用の全体像を述べる．すべてのキャラクターは，AI モードを要素とするステートマシンのアセット化されたトレイからなる（Fig. 11.5）.「ノーマルモード」にキャラクターの通常の自律型思考の AI Graph が記述される．外部からモードの指定がない場合は，このモードが実行される．それぞれの NPC に合わせて，各モードをオーバーライドすることによって，キャラクターの特色を記述する．たとえば，待機しているときの「ウェイト」モードを「頭をかく」「きょろきょろする」などの挙動をオーバーライドすることによって，各キャラクターに行動的特徴を加えることができる．

ノーマルモードにおける「モンスター」「兵士」「仲間キャラクター」において使用された AI Graph は，それぞれのキャラクター種別において，それぞれ使用傾向が異なる結果となった（Table 12.3）．これは，それぞれのキャラクターに求められる役割と知的機能が異なるからである．「モンスター」「兵士」は特殊なキャラクターを除いてプレイヤーの敵として，一度の戦闘の中にしか現れない．そこで限定されたシチュエーションの中で数秒～数分の挙動で完結する．一方で，仲間キャラクターはゲーム開始時からエンディングまで数十時間に渡る旅を，戦闘，街，車中など，さまざまなシチュエーションで対応せねばならない．この違いは，AI Graph の階層構造に現れている．モンスターや兵士の階層が 3 階層構造であることに対して，仲間キャラクターの AI Graph の階層は 15 階層とより深い階層になっているのは，仲間キャラクターが対応するべきシチュエーションが多岐に渡るからである．

以下，それぞれ「モンスター」「兵士」「仲間」キャラクターの AI Graph について説明を行う．

Table 12.3　AI Graph のキャラクター種別ごとの使用結果

キャラクター種別 / AI Graph 要素	モンスター	兵士	仲間
ステートマシン	○	○	○
ビヘイビアツリー	○	○	×
AIモード	○	○	○
階層構造	○ （3階層）	○ （3階層）	○ （15階層）
アセット化	○	○	○
（AIモード以外で） オーバーライド	×	○	×
ブラックボード	○	○	○

12.2.3　モンスター・キャラクターの AI Graph

本項では，モンスター・キャラクターの意思決定の構造について述べる．モンスターはおよそ 40 種類あり，また同じ種類でも 3 段階の大きさや様々なバリエーションがある．そこで，基本的な思考は同一でも，モンスター間のバリエーションを効率的に出す必要がある．それを実現する仕組みとして，AI Graph のアセット化が多用された [154].

知能レイヤーは，全部で 3 階層になっている．第 1 階層は，AI Graph 上のステートマシンである．これが全モンスター共通である．第 2 階層は AI Graph を呼び出すのではなく，表形式のデータとなっている．ここで，各モンスターに応じたパラメータと，第 3 階層で呼

び出す AI Graph が指定される．多数のモンスターの管理をこの表形式のデータを用いて行う開発体制である．第 3 階層として呼び出されるビヘイビアツリーは，すべてアセット化されたビヘイビアツリーであり，第 2 階層の表からアセット化 ID によって呼び出される．このような表形式による管理が必要な理由は，モンスターは他の「兵士」「仲間キャラクター」に比べて種類が多いため，一覧性による管理を必要とするためである．

オーバーライド機能はボディ・レイヤーの AI Graph で飛行モンスターと地上モンスターの差異を出すために使われただけで，知能レイヤーの中では使われなかった．ブラックボードは通常的にどの AI Graph でも活用された．また，モンスターが一斉にプレイヤーに飛び掛かって一方的な戦闘にならないように，ゲーム側から戦闘するモンスター以外，戦闘させないように「クールダウン時間」(戦闘禁止時間) がブラックボードを通じて設定している．この時間が 0 にならないうちは，モンスターはプレイヤーに襲い掛かることができない．

全体として，第 1，3 階層の AI Graph を汎用的に用意し，第 2 階層の表からパラメータと使用する第 3 階層の AI Graph を指定することで，できるだけ既存のアセット化された AI Graph を再利用し，多数のモンスターの AI Graph を量産する仕組みとなっている．

12.2.4 兵士キャラクターの AI Graph

本項では，兵士キャラクターの意思決定の構造について述べる．兵士を作るチームの担当は，「ニフル兵」と呼ばれる機械兵隊と，数 m の中型の搭乗型兵器「魔導アーマー」である [155]．両者とも，知能レイヤーは 3 階層の構造である．ニフル兵も魔導アーマーも，フィールドや基地などゲーム全域で行動し，またさまざまな作戦イベント（ゲームシナリオ上発生する戦闘イベント）に参加することが多い．そこで第 1 層はステートマシンで，作戦や戦闘のタイプなど，対応する状況を状態として分類することで，AI Graph の肥大化を全体が発散することを防ぐ．第 2 階層は，ビヘイビアツリーで，ここで固有の行動ロジックが構成される．第 3 階層はステートマシンで，ボディ・レイヤーに対して身体動作を指定する．

また異なる作戦において，類似した行動を取ることも多く，一度記述した AI Graph を「アセット化」し再利用することで，二重に記述する工程をなくし，コピー＆ペーストを避けることで保守性を高めた．一方でオーバーライドを使う機会は極めて少なかった．また，部隊として，味方同士でコミュニケーションを取る必要があり，各キャラクターAI が知能レイヤーの AI Graph において思考した結果をブラックボードに書き込み，上位のチーム AI がそれを読み取り，兵士全体の意思決定状況をリアルタイムに把握しチーム全体の管理を行った．

12.2.5 仲間キャラクターの AI Graph

本項では，仲間キャラクターの意思決定の構造について述べる．仲間キャラクターの知

能レイヤーでは，ステートマシンが多用されており，ビヘイビアツリーは使用されていない [156]．兵士やモンスターは，プレイヤーの前に数秒～数分の間，現れて倒されるか，ゲーム進行と共に画面の外へ消えてしまうので，行為の継続的連続性が必要とされない．一方仲間キャラクターは継続的にプレイヤーの周囲で活動し続ける．そこで，ステートで継続的に状態を管理する方針のもと，ステートマシンのみで 15 階層の AI Graph が構築された．これは仲間キャラクターが対応するべき状態の多さを反映している．仲間キャラクターは通常 3 人であるが，それぞれ個別の AI Graph ではなく，同一の AI Graph のアセットを使用しており，ステートの分岐でキャラクター毎に個別化されている．

第 1 階層では，AI モードごとに状態が管理されている．またその下の階層では，「身体の動作」を管理するステートと，「顔の向きや台詞など」意識的な動作を管理するステートが並行的に動作するように作られている．身体動作をブラックボードに書き込みながら，意識的動作ステートから身体状態を参照し動作を決定する形となっている．たとえば，「走る」「歩く」「剣を振る」などが身体状態であり，その時の表情や台詞は意識的動作のステートであり，身体状態を考慮した意識的動作が決定される．

末端の動作選択は表データが使用されており，各行動のトリガー条件と優先度フラグが設定されている．AI Graph の末端では，この表が呼び出され，その時の状態の条件に適合する行動を選び出す．表データが用いられたのは，キャラクターが取りえる挙動の数が十数個と多いために，AI Graph で書くには適していないからである．たとえば，「走る」にしてもいくつかのバリエーションがあり，体力や状況によってモーションを変える必要があり，その選択は表データを参照することで決定される．

12.2.6　キャラクターAI の結果と考察

AI Graph の導入によって，キャラクターの意思決定システムは AI Graph Editor 上の製作に集約され，すべてのキャラクターの意思決定システムは，統一的なフォーマットで記述されることになった．AI Graph によるキャラクターAI の製作過程において，自律型エージェントとしての思考と，ステージごとにゲームシナリオから要求されるキャラクターとしての行動を使い分ける必要が生まれ，「AI モード」が導入された．どのようなキャラクターAI を作る場合も，まず共通の「AI モード」のステートマシンのトレイを準備し，各モードをオーバーライドによってカスタマイズしていく手順が設定された．

「AI モード」をステートとするステートマシンを，すべての AI Graph の基本とすることで，AI Graph の製作者は，各モードの AI Graph の設計に集中できるようになった．また，モードごとの AI Graph が実装されることで，再利用性が高くなった．AI モードの中でも「ノーマルモード」は自律的に行動するモードであるが，キャラクターに求められる行動のタイプに応じて，AI Graph の組み方（ビヘイビアツリー，ステートマシンの組み合わせ方）や技術的要素（パス検索，表データの使用）が異なり，用意した技術の組み合わせによる効率的

な AI Graph 作成を可能にしたと同時に，人工知能の様々なバリエーションの実現を可能にした．

どのキャラクターも最上位層はステートマシンが使用された．全体として，階層型ステートマシンで状態数を発散しないように抑えて，末端の層は多様な身体的行動が取れるようにビヘイビアツリー，或いは表データが用いられた．この開発基盤の上で AI Graph に慣れていない開発者でも，迅速に AI Graph の作業に入ることが可能となった．

ビヘイビアツリーとステートマシンの効果

ビヘイビアツリーとステートマシンを組み合わせた AI Graph は，キャラクターの思考のデータに統一的なフォーマットを与え可読性を高めた．また，グラフ製作を AI Graph Editor 上で可能にしたことで，エンジニア以外のゲームデザイナー，プランナーに AI の製作に参加する体制を可能とした．また AI Graph に起因するバグは，AI Graph Editor から生み出されたバグとして，その再発を防ぐために AI Graph Editor 上に制約が加えられる形でフィードバックされ，開発の進行と共に安定性を獲得して行った．

表，及び AI モードの効果

また末端のノードにおいて，キャラクターや場所依存の分岐処理は，表データの活用によって集中的に管理され，AI Graph の記載量は大きく削減された．AI モードの導入は，全キャラクターの AI Graph の基本部分に統一的な形式を与え，AI Graph に一定の品質を担保すると同時に，その拡張によって思考のバリエーションを実装しやすい基礎を与えた．これらの開発効率化効果は，AI Graph をコンパクトにすると同時に，その品質と安定性を高め，ゲーム全体の品質の向上に貢献した．

このように，AI Graph は，キャラクターAI で最もよく使われるステートマシンとビヘイビアツリーを組み合わせることを可能とする．ステートマシン，ビヘイビアツリー単体に比べて，意思決定の表現能力を向上させている．またグラフィカルに可読性が上がり，意思決定におけるバグを AI Graph Editor に集中させ，デバッグを効率化できた．一方で，このスタイルが苦手とする表形式，ルールベース，ユーティリティ・ベースなどが，ノードの中の実装として埋め込まれている形となっている．その部分は AI Graph Editor ユーザーの負担となり，AI Graph の可読性を損なっている部分である

12.3 メタ AI

本節では，FFXV におけるメタ AI の役割と機能について述べる．第 12.3.1 項では，メタ AI と AI モードについて述べる．第 12.3.2 項では，メタ AI による戦闘のコントロールについて述べる．第 12.3.3 項では，会話におけるメタ AI の役割について述べる．

FFXV におけるメタ AI は AI モードを通じて，キャラクターへ干渉する．FFXV におけるメタ AI は，ゲームステージ上の状態を俯瞰的に観測しながら，キャラクターに指示を与える［99］［157］．一般にメタ AI には上から頻繁に制御をかける強いメタ AI と，最小限の制御に留める弱いメタ AI があるが，FFXV の場合は弱いメタ AI である．キャラクターは自律型キャラクターAI を持ち，弱いメタ AI はキャラクターの自律性を最大限利用しながら，キャラクター群をコントロールする．仲間キャラクターは常にプレイヤーの周囲におり，仲間キャラクターがユーザーから見て人間的な振る舞いを常に取るためには，キャラクターAI のみならず，メタ AI のコントロールを必要とする．メタ AI は，キャラクターAI による挙動が自然なものであるように監視し，コントロールする．また，「仲間キャラクター」がプレイヤーの仲間であることを行動によって表現するために，仲間キャラクターに指示を出す．

Table 12.4　FFXV で作られた AI モードの種類・機能［148］

種類	機能
リード	プレイヤーをリードする
フォロー	プレイヤーをフォローする
ルート	指定したルートをたどる
ワープトゥ	指定したポイントへ一瞬で移動する
プレイモーション	特定のモーションを再生する
リセット	モードをリセットしてノーマルモードへ遷移
ゴートウ	指定の場所に移動する
ウェイト/ターン	雑談など待機状態・特定の方向へ向く

12.3.1　メタ AI と AI モード

すべてのキャラクターは，「モード」と呼ばれるステートが用意されている［148］．「モード」はどのようなキャラクターでも持つ基本的な行動である．モードは複数あり「ウェイトモード」（待機），「Go To モード」（指定された地点へ向かう），「リード」（先導する），「フォロー」（ついて行く）などである．モードの種類と機能を示す（Table 12.4）［148］．モード間の遷移は共通に定義されている．このようなモードは，特に外部からキャラクターにゲームシーンに従った行動を行わせるために活用される．これらは「演技的意思決定」に対応する．また「ノーマルモード」としてデフォルト状態を規定する「通常稼働 AI」があり，これは「自律型キャラクターAI」（Fig. 12.4）に対応する．ノンプレイヤー・キャラクターは自律型エージェントとしての側面と，プレイヤーを楽しませる役者として側面があり，この両者を AI モードによって実現している．

どのようなキャラクターも「モード」群を含めた基本となる「トレイ」が最上層にある（Fig.

11.5)．基本的な実装はモードとしてされているため，どのようなキャラクターもこの層だけで基本的な動作をすることが可能となる．各モードをオーバーライドすることによって，キャラクター毎の特徴を出すことができる．この仕組みによって，モード間の遷移は共通であるが，各モード内はキャラクターごとに異なるように作ることが可能となる．AI モードは，レベルの演出で AI 行動を「ひとまとまりの行動」として指定する共通の枠組みである．このモードを利用することでレベルプランナーは，柔軟かつ簡素に AI 行動を切り替えられることができる．また AI プランナーが各キャラクターの各モードを作り込むことができる．つまりキャラクターの行動を，レベルプランナーと AI プランナーが分担して効率よく作成・編集を行うことが可能となる．

たとえば「FFXV『プラチナデモ』」（スクウェア・エニックス，2016）で登場する四足歩行のキャラクター「カーバンクル」は AI モードのみで制御されている．主人公の少年を止まりながら進んでの誘導，追尾，大きなジャンプを交えたポイント間移動などは，この AI モードの切り替えと各 AI モード内の作り込みによって実現している（Fig. 12.9）．

Fig. 12.9　AI モード「リード」で動作するカーバンクル

このような「リード」モードを実現する一つの方法は，メタ AI が基軸となり，プレイヤーを誘導したいポイントをスパーシャル AI へと指定し，パスをキャラクターAI に対して指定する．また，キャラクターAI に対しては「リード」モードを指定する．リードモードを指定されたキャラクターは指定されたパスに沿って移動しながら，常にプレイヤーとの距離を一定に保つように立ち止まりつつ進むことで，プレイヤーキャラクターを誘導することができる．

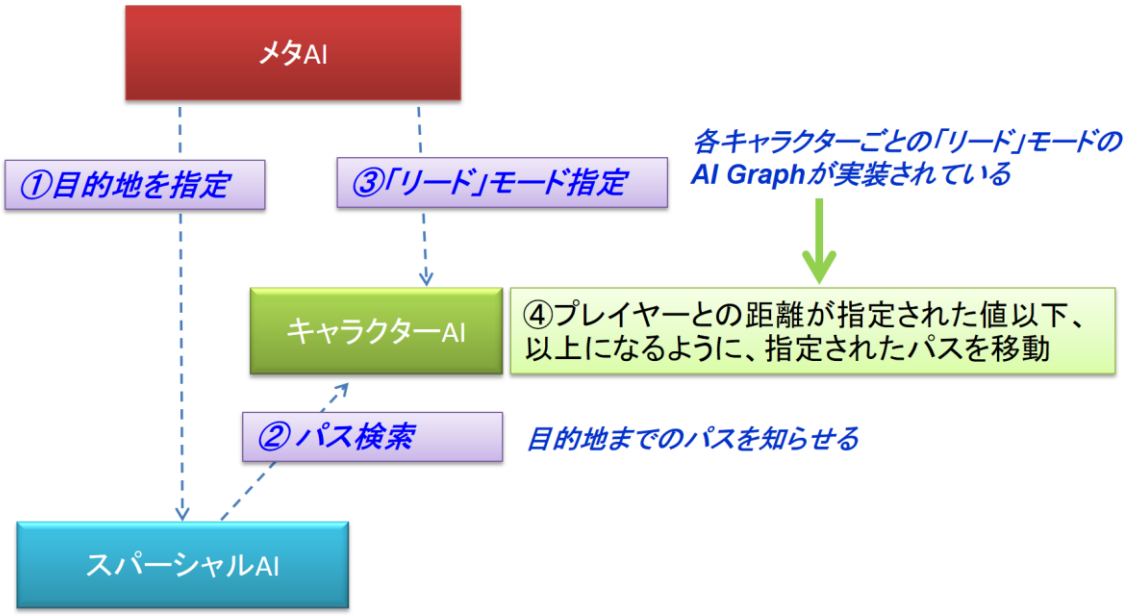


Fig. 12.10 「リード」モードの MCS-AI 動的連携モデルによる実現手法の一例

12.3.2 メタ AI と戦闘

戦闘は仲間キャラクター3 人とプレイヤー・キャラクターで行う．戦闘中はメタ AI がプレイヤー・キャラクターの状態を監視し，その状態によって仲間キャラクターに指示を与える［156].

・プレイヤーや仲間のピンチを助けよ
・プレイヤーが敵に拘束されているから助けよ
・プレイヤーが逃げているから追従せよ
・作戦が発動したのでそれに合わせた行動をせよ

などである．このような指令によって，戦闘を引き締めて，戦闘全体に緩急を与える役割を持っている．

第 1 章で述べたように，レベルとは，ステージ，キャラクター配置，イベント配置，制御など，ゲームバランス設計全体を指す言葉である．レベルデザインとはレベルを設計し調整することを言う．レベルデザインの都合上，キャラクターには特定の行動を取らなければならない場面が多数存在し，レベルから制御される．例えば「イベントが始まるので特定の場所に集合せよ」「その場で待機せよ」などである．この制御はゲーム内で絶対権限を持つ．制御中にキャラクターAI が自律的に動くとゲーム進行が滞るため，キャラクターAI を完全に止めて，強制的に，処理を実行させたい．しかし，それでは逆にキャラクターが状況にそぐわない行動をする場合が起こる．そこで「メタ AI」が，レベルからの要求とキャラクタ

ーAI の自律性の間に立って，二つを調停する役割を持つ．メタ AI は，

・適切な指示を
・適切な相手に
・適切なタイミングで命令し，
・全体をゲームデザインの方針に沿ってコントロールする

という役割を担う．例えばレベルから，「仲間の誰かにここまでプレイヤーを先導して欲しい」という指示があった場合，各キャラクターAI が判断すると，それぞれが同じ指示を受け全体の統制が取れなくなる．そこで「先導して欲しい」というレベルからの指示は，一旦メタ AI が引き受ける．その後メタ AI は各キャラクターの状態をチェックし，遠いキャラクター，他の行動で忙しいキャラクターを除外し，適切な仲間キャラクターを選択して命令を投げる．

Fig. 12.11　メタ AI からの指示でプレイヤー・キャラクターを助ける仲間キャラクター［146］

たとえば，プレイヤーが体力を失って戦闘不能近くなると，それぞれの仲間キャラクターがプレイヤー・キャラクターを回復しに近づき，プレイヤーを回復する（Fig. 12.11）．「3 人とも仲間プレイヤーを助けに来る」と，そのシーンは仲間キャラクターがとても賢くなく見える．理想的なシーンは，「一人は助けに来るが，二人はそのまま戦闘を続けること」である．そこで，回復する場合は，それぞれの仲間キャラクターがいったんメタ AI に許可の申請を出す形にし，メタ AI は各キャラクターの状態をチェックし，遠いキャラクター，他の行動で忙しいキャラクターを除外し，その時に最も戦闘していないキャラクターを選択して申請した行動の許可を出す．残された二人のキャラクターには，戦闘を継続しながら，助けに行く仲間キャラクターに「任せた」というセリフを言うようにメタ AI から指示を出

す．

このように，メタ AI を用いて，映像を用いたカットシーンを挿入するのではなく，戦闘の中でメタ AI を用いて仲間キャラクターの演出を行う．またメタ AI は，ゲームデザインの偶発的な破れを補完するという役割があり，特に開発後半で現れる諸問題をキャラクターAI の修正ではなく，メタ AI の機能の実装という方法で解消できる，という利点を持つ．

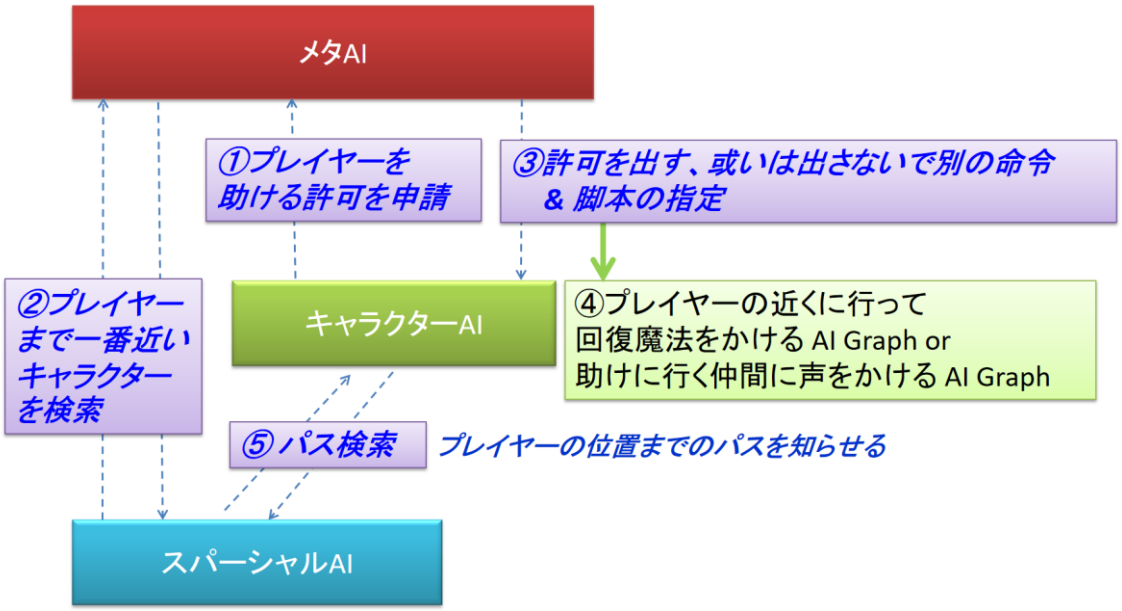


Fig. 12.12　プレイヤーを救助する MCS-AI 動的連携モデルによる実現手法の一例

このような一連のシーンの実現する一つの方法を述べる．まず複数の仲間キャラクターからメタ AI に対してプレイヤーを助けに行く許可の申請がなされる．これは第 7.2 節で述べた「提案型命令」に相当する．次にメタ AI は提案したキャラクターたちの中からプレイヤーに最も近いキャラクターを探索する，或いは，戦闘中でないキャラクターを探索する場合もある．次に選定した仲間キャラクターに対してプレイヤーを助けに行く許可を与える．許可を得た仲間キャラクターはプレイヤーの位置までパス検索によりたどりつき，そこで回復魔法をかける．メタ AI は同時に許可を出さなかった仲間キャラクターには，許可を出した仲間キャラクターに対して特定の台詞を発話するように命令する．このような台詞はたとえば「任せたぞ」などの台詞である（Fig. 12.12）．

12.3.3　メタ AI と会話システム

本項ではメタ AI が形成する会話システムについて述べる．仲間同士をはじめキャラクター同士は会話を行うが，その会話システムもメタ AI が調整する［158］．メタ AI を用いた会話システムの仕組みは以下の五段階である（Fig. 12.13）．

（1）台本再生リクエストを発行
（2）台本を抽選
（3）会話の再生
（4）各 AI へ「意識」変更の指示
（5）各 AI の意識表現行動

以下に各過程を説明する.

（1）あらかじめキャラクターの台詞が収録された複数の台本が用意されている．一つ一つの台本はキャラクター同士の会話が記述され，ボイスが用意されている．レベル，メタ AI，各キャラクターAI から，会話を再生するべきタイミングで「AI 会話量産システム」へ会話リクエストを発行する．会話リクエストには，台本の「グループ」を指定する．例えば，プレイヤーが街に入ったタイミングで，レベル側から，「街に入った台本グループ」を再生して欲しい，というリクエストが投げられる．

（2）リクエストを受けたら，一つの台本が選択される．各台本には，再生条件が設定されており，指定されたグループの中から，その再生条件を満たす台本を 1 つ抽選する．この条件チェックを，メタ AI が行う．例えば「街に入った台本グループ」がリクエストされた場合，今は「夜」で，「イグニスという仲間キャラクターが近くにいる」という条件から，その二つの再生条件を満たす「イグニスが宿泊を勧める台本」が抽選される．バトル中では，各キャラクターAI から，「プレイヤーの近くに来たから何か会話がしたい」，というリクエストが要求される．

このように，キャラクターAI やレベルは，台本再生のリクエストを投げるだけで，あとはメタ AI が自動的に選択する仕組みとなっている.再生条件は他にも「再生インターバル」（1 回再生したら再生しない時間を取る）「特定のキャラクターがいる」「時間帯」「場所」「天気，温度」「ストーリー進行度」「戦況」「どんな敵がいるか」「敵倒したばかりか」「状態異常」などである．

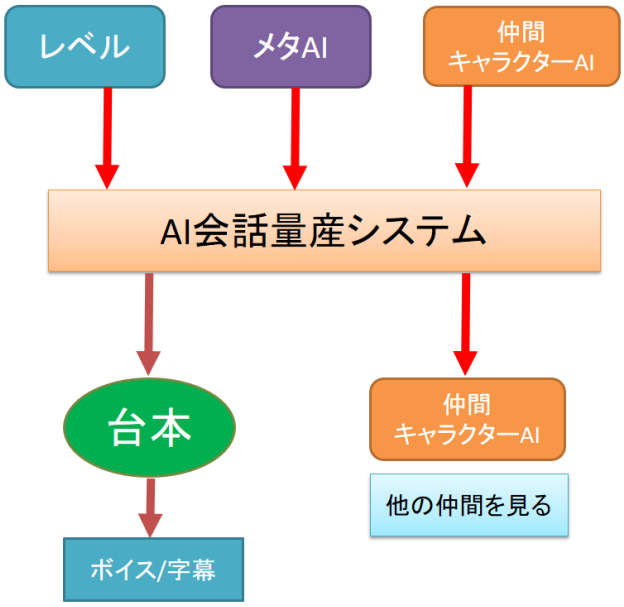


Fig. 12.13　FFXV における会話システム［99］

（3）台本が抽選されたら，指定の会話が再生され，ボイスと字幕が流れる．

（4）同時に，各キャラクターAI に対して，「意識」の変更の指示が伝えられる．ここで言及する「意識」とは，例えば，「この人に注目しなさい」「今あなたは話者だ」などといった指示になる．たとえば，イグニス（仲間キャラクターの一人）が話しているような場合には，各仲間キャラクターに対して「イグニスに注目せよ」という命令が投げられる．

（5）「意識」の指示を受け取った各キャラクターAI が，「意識表現」を行う．「意識表現」とは，体や首の向きを変え，話者としてのモーションに変更する，などといった表現である（Fig. 12.14）．例えば，「イグニスに注目しろ」と言われた AI は，イグニスに首を向けて，注目していることを表現する．次に，話者のモーション変更の例である．このモーション変更は，移動中でも行われる．話し手が変わる度に，そのキャラクターが後ろや横を向いて話すようなモーションが再生される．このように，意識が変更されたことを表現する対応は意識表現運動と呼ばれ，AI 側で臨機応変に行われる．

会話システムはこのようにあらかじめ準備された台本，音声，振る舞いを，メタ AI によって動的に組み合わせることで実現される．

Fig. 12.14　FFXV における意識表現運動［99］

12.4　スパーシャル AI

本節では, FFXV におけるスパーシャル AI について述べる. 第 12.4.1 項では「パス検索」, 第 12.4.2 項では「位置検索システム」について述べる．第 12.4.3 項では，スパーシャル AI からキャラクターを制御するスマートロケーションの技術について述べる．スパーシャル AI は，さらに小さい地形認識のための人工知能モジュールの集合である．地形を把握する方法，特徴抽出の方法はモジュール毎に異なる．FFXV ではパス検索のための世界表現は第 3.4.1 項で述べたナビゲーション・メッシュ（Navigation Mesh）である．ナビゲーション・メッシュはキャラクターの移動可能な領域を，互いに接続された多角形のポリゴンの集合によって表現したデータである（Fig. 12.15）．このポリゴンをノードとする連結ネットワークグラフ上で出発ノードから到着ノードまで A*アリゴリズムのパス検索アルゴリズムで最小コスト検索を行う．

ナビゲーション・メッシュは地形とキャラクターモデルの衝突を司る「地形の衝突モデル」に沿って生成される［159]. FFXV では, 自動生成アルゴリズムが夜間に作成するように「ナイトリー・ビルド」の形でプログラムが動いており，朝方には各マップの最新のナビゲーション・メッシュが作成される［160].

ナビゲーション・メッシュは，ノンプレイヤー・キャラクターの移動の基本となる．ところが，開発中は地形モデル，オブジェクト・モデル，キャラクターモデルなどゲームの要素がほぼ毎日追加・更新される．新しく置かれたオブジェクトにナビゲーション・メッシュが対応するためには，なるべく高い頻度のナビゲーション・メッシュの更新が必要である．開

発者が更新したアセットデータは，任意のタイミングで，データサーバーにアップロードされる．それらが一端終息するのは，開発者の退社後の夜であり，そこで夜のうちにナビゲーション・メッシュと追加・修正されたコンテンツの整合性を取るために，ナイトリー・ビルドが行われる．翌朝には，ナビゲーション・メッシュとゲームステージの整合性が取れるようになる [161].

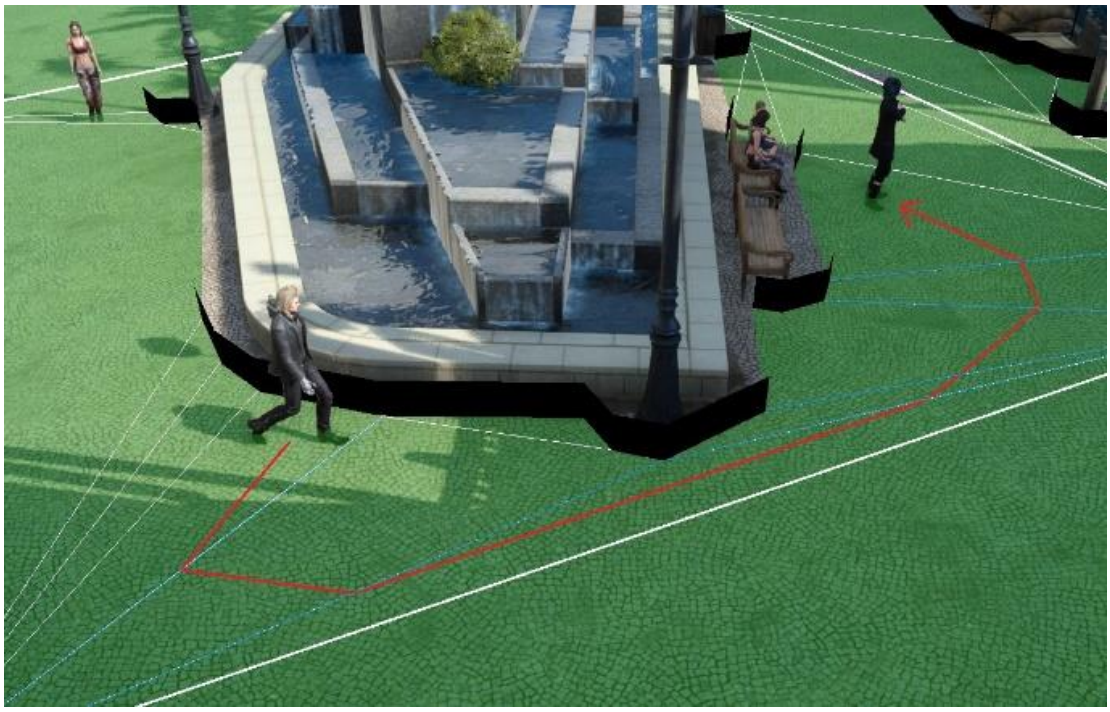


Fig. 12.15　FFXV のナビゲーション・メッシュとパス検索［160］

12.4.1　パス検索

FFXV では，パス検索モジュールはこのナビゲーション・メッシュ上の A*パス検索アルゴリズムでコストを最小にする経路を計算する．ここで導かれる経路はジグザグであるため，実際はファンネル・アルゴリズムでスムージングを行い，パスを整形する．しかし，近距離（100m 以内）の場合にはパス検索の前に目的地に向かって 2D レイキャスト（ナビゲーション・メッシュ上のレイキャスト）を行い，パス検索の代替として用いる．

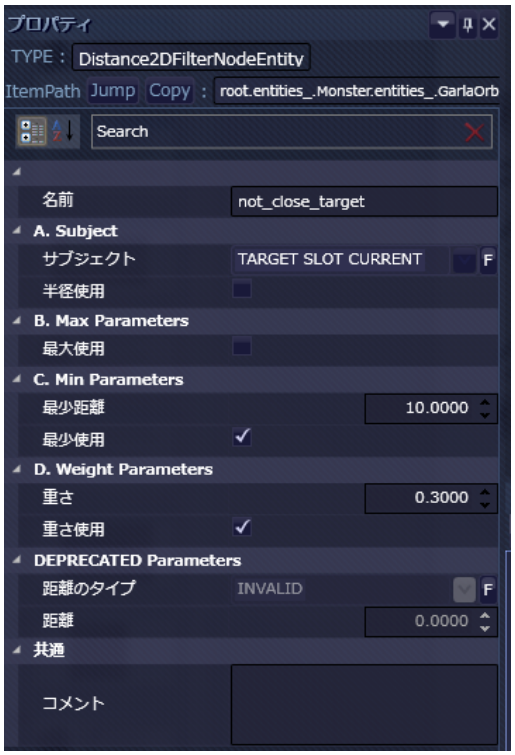


Fig. 12.16　FFXV の PQS ツール［162］

12.4.2　位置検索システム

FFXV における位置検索システムは，専用のエディタ（Fig. 12.6）によって，ポイント生成，それぞれのフィルタリング，評価関数を指定し，クエリーを構築する［162］．ポイント生成をするジェネレーター，フィルタリング，バリディエーターはモード化され，専用のエディタからモード選択とそれに伴うパラメータをセットすることで，複数ジェネレーター，複数のフィルタ，バリディエーターを組み合わせて PQS クエリーを製作する．例えば「角度フィルタ ＝ プレイヤーの前方 30 度を除外」「距離フィルタ ＝ プレイヤーから 10m 以上遠くで，出来るだけ近く」を取り，アップデートし続けると，プレイヤーから見えない場所へとプレイヤーの周囲をぐるぐる回る移動データが作成される．PQS は街中でのキャラクターの立ち位置や（Fig. 12.17），フィールド上のモンスターの運動を作るためにも用いられる（Fig. 12.18）．位置検索技術は，開発中盤から後半に従って，各キャラクターAI のための必須技術となった．

Fig. 12.17　FFXV における PQS によるポイント評価マップ

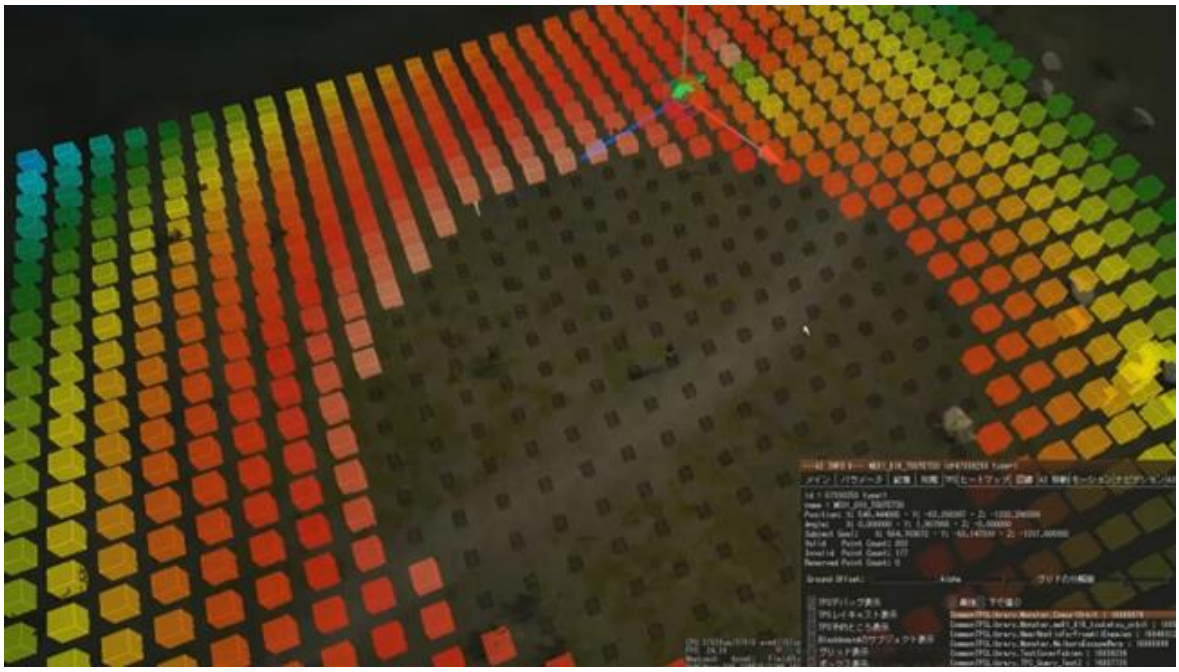


Fig. 12.18　FFXV における PQS によるポイント評価マップによる移動 [146]

Table 12.5　製作されたクエリー，バリディエーター，フォールバッククエリーの使用箇所数 [162]

種類	準備された個数
Query	135
Validator	16
Fallback Query	7

Table 12.6　ジェネレーターの種類と使用箇所数［162］

ジェネレータの種類	使用箇所
Static	3
Navigation Mesh	125
Spawning	6
Cover	7
Battle-Area	1

Table 12.7　フィルタの種類と使用箇所数［162］

フィルタの種類	使用箇所	説明
Distance3D	77	３次元距離
Distance2D	144	２次元距離（elevation なし）
Random	65	ランダム・スコアリング　(0 to 1)
Angle	126	ポイントと対象物（subject）間の角度
NavigationLineOfSight	22	ナビゲーション･メッシュ上のレイキャスティング (2D)
SteeringIsFree	20	他のエージェントと被らない
Heatmap	11	指定されたヒートマップのスコア
InAmbientRegion	11	アンビエント・フラグのある領域
InCameraView	11	カメラのビュー・フラストラム
FacingSubject	9	対象物の前または後ろ
Directness	2	方向によるフィルタリング
ReservedPoint	2	ポイントが予約されているか否か (攻撃の狙いを定める際に便利)
NearAmbientRegion	1	アンビエント・フラグのある領域が近い
DistanceMedian	0	複数の対象物間の距離の中央値
Frequency	0	対象物の周りを回転する正弦波
Visibility	0	シーンの 3D レイキャスティング

またフォールバック・クエリー(a fallback query)は，PQS のクエリーが失敗する（結果を返さない）場合に，二度目を試みるクエリーである．クエリーの条件が強く候補となるポイントがないという状況において，より緩和された通常より広い制限のフィルタリング・パラメータを持つクエリーを実行する場合に用いられる．

Table 12.5 は実際に制作されたジェネ―ター，バリディエーター，フォールバッククエリーの個数である．ゲームを通じて 135 個のクエリーが製作された．Table 12.6 はジェネレーターの種類と使用箇所数である．それぞれのジェネレーターは異なるポイント生成機能であり，ナビゲーション・メッシュ上にポイントを生成する，地形から隠れられる場所を解析して生成する，NPC をスポーニングできるポイントを地形から解析して生成する，などがあります．また Table 12.7 はフィルタの種類と使用箇所数の表である．ゲーム開発に合わせて多様なフィルタが形成されたことがわかる．

12.4.3　スマートオブジェクト，スマートロケーション

FFXV では環境側からキャラクターを操作する必要がある．特に，街などの群衆としての AI（Ambient AI）は，個々に高度なキャラクターAI を持たせるのではなく，スパーシャル AI から複数のキャラクターを制御する．オブジェクトからキャラクターを操作する手法はスマートオブジェクトと呼ばれるが，オブジェクトではなく特定の座標を中心とした領域にキャラクターを制御するシステムは「スマートロケーション」と呼ばれる［163］．スマートロケーションには，その座標近くにあるオブジェクトが所属し，動的に NPC を集めて，そのオブジェクトとインタラクションさせる．たとえば，椅子と机があり，その近くにいる NPC2 体を選択し，机まで来て，椅子に座り，お互い話し合うように命令する（Fig. 12.19）．また時間が経つと 3 体目の NPC を呼び寄せ，他の 2 体の NPC と話し合うように命令する．これによって，プレイヤーから見ると 3 体の NPC が楽しく話し合っているような演技をさせることができる．また，キャラクターがガードレールを飛び越える場合には，どのような角度で走り，どの座標に足をかけて，どのようなタイミングでアニメーションが再生するか，が環境解析によって決定される［164］．

スマートロケーションの実装

オブジェクトは状態を持ち．椅子であればあるキャラクターが座っている，テーブルであればキャラクターが占有している，などである．「タプルスペース」（Tuple Space）は，ゲーム内のそれぞれのオブジェクトと，それに対する複数の操作の集合を指す．タプルスペース上に複数のルールがあり，それぞれの NPC がルールを一つ一つチェックし，それぞれのルールはアクションを引き起こし，タプルスペースに変化を及ぼす．

Fig. 12.19　スマートロケーションによって制御される Ambient AI の例［163］

```
Rule 1:
Action:          sit(X)
Pre:             closestOf(chair,X,me) & !reserved(X,Y)
Add:             reserved(X,me)
Add-Deferred:      sitting(me)&
                   timeToGetUp(now+rand(10,15)*minute,me)
Next State:            sitting
```

Fig. 12.20　スマートロケーションの持つルールの形［163］

例えば椅子 X に対するルールは Fig. 12.20 のように書かれる．椅子に座る sit(X)，その前提条件（Pre）として自分に最も近い椅子を探し予約できる．その前提条件を実行した時に変化する状態（Add）はその椅子 X を自分が占有すること．アクションが終わった後に追加される状態（Add-Deferred）として，自分が座っており，立ち上がるまでの時間を指定する．このようなルールが複数存在し，キャラクターは一つのルールをチェックして行き実行可能条件を満たすルールを選択し実行する．このように一つの動作がタプルスペースを変化させ，そのタプルスペースの変化は用意されたルール群の前提条件の真理値を変化させ，実行可能なルールが変化する．つまり，これは STRIPS のように前提条件と効果を持つルールたちが連鎖しながらキャラクターの制御を行って行く．ルール群を管理するデバッグ画面（Fig. 12.21）ではルール群が常に監視され，前提条件が true になったルールが緑色で表示される．

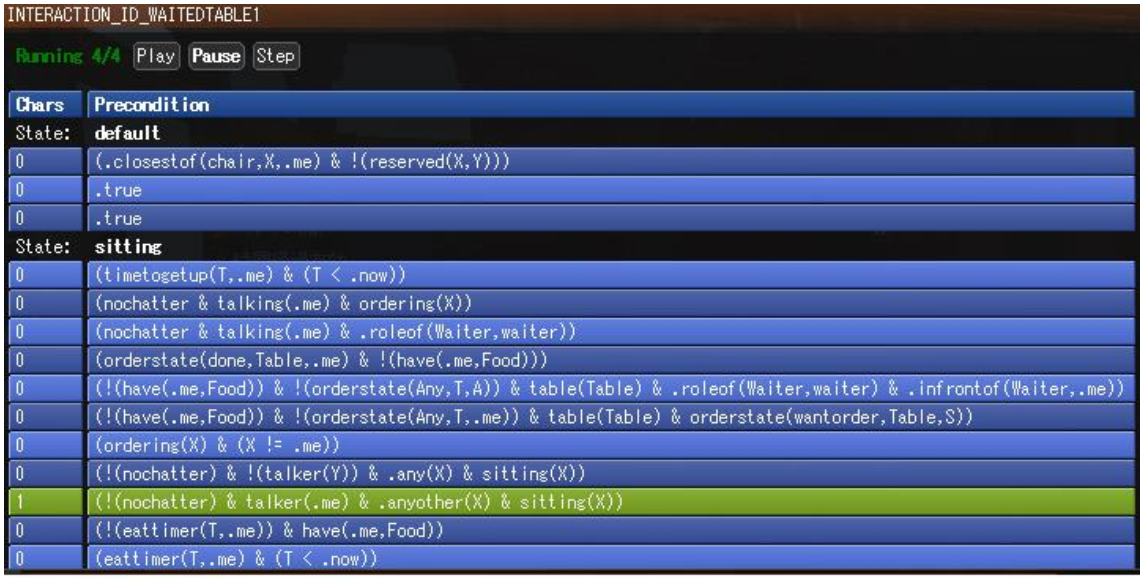


Fig. 12.21　ルール群を管理するデバッグ画面［163］

12.5　プレイヤーAI

プレイヤー・キャラクターは，当然プレイヤーの操作が基本となるが，AI 制御の対象でもある．特に 3D アクションゲームの場合，複雑な環境で自由度の高いプレイヤー・キャラクターを自在に操るためにはプレイヤーとしてのスキルが必要であり，簡単な操作でその場の状況に合った，その地形に沿った行動を自動的に行うように，人工知能がプレイヤー・キャラクターを動かす．

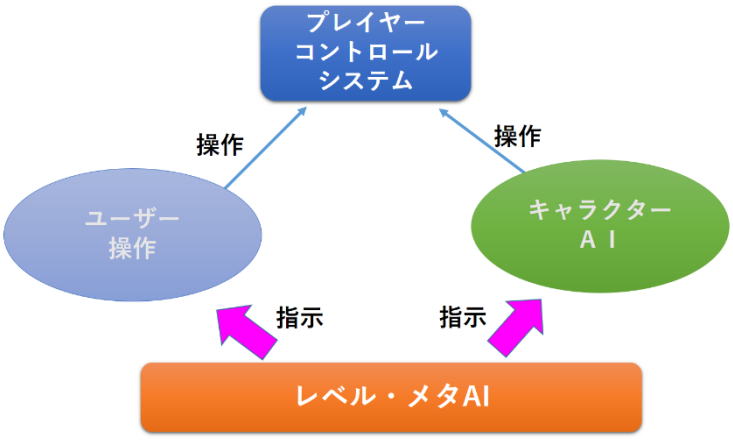


Fig. 12.22　FFXV におけるプレイヤーコントロールシステム［99］

FFXV の開発過程でプレイヤー・キャラクターに求められた 2 つの項目は「プレイヤー・キャラクターをゲームの『世界』と調和させる」「レベルやメタ AI からキャラクターを制御

する」である．本作は，主人公が仲間たちと共にリアルで広大な世界を車で旅するというゲームであり，プレイヤー・キャラクターを通し，実際に旅行しているような体験ができることを目指している．そのため，ユーザーとゲームの唯一の接点であるプレイヤー・キャラクターをゲーム世界に溶け込ませ，没入感を高めるということが求められた．2 つ目は他のキャラクターと同様に，ゲームの都合上，決まった動作をさせ，他のキャラクターと連動させるなど，柔軟かつシステム的な制御が必要である．これら 2 つの要件を満たすため，「プレイヤーコントロールシステム」が構築された（Fig. 12.22）．

プレイヤーコントロールシステムに対して，ユーザーと AI の双方が操作入力を与え，プレイヤー・キャラクターを制御する．ユーザー操作もプレイヤーAI も常に有効な状態にあり，並列に動作している．どちらか一方がプレイヤーコントロールシステムを独占するわけではない．ユーザー操作側と AI 側が直接やりとりをして，お互いに譲り合い，レベルやメタ AI からの指示で，バランスを調整しながら，プレイヤー・キャラクターを制御する．

Fig. 12.23　意識表現　AI 制御なし（上図）と AI 制御あり（下図）［99］

12.5.1 意識表現

一つ目の「世界と調和する」項目は「意識表現」として実装される．例えば，仲間プレイヤーが降ってきた雨に反応する．これは当然の反応であるが，プレイヤー・キャラクターだけ降って来た雨に反応しないと不自然でありユーザーの没入感を削いでしまう．そこでプ

レイヤー・キャラクターにもＡＩから自動的に雨に掌をかざす動作をさせる（Fig. 12.23）．また寒いダンジョンの中でパーティ全体が凍える動作を行う，なども同様である［99］．

また二つ目の「メタ AI やレベルから，プレイヤー・キャラクターを操作する」という項目は，AI モードを駆使することで行われる．たとえば，トリガーボックス（ステージ上に設置された一定の大きさを持つ空間）にプレイヤー・キャラクターが入ると，AI モードの一つである「Route」モードが発動し，ある目的地に向かって自動的に走り出す，というような制御が開始される．

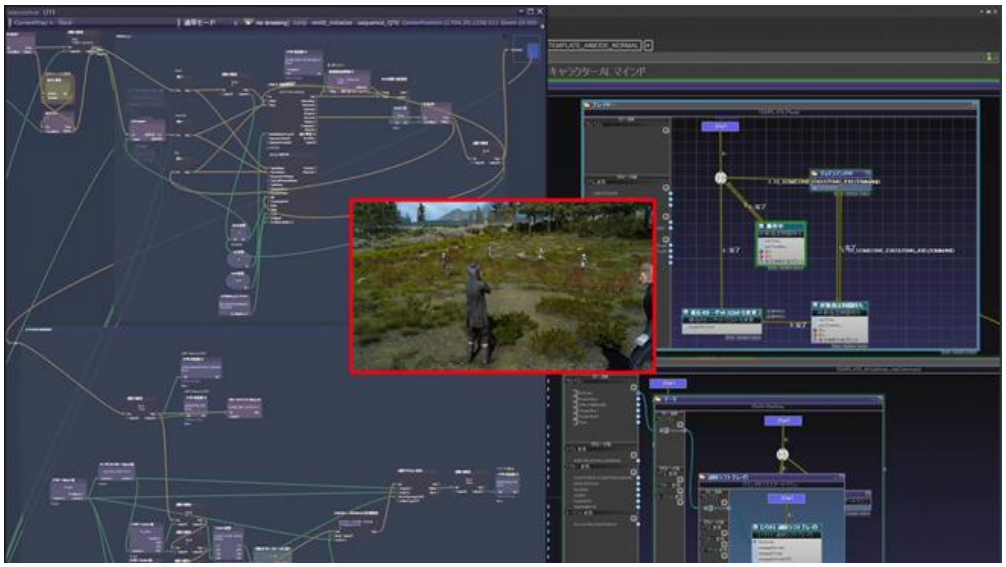


Fig. 12.24　仲間コマンドの AI Graph [99]

12.5.2 仲間コマンド

仲間コマンドは，バトル中に仲間にアビリティを使うように指示するゲームシステムである．仲間コマンドは，プレイヤー・キャラクターのユーザー操作と AI 制御の併用，そして様々な AI システムを活用して構築される［99］．たとえば，

(1)　仲間コマンド「マーク」を指定すると，仲間キャラクターの一人が複数の敵にナイフを投げてマーキングを行う．この部分は AI Graph が制御する．

(2)　プレイヤー・キャラクターが「シフトブレイク」という技を出す．マークの付いた敵にワープ攻撃を連続攻撃で行う．AI Graph によって制御される．

(3)　QTE（Quick Time Event，短い時間でボタン入力を達成条件とするイベント）でプレイヤー・キャラクターが追撃する．ここはボタン入力を感知して追撃モーションを再生する．

など，仲間コマンドはプレイヤーと仲間キャラクターの協調動作を促すものである（Fig.

12.24).

このようにプレイヤーAI は，メタ AI と AI モード，AI Graph を用いて実現される．

12.6　Face-to-Face 対話システム

F2F (Face-to-Face) 対話システムは，プレイヤー・キャラクターと NPC が向き合って対面式に会話を行うためのシステムである [99]．ゲームの進行上，プレイヤー・キャラクターと NPC，特に仲間キャラクターが対面式に対話する状況が必要となり，レベルシステムからの要請をメタ AI が受けて実行する．

Fig. 12.25　F2F 対話システム [99]

この対話イベントは地上・車内・宿などさまざまな場所で発生するため，位置合わせと動作タイミングの協調が必要である．そこで，メタ AI，キャラクターAI，スパーシャル AI の 3 者の協調のもとに行われる（Fig. 12.25）．

まずメタ AI は会話が発生する状況（地上，車内，宿）を認識する．メタ AI はそこで発生するべき台本を選択し，プレイヤー・キャラクターと NPC の間のモーション（NPC がプレイヤー・キャラクターの肩をたたく，一人で腕を組むなど）を決定する．そして NPC に対してプレイヤーの側へ駆け寄るように命令を発行する．そこでスパーシャル AI が，モーション開始位置を算出し，その地点への経路をパス検索によって経路を提供する．たとえば，車が移動しとまった後，NPC は車をパス検索で避けつつキャラクターに近寄って会話を始める（Fig. 12.26，Fig. 12.27）．

このように F2F 対話システムは，メタ AI，キャラクターAI，スパーシャル AI の連携のもとに実現される．

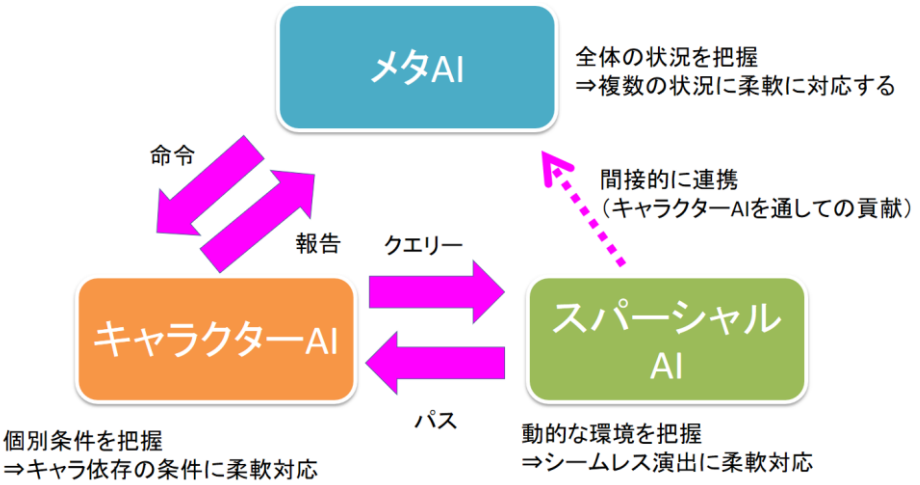


Fig. 12.26　F2F 対話システムにおける 3 つの AI の協調

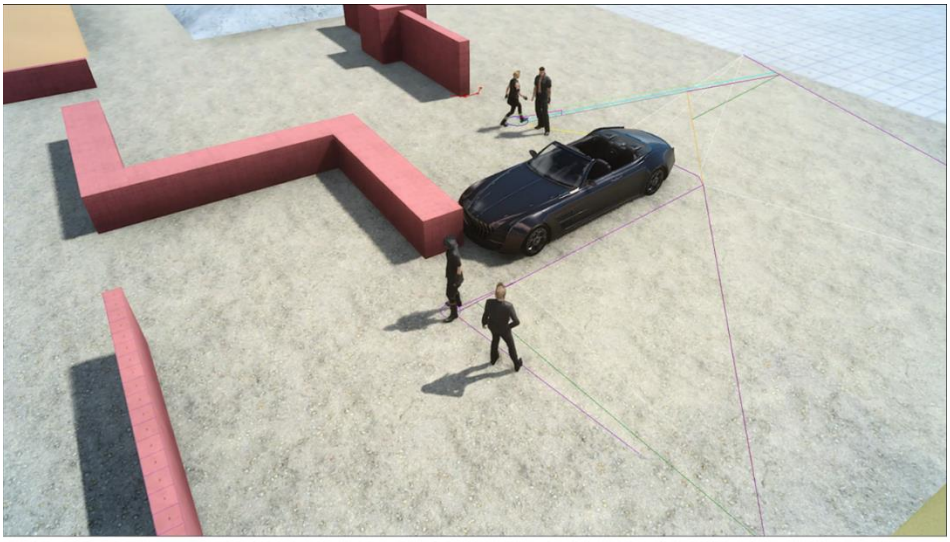


Fig. 12.27　F2F システムによって車の向こう側から NPC がプレイヤーに駆け寄る［99］

12.7　ロギングとビジュアリゼーション

本節では，FFXV におけるロギングとビジュアリゼーションについて述べる［165］．ゲームが一通り完成すると，それを実際にプレイしながらバランスを取る，バグを見つけるフェーズとなる．開発者のゲームプレイのほかに，専門のテスターが数十人から数百人の規模でプレイする．そこで，バグが見つかるたびにバグレポートが，かつては紙で，現在ではオンラインのシステムに登録されて行く．しかし，現在では，そこでデータマイニング的手法が取られている．バグレポートにタグ付けがされ，ステージやバグの種類ごとに各種ソートす

ることが可能となる．また，これとは別に，ゲームプレイログをサーバーに貯めて，そのデータからデータマイニングによってゲームの内部状態を分析するという手法が取られる[166]．

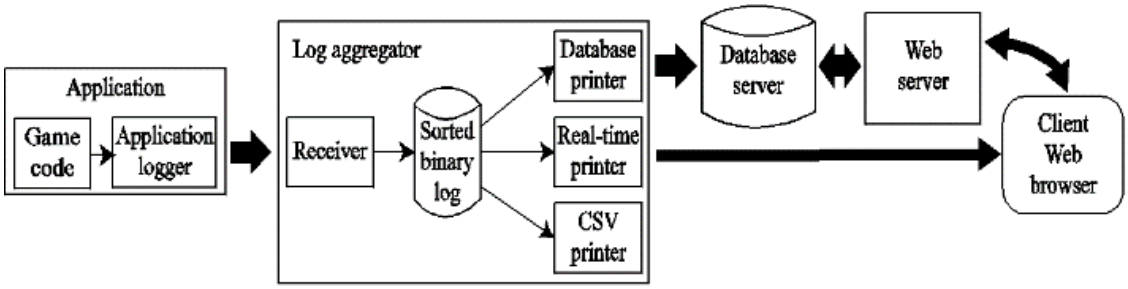


Fig. 12.28　ゲームプレイデータ自動収集・解析・表示のパイプライン［166］

FFXV においては，開発においてプレイログをサーバーに収拾し，自動的にグラフ表示までが為されるようなシステムが組まれている．これによってゲーム内のわかりにくいバグを統計的に発見することが可能となる（Fig. 12.28）．

そこで，サーバーではこのクエリーが何度発行されたかという統計情報を収集しグラフにすることで，どの位置検索クエリーが最も良く呼び出されているか，異常に呼び出されていないか，またまったく呼び出されていない位置検索クエリーがあるか，などの問題点を一目で見出すことができる（Fig. 12.29）．また，ゲーム内の台詞は，あらかじめ台本という形で準備されるが，メタ AI が状況に応じてトリガー条件を考慮して選択する．この台本に対してもゲームを通じてどれぐらいの頻度でどの脚本がどれぐらいの頻度で呼び出されているかをグラフにすることで，まったく呼び出されていない台本があった場合，そのトリガー条件の見直しを行い，迅速に問題点を見つけることができる（Fig. 12.30）．これによってプレイによってだけでは見つけにくいゲーム調整上のバグを見つけることができる［167］．

かつてデジタルゲームはプログラマが隅から隅までその変数や構造を把握できた．しかし，大規模化した現在では，全体像を把握することが難しくなっている．コンシューマーゲームはその内側に巨大な世界を内包し，携帯ゲームはネットワークの上で多数のユーザーを抱えてその関係を築いている．開発者が自分の作っているゲームの全貌を知るには，開発者の認知が難しい部分のダイナミクスの解析を，人工知能技術がデータを収集し，可視化することが必要である．単に一つの変数を追えば良いというものではなく，複雑に絡み合った数個から数十に登る変数の変化を追わなくてはならない．そのような時には可視化のテクニックが必要とされる．

現在はケース毎に作っている可視化ツールも，総合的なソリューションとしてゲームエンジンなどに組み込まれて行くことが予想される．

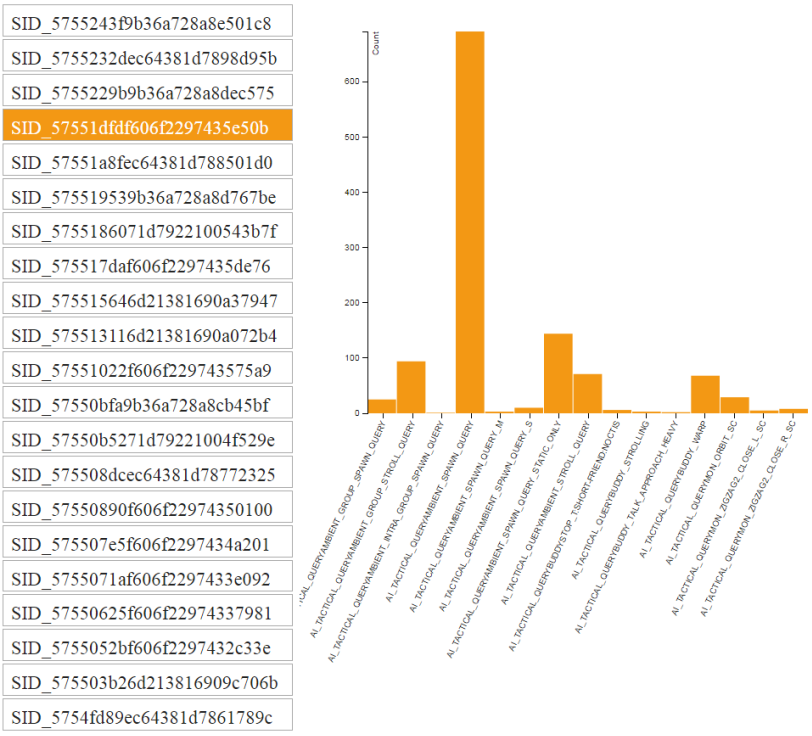


Fig. 12.29　位置検索クエリーの統計グラフ（横軸：クエリーID，縦軸：呼び出しカウント）[165]

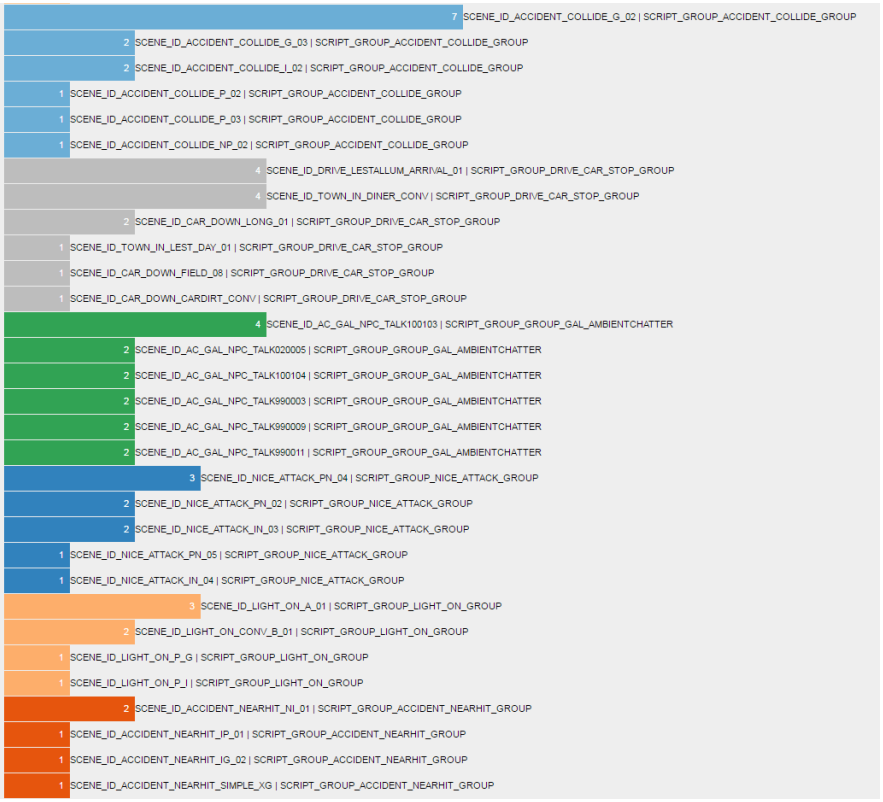


Fig. 12.30　台本の呼び出し回数　（縦軸：台本の ID，横軸; 呼び出しカウント）[165]

12.8　MCS-AI 動的連携モデルの効果

本節では，本章の全体の考察を述べる． 第 12.8.1 節では MCS-AI 動的連携モデルの統計的な結果について述べ考察を行う．第 12.8.2 項では MCS-AI 動的連携モデルによって，どれだけコストが削減されたかを検証する．第 12.8.3 項では FFXV における「MCS-AI 動的連携モデルの効果」について，第 12.8.4 項では「AI Graph の効果」について，第 12.8.5 節では総合的な考察を述べる．

12.8.1　MCS-AI 動的連携モデルの統計的結果と考察

AI モードはメタ AI とキャラクターAI への命令であり，その関係を表すものである．また PQS はメタ AI，キャラクターAI からスパーシャルへの要求であり，その関係を表すものである（Fig. 12.31）．本項では AI モードの各モードの使用数（Table 12.8）と PQS クエリーの種類（Table 12.5）から，MCS-AI 動的連携モデルの効果について考察を行う．

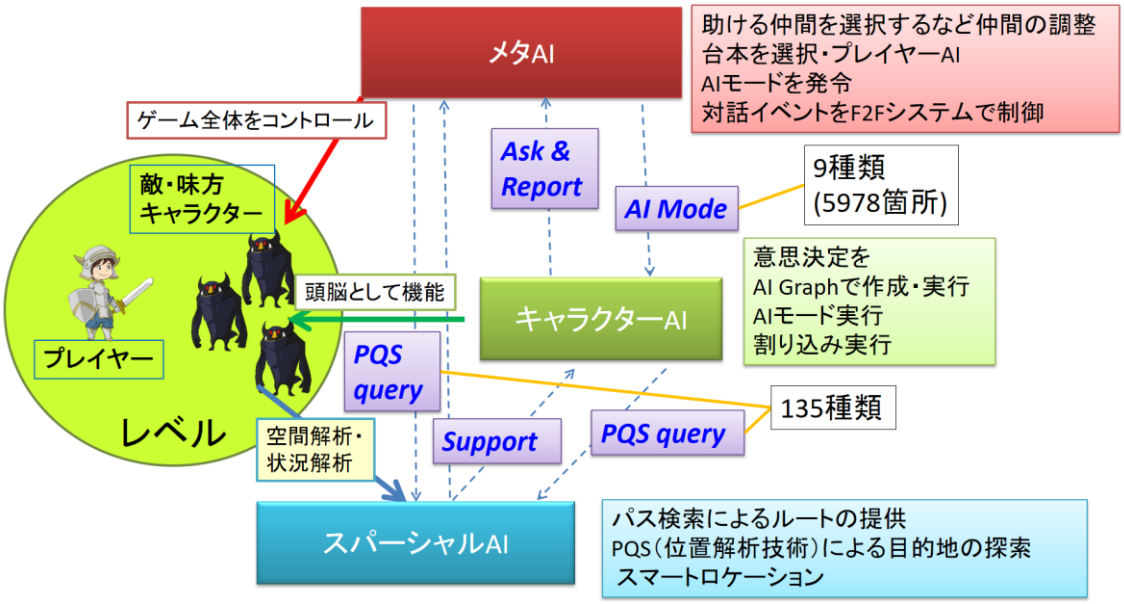


Fig. 12.31　FFXV における MCS-AI 動的連携モデル

AI モードの各モードの使用数について考察を行う．「リード」「フォロー」モードに関しては，FFXV は仲間キャラクターと旅をするゲームであり，プレイヤーと一緒に移動する際に使用され，メタ AI がその協調移動，つまりプレイヤーの前を走って誘導するか，プレイヤーに追従するか，を調整している．これらの使用箇所が 14,22 箇所と他のモードに対して少数であるのは，テンプレートの機能によって使用される箇所が限定されているからと考えられる．また敵兵士が部隊長を追従する，などがある．「ルート」モードに関しては，メ

タ AI が NPC に特定のルートをたどって登場させるなど演出としてキャラクターを用いる場合に用いられる．これらの使用箇所が 184 箇所であるのは使用される状況が限定されているからと考えられる．「プレイモーション」モードはメタ AI が NPC などに待機中のモーションやプレイヤーに向かって演技をさせる場合に用いる．様々なモーションがあるため使用箇所が 772 箇所となっていると考えられる．「ゴートゥ」モードはメタ AI が NPC の戦闘時の移動，街の中，移動など，汎用的に移動を行わせるため場合に用いられる．演技的状況や戦闘状態に合わせたキャラクターに応じた命令のため，1582 箇所と比較的多数となっていると考えられる．「ウェイト/ターン」モードに関しても演技的状況や待機中の演技のために比較的多数となっている．このようにメタ AI は AI モードを 9 種類 5978 箇所で使用し，キャラクターに演技をさせる，変化する状況への対応の行動スタイルを指定している．

Table 12.8　AI モードの種類・機能・実装における使用回数［148］

種類	機能	使用数
リード	プレイヤーをリードする	14
フォロー	プレイヤーをフォローする	22
ルート	指定したルートをたどる	184
ワープトゥ	指定したポイントへ一瞬で移動する	230
プレイモーション	特定のモーションを再生する	772
リセット	モードをリセットしてノーマルモードへ遷移	1187
ゴートウ	指定の場所に移動する	1582
ウェイト/ターン	雑談など待機状態・特定の方向へ向く	1987
モード累計		5978
総ノード数		508502

また目的地の検索には PQS が用いられるが，135 種類のクエリーが作られている（Table 12.5）．これらはキャラクターAI，メタ AI によってスパーシャル AI によって呼び出される種類である．キャラクターAI が用いるのは戦闘時における目的地の検索，街の NPC が住民らしく振舞うための位置を検索する場合に用いられる．メタ AI が PQS を用いる場合は，仲間キャラクターがプレイヤーを助けるために，攻撃しようとするモンスターとプレイヤーの中間地点を検索して，仲間キャラクターを向かわせる場所や，NPC がプレイヤーと話す位置，一緒に集団で走る場合の位置，目標地の検索を行う．ナビゲーションメッシュ上のジェネレーターが 125 個とほとんどであるのは，戦闘や街の中で使用されているからである．これらは場所を問わずナビゲーションメッシュの存在する至るところで使用可能なクエリーであり，これらのクエリーによって複雑で広大なマップの地形の複雑さが一般化された上で解析されている．FFXV の MCS-AI 動的連携モデルのメタ AI，キャラクターAI は，

この PQS の仕組みによって，マップのスケールの広大さと複雑さに対応していることがわかる．またフィルターで二次元上の距離と角度のフィルターが多用されているのは，立体的ではなく平面的なクエリーが多いことを表している．これは FFXV のレベルデザインの平面性と符合している．このようにメタ AI，キャラクターAI とスパーシャル AI の関係は 135 種類のクエリーを生み出し，多様な地形・状況解析が実行された．

このように，MCS-AI 動的連携モデルは，場面による演技の指定，状況における行動の変化を AI モードで指定し，キャラクターAI の自律性を保ちながら実行させることを実現した．またスパーシャル AI は，複雑な地形・状況による地形の使い方の違いを，PQS クエリーの解析によって吸収し，メタ AI，キャラクターAI の命令が地形・状況に沿う形に適応させた．またスパーシャル AI はスマートロケーションの仕組みによって，自律的にキャラクターAI を制御した．

AI モード，PQS クエリーの使用は，メタ AI，キャラクターAI，スパーシャル AI を相互に結び付ける形式であり，MCS-AI 動的連携モデルの上に FFXV の AI システムが精緻に作り込まれていることが AI モード，PQS クエリーの統計情報より考察される．

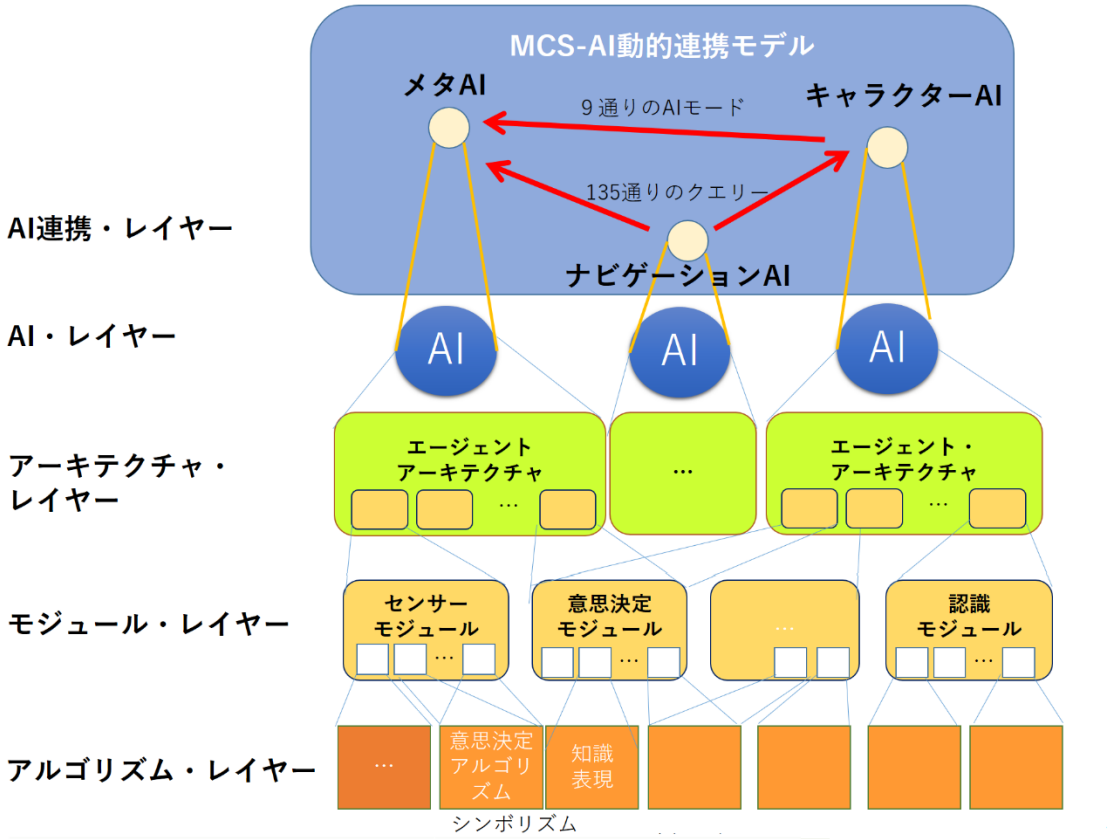


Fig. 12.32　FFXV における AI 階層図

12.8.2　MCS-AI 動的連携モデルによるコストの削減

本項ではMCS-AI動的連携モデルによってどれだけコストが削減されたかの試算を行う．MCS-AI 動的連携モデルは，AI システムの階層の最上層でそれぞれの AI がお互いを利用することで，AI システムとして，多様なバリエーションの運動を創出するシステムと捉えることができる(Fig. 12.32)．またお互いの自律性を最大限保ちながら，連携することを可能とする．

キャラクターAI はメタ AI へ 9 つのモードを用意することで，メタ AI からキャラクター AI へ 9 つの動かし方を提供する．スパーシャル AI は，135 個のクエリーのバリエーションを用意する．作られたシステムの潜在的なバリエーションは 9x135=1215 個である．もし MCS-AI 動的連携モデルでなければ，このバリエーションを作り出すためにキャラクター AI は，この 1215 個の条件分岐やスクリプトを用意する必要がある．

この 1 スクリプトのコストを C_script とすると，

$$1215 \text{ x } C_{\_script}$$

のコストとなる．またスパーシャル AI において，1 クエリーを作るコストを C_query，AI モードを用意する実装コストを C_mode とすると，全体のコストは

$$9 \text{ x } C_{\_mode} + 135 \text{ x } C_{\_query}$$

となる．C_script を 30 として，C_mode を 80，C_query を 10 とするとスクリプトで準備するコストは 36450 に対して，AI モードと PQS クエリーで準備するコストは 2070 である．実際，ここに AI Graph を作って AI モードを実装する，PQS エディタを作ってクエリーを実装する初期コストがある．スクリプトの場合，このコストがほぼない．FFXV で実現された AI のバリエーションの数を作り出すために，MCS-AI 動的連携モデルは一端ツールができた後は，このモデルを利用しない場合に比べて，0.05 倍のコストの開発となる．

また，キャラクターAI のバリエーションの作成だけに注目すると，MCS-AI 動的連携モデルでは，最初に 9 つの AI モードを用意する必要がある．さらに 5978 箇所から AI モードを呼び出すための実装コストを C_imp すると，今回の FFXV の AI モードの実装とその利用には，

$$9 \text{ x } C_{\_mode} + 5978 \text{ x } C_{\_imp}$$

だけのコストがかかることになる．キャラクターAI が．自ら 5978 個のケースにおいて AI の思考を形成する必要がある．このコストは

$$5978 \text{ x } C_{\_script}$$

である．C_imp は呼び出しだけであるから，C_imp を 5 とすると，スクリプトで実装した場合はコストが 179340 であるが，AI モードを利用した場合は 30610 となる．ツールができた後は MCS-AI 動的連携モデルは，このモデルを利用しない場合に比べて，キャラクターAI をおよそ 0.17 倍のコストでの開発が可能である．

12.8.3　FFXV における MCS-AI 動的連携モデルの効果

本項では，FFXV における MCS-AI 動的連携モデルの効果について述べる．MCS-AI 動的連携モデルは，FFXV の AI 開発において，常に技術を蓄積・発展させる場であり，各開発者の作業を連携させる場として機能する．FFXV の開発において，MCS-AI 動的連携モデルは，次の 5 つの点で効果を得た．

(1) **各 AI の連携によるユーザー体験の実現：**MCS-AI 動的連携モデルによって，各 AI を連携して活用することができる．
(2) **開発工程の構造化：**ゲーム開発初期から，AI 開発をスタートすることができる．
(3) **継続的な技術の蓄積：**複数の開発内の小チームからの要望を技術として蓄積することができる．
(4) **多様なゲームデザインの実現：**蓄積した技術を組み合わせて，多様なゲームの要求を実現することができる．
(5) **クオリティの相乗的向上：**再利用を重ねることで，各モジュールをブラッシュアップしクオリティを上げ，また，そのクオリティアップが他の要件にも還元される．

上記をさらに詳細に以下で述べる．

(1) 各 AI の連携によるユーザー体験の実現

メタ AI，キャラクターAI，スパーシャル AI の連携が導入されることで，実現したいゲーム体験をその連携によって実現することができた．また，開発後半では，その連携を活かしたゲームアイデアが提案され実現された．特にメタ AI とキャラクターAI の AI モードによる連携はゲームの要求に幅広く応じるシンプルなシステムを提供し，ゲーム全般に渡って用いられた．

(2) 開発工程の構造化

ゲーム開発が立ち上がった時点では，どのようなゲームデザインになるかは定かではな

い．開発の進展に伴い，開発の中で固まってきた要件を待って実装する．そのため，要件ごとに作成するスクリプティッド AI では，ゲーム開発後半でコンテンツの形が整ってきたところで集中的にスクリプト作業が発生する．しかし，MCS-AI 動的連携モデルは AI 機能を単位として技術を蓄積するため，ゲームジャンルが決まれば，ゲーム開発以前と初期から，技術開発を始めることができ，またゲーム要件に応じてそれらを組み合わせて実現する．また，MCS-AI 動的連携モデルの中で機能を増加させることで，実現可能な事柄を増やすことができ，ゲームコンテンツとして実現できる幅を広くし，逆にゲーム側へ提案することができる．これによってゲームコンテンツの質に貢献する．

(3) 継続的な技術の蓄積

AI 技術開発チームは，開発内の数個のチームから，実現したい要件を同時に受ける．MCS-AI 動的連携モデルによって，それぞれの要件を各 AI の技術モジュールとして蓄積することが可能であり，複数の要件を受ける場合でも，技術的に同一の要件については統一して対処することが可能となる．

(4) 多様なゲームデザインの実現

蓄積されたモジュールを組み合わせることによって，開発当初では予想できなかったゲームデザインの要件についても，モジュールの組み合わせによって実現が可能となる． 開発前半では，各モジュールの開発が主たる仕事となるが，開発後半ではそれらを組み合わせるための拡張と，組み合わせた効果の検証が主たる仕事となる．また逆に，蓄積されたモジュールの組み合わせから，新しいゲームデザインにおける効果を提案することが可能となる．たとえば，第 12.5 節のプレイヤーAI はメタ AI とキャラクターAI の組み合わせによって，第 12.6 節の Face-to-Face 対話システムはメタ AI，キャラクターAI，スパーシャル AI の連携によって，MCS-AI 動的連携モデルの上に構築された．

(5) クオリティの相乗的向上

一端作られたモジュールであっても，開発内のチームから来る要件によって，拡張しクオリティを高めることが求められる．大きな機能としては同じだが，ゲームの要件の特殊性を，モジュールの拡張によって吸収していく．そこで，それらの要件をこなすことで，各モジュールをブラッシュアップしクオリティの高いモジュールへと高めて行くことができる．また，その変更が，それを用いるすべてのゲームの箇所のクオリティアップに相乗的につながっていく．つまり一つのゲーム要件を満たすことが，他の要件に対してもクオリティアップにつながる．

このように MCS-AI 動的連携モデルは，AI 技術とゲームデザインをつなぐシステムとして機能する．MCS-AI 動的連携モデルによって技術とゲームデザイン要件を分けることが

可能となる．これはスクリプティッド AI では不可能なことであった．また MCS-AI 動的連携モデルは技術の AI 技術の蓄積の場として機能し，相乗的にゲーム開発を効率化し，開発スピードを向上させることとなった．

12.8.4　FFXV における AI Graph の効果

FFXV における AI Graph は，ゲームデザイナーがキャラクターAI を作成するためのツールを作成し，それをゲームからロードすることでキャラクターAI が動く仕組みである．AI Graph ツールによって，キャラクターAI のビヘイビアツリーとステートマシンの組み合わせからなるグラフを作成しデータとして蓄積し，ゲーム内ではそのデータをロードすることでゲームキャラクターが動作する．

AI Graph は MCS-AI 動的連携モデルへの拡張，さらに，AI Graph を使用する開発者からの要望によって発展する．

(1) **意思決定技術の自然な導入：**AI Graph を用いることで，必然的に意思決定技術を用いてコンテンツを作ることになる．
(2) **戦略的かつ滑らかな行動の実現：**最上位にあるステートマシンによって，モードの切り替えで，様々なモードの行動を実現する．
(3) **メタ AI との自然な連携：**メタ AI から AI モードが指令され，ゲームの中で複数の役割を果たすことができる．
(4) **意思決定の可視化：**すべての意思決定データが可視化される．
(5) **意思決定の作り方の統一：**意思決定の作り方とフォーマット，要素が統一化されること．また特殊な実装をなくすことができる．また，これによって AI Graph に関わるすべての人が統一の知識基盤の上にコミュニケーションが可能となる．
(6) **技術の共有：**ある要件について AI Graph の新しい機能が実装されると，AI Graph を用いるすべての開発者がそれを用いることができる．これは同時に，その技術を開発する効果の見積もりを上げることになり，そこに開発コストを割く理由となる．
(7) **相乗的なクオリティの向上：**AI Graph と同一のフォーマットによって，バグが AI Graph 上に現れ，解消できることによって，相乗的に意思決定のクオリティが上がって行く．
(8) **他の AI との連携：**メタ AI，スパーシャル AI への関係も，AI Graph 上で表現される．
(9) **開発の効率化：** アセット化とオーバーライド機能を用いて，開発を効率化する．

という利点がある．以下，詳細に各点について述べる．

(1) 意思決定技術の自然な導入

AI Graph を導入することで，ステートマシンとビヘイビアツリーを組み合わせる手法を必然的に使用することになり，意思決定技術のレベルを自然と引き上げることができる．

(2) 戦略的かつ滑らかな行動の実現

AI Graph の利点がコンテンツに反映されることになる．ステートマシンの状況ごとに思考を分割する特性と，ビヘイビアツリーの行動を滑らかに連結する手法がキャラクターの思考に反映された．

(3) メタ AI との自然な連携

AI モードによってメタ AI とキャラクターAI の連携がシンプルに実現されることで，AI モードを利用するシステムを築き，キャラクターAI の自律的な特性を簡単に利用できる仕組みを提供した．

(4) 意思決定の可視化

AI Graph による意思決定の可視化は，意思決定の保守性を上げるものである．つまり，統一された記述方法によって，誰もがその中身を理解すること可能となる．また，後から開発に入った人間でも，以前に作成された AI Graph を理解することで，AI 技術のレベルを上げることができる．

(5) 意思決定の作り方の統一

AI Graph は，AI Graph Editor を通してしか作られないため，意思決定の作り方を全開発において統一化できる．これによってイレギュラーな実装を防ぐことができ，特殊なバグや対応をなくすことができる．ただ逆に，どのような拡張要件に関しても，AI Graph 全体の拡張要件となるため，小回りの利く開発ではなくなる．

(6) 技術の共有

すべての意思決定に関わる要件が AI Graph の拡張要件となる．そのため一つの拡張機能は，AI Graph に関わるすべての人が利用可能となる．また，そのような機能が知らされることで，自らが担当する部分のデザインを改良することを促すことなる．

(7) 相乗的なクオリティの向上

AI Graph を共通の AI Graph Editor 上で作ることによって，AI Graph，および AI Graph Editor の不具合が集約され，これを改善することによって，両者の安定性とクオリティが向上する．

(8) 他の AI との連携

AI Graph と中心として，メタ AI，スパーシャル AI との連携を記述することができる．メタ AI については AI Graph による記述とは限らないが，メタ AI からメッセージを受けるノードが AI Graph 明確に定義されることで，メタ AI とキャラクターAI の関係が可視化される．また AI Graph のノードからスパーシャル AI を呼び出す，或いは情報が提供される，スパーシャル AI が実行されることで，スパーシャル AI とキャラクターAI の関係が可視化される．

(9) 開発の効率化

アセット化とオーバーライド機能を用いることで，AI Graph が再利用され，全体の効率化につながった．また意思決定を開発する開発者同士が同じフォーマットの上で方針を議論することを可能にした．

このように AI Graph は，MCS-AI 動的連携モデルの中心として機能することで，各 AI の連携の表現と実現を可能とした．また，開発に統一性とクオリティの向上を促すシステムとして機能した．また，統一的なフォーマットによる開発者の連携を促すと同時に，あとから開発に入る開発者に対して学習によるキャッチアップを可能とした．

12.8.5 総合的効果

このように FFXV における MCS-AI 動的連携モデルは AI Graph を基盤システムとして構築された．MCS-AI 動的連携モデルと AI Graph による総合的な効果を開発の時期にわけて記述すると以下のようであった．

開発の序盤

(1) AI システムの構築の指針を AI 開発チームに伝えることができた

(2) チーム分けとお互いの役割を理解し合うことができた

(3) MCS-AI 動的連携モデルの各 AI の制作を同時に進めることができた

開発の中盤

(4) 増えて行く AI への要求を MCS-AI 動的連携モデルの各 AI の機能として実装することができた

(5) AI Graph の拡張機能を活用することができた

(6) 不具合を AI Graph 上に集中させることができた

開発の終盤

(7)　「モンスター」「兵士」「仲間キャラクター」チーム間で必要とされた機能を相互に使用することで相乗効果を得た．特に AI モードは全キャラクターAI 共通の仕組みとなった．

(8)　開発終盤から開発に入ったメンバーに，各 AI の機能を活用して素早く開発できる環境を与えることができた

まず開発の序盤に関しては，開発から独立して AI 開発を進める利点がある．これは MCS-AI 動的連携モデルでは，開発コンテンツと技術が独立していることによって可能となる点である．AI 開発がゲーム開発と自律することは，AI 技術の汎用性を高め，また開発効率とスピードを高めることになった．また，これは専門性の高い技術を導入する基盤ともなる．

開発の中盤に関しては，ゲーム開発とのリンクが始まる時期となる．さまざまな AI に対する要件が，キャラクター，ミッション，イベント，戦闘システム，街作成などの担当者から伝えられる．それに対して，あらかじめ準備できていた技術に対しては，3 つの AI の連携によって対処し，また，準備されていなかった技術に関しては，各 AI のモジュールとして開発を進める．これによって MCS-AI 動的連携モデルに技術を集約することができた．

開発の後半に関しては，各開発チーム間で必要とされた機能を実装していくことで，それらのモジュールが相互に活用され，相乗的な開発の効果を得ることが可能となる．開発の末期に新しく参加した開発メンバーに対しても，AI Graph を作成する学習し，作成させることができた．

このように，FFXV における MCS-AI 動的連携モデルは，開発のさまざまなステージにおいて，異なる効果を持ち，開発内で AI 技術を集約・発信する基点としての役割を果たした．

また，第 10 章と本章から以下のことが考察される．

(11)　MCS-AI 動的連携モデルが AI を用いるアイデアを創出させる力を持つことを検証した（第 10 章）

(12)　FFXV の開発では，最初にゲームデザイナーに MCS-AI 動的連携モデルを解説しゲームデザインのアイデアを出させた（第 12 章）

(13)　序盤〜中盤で開発された AI 技術を組み合わせて，終盤では，さらに新しいゲームデザインのアイデアが実現された（第 12 章）．

このように，MCS-AI 動的連携モデルを開発の序盤で開発者，特にゲームデザイナーに話し

ておくことが効果を生み出すために必須である．

12.9 考察

本節は本章のまとめとして，本章全体を俯瞰し考察を行う．本章では，AI Graph を中心に FFXV における MCS-AI 動的連携モデルの実装について述べた．

MCS-AI 動的連携モデルは，FFXV において，ゲーム内においては AI システムの基盤システムとして機能し，開発においては技術の集積と開発者を連携する仕組みとして機能した．ビヘイビアツリーとステートマシンを融合した「AI Graph」をアセット化・オーバーライド・トレイ割り込み実行を用いて拡張することで，FFXV における MCS-AI 動的連携モデルを実装する仕組みが提供された．AI Graph は FFXV において MCS-AI 動的連携モデルの要となる技術であり，この技術の開発によってキャラクターAI に多様な知能の形を，拡張性を保って開発し続けることができた．またメタ AI は自律型 AI を持つキャラクターの行動をゲームの流れに沿った行動に調整する役割を果たした．スパーシャル AI は多様な地形のバリエーションを吸収し汎用的な移動を可能にした．本章では，MCS-AI 動的連携モデルこの 3 つの AI の連携をモデル化し，FFXV への実装とその結果を示した．

13. これからの MCS-AI 動的連携モデルの発展

本章では，MCS-AI 動的連携モデルのこれからの発展について述べる．この発展の内容は，学習・進化・プロシージャル技術と MCS-AI 動的連携モデルとの組み合わせである．第 2.3 節で述べたように，ほとんどの場合，MCS-AI 動的連携モデルはアルゴリズムレベルでは，シンボリズム（記号主義）のアルゴリズムを基礎にしており，コネクショニズムや学習・進化・プロシージャル技術のアルゴリズムを用いる場合が稀である．その理由は，これらのアルゴリズムを用いると，計算負荷が高く，メモリを消費し，結果が予想しにくく，MCS-AI 動的連携モデルが不安定になるからである．ここでは，これからの学習・進化・プロシージャル技術のアルゴリズムの MCS-AI 動的連携モデルへの取り込みを考える（Fig. 13.1，○の領域が学習・進化・プロシージャル技術）．

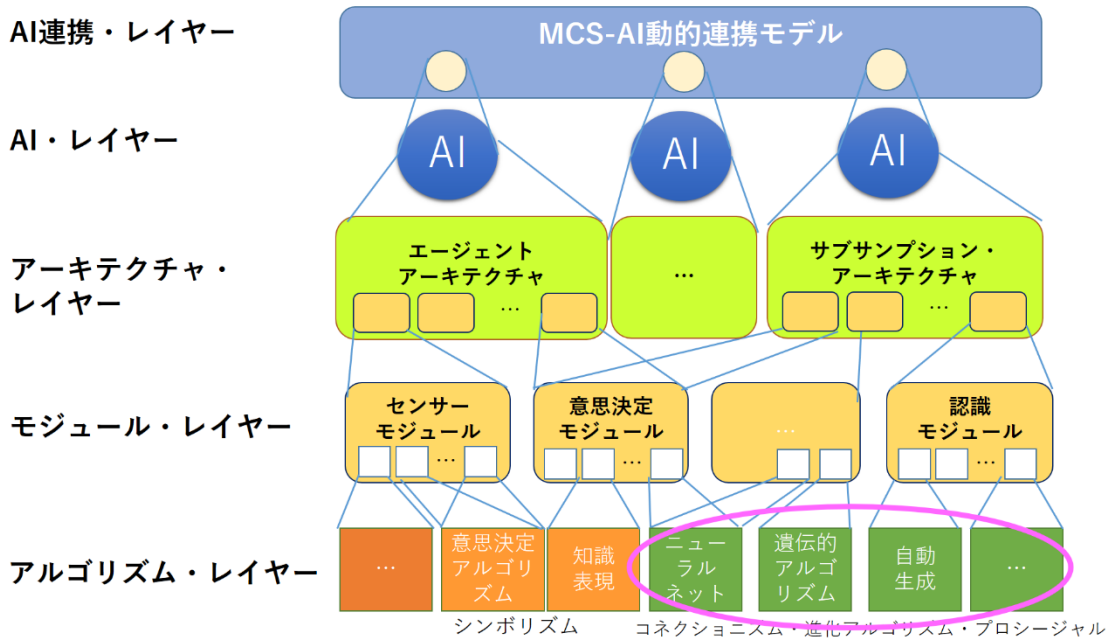


Fig. 13.1　ゲーム AI の多層構造と学習・進化・プロシージャル技術

第 13.1 節，　第 13.2 節では MCS-AI 動的連携モデルとプロシージャル技術の連携について述べる．「プロシージャル技術」（自動生成技術）はゲームの外の AI として，ゲーム開発工程に導入されてきた．将来的には「メタ AI」が取り込むことで，ゲームそのものをある程度，自動生成・自動調整することが可能となる．第 13.3 節では，学習・進化アルゴリズムと MCS-AI 動的連携モデルとの関連について述べる．第 13.4 節ではゲーム AI 技術を編

纂する仕組みとしての MCS-AI 動的連携モデルについて述べる．第 13.5 節ではスマートシティにおける MCS-AI 動的連携モデルの応用について述べる．第 13.6 節では MCS-AI 動的連携モデルの限界について述べる．第 13.7 節ではこれからの MCS-AI 動的連携モデルの研究課題について述べる．

13.1　MCS-AI 動的連携モデルとプロシージャル技術

本節では，MCS-AI 動的連携モデルとプロシージャル技術の組み合わせの構造について述べる．プロシージャル技術とは，ゲームコンテンツを生成する技術である．ゲーム中にリアルタイムに生成する場合から，開発中に何時間もかけて生成を行う場合まであり，プロシージャル・コンテンツ・ジェネレーション（PCG：Procedural Contents Generation）とも呼ばれる［126］. これが広義のゲームＡＩ技術と呼ばれる理由は,「環境や要求に合わせて，ゲーム開発者の代わりにコンテンツを生成する」からである．たとえば，第 3.4.1 項で述べたナビゲーション・メッシュやウェイポイントなど，NPC の移動を司るナビゲーション・データは自動生成される場合が多く，簡易で一時的なものはゲームプレイ中に 1 ～数秒ほどで，複雑で恒久的なものはゲーム開発の段階で数分～数時間かけて生成する．また，地形やダンジョン，植物など，フィールドマップを構成するオブジェクトの自動生成や自動分布にもプロシージャル技術は用いられる（Fig. 13.2）.

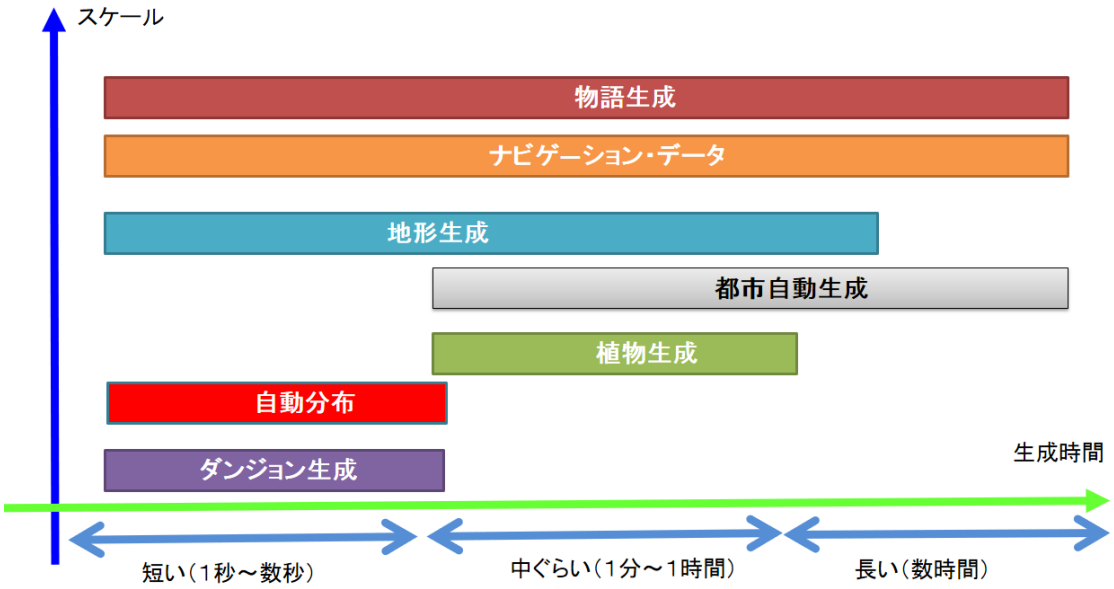


Fig. 13.2　さまざまなプロシージャル技術（生成の時間幅）

メタ AI がプロシージャル技術を用いることで，ユーザーのスキルや嗜好に応じたゲームレベルの生成がより動的になり自由度が増加する．これまでは準備されたマップに対してキャラクターを配置する機能までであったところが，メタ AI がプロシージャル技術を取り

込むことでゲームレベル（マップ，キャラクター配置）全体を生み出すシステムへと発展する．また，生成したレベルに対して，それがどのようなレベルであるか，をチェックするのは，スパーシャル AI の役割である．また，実際にマップを攻略できるかのテストを，キャラクターAI を用いて NPC にゲームをプレイさせることで検証することができる．

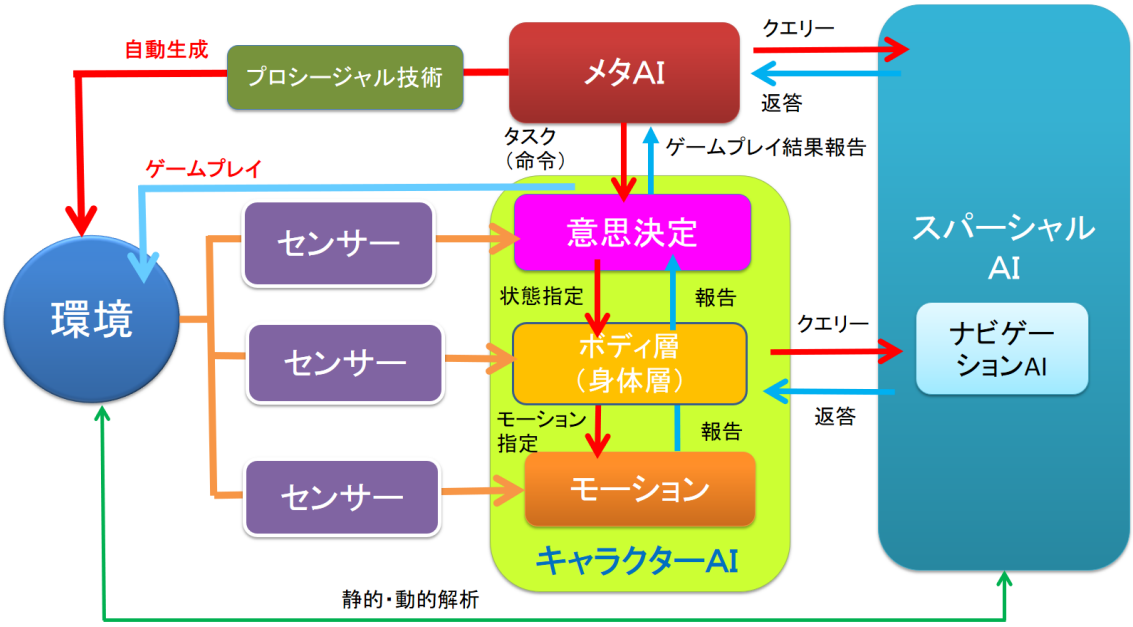


Fig. 13.3　メタ AI とプロシージャル技術

メタ AI，プロシージャル技術，スパーシャル AI が連携することで，よりゲームを詳細なレベルで自動生成することができる（Fig. 13.3）．手続きは以下のようである．

（1） メタ AI がプロシージャル技術を用いてマップを自動生成する．
（2） 自動生成したマップをスパーシャル AI が解析してマップの形状・特徴を認識し，ナビゲーション・メッシュを作る．
（3） ゲーム開始すると動的に影響マップで状況解析し，キャラクターの配置場所を決定し，動的にキャラクターを配置する．
（4） 配置されたキャラクターは自律的にプレイヤーに向かって攻撃を行う．
（5） メタ AI がプレイヤーのプレイログを解析することで，次のマップ生成の仕方を変更する．

これをフルセットで行った事例はないが，（1）～（4）を通した，以下の事例がある．

第 6.3.1 節でも言及した『Warframe』では，マップ自動生成が使われていること，またメタ AI のセンサー部分に「影響マップ」（第 8.4.2 節）が使われていることである．自動生成されたマップは解析されトポロジーが抽出される．それを用いてナビゲーション・メッシュ

が自動生成され，入り口，出口，目的地が自動的に設定される．さらに，ゲーム内では，プレイヤーを熱源としたヒートマップによる熱伝播シミュレーションが行われて各グリッドで温度が定義される．プレイヤーが近づき温度が上がって行く位置に敵キャラクターが動的に配置される．逆に温度が下がって行く位置にある敵キャラクターは消滅される（Fig. 13.4） [94] [95].

このように，MCS-AI 動的連携モデルは，プロシージャル技術を取り込むことで，ゲーム製作・ゲーム認識能力を含んだモデルへ発展することができる．

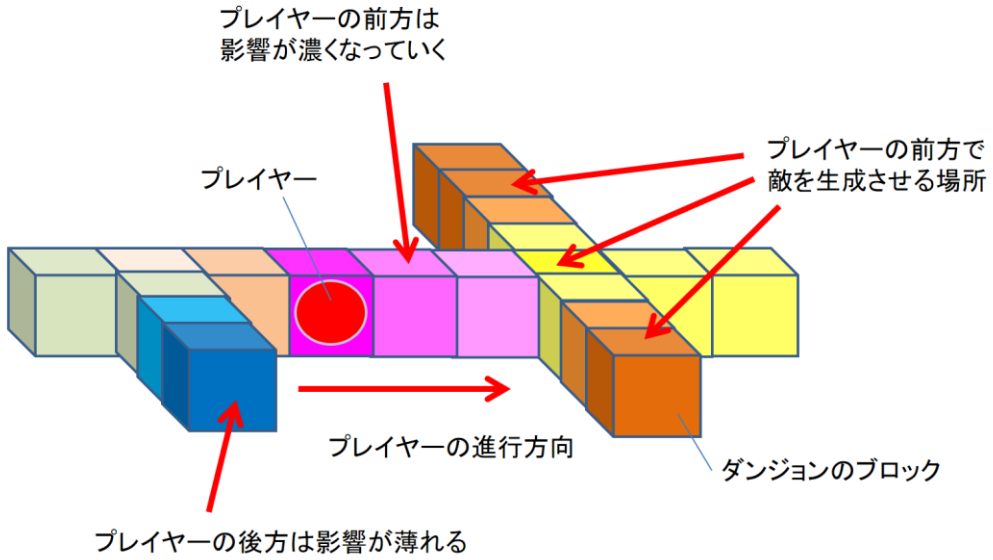


Fig. 13.4　『Warfarme』 におけるヒートマップ [94]より引用.

13.2　自動生成と自動チェック

本節では，自動生成したコンテンツの自動チェックについて述べる．プロシージャル技術によって自動生成したコンテンツを，デジタルゲーム内で使用するためには，そのゲームに応じた必要な要件を満たす必要がある．まずはそのゲームの条件を満たすかをチェックする必要がある．地形の自動生成であれば，そのゲームが求める地形的条件を満たす必要がある．満たせない場合は，再自動生成となるが，何度も揺らぎを持たせつつ自動生成して確認する．何度もくり返し生成できる点が，プロシージャル技術の良いところである．この自動生成したコンテンツが満たすべき地形的条件は「スパーシャル AI」によって確認される．

最も単純な条件は，入り口と出口までキャラクターが移動可能であるか，などである．『Age of Empires II』（Ensemble Studios，1999 年）は，プレイヤーが兵士を動かして対戦するリアルタイムストラテジーゲームである．地形が定まっていては最適な戦術が固定されてしまうため，プレイするたびに地形が生成変化する．生成された地形はゲームプレイに適しているかどうかわからないので，自動生成したマップに対して自陣と相手陣の間にパ

スがつながっているかの自動接続チェック（パスチェック）がなされる．また影響マップ，エリア分割など地形解析を施すことで，ゲームギミックを置く場所の細部を決定して行く[131]．

『Assassin's Creed Origins』（Ubisoft Montreal，2017 年）は，「自動データ解析」システムを構築して，想定する機能を正確に実行できる状態にあるかを確認する「その対象が有効かどうかを確認するテスト」（Validation Test）を行う [168]．たとえば，空中に張られたロープの途中にオブジェクトが被っていないか（あった場合キャラクターがロープの途中で渡れなくなりゲーム進行不能となる），敵キャラクター生成ポイントに対しては「その上に物が置かれていないか」（あった場合，敵キャラクターがゲームステージに出現できない），経路上に岩が転がっていないか（存在した場合，敵キャラクターが通れなくなる），ハシゴの途中で物が阻害していないか（あった場合，キャラクターがはしごを昇れなくなる），など，それぞれの対象ごとに自動チェックプログラムが毎日実行され，テストを通らなかったケースが開発者に自動報告される．

このように自動生成をし，自動チェックのフィードバックを行うことで，より高品質のゲームに合ったバグのない自動生成を行うことができる．

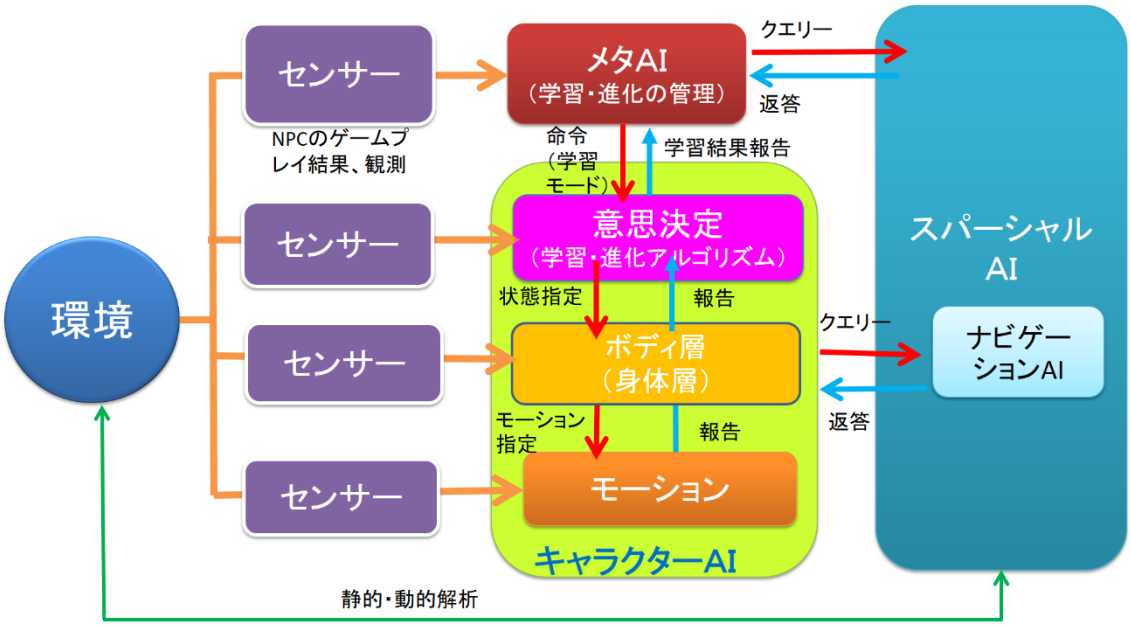


Fig. 13.5　MCS-AI 動的連携モデルと学習・進化アルゴリズム

13.3　MCS-AI 動的連携モデルと学習・進化技術

学習・進化アルゴリズムは，第 2.6 節，2.7 節で紹介したように，ゲーム開発における応用からゲームの中へ応用されようとしている．ゲームの中への応用は，計算・メモリのリソースが十分に存在すれば可能であるが，その後に必要になるのが学習・進化アルゴリズムの

管理である．MCS-AI 動的連携モデルにおいて，学習・進化アルゴリズムの管理はメタ AI が行い，キャラクターAI が学習・発展する方式が考えられる．メタ AI は学習モードを開始する起点として，キャラクターAI に命令を出し，報酬系として働き，キャラクターAI の変化を観察する．キャラクターAI は，メタ AI からの命令により，通常稼働から，学習モードへと移行し，学習・進化アルゴリズムを動作させ，ゲーム環境世界とインタラクションしながら，学習・進化させていく（Fig. 13.5）．

たとえば，MCS-AI 動的連携モデルにおいて，学習・進化アルゴリズムによるゲーム自動調整機能を行うことが可能である．NPC はその行動をパラメーターセットによって決定するとする．遺伝的アルゴリズムでは染色体とみなす．

(1) メタ AI が NPC に対してゲームをプレイする命令を与える．
(2) メタＡＩはその結果をゲームから観測する
(3) 何千という NPC にゲームをプレイさせながら遺伝的アルゴリズムでパラメーターセットを進化させる．

という方法である．第 2.6.3 項で紹介したように，ゲーム内で動作させることができる．

また本手法はゲーム全体の品質保証に用いることができる．『Candy Crash』（King，2012 年）では，1000 を超えるレベル（マップ）がゲームデザインに合っているかを検証するために，AI にゲームプレイさせた [52]．遺伝的アルゴリズムによってニューラルネットのトポロジーを進化させる「ニューロエボリューション」（Neuro Evolution）を用いて，モンテカルロ木探索（MCTS）のプレイアウトを試行ごとに自動改良し，効率良く一つ一つのレベルをクリアしていく．

このように，MCS-AI 動的連携モデルは，学習・進化アルゴリズムを，メタ AI とキャラクターAI の連携によって取り込む可能性を持つ．

13.3.1 メタ AI とニューラルネットワークとしてのキャラクターAI の関係

第 7.3 章では，メタ AI からキャラクターAI の意思決定アルゴリズムへの様々な干渉の仕方を述べた．これからの変化として，このキャラクターAI の意思決定がニューラルネットワークになるケースが増えて行くと考えられる．そこで DQN（Deep-Q-Network）のような強化学習とディープラーニングを組み合わせた事例では，ニューラルネットワークへの報酬をメタ AI が与えることによって，ゲームの各キャラクター，各ステージに適した方向に学習を行わせることが可能となると想定される．この場合，メタ AI はゲーム内でキャラクターを学習させるが，ゲーム開発中に学習させる場合と，ゲーム実行中に学習させる場合の双方が考えられる．

ゲーム開発中に学習させる場合，メタ AI はゲームデザインが要求する性能・性質に対応

する報酬を与える．たとえば，ビヘイビアツリーを作っておいて，そのビヘイビアツリーの動作と一致する場合に報酬を与えることで，ビヘイビアツリーと同様の動作をさせるニューラルネットワークを生成する［169］．またユーザーのプレイログから模倣学習によってログと同様の動作を行うNPCのディープニューラルネットワークを形成する［170］．

ゲーム実行中に開発する場合では，たとえばレーシングゲームや格闘ゲームでユーザーのログを取っておき，ユーザーのプレイの癖を学習する実例がある［51］．またストラテジーゲームにおいて，インストラクターとエグゼキューターを階層的に構築し学習することで，上位階層と下位階層のニューラルネットワークを同時に学習する研究がある［171］．このようにキャラクターAIがニューラルネットワークになった場合には，メタAIはキャラクターを学習させる教師役も担うことになる．

メタAIからニューラルネットワークを用いたキャラクターAIへの命令は，シンボルベースと異なる明示的な命令を行う方法を取ることができない．そこで学習の段階でいくつかのニューラルネットワークを作成しておき，場面ごとにニューラルネットワークを切り替えるという方法が考えられる．また，ニューラルネットワークの出力による行動選択においてバイアスをかける，具体的には，いくつかの行動評価値に数値の上乗せを行うことで特定の行動を高い確率で行わせる手法が考えられる［169］．また学習済みのニューラルネットワークから追加の学習をゲーム内で行う転移学習などが考えられる．

このようにメタAIとニューラルネットワークを用いたキャラクターAIの関係は，シンボルを用いた従来の関係と変化する必要があり，上記で挙げた例も実験的な意味合いが強く，実際のデジタルゲームで応用されている例は殆ど存在せず，これからの研究課題である．

13.4 MCS-AI動的連携モデルによるゲームAI技術の編纂

MCS-AI動的連携モデルは，第9章のようにゲーム開発，およびゲーム分析によって，各AIにAI機能として技術を蓄積していく．MCS-AI動的連携モデルは，そのモデルの元に，ゲームAI技術を蓄積したライブラリとしての役割を果たすことになる．同じモジュールをくり返し開発する「車輪の再発明」を防ぐためには，蓄積された技術リストが公開され，公開された技術は他のゲームタイトルでも利用可能なものでなければならない．

MCS-AI 動的連携モデルが真にその特性を発揮するためには，タイトルを超えて運用されることで，一つのタイトルを超えてその企業，その団体におけるAI技術を引き上げる効果を持たなければならない．

またゲームデザインから逆引きできる必要がある．ここで言うゲームデザインとは，ゲーム全体のデザインというよりは，第12章でMCS-AI動的連携モデルの例として提示したキャラクターを連携する方法など，各シーンを形成していくための仕組みのことを指している．つまり，どのような技術の組み合わせがどのような効果を持つか，とは逆に，このようなゲームデザインの効果を持つためには，どのような技術の組み合わせが必要かを検索で

きる機能である．このようなゲームデザインの効果は，実際のゲーム開発中で実装された事例や，デモなどを通して蓄積され，記述され公開される必要がある．このようなゲームデザインの効果は第 12 章のように開発における発見によって見出されるものであり，そういった発見を蓄積していくことで，ゲームデザインのバリエーションも技術と同様にライブラリ化していけるのである．このようなゲームデザインの蓄積は直接，将来のゲーム制作におけるゲームデザインに役立つものである．

13.5 スマートシティにおける MCS-AI 動的連携モデル

MCS-AI 動的連携モデルはスマートシティにおける AI システムとして導入できる可能性がある．その探求はこれからの課題である．スマートシティは都市全体を知能化することで，セキュリティ・流通・キャッシュフローまでを人工知能によって管理するシステムである [172] [173]．本節では以下の 2 点を示す．

(1) スマートシティのアーキテクチャにも，第 3 章で述べたアーキテクチャを応用して設計することができる．
(2) そのアーキテクチャの上に MCS-AI 動的連携モデルを応用することができる．

第 13.5.1 項ではスマートシティにおける人工知能アーキテクチャについて述べる．第 13.5.2 項ではスマートシティにおける MCS-AI 動的連携モデルの導入について述べ，第 13.5.3 項では実現へ向けて課題と展望について述べる．

13.5.1 スマートシティにおける人工知能アーキテクチャ

スマートシティが対象とするのは都市全体であるから，一つの人工知能で全体を観測し管理することはできない．そこでゲームキャラクターにおけるサブサンプション・アーキテクチャ（Fig. 3.16）と同様にマルチレイヤー・アーキテクチャ（第 3.5.2 項）とサブサンプション構造（第 3.5.3 項）を組み合わせたアーキテクチャが適切である．各層が独自のセンサー系統を持ち意思決定を行うが，上位下位の層と互いに連携する．以下，アーキテクチャ内の各モジュールのデザインの一例を述べる．

まず最上層に都市全体を監視・制御する人工知能を配置する．この層の役割は下から上がって来る情報をもとに最終的な意思決定を行うところにある．また，自らのセンサーを持ち都市全体の情報を監視し意思決定を行う．また都市の代表の人間と対話し必要な場合は決定の許可をあおぐ．その下の層に都市を適切なエリアに区切った領域，また交通や高速道路などエリアをまたいだインフラストラクチャーに対して監視・制御する人工知能を配置する．その領域に属する人々の安全や円滑な経済・社会活動をサポートする．同じ層に属す

る複数の人工知能モジュールは互いに連携を行う．特に災害時などでは，この層が率先して領域に属する人々の安全を確保する．また区の境界などは重複領域を作り，円滑に各モジュールが連携可能なように設計を行う．情報は定期的に上位層の人工知能に報告する．また上位層の人工知能からの命令や抑止，委任などを受け入れる．またこの層においても人が監視し制御できるように，人と対話し指示を受け入れるコミュニケーション・チャンネルを備えておく．このように各層が都市へとセンサーを持つサブサンプション・アーキテクチャとなっている（Fig. 13.6）．

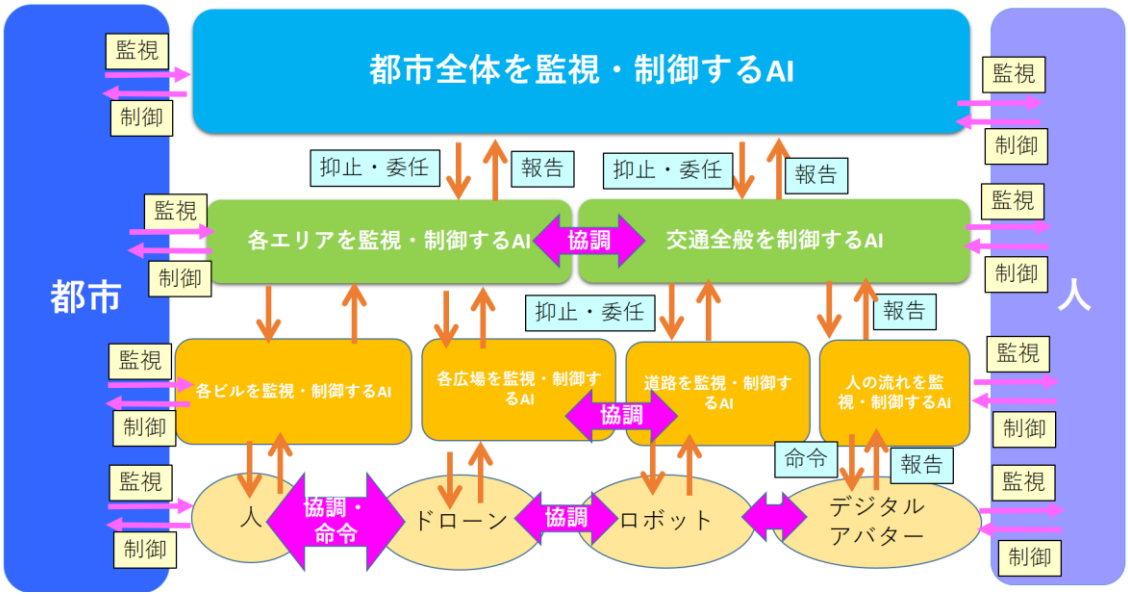


Fig. 13.6 スマートシティにおける人工知能アーキテクチャ

次に各設備・各建築，ビルや広場や会議場，発電所など公共施設を監視・制御する人工知能を配置する．施設の安全やそこに関わる人々の安全を確保すること，また適切な経済・社会活動を監視・制御することが目的である．この層の人工知能はその施設の管理者やオーナーと対話し指示を受け入れる．次に最も基本的な層として，実際に現場レベルで活動する人と人工知能が協調を行う．この層では，ドローンやロボット，デジタルアバター（デジタルキャラクター）が人の活動をエンハンスメント（拡充）することで効率的な仕事の達成が目指される．またこの層においても人が全体を監視・管理が行えるようにする．また本アーキテクチャの中で最下層はエフェクターに相当する．本アーキテクチャは，モジュールのスケール・数によって，都市の大小にかかわらず適用可能である．

スマートシティにおけるアーキテクチャは各層で人の人工知能モジュールの間の対話が用意されている．これは各層の人工知能に対して人が抑制と干渉を担保するためである．たとえば最上層の人工知能ではメタ AI は人と対話する．たとえば都市であれば，その代表と会話することができ，また都市の代表がその人工知能をコントロールすることができる．ま

た緊急停止する権限を持つ．また各層における人工知能モジュールに対して，それと対話する権利を持つ人が存在する．該当する人間は各人工知能モジュールを監視し必要に応じてコントロールする．人工知能システム全体は各層で人が干渉する権限を持つことで，システムの欠陥を補完し暴走を抑制する．

13.5.2 スマートシティにおける MCS-AI 動的連携モデルの導入

本項ではスマートシティにおける MCS-AI 動的連携モデルの導入の設計について述べる．第 13.5.1 項で述べたアーキテクチャの各 AI 層より詳細な構造として，最上層をメタ AI，また下層をキャラクターAI，さらに都市の空間的特性を抽出し提供する AI としてスパーシャル AI を導入する設計が考えられる（Fig. 13.7）．この設計によって，キャラクターAI の意思決定技術を各モジュールに適応することができ，またメタ AI，スパーシャル AI との連携も MCS-AI 動的連携モデルで構築した関係を利用可能である．以下に本設計の詳細について述べる．

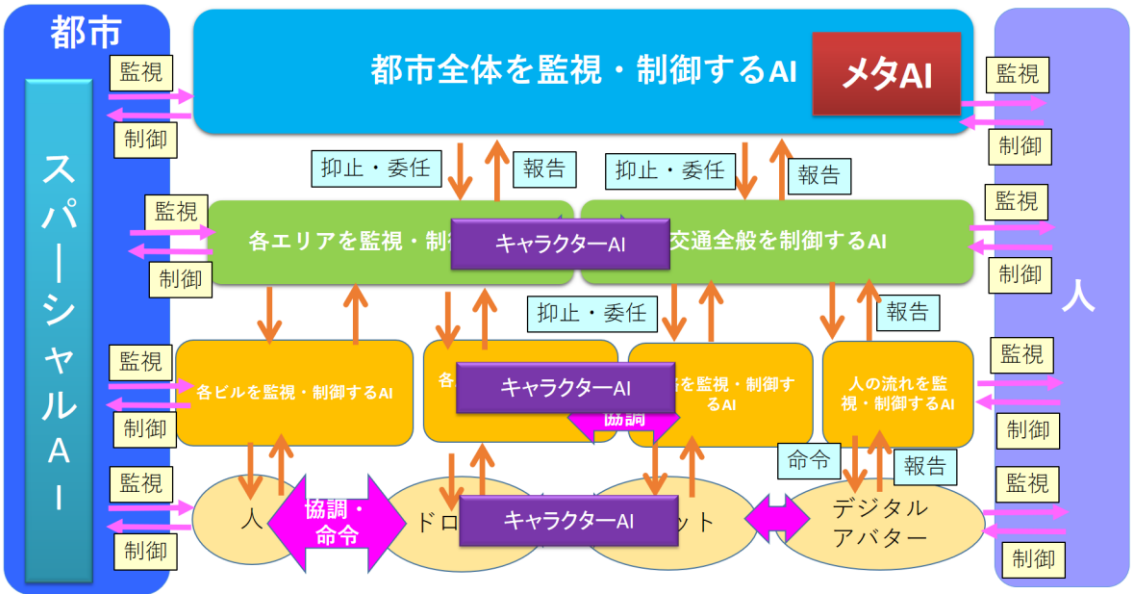


Fig. 13.7 MCS-AI 動的連携モデルを用いたスマートシティの人工知能アーキテクチャ

メタ AI は都市全体を認識し影響を与える．認識は大局・長時間に渡る都市の状態の変化の認識と，インパクトの強いイベント，たとえば災害による局所的な事故など，である．この認識をもとに，メタ AI は下位層の人工知能に影響を与える．影響の仕方は第 7.2 節で述べたように下位の人工知能の意思決定の仕方に依存するが，６つの命令タイプのうち演技型命令を除く 5 つのタイプを使用することが可能である（Table 13.1）．投機型命令はメタ AI から下位の人工知能に報告を必要としない命令である．「エリアを清潔に保て」「もし事

故があったら報告せよ」「この地域は特に注意して監視せよ」などである．提案型命令は下位層の人工知能から提案される命令である．その地区で事故が発生したときの「救援要請」やとなりのエリアを手助けしていいか「援護申請」などである．目的型命令は達成するべき目的をメタ AI から下位層に与える．下位の人工知能はこの目的を達成するように意思決定を行う．厳密な上から下への連携にはこの目的型命令が適している．モード型命令はそれぞれの第二層以降の人工知能についてモードを持たせることによって有用である．各人工知能は通常時の運転モードであり「通常モード」，事故が発生した時の「緊急モード」，大きなイベント（祭りや季節的なイベント）があるときの「警戒モード」などを設定しておき，通常は上から下へ通常モードが指定され，それ以外の場合には他のモードが発令される．複数の人工知能モジュールを連携するときには「連携命令」を発令される．たとえば高速道路で事故があれば，高速道路を担当する人工知能モジュールと，そのエリアの人工知能を協調動作させる，などである．

Table 13.1　メタ AI の命令タイプとその内容

メタ AI の命令のタイプ	メタ AI から下位の人工知能への命令の内容
演技型命令	使用しない
投機型命令	報告を必要としない命令
提案型命令	下位層からの命令の提案
目的型命令	目的を与えて，目的を解除するまでは遂行させ続ける
モード型命令	継続的に一つの方針を強いる
連携命令	複数の下位層のモジュールに命令を与えることで連携的な実行を促す

スマートシティにおけるスパーシャル AI の役割は，都市の地形的情報，都市上のオブジェクトとその配置，配置が作り出す空間的情報を抽出し，モニターし，積極的に活用するための地形・空間情報を，メタ AI，キャラクターAI に対してコミュニケーションを通じて伝えるところにある．空間解析・状況解析は第 8 章で述べた技術を用いることができる．加えて，カメラ画像や映像の解析技術を用いる必要がある．解析から得た情報をメタ AI，キャラクターAI に自発的に伝える，或いは，要求を受けて解析した情報を伝える．スパーシャル AI は，アーキテクチャの各層に情報を提供し，各層のセンサーとしての役割を持つ．

13.5.3 課題と展望

本項では，スマートシティにおける MCS-AI 動的連携モデルの導入の課題と展望について述べる．都市の中の空間・建築は社会的に所有者や管理主体が決められている．第 13.5.1 項，第 13.5.2 項で述べた人工知能の階層構造は，各空間・各建築の社会的な問題，つまり

所有権や安全性，守秘義務について考慮されていない．また，それぞれのモジュールが代表する施設や空間，建築の利益がお互いに相反しないかを考慮する必要もある．そのように都市の各部分が持つ対立と問題を解決する合意の仕方を法によって取り決めておく必要がある．特にローカルな合理性とグローバルな合理性は一致するとは限らない．たとえば，下位の層における合理性が，上位における合理性が異なる場合には，このコンフリクトを解消する合意の方法を決めておく必要がある．このような社会的な問題の解決には都市の最上位の組織からのルールの決定が必要である．

技術的な課題としてはハードウェアとソフトウェアの両面がある．ハードウェアとしては，都市を観測するカメラや赤外線など様々なセンサーの配置，そしてロボットやドローンなどエフェクターの稼動が必要とされる．ソフトウェアはそれらのハードウェア間のデータリンクの構築が必要とされる．このようなシステムの実現には，まず小規模な実験，大学のキャンパスや一つのビル内での実現から始めて徐々にスケールアップしながら実験を繰り返す必要がある．

スマートシティの研究は，まず 3D グラフィクス上で実現された仮想都市空間において行うことでソフトウェア部分のテストを行うことができる．そこで見つけた課題を一つ一つ設計に織り込んで行くことでソフトウェアの大半の部分をテストし構築することが可能である．同時にハードウェアとしてのセンサー，エフェクター（ロボット，ドローン）は小規模な環境，つまり一つのビルや大学の中などから実験を始める．そういった小規模な実験から次第にスケールをアップしていく．そして，現実世界を模したデジタル世界である「デジタルツイン」の応用がコンテンツレベルの研究課題として存在する．

13.6 MCS-AI 動的連携モデルの限界

MCS-AI 動的連携モデルの限界は，各 AI の限界と，総合的な限界がある．

まず各 AI の限界について述べる．メタ AI の限界は，ユーザー理解の限界である．ユーザーの心理状態を推定する場合など，ユーザーをどれだけ深く認識できているかが，MCS-AI 動的連携モデルの効果の深さを決める．キャラクターAI は，ユーザーの動作にどれだけリズムを合わせられるかがユーザーのゲーム体験を決定する．キャラクターAI は，プレイヤー・キャラクターの動きに対する反応するが，単なる反射的な反応だけではリズムを作ることはできない．ある場合には，NPC が動作のリズムを作り，ある場合には，ユーザーのリズムに合わすことで，相互に創発的にアクションのリズムを作り出すことができる．このようなキャラクターの動作にメタ AI によるユーザーの分析，スパーシャル AI による空間の解析が必要であるが，キャラクターAI はユーザーの分析の限界，空間的特報の抽出の限界によって，ユーザーとの協調関係の限界を持つ．スパーシャル AI の限界は空間・状況の分析の解像度の限界でもある．細かいスケールでの分析から，大局的な地形解析までスパーシャル AI による空間認識はユーザーの行う柔軟なマルチスケールの空間認識に追随するこ

とは難しい．この限界が MCS-AI 動的連携モデルの空間的性能の限界を決定する．

これらの限界は，特にスマートシティなど現実空間において MCS-AI 動的連携モデルを適用するときに現れることになる．まず現実空間におけるメタ AI は生身の人間，或いは，人間の集団を認識し，意思決定を行う必要がある．自由な人間の心理推定，事故時の多数の人間の行動推定など，メタ AI は現実の不定性の中で人間や人間の集団を理解する必要に差迫られる．その推定の限界は，そのまま MCS-AI 動的連携モデルの限界でもある．

13.7 これからの MCS-AI 動的連携モデルの研究課題

本節では, MCS-AI 動的連携モデルのこれからの研究課題について述べる. 現在の「MCS-AI 動的連携モデル」は主に記号主義型人工知能技術の構造化による階層化によって成り立っている．これからの研究課題は，学習・進化・プロシージャル技術のアルゴリズムを，「MCS-AI 動的連携モデル」に組み込むことで，本モデルをより発展させ，これまで届かなかったユーザー体験を実現する土台とすることである．

MCS-AI 動的連携モデルは，ゲーム AI の総合システムとして提案された．プロシージャル技術は，そのモデルをゲームの自動生成・自動品質チェックのシステムとして発展させる可能性を持つ. 現段階では, そのシステムの一部が, さまざまなタイトルで実現されている. 第 11 節，第 13.2 節で示したように，プロシージャル技術を含んだ MCS-AI 動的連携モデルは，ゲーム AI の総合システムから，ゲーム開発の総合システムとして発展する可能性を持っている．

MCS-AI 動的連携モデルはまた, 学習・進化アルゴリズムをキャラクターAI に組み込み，メタ AI がその学習・進化を管理する，という方式の可能性を述べた．今後は，この 2 つの可能性を実際に検証して行くことで，MCS-AI 動的連携モデルを発展させていく．

14. 全体の結論

本論文では，デジタルゲームにおける「MCS-AI 動的連携モデル」を提案し，実装例を示した．本章では，本論文の各章を振り返り，全体の論旨をまとめる．第 14.1 節では各章の主題を振り返りまとめる．第 14.2 節では，第 1 章で述べた「MCS-AI 動的連携モデル」の効果について検証する．第 14.3 節は本論文をまとめる．

14.1 各章のまとめ

本節では，本論文全体について各章の内容をまとめる．

第 1 章では，本論文の主旨と全体図を示した．「MCS-AI 動的連携モデル」（Meta-Character-Spatial AI Dynamic Model）は，ゲーム開発の歴史の中で形成された「メタ AI」，「キャラクターAI」，「スパーシャル AI」の連携モデルである．「物語的ゲーム」と「アクションゲーム」の二つの異なる AI システムが融合したモデルである．

第 2 章では，「MCS-AI 動的連携モデル」の人工知能技術分野の中での位置づけを行った．アルゴリズムからモジュール，アーキテクチャ，そして AI 連携レベルへと，ゲーム AI 技術が構造化されて来た発展を説明した．前半では「MCS-AI 動的連携モデル」は，AI 連携層のモデルであり，デジタルゲームの人工知能の最も抽象度の高いモデルであることを示した．「MCS-AI 動的連携モデル」の発展はゲーム開発の構造化の大きな流れではあるが，一方でアルゴリズムでも新しく「学習・進化アルゴリズム」のデジタルゲームへの導入が始まっている．後半では，この流れをまとめた．

第 3 章では，本論文の理解に必要なデジタルゲームの基本知識をまとめた．それぞれの知識は，人工知能の汎用的知識であるが，ゲーム産業の中でそれぞれ発展してきた知識でもあり，それぞれ組み合わされて新しくデジタルゲームの構造を作ってきた．特に，メタ AI はデジタルゲームのオリジナルの概念であり，知見が積み重ねられてきた．

第 4 章では，ゲーム産業で形成されてきた従来のゲーム AI 理論と課題を述べた．デジタルゲームが物語的ゲームとアクションゲームという分類を超えて，両者を融合したゲームへと発展する時に，ゲーム AI システムも発展する必要があり，そこで「LCN-AI 連携モデル」（Level-Character-Navigation-AI co-operation model）が形成された．しかし，ゲーム AI システムに求められたのは，レベルスクリプトのように，それぞれ想定された状況ごとにコンテンツを作り込む手法ではなく，オープンワールドにおいてより動的にゲームコンテンツを生成するシステムだった．そこで，レベルスクリプトはメタ AI へ，ナビゲーション AI はスパーシャル AI へと発展した．

第 5 章では，MCS-AI 動的連携モデルの導入と解説を行った．LCN-AI 連携モデルが持つ欠点が，MCS-AI 動的連携モデルにおいてどのように解決するかを述べた．また，LCN-AI

連携モデルが，デジタルゲームの AI に対する要求を実現するシステムであるが，MCS-AI 動的連携モデルはユーザー体験のレベルで体験を作って行くモデルであることを述べた．MCS-AI 動的連携モデルは各 AI に対する役割を明確にし，それぞれが高度な AI になることで，それらの相乗効果をゲーム全体にもたらす仕掛けでもある．

第 6 章では，MCS-AI 動的連携モデルにおけるメタ AI の持つ構造と役割と機能について述べた．メタ AI の内部構造について述べた．されに，メタ AI のゲームに対する関わり方について分類を行った．メタ AI には「キャラクター」「ゲーム全体」「コンテンツ生成」をコントロールするレベルがあり，それぞれがユーザー体験上で異なる効果を持つことを述べた．

第 7 章では，MCS-AI 動的連携モデルにおけるキャラクターAI の持つ役割と機能について述べた．本モデルにおけるキャラクターAI の意思決定には，自律的意思決定と演技的意思決定があるが，これを別々に実装するのではなく，メタ AI からの命令を受けて，シームレスにお互いを行き来する仕組みを述べた．意思決定アルゴリズムには 7 つの種類があり，MCS-AI 動的連携モデルにおいて，メタ AI とどのような関係性を持つかを整理した．

第 8 章では，MCS-AI 動的連携モデルにおけるスパーシャル AI の持つ役割と機能について述べた．スパーシャル AI は「空間解析」と「状況解析」の機能を持つ．メタ AI，キャラクターAI が思考する土台となるフレームのための情報を準備し，またキャラクターAI の行動形成において必要な情報を提供する．

第 9 章では，MCS-AI 動的連携モデルの既存のゲームへの適用を検討した．『パックマン』における AI システムは，アルゴリズムレベルで高い完成度で構成されていが，MCS-AI 動的連携モデルは，その構造を 3 つの AI として引き継ぎ，より拡張性のあるゲーム AI の設計として再構成した．『クロムハウンズ』における AI システムは，チーム AI，キャラクターAI，ナビゲーション AI，の組み合わせである．これを MCS-AI 動的連携モデルへ拡張することで，戦闘の創出から，面白い戦局を創出するシステムへと発展させることができることを示した．

第 10 章では，MCS-AI 動的連携モデルがゲームデザインに多様性をもたらすことを検証するために，比較検証実験を行った．MCS-AI 動的連携モデルの解説をする被験者と，MCN-AI 連携モデルのみを解説する被験者の間で，アイデアの数の相違は見られなかったが，前者は後者に比べ各 AI を用いたアイデア，さらに各 AI を連携させたアイデアを出す，という結果が得られた．

第 11 章では，AI Graph と MCS-AI 動的連携モデルの関係について述べた．AI Graph は複数の意思決定アルゴリズムを組み合わせる仕組みであり，この方針に基づいて実際に実装された意思決定作成ツールが AI Graph Editor である．この基本的な意思決定のシステムを MCS-AI 動的連携モデルへと適応させるための拡張について述べた．

第 12 章では，MCS-AI 動的連携モデルに基づくゲーム開発事例として，FFXV の AI システムを取り上げた．MCS-AI 動的連携モデルを大規模開発で実装するための，コア技術とな

るのが，キャラクターの意思決定を構築する AI Graph である．AI Graph はキャラクター AI の量産を可能にするさまざまな機能を有しており，この機能を用いてバリエーションのある AI Graph 群を製作することができた．また，AI Graph からメタ AI，スパーシャル AI と連携を行うことができた．また MCS-AI 動的連携モデルが，アルゴリズム，モジュール，アーキテクチャ, AI 層の, 技術階層を内包し活用するフレームとなっていることを述べた．

第 13 章では，MCS-AI 動的連携モデルが学習・進化・プロシージャル技術と結びつくことで，ゲーム開発・チェック機能を本モデルが吸収し，ゲームとゲーム開発全体をコントロールする AI システムとなる可能性を示した．また，学習・進化アルゴリズムも，MCS-AI 動的連携モデルに組み込める可能性を示した．

14.2 MCS-AI 動的連携モデルの効果の検証

第 1 章で MCS-AI 動的連携モデルの効果ついて以下のように述べた．

(1) ゲーム AI 開発を効率化する
(2) ゲーム AI 開発を単なる技術ではなく，ユーザー体験の形成を行うゲームデザインと結びついたシステムとする．
(3) 過去のゲームの AI システムを本モデルから捉え直すことができる．

本節では，本論文の内容をふまえて，この 3 点を検証する．

(1) ゲーム AI 開発を効率化する

MCS-AI 動的連携モデルは，ゲームの各要求に対してコンテンツをスクリプトで記述するレベルスクリプト，或いはスクリプティッド AI に対して，各 AI（「メタ AI」「キャラクターAI」「スパーシャル AI」）の機能として蓄積する（Fig. 14.1）．この蓄積された機能を掛け合わせることで, 必要とされる知的機能を実現して行くため, 技術の蓄積が終わったあとは, 新規に機能を開発することなく, 組み合わせるだけでゲームの要求に応えることができる．第 12 章では，MCS-AI 動的連携モデルが技術を蓄積し組み合わせる仕組みであり，同時に開発者の仕事を組み合わせるシステムとして機能していることを, FFXV の開発を通して示した．

(2) ゲーム AI 開発を単なる技術ではなく，ユーザー体験の形成を行うゲームデザインと結びついたシステムとする．

MCS-AI 動的連携モデルは，技術や機能は各 AI の中に蓄積されている．そして，MCS-AI 動的連携モデルは, 実現の求められるユーザー体験から必要な技術を 3 つの AI に分解して実装を促す．各 AI が連携するレベルで生み出されるのは，ユーザーとゲームとのインタ

ラクションであり，各 AI や組み合わせ方を調整することで，ユーザー体験を調整することができる．第 6 章ではメタ AI が中心となりユーザー体験を形成すること解説した．第 12 章の FFXV では，ゲーム経験のデザインを「メタ AI」「キャラクターAI」「スパーシャル AI」の機能要件に分解し，技術開発を行い，それらを組み合わせることで，ゲーム体験を形成する過程を示した．

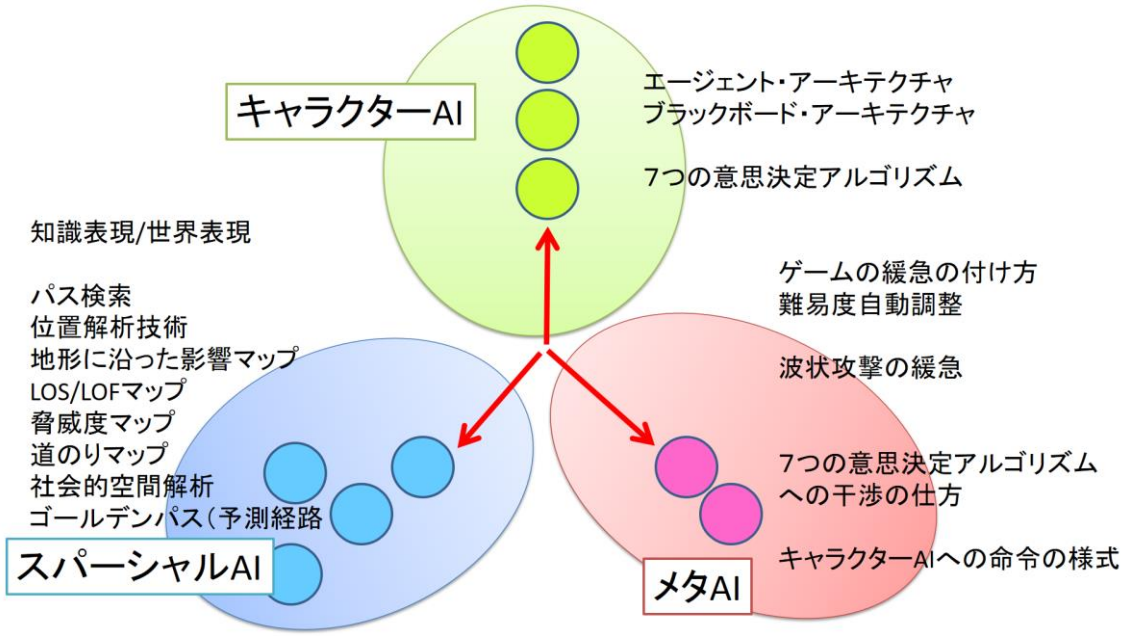


Fig. 14.1　MCS-AI 動的連携モデルにおける各 AI の機能

(3) 過去のゲームの AI システムを本モデルから捉え直すことができる．

第 9 章では，MCS-AI 動的連携モデルを『パックマン』と『クロムハウンズ』に適用することで，各 AI（「メタ AI」「キャラクターAI」「スパーシャル AI」）における技術を抽出することができた．抽出すると同時に，各 AI にどのような技術を準備すれば，AI システムを発展させ，ゲームを発展できるかを考察することができた．これは様々なゲームを MCS-AI 動的連携モデルから捉えることで，新しくそのゲームを発展させる AI のポイントを見つけだす可能性を示している．第 10 章では MCS-AI 動的連携モデルがゲームデザインに多様性をもたらすことの検証実験として，被験者に AI を用いた新しい『パックマン』のアイデアを記述させる実験を行った．結果として，MCS-AI 動的連携モデルは，被験者のアイデアをより AI と結びついたアイデアとして再構成することがわかった．

14.3　本論文のまとめ

本節は，本論文の総括を行う．デジタルゲームの AI システムは，タイトル開発の中でさまざまな試行錯誤がなされ，蓄積された技術がその度は構造化されてきた．MCS-AI 動的連

携モデルは，くり返し構造化されてきたデジタルゲーム AI の最大の構造の一つである．逆に，MCS-AI 動的連携モデルから，各 AI の役割と機能が明確に定義される．MCS-AI 動的連携モデルを導入して開発することで，開発タイトルに大きな構造を入れると同時に，チームの力を集約し，効率的に AI 開発を進めることが可能となる．

MCS-AI 動的連携モデルは，ユーザー体験を形成するモデルである．各 AI の機能の連携が，一つのユーザー体験に集約される．MCS-AI 動的連携モデルの上に，どのような組み合わせで，何を生み出すかについて，事例を紹介することで示した．どのような組み合わせが，どのようなユーザー体験を生み出すか，が AI 連携レベルの技術である．つまり技術の組み合わせと，ユーザー体験の対応が技術として蓄積される．MCS-AI 動的連携モデルを通じたデジタルゲームの研究は，ゲームデザインと AI 技術を結ぶ研究である．

MCS-AI 動的連携モデルは，それぞれの AI が自律的に動作しつつ，ダイナミックに連携するモデルである．それぞれの AI は，ゲームの異なる様相を観察しつつ，意思決定を行い，動作する．メタ AI はユーザーとゲーム全体の変化に注目し，キャラクターAI は，その周囲の状況変化を注目し，スパーシャル AI はゲームの環境世界について観察する．MCS-AI 動的連携モデルによって，ムービーを使うことなく，インタラクティブなゲームの中で，NPC に自律型思考を与えながらも，指示によって役を演じさせることが可能となる．

MCS-AI 動的連携モデルは既存のゲームに適用することで，そのゲームの AI の機能を抽出することができる．また，MCS-AI 動的連携モデルに沿って新しくゲーム AI の機能を考案することで，既存のゲームを新しく拡張するアイデアを得ることを可能にする．

MCS-AI 動的連携モデルの実装には，より具体的な構造を入れる必要がある．「AI Graph」はグラフ形式の意思決定アルゴリズムであり，ステートマシンとビヘイビアを階層的に組合せる方法である．FFXV の意思決定はすべて AI Graph Editor で製作され，このツールを使用することで AI 開発者の作業を集中することで効率化することができた．キャラクター AI とメタ AI とはモード型命令「AI モード」を通じて連携し，メタ AI，キャラクターAI とスパーシャル AI は「PQS クエリー」「パス検索」によって連携した．また，キャラクター AI とスパーシャル AI は「スマートロケーション」を通じても連携した．

MCS-AI動的連携モデルは，これまで学習・進化・プロシージャル技術を取り込むことができていなかった．これからの研究課題は，学習・進化・プロシージャル技術のアルゴリズムを，「MCS-AI動的連携モデル」に組み込むことで，本モデルをより発展させ，これまで届かなかったユーザー体験を実現する土台とすることである．

（本文部分終了）

15. 付録 1:メタ AI の形成の歴史と背景

「メタAI」という言葉は, 2005年のウィル・ライトの講演で最初に使われた [1]. しかし, それ以前から特定の用語はなくとも「メタAI」と同等の機能が実装されていた. またゲームをコントロールするAIとしては, メタAIと類似した「AI ディレクター」という概念が2008年に提案されゲーム産業内に浸透している. 本章では, メタAIとAI ディレクターの事例の発展を追い, メタAIの定義を明らかにする.

15.1 メタ AI と AI ディレクターの関係

第6.1節で示したエージェント・アーキテクチャの構造をもとに, 「使っている名称」「センサー領域」「意思決定事項」「エフェクターの影響範囲」をTable 15.1に年代順にまとめた.

AI ディレクターは, エフェクターとしてキャラクターに特化しており, メタAIのエフェクターはキャラクター及びゲーム世界全体である. そこで形式的には, AI ディレクターはメタAIに含まれる概念と言うことになる. AI ディレクターは, 戦闘と非戦闘時間の緩急をつける仕組として始まり, それ以降も戦闘を中心としたイベントレベルの調整役を果たしている.

一方, メタAIの始まりはゲームシステム全体或いは一部の自律化にあった. つまりゲームデザイナーがスクリプトを用意するのではなく, ゲームがそれ自身の自律性で動作し発展するのが『シムシティ』『The Sims』『Spore』である. 『Assassin's Creed Origins』（Ubisoft,2017年）のメタAIは, ゲーム全域のオブジェクトとキャラクターをコントロールしている [174]. つまりメタAIはゲームシステムレベルの自律化である. また, AI ディレクター とメタAIでは指向される目標が異なる. AI ディレクターはユーザーの体験を指向しているのに対して, メタAIはゲーム全体の整合性（consistency）が指向されている. ここで言う整合性とは展開が首尾一貫している, 世界観としてリアリティがある, という意味である. FFXVでは場面に沿った会話が, 『Assassin's Creed Origins』 では場所に応じたキャラクターの振る舞いがメタAIを通じて実現されている.

Table 15.1　メタ AI，AI ディレクターの歴史的年表

ゲームタイトル 分類名	名称	センサー範囲	意思決定 （決定すること）	エフェクター （影響範囲）
パックマン	波状攻撃	ゲーム状態	プレイヤーを包囲する・しない	敵キャラクター
岩谷徹	レベル自動コントロールシステム	ユーザーのスキル（命中率や到達率）	ゲームの難易度	敵キャラクター
ゼビウス	AI	ユーザーの進行度	出現する敵テーブルの開始点	敵キャラクター
シムシティ	メタAI	ゲーム状態	街の発展	ゲーム世界全体(街)
The Sims	メタAI	ゲーム状態	キャラクター生成・街の発展	ゲーム世界全体（街・人）
Spore	メタAI	ゲーム状態	キャラクター、森、惑星の創造	ゲーム世界全体（宇宙）
LEFT 4 DEAD	AI Director	ユーザーの緊張度・位置 ユーザーのスキル	予測経路・配置キャラクター・配置タイミング・配置場所・	敵キャラクター
Warframe	AI Director	ユーザーの緊張度・位置	予測経路・配置タイミング・配置場所	敵キャラクター
FarCry4	AI Director	ユーザーの行動ログ 場所の情報	発生させるイベントの種類・場所・タイミング	プレイヤー以外の全てのキャラクター
FINAL FANTASY XV	メタAI	プレイヤー・キャラクターと仲間、周囲の状況	仲間キャラクターの振る舞い	仲間キャラクター
LOST REAVERS	AI Director	ユーザーの感情値 プレイヤーの周囲の状況	敵キャラクターの出現位置・数	敵キャラクター
Assassin‘s Creed Origins	メタAI	プレイヤー・キャラクターの位置	発生させるエリア・オブジェクト・ キャラクター	ゲーム世界全体 （オブジェクト・キャラクター）

15.2　メタ AI と AI ディレクターの定義

AI ディレクターは『LEDT 4 DEAD』という明確な出発点が常に参照されながら各ゲームにおいて発展しているのに対して，メタAIは，岩谷徹 [93] [97]，遠藤 [98]の仕事以来，断続的に同じアイデアが再生産されながら発展している．その結果，メタAIは事例ごとに概念の揺れ幅が大きいが，メタAIを多様かつ抽象的な領域を指す言葉としている．ここで，これまでの歴史的経緯から，メタAIとAI ディレクターに再定義を与える．

メタAI：ユーザーを含めたゲーム全体の状況を認識し，目的に沿ったゲームシステム（ゲームの流れ，ゲームデザイン，レベルデザイン）の整合性，難易度を実現するために，ゲーム

全域（キャラクターやオブジェクトの配置，動作，イベント，地形）に命令を与え変化させる．

AI ディレクター：戦闘を始めとするプレイヤー・キャラクターの周囲の局所的かつ限定された時間の状況を認識し，敵・味方を含めたNPCに指示を与えることで，目的に沿ったユーザー体験を実現する．

15.3 定義の適用性

第15.2節で与えた定義から，デジタルゲームの歴史を調査することによって，これまで「メタAI」「ディレクターAI」という名称で呼ばれていなかったものの，本定義に当てはまる事例を見いだす．『The Elder Scrolls IV: Oblivion』（Bethesda Game Studio, 2006年），『The Elder Scrolls V: Skyrim』(Bethesda Game Studio, 2011年）では「Radiant AI」「Radiant Stories」が実装されている［107］．「Oblivion」から「Skyrim」へ向けて「Radiant AI」から「Radiant Stories」が発展した．「Radiant AI」は，プレイヤーの行動履歴からNPCの行動を変化させるAIである．「Radiant AI」によってNPCたちは一日のスケジュールを持ちながらも，プレイヤーの行動履歴と特性に応じてプレイヤーとのインタラクションを変化させることができる．「Radiant Stories」はプレイヤーの行動履歴からクエストを自動生成し，NPCにロールを割り当てる仕組みである．「Radiant AI」はAI ディレクターレベルの機能であり，「Radiant Stories」はメタAIレベルの機能である．

『Shadow of Mordor』（Monolith Productions, 2014年），『Shadow of War』（Monolith Productions, 2017年）は，一度倒した敵キャラクターが，プレイヤーとの戦闘の履歴を反映した経験と外見を持って復活し復讐しに来る．このシステムはNemesis Systemと呼ばれる［108］．プレイヤーの倒し方が外見に残され，さらにプレイヤーに突かれた弱点が強化される．これは，本論文の定義の「メタAI」に相当する．

このように，前節の定義からデジタルゲームを見ると，「メタAI」「AI ディレクター」に該当する事例の系譜を新しく見出すことができる．このように「メタAI」「AI ディレクター」の定義は，ゲーム開発の歴史の中に該当する流れを明らかにする．さらに，これからの応用の在り方を示唆する．

16. 付録 2:比較検証実験のアンケート文書

本章では，第10章の比較検証実験で用いたアンケート文書を示す

16.1 MCN-AI 連携モデルの解説文

本章では，アンケートに使用した文書を示す．第 16.1.1 項では MCN-AI 連携モデルの解説文を示す．第 16.1.2 項では MCS-AI 動的連携モデルの解説文を示す．第 16.1.3 項では回答シート A，第 16.1.4 項では回答シート B を示す．

16.1.1 MCN-AI 連携モデルの解説文

ゲームＡＩ解説シート

以下の３つの AI の解説を読んでください．（５分）

メタ AI

メタ AI は，ゲームに干渉するためにさまざまなインターフェースを持ち，ゲーム内のあらゆる要素をコントロールする能力を持つが，主にキャラクターをコントロールする．たとえば，キャラクターAI に指令を送ることで，ゲームの状況を動的に変化させる．敵キャラクターの生成する場所・タイミングを動的に決定する，などである．

メタ AI はゲーム全体を把握し，プレイヤー・キャラクター，敵キャラクター，ゲームギミックなど，すべての動きを把握しており，ゲームを自在に意図する方向へコントロールすることができる.特にプレイヤーの挙動から，プレイヤーの心理を推定し，プレイヤーの心理に影響するような変化をゲームにもたらす.多くの場合，メタ AI は敵キャラクターをコントロールすることで，プレイヤーにプレッシャーをかけたり，緊張を緩和させたりする．

メタ AI は，言うなればゲームの「神様」としての人工知能である．

キャラクターAI

「キャラクターAI」は，キャラクター全般のコントロールを担う AI である．主に意思決定，身体状態管理，モーションの決定の役割がある．ゲーム世界を認識し，抽象的な意思決定を行い，身体モーションを出力する．特にプレイヤーに対して，敵であれば，距離をつめ

て剣を振る，味方であれば，プレイヤーと周囲の敵の間に割って入って戦う，などをする．

キャラクターAI はキャラクターの身体の周囲を精緻に認識し，自らの行動を作り出す．このようなキャラクターAI は自律型人工知能と呼ばれる.つまり，ゲーム世界の中で，自分で感じて，自分で考え，自分で行動するキャラクターのことである．

言うなればキャラクターAI はキャラクターの頭脳に相当する．

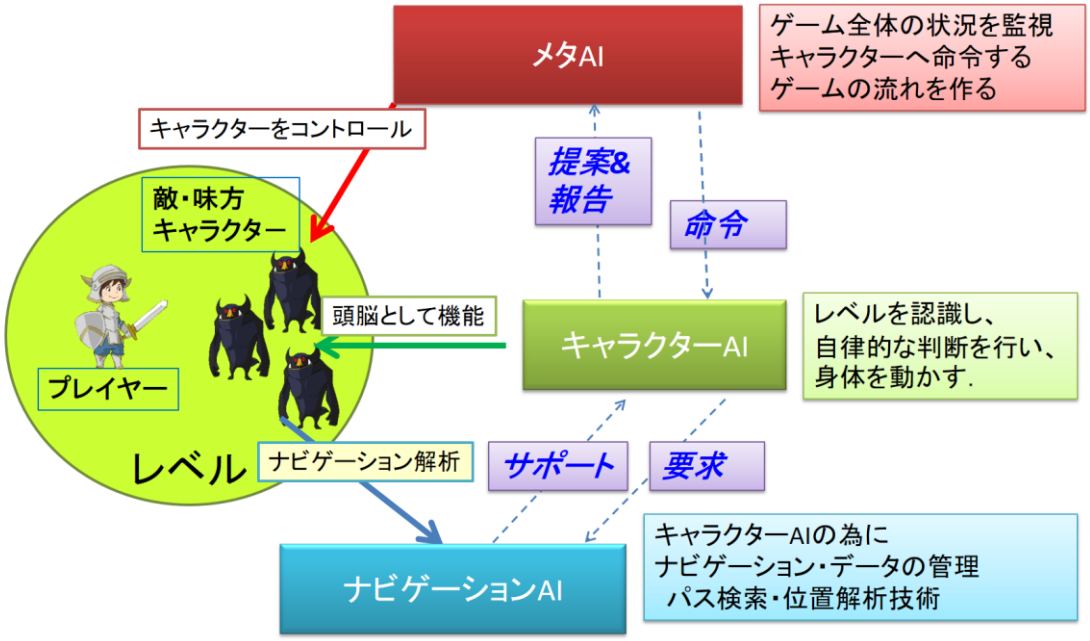
ナビゲーション AI

「ナビゲーション AI」は，地形や空間を認識し，それに沿った思考を行う．たとえば，「パス検索」や条件に合う目的地を自律的に見つける.ナビゲーション AI には，キャラクターAI からの要求を受けて, キャラクターAI が行動するために必要な空間的情報を提供する，という役割がある．

ナビゲーション AI のパス検索は，出発点からゴール地点までのパス検索を行うモジュールとして出発した.そこから，目的地を目的に応じて探索する位置解析技術などを含むようになった.この 2 つの機能があれば，キャラクターは自律的にマップ内を移動することができる．

MCN-AI 連携モデル（メタ AI-キャラクターAI-ナビゲーション AI 連携モデル）

この 3 つの AI は，ゲーム実行中に動的に連携する，ナビゲーション AI はゲーム実行中に動的に地形を解析し，キャラクターAI が必要とするパスや目的地などナビゲーション情報をリアルタイムに提供する．キャラクターAI は，自律型エージェントとしての人工知能であり，自分自身の感覚によって環境を認識し，意思決定し，身体を動かす.「メタ AI」は，キャラクターAI をコントロールして，ゲームが必要とする状況を動的に作り出す．



※レベルとは，ゲームの地形・キャラクター配置・仕掛けなどの総称

- メタ AI はキャラクターAI に命令を与えることができる.
- ナビゲーション AI はキャラクターAI から要求を受けて，必要な情報を提供することでサポートすることができる.
- キャラクターAI はメタ AI に「こういう行為をして良いか」という提案を質問する，或いは，状況を伝えることができる.複数の敵キャラクターが，或いは味方キャラクターがいる場合，あるキャラクターたちには攻撃などのアクションの許可を出し，あるキャラクターたちには許可を出さない，などによって全体のキャラクターの動作を調整することができる.

[ご質問があれば５分以内であればお答えいたします]

16.1.2 MCS-AI 動的連携モデルの解説文

MCS-AI 動的連携モデルの解説文は以下のようである.

ゲームＡＩ解説シート

以下の３つの AI の解説を読んでください.（５分）

メタ AI

メタ AI は，ゲームに干渉するためにさまざまなインターフェースを持ち，ゲーム内のあらゆる要素をコントロールする能力を持つ．たとえば，キャラクターAI に指令を送ることで，ゲームの状況を動的に変化させる．敵キャラクターの生成する場所・タイミングを動的に決定する，地形や天候を動的に変更するなど，などである．

メタ AI はゲーム全体を把握し，ゲーム全体に対する影響力を持つ．プレイヤー・キャラクター，敵キャラクター，ゲームギミックなど，すべての動きを把握しており，ゲームを自在に意図する方向へコントロールすることができる．特にプレイヤーの挙動から，プレイヤーの心理を推定し，プレイヤーの心理に影響するような変化をゲームにもたらす．多くの場合，メタ AI は敵キャラクターをコントロールすることで，プレイヤーにプレッシャーをかけたり，緊張を緩和させたりする．

またスパーシャル AI からの空間の情報を取得し，その情報をもとにゲーム世界の動的動向を読み取り，ゲーム世界の各部分を変化させる．スパーシャル AI から提供される動的・静的なゲーム状態・地形の情報により，メタ AI は，キャラクター以外に対しても，より詳細にゲーム世界に影響を与えることができる．

メタ AI は，言うなればゲームの「神様」としての人工知能である．

キャラクターAI

「キャラクターAI」は，キャラクター全般のコントロールを担う AI である．主に意思決定，身体管理，モーションの決定の役割がある．ゲーム世界を認識し，抽象的な意思決定を行い，身体モーションを生成する．特にプレイヤーに対して，敵であれば，距離をつめて剣を振るとか，味方であれば，プレイヤーと周囲の敵の間に割って入って戦うなどをする．

キャラクターAI はキャラクターの身体の周囲を精緻に認識し，自らの行動を作り出す．このようなキャラクターAI は自律型人工知能と呼ばれる．つまり，ゲーム世界の中で，自分で感じて，自分で考え，自分で行動するキャラクターのことである．

言うなればキャラクターAI はキャラクターの頭脳に相当する．

またスパーシャル AI からの空間の情報を取得し，その情報をもとにゲーム世界の動的動向を読み取り，ゲーム世界の変化に対して，戦略的・戦術的な行動を行うことができる．スパーシャル AI から提供される動的・静的なゲーム状態・地形の情報により，キャラクターAI は，より詳細にゲーム世界の状態・地形の変化に沿った行動をとることができる．

スパーシャル AI

「スパーシャル AI」（スパーシャル＝空間的）は，地形や空間を認識し，それに沿った思

考を行う．たとえば，「パス検索」や条件に合う目的地を見つける．スパーシャル AI には，キャラクターAI が行動するために必要な空間的情報を要求に応じて提供する，また自律的に提供する．

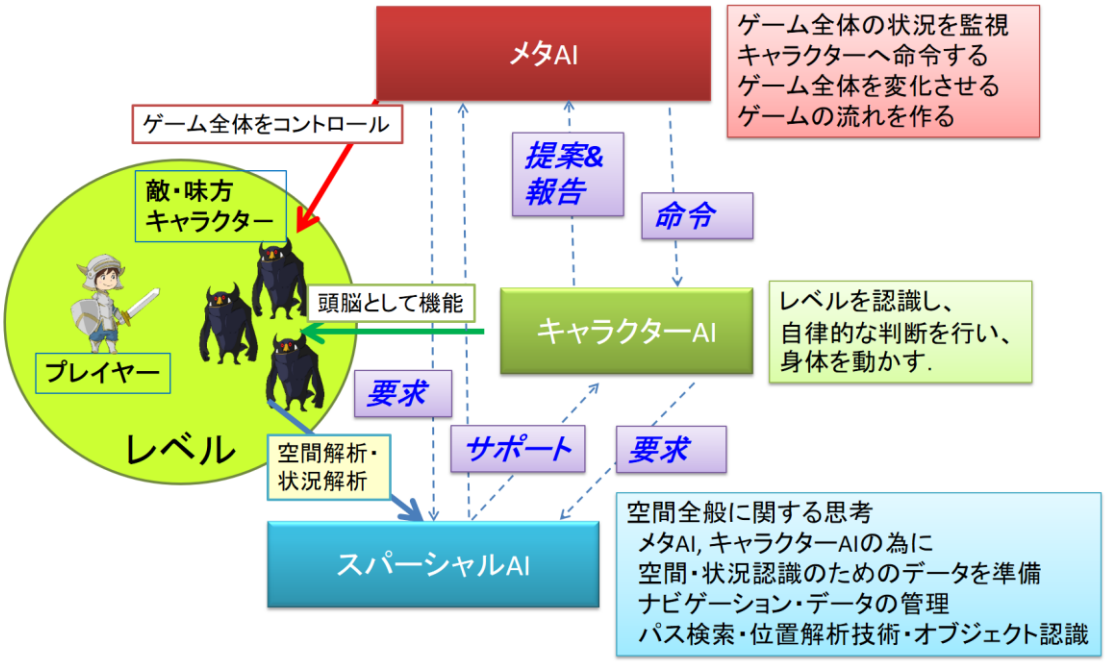
スパーシャル AI は，出発点からゴール地点までのパス検索を行うモジュールとして出発した．そこから，目的地を探索する位置解析技術などを含むようになり，さらに，ゲームの空間的状態全般を把握する人工知能となった．敵がいる場所を温度分布のように解析する情勢解析技術や，勢力均衡線を見出す機能もある．また，メタ AI，キャラクターAI の目的に沿った地形解析なども行う．たとえば，待ち伏せをする位置，戦術上重要なポイントなど，意味を持つ場所をゲーム内でリアルタイムに見つけ出す．

スパーシャル AI は自律的に，メタ AI，キャラクターAI を，より深くゲームの状況・地形に関わらせる役割を持つ．

MCS-AI 動的連携モデル（メタ AI-キャラクターAI-スパーシャル AI 動的連携モデル）

この 3 つの AI は，ゲーム実行中に動的に連携する．スパーシャル AI は自律的に，静的な地形データから抽出したデータを用いつつ，ゲーム実行中に動的に地形を解析し，メタ AI，キャラクターAI が必要とする空間的情報をリアルタイムに両者に提供する．キャラクターAI は，自律型エージェントとしての人工知能であり，自分自身の感覚によって環境を認識し，意思決定し，身体を動かすが，さらにスパーシャル AI からの情報によって，ゲームの状況・地形に合った行動を取ることが可能になる．「メタ AI」は，キャラクターAI を始め，ゲームの全要素（天候，地形，台詞など）をコントロールして，ゲームが必要とする状況を動的に作り出す．キャラクターからの報告や自分のセンサーからの情報のみならず，ゲーム全域に関わるスパーシャル AI からの状況・地形の動的変化・静的特徴を提供されることで，より深く細かくゲーム状況を変化させることができるようになる．

MCN-AI 連携モデルとの明確な違いは，それぞれの AI が自律的に世界に対して働きかけ，連携してゲームを積極的に変更する点である．特に，スパーシャル AI はナビゲーション AI と違い，メタ AI，キャラクターAI の要求を待たずに世界を解析し続けて，その場，との時に必要な地形・状況の情報を両者に伝える．またメタ AI は，スパーシャル AI を通じて，さまざまな世界の部分に影響を与えることができる．



※レベルとは，ゲームの地形・キャラクター配置・仕掛けなどの総称

- メタ AI はキャラクターAI に「命令」を与えることができる．
- メタ AI はスパーシャル AI に対してゲーム全体を把握するために必要な情報を「要求」し，ゲーム全体をコントロールする．キャラクターに「命令」する場合もあれば，ゲームのさまざまな要素（岩などのオブジェクトや天候，河の流れなど）を運動させる場合もある．メタ AI は，ユーザーの一連の動きやゲーム全体を認識し，ゲーム全体を動的に変化し続けることができる．
- キャラクターAI はメタ AI に「こういう行為をして良いか」という提案を尋ねる，或いは，状況を伝えることができる．複数の敵キャラクターが，或いは味方キャラクターがいるが場合，あるキャラクターたちには攻撃などのアクションの許可を出し，あるキャラクターたちには許可を出さない，などによって全体のキャラクターの動作を調整することができる．
- スパーシャル AI はキャラクターAI，メタ AI から「要求」を受けて，必要な情報を提供することで「サポート」することができる．
- スパーシャル AI はメタ AI，キャラクターAI からの要求がなくても，常に地形や空間を解析し，メタ AI，キャラクターAI に自分から情報を提供することで，メタ AI，キャラクターAI の地形・空間認識をアップデートする．そのため，メタ AI，キャラクターAI は地形・空間の詳細な情報を用いて，メタ AI はゲーム全体を変化させ，キャラクターAI はより巧みに地形・空間を用いた運動を構成することができる．

[ご質問があれば５分以内であればお答えいたします]

16.1.3 回答シート A

回答シート（20 分以内）

以下の情報は匿名化した上で，統計的に扱います．誰が何を書いたかわからない状態にします．試験ではないので，思いつくままに書いてください．

（1） ゲーム開発経験はありますか？

a. 1 年以内　　b. 1 年以上～3 年以内　　c. 3 年以上～

回答欄（　　）

（2） 職種はなんですか？　（複数回答可）

a. エンジニア　　b. プランナー　　c.アーティスト　　d.その他（）

回答欄（　　）

（3） どんなゲーム開発経験がありますか？（複数回答可）

a. コンシューマーゲーム　b. ＰＣゲーム　c. 携帯電話ゲーム　d.その他（）

回答欄（　　）

（4）ゲーム開発でＡＩ技術を用いたことがありますか？

a. ある　　b. なし　c. わからない

回答欄（　　）

[次ページへ]

以下に，AI を使ったパックマンのゲームを変化させるゲームのアイデアを 1 つ以上，5 つまで書いてください．それぞれ 500 文字以内とします．体系的に書く必要はありません．

[パックマン]

パックマン…プレイヤーが動かすキャラクター

敵キャラクター … 中央の巣から出現する四種類のモンスター（赤，シアン，ピンク，オレンジ色で区別される）．パックマンを追いかける．接触すると，ゲームオーバー．

ドット…マップの通路に沿って置かれている．

パワーエサ…プレイヤーが食べると，立場が逆転して，パックマンが敵キャラクターを捕食することができる．敵キャラクターは食べられると消滅し時間を置いて巣から再発生する．

クリア条件…パックマンがマップ上のすべてドットとパワーエサを食べるとクリアとなる．

---

(1)

---

(2)

---

(3)

---

(4)

---

(5)

※５個以上でも，あれば書いてください.

※20 分以内でも書き終わったらお声がけください.

[文書おわり]

16.1.4 回答シート B

回答シート（20 分以内）

以下の情報は匿名化した上で，統計的に扱います．誰が何を書いたかわからない状態にします．試験ではないので，思いつくままに書いてください．

以下に，読まれた解説シートを踏まえて，AI を使ったパックマンのゲームを変化させるゲームのアイデアを 1 つ以上，5 つまで書いてください．それぞれ 500 文字以内とします．体系的に書く必要はありません．

回答シート A で記入した事柄と，まったく同じアイデアは記入しないでください．
ほんの少しでも違っていれば問題ありません．

[パックマン]
パックマン…プレイヤーが動かすキャラクター
敵キャラクター … 中央の巣から出現する四種類のモンスター（赤，シアン，ピンク，オレンジ色で区別される）．パックマンを追いかける．接触すると，ゲームオーバー．
ドット…マップの通路に沿って置かれている．
パワーエサ…プレイヤーが食べると，立場が逆転して，パックマンが敵キャラクターを捕食することができる．敵キャラクターは食べられると消滅し時間を置いて巣から再発生する．
クリア条件…パックマンがマップ上のすべてドットとパワーエサを食べるとクリアとなる．

---

(6)

(7)

---

(8)

---

(9)

---

(10)

※５個以上でも，あれば書いてください．
※20 分以内でも書き終わったらお声がけください．

[文書おわり]

17. 参考文献

[1] W. Wright, “AI : A Design Perspective,” AIIDE(AI and Digital Entertainment 2005, 2005.

[2] 三宅陽一郎, 水野勇太 , 里井大輝, “「メタ AI」と「AI Director」の歴史的発展,” デジタルゲーム学研究, 第 13 巻, 第 2 号, 日本デジタルゲーム学会, 2020.

[3] 三宅陽一郎, “ディジタルゲームにおける人工知能技術の応用の現在,” 人工知能, Vol.30, No.1, pp.45-64, 2015.

[4] “KONAMI、スクエニ、セガ、バンナム、コーエーの大手 5 社がゲーム開発現場の未来を再び討議,” 16 9 2008. [オンライン]. Available: https://game.watch.impress.co.jp/docs/20080916/cedec_dev.htm. [アクセス日: 17 2 2021].

[5] D. Pottinger, “Coordinated Unit Movement,” UBM Technology Group, 1999.

[6] G. Snook, “ナビゲーション メッシュによる 3D 移動とパス発見の単純化,” Game Programming Gems, Vol.1, No.3.6, 2001, pp. 279-294.

[7] P. Xu , J. Feng, “Proceedings of International Conference on Computer Science and Network Technology,” Proceedings of International Conference on Computer Science and Network Technology, 2011.

[8] 川中真耶, “ぷよぷよ AI 人類打倒に向けて,” ゲームプログラミングワークショップ, 2015.

[9] E. L.-C. Law , e. al., “Understanding, scoping and defining user experience: a survey approach,” CHI '09: Proceedings of the SIGCHI Conference on Human Factors in Computing System, 2009.

[10] P. ". Prasertvithyakarn, “キャラクター・エクスペリエンス(CX)』デザイン入門 〜FINAL FANTASY XV の仲間達に生命を与えるデザイン論とその応用〜,” CEDEC 2018, https://cedil.cesa.or.jp/cedil_sessions/view/1878, 2018.

[11] D. Isla, R. C. Burke, M. Downie , B. Blumberg, “A Layered Brain Architecture for Synthetic Creatures,” Proceedings of IJCAI, 2001.

[12] S. Rabin, AI Game Programming Wisdom, Charles River Media, 2002.

[13] S. Rabin, AI Game Programming Wisdom 2, Charles River Media, 2003.

[14] S. Rabin, AI Game Programming Wisdom 3, Charles River Media, 2006.

[15] S. Rabin, AI Game Programming Wisdom 4, Charles River Media, 2008.

[16] S. Rabin, Game AI PRO, Charles River Media, 2013.

[17] S. Rabin, Game AI PRO 2, Charles River Media , 2015.

[18] S. Rabin, Game AI PRO 3, Charles River Media, 2017.

[19] 三宅陽一郎,“ディジタルゲームにおける人工知能技術の応用,”人工知能学会誌 Vol. 23, No. 1, 2008.

[20] 三宅陽一郎,“デジタルゲーム AI,”著: デジタルゲームの教科書, 第 23 章, ソフトバンク パブリッシング, 2010, pp. 214-218.

[21] A. Barbuta,“'Horizon Zero Dawn': A QA Open World Case Study,”GDC 2018, https://www.gdcvault.com/play/1025326/-Horizon-Zero-Dawn-A, 2018.

[22] J. Paredes , P. Jones,“Automated Testing: Using AI Controlled Players to Test 'The Division',”GDC 2019, https://www.gdcvault.com/play/1026382/Automated-Testing-Using-AI-Controlled, 2019.

[23] J. Gillberg,“AI for Testing: The Development of Bots that Play 'Battlefield V',” GDC 2019, https://www.gdcvault.com/play/1025905/AI-for-Testing-The-Development, 2019.

[24] 三宅陽一郎,“メタ AI――ユーザーを楽しませるために,”著: ゲーム AI 技術入門, 技術評論社, 第 9 章, 2019.

[25] 峯恒憲,“仲介（メディエーション）,”著: 知識ベース, 7 群－7 編－3 章 3-12 , 電子情報通信学会, 2010, pp. 20-21.

[26] 三宅陽一郎, 横山貴規 , 北崎雄之,“エージェント・アーキテクチャに基づくキャラクターAI の実装,”第 4 回デジタルコンテンツシンポジウム予稿集 2-2, 2008.

[27] M. Booth,“Replayable Cooperative Game Design: Left 4 Dead,”GDC （2009） http://www.valvesoftware.com/company/publications.html, 2009.

[28] T. Graepel , R. H. J. Gold,“Learning to Fight," Proceedings of the International Conference on Computer Games,”Artificial Intelligence, Design and Education, https://www.microsoft.com/en-us/research/publication/learning-to-fig, 2004.

[29] M. Robbins,“Neural Networks in Supreme Commander 2,”GDC 2012, 2012.

[30] M. Robbins,“Using Neural Networks to Control Agent Threat Response,”著: Game AI PRO, Charles River Media, chapter 30, http://www.gameaipro.com/GameAIPro/GameAIPro_Chapter30_Using_Neural_Networks_to_Control_Agent_Threat_Response.pdf, 2013.

[31] S. Grand, D. Cliff , A. Malhotra,“Creatures: Artificial Life Autonomous Software

Agents for Home Entertainment,” Autonomous Agents and Multi-Agent Systems, Volume 1, Issue 1, pp 39–57, 1998.

[32] 森川幸人, “テレビゲームへの人工知能技術の利用,” 人工知能学会、Vol.14 No.2, http://www.ai-gakkai.or.jp/whatsai/PDF/article-iapp-7.pdf, 1998.

[33] R. Evans, “Varieties of Learning,” 著: AI Programming Wisdom, 11.2, Charles River Media, 2002.

[34] J. Chung , S. Rho, “Reinforcement Learning in Action: Creating Arena Battle AI for 'Blade & Soul',” GDC 2019, 2019.

[35] I. Oh , et al., “Creating Pro-Level AI for Real-Time Fighting Game with Deep Reinforcement Learning,” https://arxiv.org/abs/1904.03821, 2019.

[36] R. Herbrich, T. Graepel , J.Q.Candela, “Halo, Xbox Live The Magic of Research in Microsoft Products,” Microsoft Research, https://www.microsoft.com/en-us/research/project/video-games-and-artificial-intelligence/, 2008.

[37] 甲野佑, 田中一樹, 岡田健 , 奥村エルネスト純, “逆転オセロニア”における深層強化学習応用,” デジタルプラクティス』情報処理学会、Vol.10 No.2、https://www.ipsj.or.jp/dp/contents/publication/38/S1002-S05.html, 2019.

[38] K. O. Stanley, B. D. Bryant , R. Miikkulainen, “Real-time neuroevolution in the NERO video game,” IEEE Transactions on Evolutionary Computation, vol. 9, no. 6, pp. 653-668, 2005.

[39] E. J. Hastings, R. K. Guha , K. O. Stanley, “Evolving Content in the Galactic Arms Race Video Game,” Proceedings of the IEEE Symposium on CIG09, 2009.

[40] R. Coulom, “Monte-Carlo Tree Search in Crazy Stone,” 12th Game Programming Workshop, 2007.

[41] D. Silver, J. Schrittwieser, K. Simonyan , et al., “Mastering the game of Go without human knowledge,” Nature 550, 354–359, 2017.

[42] 大槻知史, 最強囲碁 AI アルファ碁 解体新書 深層学習、モンテカルロ木探索、強化学習から見たその仕組み, 翔泳社, 2017.

[43] P. Andruszkiewicz, “Optimizing MCTS Performance for Tactical Coordination in Total War: Attila,” nucl.AI conference, 2015.

[44] T. Thompson, “Make Peace, Not War | The AI of Total War (Part 4), Gamasutra,” 2018. [オンライン]. Available: https://www.gamasutra.com/blogs/TommyThompson/20180226/314400/Make_Peace_Not_War. [アクセス日: 22 2 2020].

[45] G. Mountain, “Tactical Planning and Real-time MCTS in Fable Legends,” nucl.ai, 2015.

[46] OpenAI Five. [オンライン]. Available: https://openai.com/blog/openai-five/. [アクセス日: 17 2 2021].

[47] DeepMind, “The AlphaStar team, "AlphaStar: Mastering the Real-Time Strategy Game StarCraft II,",” 2019. [オンライン]. Available: https://deepmind.com/blog/article/alphastar-mastering-real-time-strategy-game-starcraft-ii. [アクセス日: 17 2 2021].

[48] J. McDonald, “Robocalypse Now: Using Deep Learning to Combat Cheating in 'Counter-Strike: Global Offensive',” GDC 2018, 2018.

[49] J. Hwang, “Beating Wallhacks using Deep Learning with Limited Resources,” GDC 2019, 2019.

[50] F. Nadeau, “Tools Tutorial Day: 3D Asset Recognition with Deep Learning,” GDC 2019, 2019.

[51] 泊久信, “ニューラルネットワークを用いた AI の格闘ゲームへの組み込み,” Game Creators Conference 2019, 2019.

[52] A. Andelkovic , S. Freyr, “How King Uses AI in 'Candy Crush',” GDC 2016, 2016.

[53] 森田和則, “『ベヨネッタ 2』におけるゲーム品質を上げる為の自動化――オートプレイと継続的なパフォーマンス計測」,” CEDEC 2016, 2016.

[54] M. Booth, “The AI Systems of Left 4 Dead,” AIIDE（2009） http://www.valvesoftware.com/company/publications.html.

[55] D. Holden, T. Komura , J. Saito, “Phase-functioned neural networks for character control,” ACM Transactions on Graphics 36(4):1-13, 2017.

[56] S. Russell , P. Norvig, “Intelligent Agents,” 著: Artificial Intelligence: A Modern Approach, 3rd Edition, chapter 2, Pearson, 2016.

[57] 三宅陽一郎, “エージェント・アーキテクチャーから作るキャラクターAI,” CEDEC 2007, 2007.

[58] J. Griesemer, “The Illusion of Intelligence: The Integration of AI and Level Design in Halo,” GDC 2002 Proceedings Archive, 2002.

[59] 三宅陽一郎, “デジタルゲームのための人工知能の基礎理論,” 日本バーチャルリアリティ学会誌, Vol.18, No.3, pp.28-33, 2013.

[60] 三宅陽一郎, “ディジタルゲームにおける人工知能エンジン,” 映像情報メディア学会誌, Vol.68, No.2, pp.125-130, 2014.

[61] MIT Synthetic Character Group, 2003. [オンライン]. Available: http://characters.media.mit.edu/. [アクセス日: 17 2 2021].

[62] R. Burke, D. Isla, M. Downie, Y. Ivanov , B. Blumberg, “Creature Smarts: The Art and Architecture of a Virtual Brain,” Proceedings of the Game Developers Conference, 2001.

[63] J. Orkin, “3 States & a Plan: The AI of F.E.A.R.,” Game Developer’s Conf. Proc., http://web.media.mit.edu/~jorkin/, 2006.

[64] J. Orkin, “Agent architecture considerations for realtime planning in games,” AIIDE 2005, http://web.media.mit.edu/~jorkin/, 2005.

[65] R. Straatman, T. Verweij , A. Champandard, “Killzone 2 Multiplayer Bots,” Paris Game/AI Conference, 2009.

[66] H. P. Nii, “Blackboard Application Systems, Blackboard Systems and a Knowledge Engineering Perspective,” AI Magazine, Vol.7 No.3, pp82-107, 1986.

[67] H. P. Nii, “The Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures,” AI Magazine, Vol.7 No.2, pp38-53, 1986.

[68] 石田亨, 桑原和宏 , 片桐恭弘, “黒板モデル,” 著: 分散人工知能, コロナ社, 1996, p. 第 4 章.

[69] D. Isla , B. Blumberg, “Blackboard Architectures,” 著: AI Game Programming Wisdom, 2002, pp. Vol.1, No.7.1, pp.333-344.

[70] 三宅陽一郎, “次世代デジタルゲームにおける人工知能の研究課題について,” ゲームプログラミングワークショップ 2012 論文集, Vol. 2012, No. 6, pp.108-113, 2012.

[71] I. Millington, “World Representations,” 著: AI for Games, Third Edition, 4.4. , CRC Press, 2019.

[72] B. Schwab, “Location based information systems,” 著: AI Game Engine Programming, Charles River Media; 2nd Edition, 2008.

[73] M. Buckland, “Practical Path Planning,,” 著: Programming Game AI by Example, Wordware, 2004.

[74] 岡村信幸, “ARMORED CORE V のパス検索,” CEDEC 2011, http://cedil.cesa.or.jp/session/detail/593, 2011.

[75] 岡村信幸, “ARMORED CORE V の階層型ゴール指向プランニングと機体制御,” CEDEC 2011, http://cedil.cesa.or.jp/session/detail/591, 2011.

[76] F. Gravot, T. Yokoyama , Y. Miyake, “パス移動（動画）,” 2012. [オンライン].

Available: https://www.youtube.com/watch?v=9_vdpndn4jI.

[77] F. Gravot, T. Yokoyama , Y. Miyake, “落下移動（動画）(2012）,” 2012. [オンライン]. Available: https://www.youtube.com/watch?v=o3ihN2kSzMs. [アクセス日: 17 2 2021].

[78] 米田聡, “［SQEXOC 2012］FFXIV で使われている AI 技術〜敵 NPC はどうやって経路を探索しているのか？,” 2012. [オンライン]. Available: http://www.4gamer.net/games/032/G003263/20121205079/. [アクセス日: 17 2 2021].

[79] R. Straatman, A. Beij , W. v. d. Sterren, “Killzone's AI : Dynamic Procedural Combat Tactics,” 2005. [オンライン]. Available: http://www.cgfai.com/docs/straatman_remco_killzone_ai.pdf. [アクセス日: 22 2 2020].

[80] 三宅陽一郎, “クロムハウンズにおける人工知能開発から見るゲーム AI の展望,” CEDEC 2006, 2006.

[81] M. Walsh, “Modeling AI Perception and Awareness in Splinter Cell: Blacklist,” GDC 2014, 2014.

[82] D. Isla, “Managing Complexity in the Halo2 AI,” GDC 2005, 2005.

[83] D. Isla, “Dude, Where’s my Warthog? From Pathfinding to General Spatial Competence,” AIIDE 2005, 2005.

[84] 並木幸介, “ぽかぽかアイルー村における，アフォーダンス指向の AI 事例，AI に多様な振る舞いをさせる手法,” CEDEC 2011, https://cedil.cesa.or.jp/cedil_sessions/view/697, 2011 年.

[85] J. Abercrombie, “Bringing BioShock Infinite's Elizabeth to Life: An AI Development Postmortem,” GDC 2014, 2014.

[86] D. Isla , P. Gorniak, “Beyond Behavior: An Introduction to Knowledge Representation,” AI Summit, GDC 2009, 2009.

[87] S. Rabin, “Agent Awareness and Knowledge Representation,” 著: Game AI Pro, CRC Press, 2013.

[88] 三宅陽一郎, “第 5 夜　メルロ＝ポンティと知覚論,” 著: 人工知能のための哲学塾, BNN 新社, 2016.

[89] 今村紀之, “AI とアニメーション,” 著: FINAL FANTASY XV の人工知能, CHAPTER 4, ボーンデジタル, 2019, pp. 61-66.

[90] R. A. Brooks, “A Robust Layered Control System for a Mobile Robot,” MIT AI Lab

Memo No. 864, 1985.

[91] 三宅陽一郎, “キャラクターの身体システム,” 著: ゲーム AI 技術入門, 技術評論社, 10.2 , 2019.

[92] T. Mathews, “Making "Big Data" Work for 'Halo': A Case Study,” GDC 2017, 2016.

[93] 岩谷徹, 高橋ミレイ , 三宅陽一郎, “ゲーム AI の原点『パックマン』はいかにして生み出されたのか？：岩谷 徹インタビュー,” 人工知能,Vol.34, No.1 pp.86-99., 2019.

[94] D. Brewer, “AI Postmortems: Assassin’s Creed III, XCOM: Enemy Unknown, and Warframe,” GDC 2013, 2013.

[95] D. Brewer, “The living AI in warframe’s procedural space ships,” Game AI Conference, 2014.

[96] 長谷洋平, “LOST REAVERS における AI Director の試み,” CEDEC 2016, 2016.

[97] “セルフゲームコントロールシステム,「パックマン」岩谷氏,「Rez」水口氏ら 4 人のクリエイターが語る世界のゲームデザイン論「International Game Designers Panel」(講演記事),” 2005. [オンライン]. Available: http://game.watch.impress.co.jp/docs/20050312/gdc_int.htm. [アクセス日: 16 2 2021].

[98] 遠藤雅伸, “ 糸井重里の電視遊戯大展覧会」『遠藤雅伸ゼビウスセミナー』,” フジテレビ, 1987.

[99] 上段達弘, 下川和也, 高橋光佑 , 並木幸介, “FINAL FANTASY XV におけるレベルメタ AI 制御システム,” CEDEC 2016, https://cedil.cesa.or.jp/cedil_sessions/view/1544, 2016.

[100] P. Prasertvithyakarn, “Walk Tall, My Friends: Giving Life to AI-Buddies in 'Final Fantasy XV',” GDC 2018, 2018.

[101] 三宅陽一郎 , et. Al, “大規模ゲームにおける人工知能. ―ファイナルファンタジー XV の実例をもとに―,” 人工知能学会誌, Vol.32, No.2, pp.197-213, 2017.

[102] ウィル・ライト , 多摩豊, “「シムシティの仕組み」 ウィル・ライトが明かすシムシティーのすべて（コンプコレクション),” 角川書店 .

[103] F. D. Kenneth, “Simulation and Modeling: Under the hood of the Sims, Northwerstern University, Lecture note,” 2002. [オンライン]. Available: http://users.cs.northwestern.edu/~forbus/c95-gd/lectures/The_Sims_Under_the_Hood_files/v3_document.htm. [アクセス日: 24

2 2020].

[104] C. Hecker , et al.,“Real-time motion retargeting to highly varied user-created morphologies,”Proc. ACM SIGGRAPH 2008, Vol. 27, Issue 3, 2008.

[105] C. Hecker,“Spore Behavior Tree Docs. ,”2009. [オンライン]. Available: http://chrishecker.com/My_liner_notes_for_spore/Spore_Behavior_Tree_Docs. [アクセス日: 17 2 2021].

[106] J. Varnie,“Far Cry’s AI: A Manifesto for Systemic Gameplay,”Game / AI Conference , 2014.

[107] B. Nesmith,“Radiant Story, Game Design Expo, Vancouver Film School,”2012. [オンライン]. Available: https://www.youtube.com/watch?v=Ou6SB8dWKjw.

[108] B. Francis,“Upgrading the Nemesis system for Middle-earth: Shadow of War,” 2017. [オンライン]. Available: https://www.gamasutra.com/view/news/305601/Upgrading_the_Nemesis_system_for_Middleearth_Shadow_of_War.php. [アクセス日: 17 2 2021].

[109] B. Hayles,“Case-based Reasoning for Player Behavior Cloning in Killer Instinct,” nucl.ai Conference, 2015.

[110] 三宅陽一郎,“はじめてのゲーム AI ～意思を持つかのように行動するしくみ～,”著: WEB＋DB PRESS, Vol. 68,, 技術評論社, 2012, pp. 87-120.

[111] J. Gregory, Game Engine Architecture, A K Peters/CRC Press, 2009.

[112] J. Gregory,“State-Based Scripting in Uncharted 2: Among Thieves,,”GDC（2009）, http://www.gdcvault.com/play/1730/State-Based-Scripting-in-UNCHARTED, 2009.

[113] J. Orkin,“Applying Goal-Oriented planning for Games,”著: AI Game Programming Wisdom, Vol. 2, No. 3.4, Charles River Media, 2003, pp. 217-227.

[114] D. Nau , et al.,“SHOP2: An HTN Planning System,”Journal of Artificial Intelligence Research, Vol.20, pp.379-404, 2003.

[115] R. Straatman, T. Verweij, A. Champandard, R. Morcus , H. Kleve,“Hierarchical AI for Multiplayer Bots in Killzone 3,”著: GAME AI PRO, Chapter 29, Charles River Media, 2013, pp. 377-390.

[116] J. P.Bigus, J. Bigus , 井田昌之訳,“Java による知的エージェント入門,”SB クリエイティブ, 2002, p. 82.

[117] R. Evans,“Modeling Individual Personalities in The Sims 3,”GDC 2010, 2010.

[118] M. Buckland,“Chapter 2: State Driven Agents,”著: Programming Game AI by

Example, Jones & Bartlett Learning, 2004.

[119] F. Sanglard, “5.18 Artificial Intelligence,” 著: GAME ENGINE BLACK BOOK DOOM, CreateSpace Independent Publishing Platform, 1993, 2018.

[120] R. Pillosu, “Coordinating Agents with Behavior Trees, Paris Game Conference,” 2009.

[121] D. Isla, “GDC 2005 Proceeding: Handling Complexity in the Halo 2 AI,” 11 3 2005. [オンライン]. Available: http://www.gamasutra.com/view/feature/130663/gdc_2005_proceeding_handling_.php. [アクセス日: 17 2 2021].

[122] S. Ocio, “Adapting AI behaviors to players in driver san francisco hinted-execution behavior trees,” AIIDE'12: Proceedings of the Eighth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 2012.

[123] Yager, “Behavior Tree Architecture of Spec Ops: The Line,” Vienna Game/AI Conference 2012 Behavior Workshop, 2012.

[124] M. Buckland, “Hierarchical Goal Based Agents,” 著: Programming Game AI by Example, Chapter 9, Jones & Bartlett Learning, 2004.

[125] S. Russell , P. Norvig, “Classical Planning,” 著: Artificial Intelligence: A Modern Approach 3rd Edition, chapter 10, 2016.

[126] 三宅陽一郎, “オンラインゲームにおける人工知能・プロシージャル技術の応用,,” 知能と情報, 日本知能情報ファジィ学会誌 Vol. 22, No.6, pp.745-756, 2010.

[127] 安藤毅, “「サカつく」のサッカー試合ＡＩシステム,” CEDEC 2010, 2010.

[128] 安藤毅, “リアルタイムサッカーシミュレーションゲームの AI システムの一手法について,” 人工知能 32(2), 2017-03-01, 2017.

[129] S. Russell , P. Norvig, “Planning and Acting in the Real World,” 著: Artificial Intelligence: A Modern Approach, 3rd Edition, chapter 11, Pearson, 2016.

[130] P. Dunstan , D. Rechner, “Blending Autonomy and Control: Creating NPCs for 'Tom Clancy's The Division',” GDC 2016 AI Summit, 2016.

[131] D. C. Pottinger, “Terrain Analysis in Realtime Strategy Games,” GDC 2000, 2000.

[132] M. Jack , M. Vehkala, “Spaces in the Sandbox: Tactical Awareness in Open World Games,” GDC 2013, https://www.gdcvault.com/play/1018136/Spaces-in-the-Sandbox-Tactical, 2013.

[133] M. Jack , M. Vehkala., “Spaces in the Sandbox: Tactical Awareness in Open World Games,” GDC 2014, 2014.

[134] C. Pedica , H. Vilhjálmsson, “Social perception and steering for online avatars,” international Workshop on Intelligent Virtual Agents, Springer, Berlin, Heidelberg, 2008.

[135] C. Pedica , H. H. Vilhj ´ almsson, “Spontaneous Avatar Behavior for Human Territoriality,” Applied Artificial Intelligence, Volume 24 Issue 6, 2010.

[136] 長谷川誠, “『LEFT ALIVE』における地形表現とナビゲーション AI,” CEDEC 2019, 『LEFT ALIVE』における地形表現とナビゲーション AI, 2019.

[137] Y. Miyake, “Researching AI Technologies Created in Japan in the 1980s and 1990s (accepted),” Replaying Japan, Vol.2, 2020.

[138] S. Rabin, “: #define GAME_AI,” GDC 2009, 2009.

[139] R. Thawonmas , H. Matsumoto, “Automatic Controller of Ms. Pac-Man and Its Performance:Winner of the IEEE CEC 2009 Software Agent Ms. Pac-Man Competition,” Asia Simulation Conference 2009 (JSST 2009), 2009.

[140] 池畑望 , 伊藤毅志, “Ms. Pac-Man におけるモンテカルロ木探索,” 情報処理学会誌、Vol.52, No.12, 3817-3827, 2011.

[141] “Competition, Ms. Pac-Man Vs. Ghost Team,” CIG 2018, 2018.

[142] 岩谷徹, “パックマンのゲーム学入門,” エンターブレイン, 2005, p. 101.

[143] 樋口耕一, “KH Coder,” https://khcoder.net/. [オンライン]. [アクセス日: 17 2 2021].

[144] 樋. 耕一, 社会調査のための計量テキスト分析—内容分析の継承と発展を目指して【第 2 版】, ナカニシヤ出版, 2020.

[145] 白神陽嗣 , 下川和也, “意思決定ツール,” 著: FINAL FANTASY XV の人工知能, CHAPTER 2. , ボーンデジタル, 2019.

[146] 白神陽嗣, 並木幸介, 横山貴規 , 三宅陽一郎, “FINAL FANTASY XV -EPISODE DUSCAE-におけるキャラクターAI の意思決定システム,” CEDEC 2015, https://cedil.cesa.or.jp/cedil_sessions/view/1437, 2015.

[147] 三宅陽一郎 , et al, “AI 最前線の現場から【スクウェア・エニックス】ゲーム・キャラクターはどのように意志決定するのか, Think IT,” 2016. [オンライン]. Available: https://thinkit.co.jp/article/10012. [アクセス日: 17 2 2021].

[148] P. Prasertvithyakarn , 高橋光佑, “AI モード,” 著: FINAL FANTASY XV の人工知能, CHAPTER:12, ボーンデジタル, 2019.

[149] 三宅陽一郎, “大規模 デジタルゲームにおける人工知能の一般的体系と実装,” 人工知能学会論文誌 35 巻 2 号 p. B-J64_1-16, 2020.

[150] Y. Miyake, Y. Shirakami, K. Shimokawa, K. Namiki, T. Komatsu, J. Tatsuhiro, P. Prasertvithyakarn , T. Yokoyama, “Chapter 11: A Character Decision-Making System for FINAL FANTASY XV by Combining Behavior Trees and State Machines,,” A K Peters/CRC Press, 2017.

[151] N. Imamura , et al. , “Final Fantasy XV: Pulse and Traction of Characters,” Proceeding siggraph '16 acm siggraph 2016 Talks Article No. 4, 2016.

[152] 三宅陽一郎 , et al., “AI 最前線の現場から【スクウェア・エニックス】 キャラクターの身体を作る, Think IT,” 2016. [オンライン]. Available: https://thinkit.co.jp/article/10015. [アクセス日: 17 2 2021].

[153] 三宅陽一郎, “AI 最前線の現場から【スクウェア・エニックス】デジタルゲームのための人工知能入門, Think IT,” 2016. [オンライン]. Available: https://thinkit.co.jp/article/10010. [アクセス日: 17 2 2021].

[154] 並木幸介, “モンスターAI,” 著: FINAL FANTASY XV の人工知能, Chapter:7, ボーンデジタル, 2019.

[155] 遠矢司, “兵士 AI,” 著: FINAL FANTASY XV の人工知能, CHAPTER:8 , ボーンデジタル, 2019.

[156] 上段達弘 , P. Prasertvithyakarn, “仲間 AI,” 著: FINAL FANTASY XV の人工知能, CHAPTER:6 , ボーンデジタル, 2019.

[157] 三宅陽一郎 , et al. , “AI:最前線の現場から【スクウェア・エニックス】 ゲームそのものを認識するメタ AI, Think IT,” 2016. [オンライン]. Available: https://thinkit.co.jp/article/10016. [アクセス日: 17 2 2021].

[158] P. Prasertvithyakarn, “会話 AI,” 著: FINAL FANTASY XV の人工知能, CHAPTER:11, ボーンデジタル, 2019.

[159] F. Gravot, T. Yokoyama , Y. Miyake, “Precomputed Pathfinding for Large and Detailed Worlds on MMO Servers,” GAME AI PRO, Chapter 20, 2013, pp. 269-287.

[160] Fabien Gravot, “ナビゲーションシステム,” 著: FINAL FANTASY XV の人工知能, ボーンデジタル社, 2019, pp. Chapter 3, .

[161] グラヴォ・ファビアン , 下川和也, “FINAL FANTASY XV におけるキャラクターナビゲーションパイプライン,” CEDEC 2017, 2017.

[162] マシュー・ジョンソン , 三宅陽一郎, “位置検索システム,” 著: FINAL FANTASY XV の人工知能, CHAPTER:5 , ボーンデジタル, 2019.

[163] H. Skubch, “Ambient Interactions: Improving Believability by Leveraging Rule-

Based AI,,” pp.411-422, chapter 35, Game AI Pro 3, CRC Press, 2017.

[164] 川地克明, “障害物を乗り越えるアニメーションの制御手法とその応用,” CEDEC 2017, https://cedil.cesa.or.jp/cedil_sessions/view/1733, 2017.

[165] マシュー・ジョンソン, ファビアン・グラヴォ, 南野真太郎, インギマール・グードムンソン, ハンドリック・スクバ , 三宅陽一郎, “ロギングと可視化,” 著: FINAL FANTASY XV の人工知能, CHAPTER:13, ボーンデジタル, 2019.

[166] M. W. Johnson, F. Gravot, S. Minamino, I. H. Guðmundsson, H. Skubch , Y. Miyake, “Chapter 3: Logging Visualization in FINAL FANTASY XV,” 著: Game AI Pro 3, A K Peters/CRC Press, 2017.

[167] 三宅陽一郎, “デジタルゲームにおける可視化技術,” 可視化情報学会誌, Vol.38(151), pp.158-163, 2018.

[168] N. Routhier, “Assassin’s Creed Origins‘: Monitoring and Validation of World Design Data,” GDC 2018, https://www.gdcvault.com/play/1025054/-Assassin-s-Creed-Origins, 2018.

[169] 上段達弘, “「強い」を作るだけが能じゃない！ディープラーニングで 3D アクションゲームの敵 AI を作ってみた,” CEDEC 2019, https://cedil.cesa.or.jp/cedil_sessions/view/2051, 2019.

[170] M. Nordin, “Deep Learning: Beyond the Hype,” GDC 2018, https://www.gdcvault.com/play/1025098/Deep-Learning-Beyond-the, 2018.

[171] H. Hu , et al., “Hierarchical Decision Making by Generating and Following Natural Language Instructions,” Advances in Neural Information Processing Systems 32 (NIPS 2019), 2019.

[172] 池田伸太郎 , 大岡龍三, “日本国内におけるスマートシティ・スマートコミュニティ実証事業の最新動向,” 生産研究 66 巻 1 号 p. 69-77, 2014.

[173] 萩田紀博, “街まるごとロボット化,” 日本ロボット学会誌 32 巻 3 号 p. 259-262, 2014.

[174] C. Lefebvre, “Virtual Insanity: Meta AI on 'Assassin's Creed: Origins',,” 2018, GDC 2018 .

画像掲載許諾・商標について

Fig. 9.3, Fig. 9.6, Fig. 9.11: (C)SEGA / FromNetworks,Inc./ FromSoftware,Inc., 2006

Fig. 12.1,Fig. 12.9,Fig. 12.10,Fig. 12.12,Fig. 12.13,Fig. 12.14,Fig. 12.15,Fig. 12.17,Fig. 12.21,Fig. 12.22,Fig. 12.23,Fig. 12.25,Fig. 12.27：© 2016-2021 SQUARE ENIX CO., LTD. All Rights Reserved. MAIN CHARACTER DESIGN：TETSUYA NOMURA

Fig. 11.1, Fig. 11.2,Fig. 11.3,Fig. 11.4,Fig. 11.5,Fig. 11.6,Fig. 12.2Fig. 12.3,Fig. 12.4, Fig. 12.5, Fig. 12.6, Fig. 12.7, Fig. 12.8, Fig. 12.11, Fig. 12.16,Fig. 12.18,Fig. 12.19,Fig. 12.20, Fig. 12.24, Fig. 12.26,Fig. 12.28,Fig. 12.29,Fig. 12.30,Fig. 12.31,Fig. 12.32,Table 12.1, Table 12.2, Table 12.3, Table 12.4, Table 12.5, Table 12.6, Table 12.7, Table 12.8：© 2021 SQUARE ENIX CO., LTD. All Rights Reserved.

その他掲載されている会社名，商品名は，各社の商標または登録商標です.
All other trademarks are the property of their respective owners.

謝辞

本論文の主査を引き受けて頂きました東京大学大学院工学系研究科システム創成学専攻の鳥海不二夫先生に深く感謝申し上げます．鳥海先生は度重なる相談に丁寧に答えて頂き，優れたご指導の数々を賜りました．深く感謝申し上げます．また，副査を引き受けて頂きました，同専攻の青山和浩先生，和泉潔先生，藤井秀樹先生，東京大学大学院情報理工学系研究科知能機械情報学専攻の鳴海拓志先生に感謝いたします．予備審査会において的確なご指摘を頂き，研究をさらに発展させる示唆をいくつも頂きました．先生方にご指導頂いた一つ一つのご指摘をこれからの研究に活かしていけたらと考えております．

またこのような挑戦の契機を頂いた神戸大学名誉教授の松田卓也先生に深く感謝申し上げます．さまざまな道を明るく示して導いてくださった東京大学大学院工学系研究科（現・戸田建設 ICT 統轄部顧問）白山晋先生に深く感謝申し上げます．

画像の掲載許諾を頂いた（C)SEGA 様，株式会社フロム・ソフトウェア様に感謝いたします．

本論文における第 12 章で取り上げました『FINAL FANTASY XV』の開発チームの皆様に感謝いたします．特に，人工知能開発に深く携わられた今村 紀之さん，Ingimar Gudmundsson さん，下川和也さん，上段達弘さん，白神陽嗣さん，関本大嗣さん，高橋光佑さん，遠矢司さん，並木幸介さん，平山創大さん，Fabien Gravot さん，Prasertvithyakarn Prasert さん，Hendrik Skubch さん，Matthew Johnson さん，南野真太郎さん，松尾祐樹さん，横山貴規さん，Wan Hazmer さんに感謝いたします．また，本作品の画像などの掲載許可を頂きました株式会社スクウェア・エニックス様に感謝いたします．

第 10 章で行った比較検証実験では 30 名近い方にお時間を頂きました．熱心にアイデアを書いて頂きました皆様に深く感謝申し上げます．

常に優しい言葉と美味しい料理で励ましてくれた妻に感謝いたします．

学びの機会を与えてくれた両親と姉に感謝いたします．