Vol.2014-GI-31 No.5
2014/3/17

プレイヤ行動の模倣に基づく AI キャラクタ行動ルールの自動生成

張輝陽[†1]　星野准一[†2]

ビデオゲームにおいて，AI キャラクタの行動がプレイヤのプレイ体験に大きい影響を与えるため，ゲーム AI の開発が非常に重要とされている．しかし，Multiplayer Online Battle Arena（MOBA）のようなゲームではチェスなどのボードゲームより複雑で，可能なゲーム場面数が厖大な故，ゲーム AI の開発コストも高くなると考えられる．ゲーム AI の開発コストを削減に関連する手法として，機械学習手法を用いた AI キャラクタの行動を自動的に生成する研究はあった．しかし，実際のゲームデータの中にあるプレイヤの行動には必要がない行動つまりノイズ行動も入っていると考えられる．本稿では MOBA ゲームにおいて自己増殖ニューラルネットワーク SOINN を用いた AI キャラクタの行動ルールの自動生成及びノイズ行動除去の効果について検討する．

Automatic Generation of AI Characters' Behavior Rules Based on the Imitation of Player's Behavior

HUIYANG ZHANG[†1]　JUNICHI HOSHINO[†2]

In video games, for AI characters' behavior will greatly impact on players' play experience, developers paid a lot of attention to the development of game AI. However, games like Multiplayer Online Battle Arena which are considered to be much more complex than board games such as chess have a great number of game situation and the development of game AI for these games is quite costly. Aimed saving cost of developing game AI, there are some researches on automatic generating behavior rules for AI characters by using machine learning. However, it is needed to take into account of players' behavior in real games' data that are not necessary. These behaviors are also called as noisy data. This paper will presents a method of automatic generating AI characters' behavior rules and discusses about the noisy excepting result of Self-Organizing Incremental Neural Network.

1. はじめに

現在欧米や中国で非常に流行っている Multiplayer Online Battle Arena（MOBA）という新しいゲームジャンルがある．MOBA の代表作の一つ League Of Legends では毎月のアクティブユーザ数が 3000 万を超えている．MOBA とは RTS ゲームを基本として敵を狩って経験値やお金を稼ぎ、キャラクタを育成するような RPG の要素が含まれたゲームである．MOBA ゲームのルールは基本的に 10 人のプレイヤが二つのチームに分けて，一人のプレイヤが一つのキャラクタを操って対戦を行い敵のアジトを潰したら勝利することである．また，プレイヤの人数が少ない時，ゲーム AI がプレイヤの代わりにゲームに参加することもある．この場合，ゲーム AI が操っているキャラクタはプレイヤのキャラクタと同じ役割を担っている．AI キャラクタの行動次第で，プレイヤのプレイ体験が大きく影響されるので，ゲーム AI の開発が非常に重要とされている．

しかし，MOBA ゲームは RTS ゲームを基本としたので，ゲーム中可能なゲーム場面の数は厖大で，ゲームの複雑さはチェスなどのボードゲームより高いと言われている[1]．また，MOBA の選択可能なキャラクタ数は 100 を超えており，個々違うステイタスやスキルを持っている．開発者は個々のキャラクタに特化したゲーム AI を開発する必要もあり，開発コストが非常に高くなると考えられる．もし AI キャラクタの行動が自動的に生成できると，MOBA ゲームの AI 開発コストが削減できると考えられる．

AI キャラクタの行動を自動的に生成することに関して，人間プレイヤのゲームプレイの記録あるいはプレイログを利用した研究は幾つかある．P．H．F． Ng（2009）らは RTS ゲームに対して Case-Based Reasoning と Dynamic Bayesian Network を用いて，ゲームのプレイログからプレイヤの行動モデルを構成し，プレイヤの行動を予測することができた．また，この予測によって，AI キャラクタの行動がより変化的になることができた[2]．そして，Ortega（2013）らは「Super Mario Bros」において AI キャラクタが三つの手法によるプレイヤ行動を模倣する効果を検討した[3]．また，ゲーム AI の自律的獲得について藤井（2013）らは生物学的制約を課した強化学習と経路探索の手法を用いて，ビデオゲームにおいて人間らしい NPC の振る舞いを自律的に獲得することが成功した[4]．濱名（2008）らは格闘ゲームにおいて，プレイヤの行動パターンを模倣することで試合ごとに自分の行動パターンを拡張し，成長していける COM を生成することができた[5]．これ以外に，機械学習手法を用いたゲーム AI の研究は[6][7][8][9][10]もある．

しかし，実際のゲームのプレイログにあるプレイヤの行動は全部必要な行動とは限らないので，そのまま教師データとして使用すると AI の学習に誤差が生じることがあり得る．学習データを前処理して必要のない行動データを削

†1 筑波大学大学院システム情報工学研究科
Graduate School of Systems and Information Engineering, University of Tsukuba
†2 筑波大学システム情報系
Faculty of Engineering, Information and System, University of Tsukuba

除する必要があると考えられる．

そこで，本稿では前述した MOBA ゲームの AI の自動的生成について，プレイヤがプレイしたゲームのプレイログを利用し，その中から抽出したゲーム場面とプレイヤの行動を模倣することで生成した AI キャラクタの行動ルールをノイズ耐性の高い機械学習手法の一つ自己増殖ニューラルネットワーク（Self-Organizing Incremental Neural Network, SOINN） によっての学習効果について検討する．

2. プレイログによる行動ルールの自動生成

MOBA ゲームではプレイログを生成する機能があり，プレイログからゲームの状況を完全に再現することができる．本手法では MOBA ゲームのプレイログを利用して，プレイヤが取った個々の行動に対して，行動の詳細と行動を取る時のゲーム場面の状況を抽出する．プレイヤがあるゲーム場面である行動を取ったということを一つの行動ルールとして記録する．そして，記録した全部の行動ルールを SOINN の入力データとして SOINN に学習させる．SOINN が学習した行動ルールのノードを AI キャラクタの行動ルールとして使用する．AI キャラクタは次の行動を選択する時，まずゲーム場面を確認し，SOINN のネットワーク中から現在のゲーム場面との距離が最も近い幾つかのノードの中から評価値の一番高いノードの行動を次の行動として選択する．本章では提案手法の詳細について説明する．

2.1 行動ルールの定義

行動ルールとはゲーム場面と行動の二つの部分によって構成される．ゲームのジャンルによって，ゲーム場面の意味は異なるが，本稿では MOBA において各キャラクタのステイタスや位置などの情報をゲーム場面とする．MOBA のキャラクタの基本的な行動として，移動，攻撃，スキルの使用の三種類がある．

ステイタスは特にキャラクタの HP， MP， Damage，Armor 四つを指している．他にもいろいろなステイタスがあるが，行動の意思決定に一番影響の高いのはこの四つのステイタスである．そして，位置を二つキャラクタ間の距離で表し，ゲーム場面を次のように定める．

（（HP， MP， Damage， Armor）* n ，（距離）*（n-1））

n はキャラクタの数である．複数キャラクタのステイタスと位置を含めたゲーム場面の情報は多次元のデータになり，その次元数 5n – 1 次元である．

行動について，移動行動は移動命令と移動の目的を記録している．攻撃行動は攻撃命令と攻撃の対象を記録している．スキル行動はスキル命令，スキルの名前，スキルの目標（他のキャラクタあるいは目標の座標）を記録している．

2.2 プレイログの分析

本手法はプレイヤのゲームプレイログを利用し，その中からプレイヤが各ゲーム場面で取った行動を抽出して，AI キャラクタの行動ルールとして記録する．プレイログを分析する時，一番重要なのはログから 2.1 に定義したゲーム場面と行動のペアを抽出することである．

ゲーム場面の情報はその時の各キャラクタのステイタスを記録することができるけど，行動の座標や目標をそのまま記録すると AI キャラクタがこの行動ルールによって行動する時，行動の目的地が毎回同じ座標になってしまい，行動がゲーム場面と合わない行動になってしまう可能性がある．そこで，本稿では移動行動とスキル行動の目的地座標を絶対位置ではなく，相対位置で記録する．相対位置は行動の目的地座標から目標の敵の位置の座標を引くことによって得られたベクトルである．そして，AI キャラクタは敵の位置によって行動の目的地を変化することができ，行動の正確性も高まる．これにより，プレイログから大量の行動ルールを生成することができる．

2.3 行動ルールの評価

AI キャラクタは評価によって，適切な行動ルールを選択する必要がある．

行動ルールの評価はその行動によるダメージの量と次の行動の評価によって決められる．次の行動の評価を考慮に入れるのは，プレイヤが取った行動は連続的であるからである．例えば，プレイヤは先に移動してから攻撃する場合に二つの行動ルール，移動の行動に続いて攻撃の行動がある．ダメージが与えたのは攻撃の行動だが，移動の行動によって，攻撃行動が成功したとも考えられる．この場合，前の移動行動の評価値を攻撃の行動より少し高めに設定して，AI キャラクタが同じゲーム場面で行動ルールを選択する時先に評価値の高い移動行動を選択してから，次に攻撃行動を選択する．これにより，AI キャラクタの行動はプレイヤの行動の連続性を一定程度に保つことができる．

2.4 SOINN による行動ルールの学習

SOINN は追加学習可能な教師なし学習手法であり，事前ネットワークの構造を決定する必要がないほか，高いノイズ耐性を有し，計算が軽いなどの特長がある．SOINN 入力パターンに対して，分布の近似が十分な領域に入力される場合，周辺のノードの更新に利用される．分布の近似が不十分な領域へ入力する場合，ノイズである可能性あるいは現在のネットワーク構造がまだ不十分である可能性があるのでノードをネットワークに取り込み，ネットワーク構造を変化させることができる[11]．また，関係するノードをエッジで連結する，ノイズを除去するなどの処理によって，効率的に学習データを学習することができる．

行動ルールを SOINN の入力データとして SOINN に学習させる時，SOINN は入力データ毎に一つのノードを生成する．ノードの中身は行動の詳細を記録しており，ノードのネットワーク中の位置はゲーム場面のデータによって決める．行動ルールの学習とともに，分布がスパースのノードをノイズノードとして削除することができる．そして，分布が近いノードを有効ノードとして学習して，AI キャラク

タの行動ルールとして利用する．

プレイヤの行動を見ると，特定のゲーム場面において，多く使われている行動，つまり分布の近似が十分な行動は意味ある行動と考えられる．その一方，あまり使われていない行動，つまり分布の近似が不十分な行動は無意味な行動である可能性が高く，SOINN によって除去することができる．そして，2.2 の方法で得た行動ルールのノードの中で，多数ノードの分布が十分近似していて，少数のノードの分布が近似していないと考えられる．

第 3 章ではこの仮説及び本手法の効果を検証するための実験について説明する．

3. SOINN を用いた評価実験

市販のゲームの多くはソースがオープンではないので，それらのプレイログを用いて AI キャラクタの行動ルールを作るのは非常に難しい．したがって，本稿では一つオープンソースのゲーム（Simple MOBA）を作り，提案手法の実装を行った．

3.1 Simple MOBA

Simple MOBA は MOBA の形に作ったゲームである．キャラクタ数は二体で，各キャラクタが四つのステイタス（HP， MP， Damage ，Armor）と四つのスキルを持っている．二つのキャラクタの内，一つはプレイヤが操り，もう一つは AI が操る．ゲームが開始すると，二つのキャラクタが長方形のフィールドに出現し，移動，攻撃，スキルなどの行動を取りながら対戦を始める．どちらかの HP が 0 になると，ゲーム終了とする．この間プレイヤが取った行動は全部プレイログに記録される．

3.2 実験手順と結果

実験はプレイヤー人対 AI 一人の戦闘場面で，AI キャラクタが完全に自動生成した行動ルールに従って行動するように設定する．ゲームを 25 回行い，毎回のプレイログをすでに行ったゲームのプレイログに加えて AI に学習させ，生成した行動ルールを次回のゲームに使用する．

SOINN の二つのパラメータλと age の調整については参考文献[11]に書いた表 1 を目安に今回の実験データは低次元で少クラス数のデータであり，より多くの行動ルールを学習できるためλと age をそれぞれ最大値の入力データ数とその百分の一に設定する．そして，三種類の行動ルールのノードを別れて SOINN に学習させる．25 回の全部のプレイログの学習結果は表 2 に示したように，Total，Learned，Updated， Deleted はそれぞれ全部のノード，生成した行動ルールノード，ノードのベクトルを行動するために使ったノードと削除されたノードである．

表 2 でプレイログから生成した行動ルールの半分以上がノイズノードとして SOINN に削除されたことが分かった．削除されたノードはどのようなノードかを知るため，25 回のプレイデータの中で 5 回ごとに削除されたノードの評価値と学習したノードの評価値の比較を表 3 に示す．

表 1　SOINN の学習データに関する目安表

データ量	多	数百万～数千万
	中	数千～数十万
	少	数十～数百
ノイズ量	多	全体の 30%程度
	少	全体の 10%程度
次元数	高次元	～数百次元
	低次元	～数十次元
クラス数	多	数十クラス程度
	少	十クラス程度

表 2　全部データの SOINN による学習結果

	Total	Learned	Updated	Deleted
All	2324	424	433	1467
Move	1475	278	282	915
Attack	413	55	60	298
Skill	436	91	91	254

表 3　五回毎の平均評価値の比較

Average Evaluation		5	10	15	20	25
Learned Nodes	Move	108	134	96	83	104
	Attack	321	207	294	260	250
	Skill	164	165	137	127	134
Deleted Nodes	Move	134	141	121	127	121
	Attack	194	177	183	155	167
	Skill	120	139	134	143	141

表 3 の結果から見ると，学習したノードの中攻撃ノードの平均評価値だけは削除された攻撃ノードの平均評価値より高いことがわかった．移動ノードとスキルノードでは削除された方の評価値が高い．

3.3 考察

表 2 に示したように，今回のプレイログから生成した 2324 個の行動ルールのノードの内 1467 個がノイズと認識され SOINN に削除された．プレイログから抽出したプレイヤの行動に無意味な行動があり，その割合が非常に少ないと予想したが，SOINN はその半分以上を無意味な行動と認識してしまった．SOINN の特徴により，つまり半分以上の行動ルールノードの分布が十分ではなく，非常にスパースであることが考えられる．

今回の実験で SOINN に学習されるデータは行動ルール（ゲーム場面，行動）の中のゲーム場面を示す部分（(HP，MP， Damage， Armor)，(距離)）によって構成されたノードである．Simple MOBA は二つのキャラクタだけなので，ノードデータの次元数は二つキャラクタのステイタス数とキャラクタ間の距離を含めて九次元になる．また，データの中で Damage と Armor は特別なスキルを使用した場合以

外にあまり変化はなく,その値の範囲も 30 から 100 までとなっている．よく変化するのは HP， MP と距離である．HP と MP の値の範囲はゲーム設定により 0 から 2000 までとなっている．その一方，距離の値の範囲は 0 から数千までとなり，HP や MP より高い．つまり，距離の値が九つのデータ中で一番影響力が高い．距離が少し変化すると，ノードが九次元区間内の位置は大きく変わることになる．そして，実際のゲーム中で HP や MP の滑らかな変化と比べて，キャラクタ間の距離は常に大きく変化している．そのため，プレイログから生成した行動ルールのノードの分布は非常にスパースになっていて，SOINN にノイズノードとして削除されやすいようになると考えられる．

スパースの問題を解決するには SOINN の学習の前に全部の行動ルールのノードのデータをスケーリングする必要がある．各データの重要性によって，スケーリングの範囲を設定する．例えば，距離の重要性はそれほど重要ではないので，範囲を HP の範囲と同じくらいに縮小したり，Damage と Armor の重要性を考慮して，値の範囲を少し大きめに設定したりする．また，ノードがスパースにならないように，実験で各データの値の範囲を確定する必要もある．

表 3 の結果もノードのスパースの影響を受けていると考えられるが，Attack ノードについては少し有益な結果が出ている．五回の比較で， SOINN によって学習された Attack ノードの評価値は全部削除された Attack ノードの評価値より高い．Simple MOBA の中で，二つのキャラクタが十分近い時だけ攻撃できるので，攻撃ノードのデータの中で「距離」の値の範囲は他二種類のノードより小さく，相対的にスパースの影響が小さいため，SOINN に正しく学習されたと考えられる．表 3 に示した結果もちょうど表 2 に対する検討を検証できている．そして，もしデータをスケーリングしたら，削除されたノードの評価値はより小さくなると考えられる．

また，一部のノードは高い評価値を持っているが，その分布が十分ではないため，SOINN で削除してしまうことも可能である．このようなノードを正しく学習できるように，SOINN がノードを削除する時ノードの分布だけではなく，ノードの評価値も考慮に入れる必要があると考えられる．

4. おわりに

本稿ではプレイログを利用し，AI キャラクタの行動ルールを自動的に生成する手法の提案及び自己増殖ニューラルネットワーク SOINN を用いた効果の検討を述べた．

プレイログを利用して，プレイヤ行動の模倣によって自動的に AI キャラクタの行動ルールを生成することができた．

しかし，ノイズデータを削除するための SOINN が予想通りに動かなかった．その原因として，行動ルールノードのデータの値の範囲の違いによって，データ全体がスパースになっていることが判明できた．スパースの問題を解決するため，データについてスケーリングの必要性について述べた．また，一部の高評価値のノードが誤って削除されないように，SOINN についての改良も説明した．今後は前述した点について，本手法を改良していきたいと思う．

参考文献

[1] Santiago Ontanon, Gabriel Synnaeve, Alberto Uriarte, Florian Richoux, David Churchill, Mike Preuss: A Survey of Real-Time Strategy Game AI Research and Competition in StarCraft, Computational Intelligence and AI in Games, Vol. 5, pp. 293-311, 2013.

[2] P.H.F. Ng, S.C.K. Shiu, H. Wang: Learning Player Behaviors in Real Time Strategy Games From Real Data, 2009.

[3] Ortega, J., Shaker, N., Togelius, J. and Yannakakis, G. N.: Imitating human playing styles in Super Mario Bros， Vol. 4， pp. 93–104.

[4] 藤井叙人，佐藤祐一，中嶌洋輔，若間弘典，風井浩志，片寄晴弘：生物学的制約の導入による「人間らしい」振る舞いを伴うゲーム AI の自律的獲得，GPW2013， pp. 73-80(2013).

[5] 濱名克季，田中彰人，星野准一 : 模倣学習により成長する格闘ゲームキャラクタ，情報処理学会論文誌，Vol. 49，No.7，pp. 2539-2548，2008.

[6] Siddhesh V. Kolwankar: Evolutionary Artificial Intelligence for MOBA/Action-RTS Games using Genetic Algorithms, 2012.

[7] Ontanon, S., Mishra, K., Sugandh, N., & Ram, A.: Case-based planning and execution for real-time strategy games. In Case-Based Reasoning Research and Development, pp. 164-178.

[8] Hsieh, J.L., Sun, C.T.: Building a player strategy model by analyzing replays of real-time strategy games. In: IJCNN, WCCI, pp. 3106–3111, 2008.

[9] Weber, B. G., Mateas, M., & Jhala, A.: Building human-level ai for real-time strategy games. In Proceedings of the AAAI Fall Symposium on Advances in Cognitive Systems, pp. 329-336, 2011.

[10] Hsieh, Ji-Lung, and C-T. Sun: Building a player strategy model by analyzing replays of real-time strategy games, IEEE World Congress on Computational Intelligence, 2008.

[11] 山崎和博，巻渕有哉，申富饶，長谷川修：自己増殖ニューラルネットワークSOINNとその実践，日本神経回路学会誌，Vol.17，No.4，2010.