

# 無限1UPを題材としたアクションゲームの裏技を発見する 自己学習手法の提案

高田 亮介<sup>†1,a)</sup> 橋本 剛<sup>†1,b)</sup>

**概要:** アクションゲームには製作者の意図しない動作をする裏技が多く存在するが、一般には人間がプレイすることによって発見される。機械が裏技を発見できるようになれば、ユーザの楽しみ方が広がるだけでなく、ゲームのデバッグ補助にも役立つと考えられる。本稿では、エージェントが試行を繰り返して成長していく強化学習を用いることで、裏技を教えることなく発見する手法を提案する。実験として、スーパーマリオブラザーズの裏技の1つである「無限1UP」を題材とし、進化戦略と多層パーセプトロンを用いた深層強化学習によって発見するAIを作成した。学習の結果、無限1UPのような動作を行うエージェントが確認できた。提案手法により、他のゲームでも裏技の発見が可能であると考えられる。

**キーワード:** 裏技, 強化学習, ニューロエボリューション, 無限1UP, マリオ

## Proposal of A Self Learning Method to Discover Secret Tricks Using Infinite 1-up as The Theme

TAKATA RYOSUKE<sup>†1,a)</sup> HASHIMOTO TSUYOSHI<sup>†1,b)</sup>

**Abstract:** There are many tricks in action games that make behaviors unintended by the producers, but in general they are discovered by humans playing. If machines can find tricks, it can not only expand user's way of enjoying but also help to debug of the game development. In this paper, by using the reinforcement learning by which agents repeatedly try and try to grow. A method to discover tricks without teaching is proposed here. As a theme, "Infinite 1-up" is used and as an experiment, an AI to be discovered by the deep reinforcement learning by using MLP and ES. As a result, the agent performed action like Infinite 1-up is confirmed. Likewise, it is considered possible to discover tricks in other games.

**Keywords:** Trick, Reinforcement Learning, Neuro-evolution, Infinite 1-up, Mario

### 1. はじめに

コンピュータゲームにおいて、裏技はユーザの楽しみ方を広げるひとつの要素として知られている。特にアクションゲームにおいては行動の自由度が高いことから多くの裏技が存在し、中にはゲーム性を損ねてしまうものもある。そのひとつに「無限1UP」という裏技がある。これは、任

天堂が発売したスーパーマリオブラザーズというアクションゲームの裏技であり、特定の位置で特定の操作をすることで、得点が増え続けるという裏技である。これはバグによる動作ではなく、仕様上での動作であり、製作者の意図しない裏技だと考えられる。本研究では、このような製作者の意図しない仕様動作上の裏技を研究対象とする。

通常、裏技は人間のプレイによって発見されるものであり、自動で発見する仕組みは報告されていない。機械が裏技を自動で発見できるようになると、ゲームに対する楽しみ方が増えるだけでなく、ゲーム開発者がゲーム性を損なうような裏技を事前に見つけることができる。そこで本研

<sup>†1</sup> 現在, 松江工業高等専門学校  
Presently with National Institute of Technology, Matsue College

<sup>a)</sup> j1318@matsue-ct.jp

<sup>b)</sup> hashimoto@matsue-ct.jp

究では、スーパーマリオブラザーズの無限 1UP を題材として、機械が自動で裏技を発見する手法について考察する。スーパーマリオブラザーズは NP 困難であることが科学的に証明されており [1], 探索により発見する方法も考えられるが、特定の条件を満たした際にのみ発生する裏技を探索するのはあまり現実的でないと予想し、今回は機械学習によるアプローチを試みる。DeepMind が発表した DQN や A3C などの汎用的な強化学習の手法は簡単なゲームでしか成功しておらず [2][3], スーパーマリオブラザーズなどある程度の複雑さを持つゲームでは学習が局所解に収束してしまう可能性があるため、本研究では進化戦略によるニューロエボリューションを用いる。これにより、局所解からの脱出が期待できる。今回は Mario AI の国際競技大会である Mario AI Competition[4] で用いられた Mario AI Benchmark[5] を実験環境として、ニューロエボリューションによって学習するエージェントを構築し、様々な状況のステージで学習させることでその結果を考察する。

## 2. 裏技

本章では裏技の判定方法と無限 1UP について説明する。

### 2.1 判定方法

コンピュータゲームにおける裏技は、ここでは「通常プレイでは実現できない動作を行うテクニック」と定義する。特に無限 1UP など仕様上の裏技では、あるパラメータが通常では起こりにくい異常な値になると考えられる。そこで、本研究では「ある 1 つのパラメータが異常に高くなった状態」を裏技発生の判定方法とすることを提案する。

### 2.2 無限 1UP

本研究の題材である無限 1UP が発生している様子を図 1 に示す。無限 1UP はスーパーマリオブラザーズの代表的な裏技であり、発生する手順を以下に示す。

- (1) 階段状に積まれたブロックの上方から敵キャラクター「ノコノコ」が降りてくる。
- (2) マリオがノコノコを 1 度踏みつける。
- (3) ノコノコは「甲羅」の状態となり、静止する。
- (4) マリオが甲羅の左端を踏みつける。
- (5) マリオに踏みつけられた甲羅は階段側に移動する。
- (6) 甲羅は階段のブロックで跳ね返り、マリオに向かって移動してくる。
- (7) 跳ね返ってきた甲羅を再びマリオが踏みつける。  
このとき、(4) の状態に戻る。

## 3. 実験環境

本章では実験環境である Mario AI Benchmark について説明する。

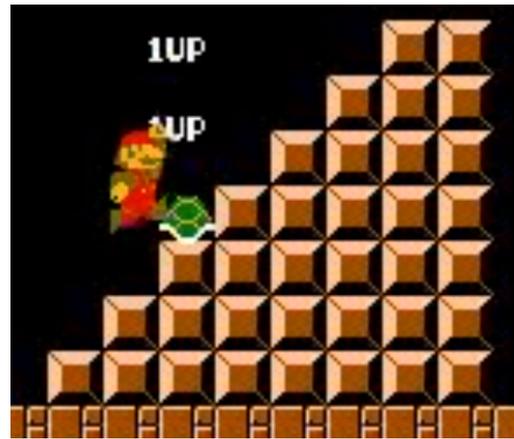


図 1 無限 1UP (任天堂, スーパーマリオブラザーズより)

Fig. 1 Infinite 1-Up(Nintendo, Super Mario Bros.).

## 3.1 Mario AI Competition

IEEE 主催の Computing Intelligence and Games (以下, CIG) という国際会議がある。CIG は 2005 年から毎年、全世界で開催されており、ボードゲームや戦略ゲーム、格闘ゲーム、アクションゲームなど多種にわたるゲーム AI の競技が行われている。中でも 2009 年から 2012 年まで Mario AI Competition という Mario AI の対戦競技が CIG で行われており、Mario AI の研究は盛り上がりを見せた [4]。

## 3.2 Mario AI Benchmark

本研究で使用するのは Mario AI Competition で用いられた Mario AI Benchmark と呼ばれる環境 [5] である。これはスーパーマリオブラザーズをトリビュートしたゲームが Java で動作し、専用のエージェントを読み込むことでマリオを操作することができる。マリオの周囲の情報はプログラム上で取得でき、行動は全て配列で表現される。また、ステージマップは Benchmark の機能により自作できるようになっている。これにより、無限 1UP の発生しうるステージを自作し、学習によって行動を選択することが可能になっている。

## 3.3 Mario AI Benchmark での無限 1UP

Mario AI Benchmark でのマリオの動作は実際のスーパーマリオブラザーズと異なる点が多い。特に、マリオが甲羅を踏みつけた時の動作が大きく異なる。具体的には、Mario AI Benchmark では静止している甲羅を踏みつけた際にマリオはジャンプするが、スーパーマリオブラザーズではジャンプしない。この差異は無限 1UP の動作に大きく関わる事象であるため、Mario AI Benchmark では実際の無限 1UP と全く同じ動作が発生しないと考えられる。そこで本研究ではこれより「スコアが異常に増え続ける動作」を無限 1UP として議論する。ここで無限 1UP が発生すると考えられる条件を以下に示す。

- (1) 踏んで倒されると甲羅になる敵キャラクターが存在
- (2) 甲羅が跳ね返るブロックが存在

#### 4. 提案手法

本章では無限 1UP を発見するためのアイデアと学習手法について説明する。

##### 4.1 アイデア

今回はあるパラメータが異常に高い状態を裏技とみなす前提で、1つのパラメータに注目して学習させることを考える。注目しているパラメータが最大になるように学習を続けることで、裏技を発見した時のパラメータは異常値を示すと予想できる。学習の種別としては、線形として扱えないゲームであることと、裏技は試行錯誤を通して発見されるものであると考えられることから深層強化学習を用いる。これにより、エージェントの行動の結果から報酬をフィードバックし、それに応じて報酬を最大化するように学習する。

無限 1UP はマリオの裏技の中でも特にスコアが増加し、最終スコアは通常プレイでは取りえない値になることが知られている。これを利用して、最終スコアを注目するパラメータとして設定し、最大化するように強化学習させれば発見できると考えられる。

##### 4.2 ニューロエボリューション

ニューロエボリューションはニューラルネットワーク(以下, NN)に進化的アルゴリズム(以下, EA)を組み合わせた強化学習の手法である。今回はニューロエボリューションの中でも HyperGP と呼ばれる手法 [6] を用いる。これはユニットの位置を固定し、重み決定に遺伝的プログラミングを用いる手法である。具体的には、NN の重み計算に EA を用いることで、報酬が最大化するような NN に収束させる。今回は、NN として多層パーセプトロンを、EA として進化戦略を用いる。

###### 4.2.1 多層パーセプトロン

多層パーセプトロン(以下, MLP)は代表的な NN で、単一方向にのみ伝播する順伝播型 NN である。MLP のモデル図を図 2 に示す。通常の MLP は誤差逆伝播法によってユニット間の重みを更新するが、今回は EA によってユニット間の重みを更新させる。

###### 4.2.2 進化戦略

進化戦略(以下, ES)は、EA の中でも実数関数の非線形を扱う際に用いられることが多く、突然変異を主な操作として最適化を行う。突然変異には標準偏差の正規分布を持つ乱数を用いる。今回は親集団と子集団の両方から次世代の個体を選択する  $(\mu + \lambda) - ES$  を用いる。このモデル図を図 3 に示す。

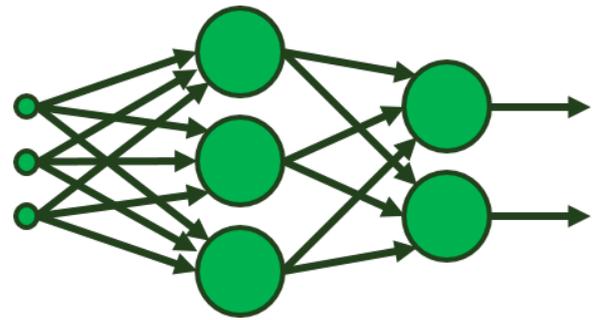


図 2 MLP モデル図  
Fig. 2 MLP model.

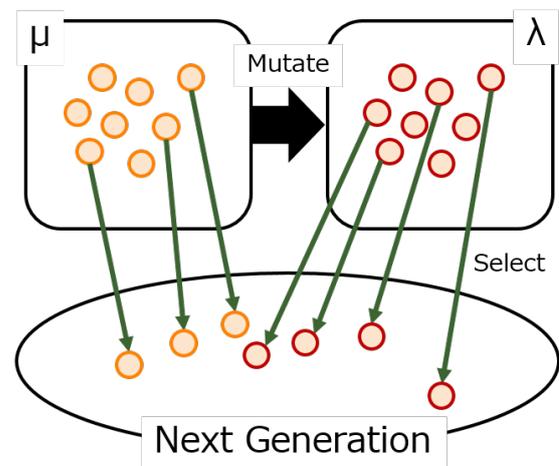


図 3  $(\mu + \lambda) - ES$  モデル図  
Fig. 3  $(\mu + \lambda) - ES$  model.

##### 4.3 裏技の難易度

実際の無限 1UP はごく限られた状況でしか発生しないことから、学習によっては発見できないことも考えられる。そのため、3.3 章に挙げた無限 1UP の発生する条件を満たすステージを、発見しやすいと考えられるステージから順に実験し、どの程度の難易度まで発見できるのかを検討する。

#### 5. 実装

本章では実際に作成したステージや学習エージェントについて説明する。

##### 5.1 エージェントの構築

4.2 章のニューロエボリューションを Mario AI Benchmark のエージェントとして構築した。MLP と ES は、Mario AI Benchmark に組み込まれているプログラム [7] を使用した。今回は敵や甲羅を踏みつけた際の得点をスコアとし、報酬は最大スコアとして設定した。表 1 に Mario AI Benchmark のパラメータと実際に設定した報酬を示す。また、ニューロエボリューションに用いた学習パラメータ

表 1 報酬パラメータ

Table 1 Reward parameters.

パラメータ	報酬
進んだ距離	0
ゴール	0
マリオの状態	0
コイン	0
ファイアフラワー	0
倒した敵の数	0
キノコ	0
残り時間	0
隠しブロック	0
1UP キノコ	0
踏みつぶし	10

表 2 MLP のパラメータ

Table 2 MLP parameters.

パラメータ	値
入力ユニット数	28
中間ユニット数	10
出力ユニット数	6

表 3 ES のパラメータ

Table 3 ES parameters.

パラメータ	値
個体数	100[個体]
世代数	1000[世代]

について、表 2 に MLP を、表 3 に ES を、それぞれ示す。なお、表 3 に示すパラメータは Mario AI Competition の Learning Track において定められている学習回数によって設定した。

## 5.2 ステージの生成

本研究では、無限 1UP が発生すると考えられる 6 ステージと、ランダムに生成した 10 ステージの計 16 ステージを作成した。

### 5.2.1 無限 1UP が発生するステージ

3.3 章の無限 1UP が発生する条件を満たした難易度の違うステージを 6 つ作成した。実際に作成したステージの無限 1UP が発生する要素の部分を図 4、図 5、図 6、図 7、図 8、図 9 に、それぞれ示す。ステージ 1 (図 4) ではノコノコが静止しており、エージェントはノコノコの位置で甲羅を踏むことによって無限 1UP できるように作成した。ステージ 2 (図 5) ではノコノコが移動するようにして、正しいタイミングでノコノコを踏むことを要求するよう作成した。その他のステージも、順に要求する動作が複雑になるように作成し、実際の無限 1UP に近い状況であるステージ 6 (図 9) を最後の難易度とした。なお、ステージ 4 (図 7) からステージ 6 (図 9) にかけて、階段状に積まれたブロックの手前に穴を設けたのは、階段を降りた平地でなく階段の中段で無限 1UP を発見させるためである。これら無限 1UP の発生する要素はマリオの初期位置から 100 ブロック以内に配置し、学習によって十分到達できるものと考えて作成した。

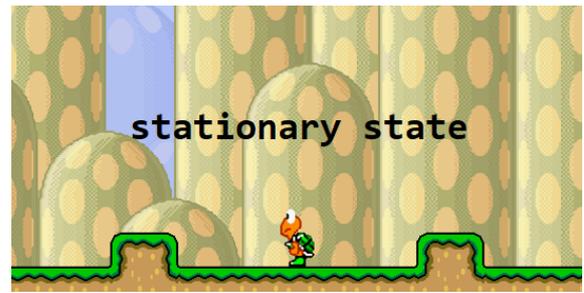


図 4 ステージ 1

Fig. 4 Stage 1.



図 5 ステージ 2

Fig. 5 Stage 2.



図 6 ステージ 3

Fig. 6 Stage 3.

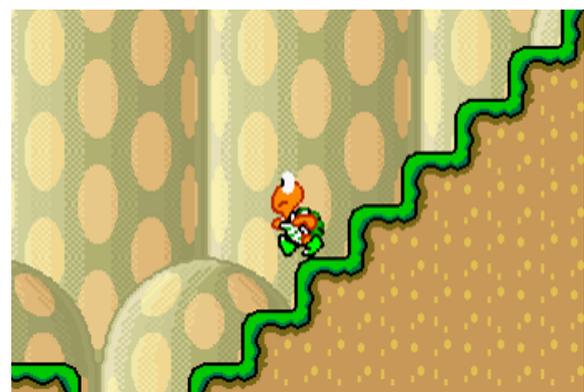


図 7 ステージ 4

Fig. 7 Stage 4.

### 5.2.2 ランダム生成したステージ

5.2.1 章の自分で用意した要素のみでは敵の数やブロックの配置が単純であると考えられる。しかし実際のゲームでは地形や敵の種類や数が多く、さらに複雑である場合が多い。そこで、より複雑なステージで裏技を発見させるため、Mario AI Benchmark のランダムステージを生成する機能により、Seed 値を変えた 10 ステージを作成した。実際に

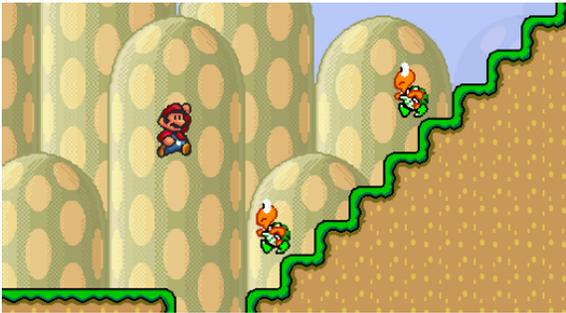


図 8 ステージ 5  
 Fig. 8 Stage 5.

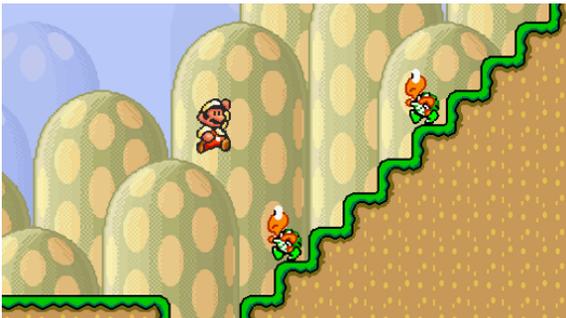


図 9 ステージ 6  
 Fig. 9 Stage 6.



図 10 ランダムステージ 1  
 Fig. 10 Random stage 1.



図 11 ランダムステージ 2  
 Fig. 11 Random stage 2.



図 12 ランダムステージ 3  
 Fig. 12 Random stage 3.



図 13 ランダムステージ 4  
 Fig. 13 Random stage 4.

作成したステージの一部を図 10, 図 11, 図 12, 図 13, 図 14, 図 15, 図 16, 図 17, 図 18, 図 19 に, それぞれ示す。

## 6. 実験

本章では実験として行った内容を説明する。

### 6.1 学習

5.1 章で構築したエージェントを 5.2 章で作成したステー



図 14 ランダムステージ 5  
 Fig. 14 Random stage 5.



図 15 ランダムステージ 6  
 Fig. 15 Random stage 6.



図 16 ランダムステージ 7  
 Fig. 16 Random stage 7.



図 17 ランダムステージ 8  
 Fig. 17 Random stage 8.



図 18 ランダムステージ 9  
 Fig. 18 Random stage 9.



図 19 ランダムステージ 10  
 Fig. 19 Random stage 10.

ジで学習させた。学習回数は表 3 における個体数 × 世代数であり, 今回は 100000 回とした。なお, この数字は Mario AI Competition で定められた学習回数である。

### 6.2 裏技の判定

学習の経過途中で, 世代単位ごとの最大スコアを記録し, 後にグラフ化して遷移の様子を確認した。

## 7. 結果

本章では実験の結果を説明する。

### 7.1 無限 1UP が発生するステージ

5.2.1 章で作成した, 無限 1UP の発生する 6 ステージで学習させた結果を図 20 にまとめて示す。また, 目視によって確認したところ, 全てのステージで無限 1UP の動作が確認できた。特にステージ 6 (図 9) では, 実際の無限 1UP に近い動作を確認した。この様子を図 21 に示す。

### 7.2 ランダム生成したステージ

5.2.2 章のランダム生成した 10 ステージで学習させた結果, 最終スコアが上昇した範囲において大きく 4 つに分類された。具体的には, 上昇値が 100 程度, 1000 前後, 3000

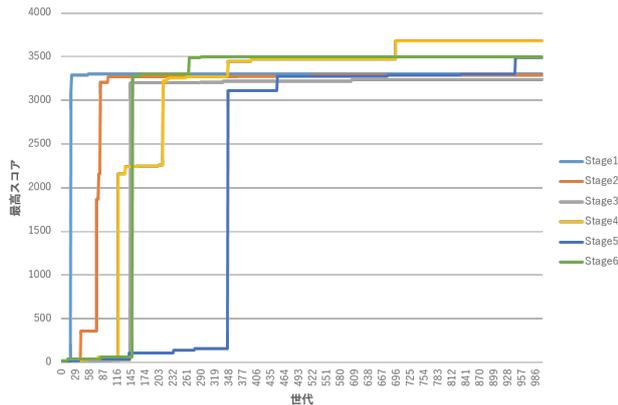


図 20 無限 1UP 各ステージ学習世代毎の最高報酬推移

Fig. 20 Transitive graph of max reward for each generation at Infinite 1-up stage.



図 21 ステージ 6 で無限 1UP が発生している様子

Fig. 21 Appearance of Infinite 1-up in stage 6.

前後, 9000 前後の 4 種類であった。これを上昇値の順に図 22, 図 23, 図 24, 図 25 に, それぞれ示す。また, 目視によって確認したところ, 図 22 の 2 ステージで結果を出したエージェントは無限 1UP らしい動作は確認できず, 残りの 8 ステージで無限 1UP の動作が確認できた。図 25 の結果を出したエージェントが無限 1UP を行っている様子を図 26 に示す。

## 8. 考察

本章では実験結果から裏技が発生する条件と学習手法について考察する。

### 8.1 無限 1UP が発生するステージ

7.1 章の結果から, 無限 1UP の発生する条件を満たす全てのステージで最終スコアのパラメータが急上昇していることがわかる。これは, タイムアウトまで無限 1UP を続けたことにより最終スコアが異常に高くなったと考えられ, 裏技を発見できたと言える。学習エージェントは何も

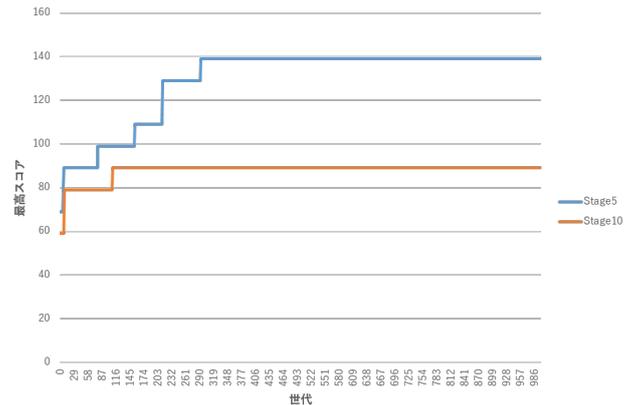


図 22 ランダムステージ学習世代毎の最高報酬推移

(上昇値 100 程度)

Fig. 22 Transitive graph of max reward for each generation at random stage (rise value around 100).

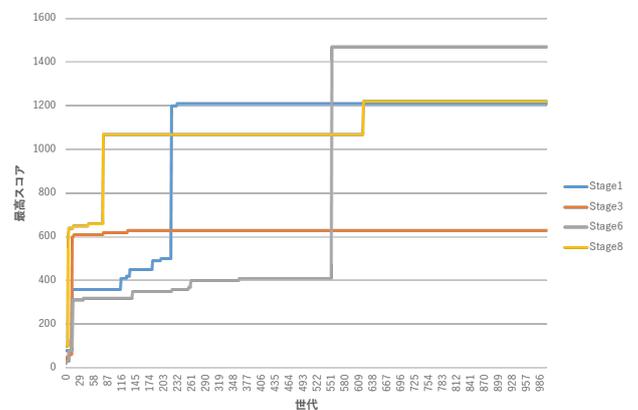


図 23 ランダムステージ学習世代毎の最高報酬推移

(上昇値 1000 前後)

Fig. 23 Transitive graph of max reward for each generation at random stage (rise value around 1000).

知らない状態からランダムに動き回り, 偶然スコアが増えたエージェントを進化的計算によって次世代のエージェントに引き継ぐ。こうして世代を重ねる度に甲羅を踏み続けてスコアを増やす方法を学習し, 次第にタイムアウトまで甲羅を踏み続けるようになったと考えられる。

### 8.2 ランダム生成したステージ

7.2 章の結果, 図 23, 図 24, 図 25 から, ランダム生成した 10 ステージのうち 8 ステージで最終スコアのパラメータが急上昇していることがわかる。これは, 8.1 章で述べた考察と同様のことが推論される。また, 図 22 のように, 最終スコアが急上昇しなかったランダムステージ 5 (図 14) とランダムステージ 10 (図 19) の結果においては, 学習エージェントの到達できた範囲に無限 1UP の発生する条件を満たした地形や敵が存在しなかったからであると考えられる。この 2 つのステージにおいては, ブロックの配置

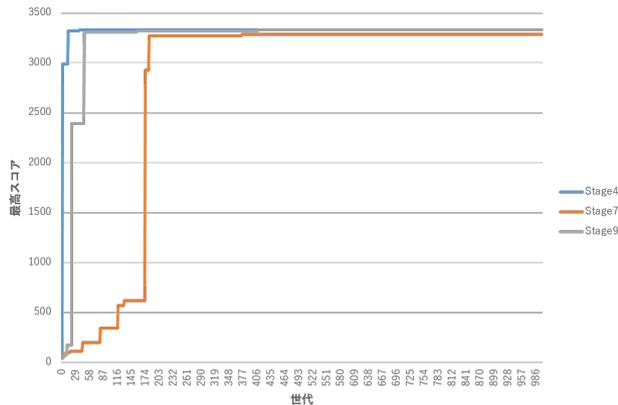


図 24 ランダムステージ学習世代毎の最高報酬推移 (上昇値 3000 前後)

Fig. 24 Transitive graph of max reward for each generation at random stage (rise value around 3000).

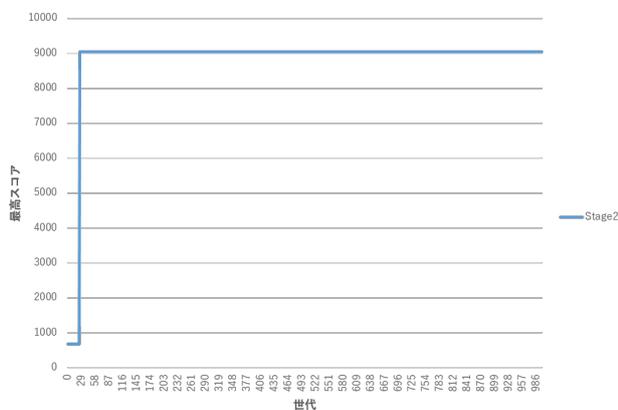


図 25 ランダムステージ学習世代毎の最高報酬推移 (上昇値 9000 前後)

Fig. 25 Transitive graph of max reward for each generation at random stage (rise value around 9000).



図 26 ランダムステージ 2 で無限 1UP が発生している様子  
Fig. 26 Appearance of Infinite 1-up in random stage 2.

や敵などステージの構造で複雑な要素の分散が大きく、学習回数が足りなかったことも予想される。

以上のことから、裏技の発生する条件を満たしており学習回数が十分である場合、複雑なステージでも裏技が発見できると示唆される。

### 8.3 学習手法

他の汎用的な深層強化学習の手法でも、ある 1 つのパラメータに注目して学習させることで裏技を発見できる可能性があると考えられるが、DQN など他の汎用的な学習手法は単純なゲームでしか成功しておらず、複雑なステージだと学習が局所解に収束してしまう可能性が高い。今回は学習エージェントにニューロエボリューションを用いることで、進化計算により複雑なステージでも学習が局所解に収束することなく、報酬である最終スコアを最も増やす最適解である裏技の発見に至ったのだと考えられる。

## 9. おわりに

現状、裏技は人間のプレイによって発見されるものであり、機械が発見する仕組みは報告されていないため、本稿では機械に教師データとして教えることなく強化学習させて発見させる手法を提案した。具体的には、MLP と ES を用いたニューロエボリューションで学習エージェントを構築し、無限 1UP を発見することができた。これは汎用的な手法であり、MLP の入力や報酬とするパラメータを変えることで他の裏技にも適用できると考えられる。今後の課題として、無限 1UP 以外の裏技の発見、DQN や A3C など他の深層強化学習の手法を用いた実験と考察などが挙げられる。そして、それらを踏まえて裏技を発見する方法論を確立し、自動で裏技を発見する理論的に完全な手法の実現を目指したい。

### 参考文献

- [1] Greg Aloupis, et al.: Classic Nintendo Games are (Computationally) Hard, pp. 1-36, 2015.
- [2] Volodymyr Mnih, et al.: Playing Atari with Deep Reinforcement Learning, NIPS 2014 Deep Learning Workshop, pp. 19, 2013.
- [3] Volodymyr Mnih, et al.: Asynchronous Methods for Deep Reinforcement Learning, ICML 2016, pp. 1-19, 2016.
- [4] Julian Togelius, et al.: The Mario AI Championship 2009-2012, AI Magazine 34, pp. 89-92, 2013.
- [5] Sergey Karakovskiy, et al.: The mario AI Benchmark and Competitions, IEEE Transactions on Computational Intelligence and AI in Games, volume 4 issue 1, pp. 55-67, 2012.
- [6] Zdenek Buk, et al.: NEAT in HyperNEAT Substituted with Genetic Programming, Adaptive and Natural Computing Algorithms, pp. 243-252, 2009.
- [7] Julian Togelius, et al.: Super Mario Evolution, Proceedings of the IEEE Symposium on Computational Intelligence and Games, pp. 1-6, 2009.