弾幕シューティングゲームにおけるQ学習を用いた自機の制御
Q-Learning-Based Avatar Control in a Bullet-Hell Shooter Game

須賀　貴大
Takahiro Suga
法政大学情報科学部ディジタルメディア学科
E-mail: takahiro.suga.6m@stu.hosei.ac.jp

Abstract

AI research using machine learning for board games such as Go is becoming popular, and some research produces stronger AI than professional players. On the other hand, research using machine learning for bullet-hell shooter games is still few, and past research did not result in producing AI comparable with human players. There are AI studies that use influence maps to predict positions of enemy bullets in bullet-hell shooter games. However, such AI is not sufficient for shooting at enemies. This paper proposes two methods for creating bullet-hell shooter AI by combining an influence map with Q-learning, an instance of machine learning. The proposed methods are called DCI and ICI. Both methods calculate the relative position of the avatar and the enemy and use it when the player AI decides its action. DCI combines an influence map with Q-learning directly while ICI combines an influence map with Q-learning indirectly. The paper presents a comparison experiment on three AI methods: DCI, ICI, and a simple influence-map method. The enemies used in the experiment were reproduced from enemies of actual bullet-hell shooter games called Toho. The result of the experiment shows that ICI worked most successfully.

1. はじめに

機械学習を用いたAIの研究が行われている [1] [2]．特に囲碁などのボードゲームの研究はさかんであり，2015年に囲碁ソフトの AlphaGo がプロに勝利する実績をあげている [2]．しかし，弾幕シューティングゲーム(弾幕SG)における機械学習の研究は少ない．弾幕 SG はアクションゲームの 1 種であり，その最大の特徴は“避ける”要素にある．弾幕 SG は敵弾が非常に多く，それを避けながら敵に自弾を当てる．野村ら [1]は機械学習を用いた弾幕SGの AI 研究を行ったが，人間のプレイには及ばすまだ改善の余地がある．一方で栗谷ら [3]は影響マップにより敵弾の位置を予測することで回避性能を向上する弾幕SGの AIを開発した．しかしこの手法は回避を優先させるため，敵を倒すことを後回しにしてしまう．

本研究では敵の位置予測に機械学習を用いて自弾の当たる射線上に自機を移動させる手法と，栗谷らの影響マップを利用した敵弾回避の手法を組み合わせて，よりハイスコアを狙う手法を提案する．手法は Q 学習と影響マップを直接的に組み合わせる手法(DCI)と間接的に組み合わせる手法(ICI)の 2 種類あり，これらに加えて影響マップのみで敵弾を避ける AI の 3 種類を比較した．その結果 ICI は勝率が高くより素早く敵を倒せる結果となった．

2. 関連研究

弾幕 SG に関する研究は少ない．論文 [3]では弾幕 SG の AI 設計に影響マップの実装を提案している．論文 [4]では人間の見る焦点の外が正確な認識ができない特徴をAI に実装することを目標としている．その第一歩として弾幕 SG の AI で実装を行っている．論文 [5]では影響マップを利用した人間らしい動作をする弾幕 SG の AI 実装を行っている．

また弾幕 SG の機械学習に関する研究も行われており，論文 [1]では Deep Q-Network と呼ばれる機械学習の手法で弾幕 SG を学習させている．しかしこの研究では人間のプレイと比較すると十分にゲームを学習できたとはいえない結論に終わっている．

ゲームの機械学習に関する研究が様々行われている．論文 [2]では囲碁を Deep Learning で学習させてプロにも勝てるほど強い囲碁 AI を実装させることに成功した．

3. 準備

3.1. 影響マップ

影響マップとは，画面を等間隔な格子に分割してそれぞれに問題とする性質の評価値を記録していく方法である．本研究では弾幕 SG において問題とする性質を自機がその位置にいるとどのくらい被弾する危険があるかの危険度とする．敵と敵弾の周りの危険度を高く設定することで危険度の低い場所が自機の移動すべき安全地帯となる．

本研究で以下の構造の影響マップを使用する．影響マップの格子の大きさを20×20ピクセルの正方形とする．格子cごとの危険度を定義する．敵Eの周りを危険度$d_{\text{enemy}}(E, c)$，全体の敵弾の集合Bの中の 1 つの敵弾bの周りの危険度を$d_{\text{nearb}}(b, c)$と定義し，その範囲に格子がある場合，危険度を 50 追加する．また敵弾の進行方向の危険度を$d_{\text{gob}}(b, c)$と定義し，10 フレーム後までの敵弾の移動先を予測する．各n $(1 \leq n \leq 10)$に対して，nフレーム後に格子が敵弾進行内にあるとき危険度を 5 追加する．従ってある格子cの危険度をd_{c}は以下となる．

$$d_{\text{c}} = d_{\text{enemy}}(E, c) + \sum_{b \in B} \left(d_{\text{nearb}}(b, c) + d_{\text{gob}}(b, c) \right)$$

Supervisor: Prof. Hiroshi Hosobe

本研究では格子ごとの危険度を可視化することができ，図 1 にあるように格子ごとの危険度を色で表示している．危険度が高いほど赤色が目立つように表示させて危険度が全くないときは白色になるように表示させている．

図 1 自機周りの影響マップの危険度

3.2. Q 学習

Q 学習とは強化学習の手法の 1 つである．ある状態sにおいて選択可能な行動の集合$A(s)$の中の行動aにQ値$Q(s,a)$をつけて次の行動を決定する．行動のたびに報酬rに基づいて$Q(s,a)$を以下の式で更新する．

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma \max_{a' \epsilon A(s')} Q(s',a') - Q(s,a)\right)$$

αは学習率と呼ばれ，$0 < \alpha \leq 1$の範囲で表される．γは割引率とよばれ，$0 < \gamma \leq 1$の範囲で表される．$\max_{a' \epsilon A(s')} Q(s',a')$は行動$a$をとった後の状態$s'$において選択可能な行動の集合$A(s')$の中で最大の価値をもつ行動$a'$の値である．

4. 提案手法

本研究では，Q 学習と影響マップを利用してよりハイスコアを狙う AI として以下の 2 つの手法を提案する．

- Direct combination of an influence map with Q-learning (DCI)
- Indirect combination of an influence map with Q-learning (ICI)

2 つの手法の共通点は以下の通りである．状態sを表現するために自機と敵機の相対位置を利用する．相対位置は自機の x，y 座標からみて敵の x，y 座標がどの位置にあるかできまる．相対位置の組み合わせは自機からみて敵の x 座標が–200 未満，–200 以上–150 未満，–150 以上–100 未満，–100 以上–90 未満，–90 以上–80 未満，–80 以上–70 未満，–70 以上–60 未満，–60 以上–50 未満，–50 以上–40 未満，–40 以上–30 未満，–30 以上–20 未満，–20 以上–10 未満，–10 以上–0 未満，0，0 超過 10 以下，10 超過 20 以下，20 超過 30 以下，30 超過 40 以下，40 超過 50 以下，50 超過 60 以下，60 超過 70 以下，70 超過 80 以下，80 超過 90 以下，90 超過 100 以下，100 超過 150 以下，150 超過 200 以下，200 超過，の 27 通りあり，同様にして y 座標に対しても 27 通りもつ．従って合計27×27の相対位置の組み合わせをもつ．

8 方向に静止を加えた 9 通りの動きの集合をAとしてその中の 1 つを行動aとする．ある状態sの中のAからQ値の高い行動aを選び選択して行動する．その結果に応じてQ値を更新する．自弾が命中したときに正の報酬$r = 1$を与え，外れたときに負の報酬$r = -1$を与える．さらに敵に接触または被弾した場合負の報酬$r = -100$を与える．自機の y 座標が敵機の y 座標より小さい場合，自弾が当たらないことよりこの場合のみ自機が下方向へ移動したときに正の報酬を$r = 1$与え，それ以外の行動をしたときは負の報酬$r = -1$を与える．また 6.1 節で述べられるように敵機の大きさが半径 25 ピクセルのため自機と敵の x 座標が 25 ピクセル以上離れているとき，y 軸方向のみ移動する自弾が当たらないとして自機の x 座標が敵の x 座標に近づく動きをしたときに正の報酬$r = 1$を与え，それ以外の行動をしたとき負の報酬$r = -1$を与える．

DCI と ICI の最大の違いは状態sに影響マップを直接取り入れているかである．DCI では計算量は増えるがよりハイスコアを狙える可能性がある一方で，ICI は DCI と比べて計算量が少なくなる分ハイスコアを狙えなくなる恐れがある．以下にそれぞれの手法の特徴と相違点を述べる．

4.1. DCI

DCI は Q 学習と影響マップを直接組み合わせる手法である．状態sを表現するために相対位置を利用することに加えて，自機周りの影響マップの情報Iも状態sに加える．情報Iは自機周りの格子 9 つのそれぞれ危険度が 0 とそれ以外の 2 パターンの2^9通りとする．従って合計$27 \times 27 \times 2^9$通りの状態sの組み合わせがある．

この手法の利点は，状態sに影響マップの状況を取り入れているため，様々なシチュエーションを想定できてよりハイスコアの狙える AI を開発できる可能性が高い．一方で，相対位置と影響マップ両方の全ての状況を取り入れるため状態sが非常に多くなり，計算量が増えて学習しきれない恐れがある．

4.2. ICI

ICI は Q 学習と影響マップを間接的に組み合わせる手法である．相対位置のみを利用して状態sを表現する．その中の行動の集合Aの中から自機周りの影響マップの中で危険度が最も低い行動方向の集合A'を求める．A'の中からQ値の高い行動a決める．

この手法の利点は，状態sに影響マップを取り入れていない分状態sが27×27通りとなり DCI と比べて少なくなり同時に計算量が少なくなる．一方で影響マップによる行動aの絞り込みを先に行うため避けることを最優先してしまうため積極性に欠けてハイスコアを狙えなくなる恐れがある．

5. 実装

Java ベースの Processing ライブラリ上で実装した．Processing は可視化に特化した言語であり，ゲームのアニメーションを画面に映し出すために使用している．学習している間は処理速度の向上のため画面は表示せず，十分な学習を終えた後の実験のときのみ画面を表示する．

6. 実験

DCI および ICI により開発された AI と影響マップのみで敵を避ける AI の 3 種類で比較実験を行う．DCI および ICI は十分に学習させた AI を実験対象とする．評価対象はある回数敵と対戦させたときの勝率，クリアまでにかかった時間とする．

表 1 敵の詳細

敵の名前	秋穣子(あきみのりこ)	河城にとり(かわしろにとり)	橙(ちぇん)	魂魄妖夢(こんぱくようむ)
出現するゲーム名	東方風神録 1 面ボス	東方風神録 3 面ボス	東方妖々夢 2 面ボス	東方妖々夢 5 面ボス
行動パターン	自機に向かう螺旋状に発射する弾と敵の中心から円形状に 2 回発射する弾を交互に発射する．弾の発射直前に画面上部でランダムな 1 方向に動く．	一定の距離感で画面を横切る弾を画面全体に展開しながら敵の中心から自機に向かう弾を放物状に発射する．放物状の弾を発射する直前に敵は画面上部でランダムな 1 方向に動く．	敵が縦横無尽に移動しながら弾を放つ．斜め移動のときは放物状に，横移動のときは円形状に発射する．最終的に中心に戻り，少し間隔をあけたあと画面上に存在する敵弾全てが下方向に加速しながら動く．これを繰り返す．	敵が画面左右どちらかの最下部に移動して画面端から端まで横切り，最下部から上方向に弾をランダムに発生させるフィールドをつくる．そのあと画面上部で動き周りながら円形状に弾を無数に発射する．
特徴	他の敵と比べて弾及び行動パターンが単純である．	画面上に常に横へ移動する弾があり，自機の行動範囲が狭められる．	敵の移動範囲が他の敵に比べて広く，自弾を最も当てにくい敵といえる．	他の敵と違い，下から上方向へ移動する敵弾が常に存在する．

表 2 実験結果

AI の種類	秋穣子				河城にとり				橙				魂魄妖夢			
	勝率 (%)	平均クリア時間 (秒)	平均生存時間 (秒)	全自弾の命中率 (%)	勝率 (%)	平均クリア時間 (秒)	平均生存時間 (秒)	全自弾の命中率 (%)	勝率 (%)	平均クリア時間 (秒)	平均生存時間 (秒)	全自弾の命中率 (%)	勝率 (%)	平均クリア時間 (秒)	平均生存時間 (秒)	全自弾の命中率 (%)
影響マップ	19	25.1	28.7	9.5	0	–	9.7	27.7	8	33.6	32.5	5.8	0	–	11.8	2.8
DCI	4	6.1	2.3	69.6	0	–	3.8	80.9	0	–	3.7	67.0	0	–	5.1	8.3
ICI	76	14.2	14.7	43.6	6	7.4	7.0	50.3	62	21.3	17.5	25.6	1	10.9	9.0	16.8

6.1. 実験方法

600×600 のピクセル上で自機と敵機を対戦させる．自機を半径 5 ピクセルの大きさの円とする．その行動パターンは x，y 軸方向へそれぞれ 5 ピクセルの速さの 8 方向に静止を加えた 9 通りとする．自弾は 3 フレームごとに発射されて上方向 30 ピクセルの速さで進むものとする．

敵機は実際の弾幕 SG の東方シリーズの敵を再現したものとする．本実験に使用する敵は 4 種類であり詳細は表 1 にある．それぞれ大きさを共通して半径 25 ピクセルの円とする．

DCI，ICI での学習率$\alpha = 0.1$ 割引率$\gamma = 0.9$とする．1 フレームの行動を 1 エピソードとして学習回数を 20 分に相当する 72000 回とする．学習対象は 1 つの敵のみとして対戦相手も学習させた敵のみとする．学習させた AI をそれぞれの敵に対して 100 回ずつ対戦して勝敗等のデータを記録する．敵の体力を 100 として，自弾が当たるたびに 1 ずつ減らす．一方で自機の体力を 3 として敵及び敵弾に被弾するたび 1 ずつ減らす．敵の体力を 0 にした場合勝利として，自機の体力が 0 になった場合敗北とする．

6.2. 実験結果

実験結果は表 2 のようになった．表 2 はそれぞれの敵に対しての AI の勝率，勝利したときの平均クリア時間，平均生存時間，全ゲームの自弾の命中率がまとめられている．どの AI も秋穣子，橙に対しては勝利を収めることもできたが，河城にとり，魂魄妖夢に対してはほぼ勝利することができなかった．以下にそれぞれの AI の傾向を述べる．

影響マップは 3 つの AI の中で平均生存時間が最も長いが，全自弾の命中率が最も低いことから避ける行動のみに徹していることがわかる．DCI はどの敵にも対して勝率がほぼ 0 であるが魂魄妖夢を除く敵に対して全自弾の命中率が 3 つの AI の中で最も高い．従って 3 つの AI の中で最も自弾を当てるための行動をしているが，敵弾を避ける行動をすることに欠けているがわかる．一方で ICI は 3 つの AI の中でどの敵に対しても最も高い勝率を収めており，特に影響マップの勝率が 0 でない敵に対して高い勝率を収めている．加えて平均クリア時間および全自弾の命中率が影響マップと比べて高いことから避けるだけでなく自弾を当てるための行動もしていることがわかる．3 つの AI の中で総合的に ICI が強いことがわかった．

6.3. 応用実験

6.2 節の実験結果より ICI がうまくいったことからどれほど有用性があるかの実験する．6.1 節でそれぞれの敵で学習させた 4 つの ICI を学習の対象外の敵と戦わせ，総当たり戦の形で実験を行った．実験環境は 6.1 節と同様のものとする．

実験結果は表 3 のようになった．4 つの中で橙を学習させた AI が最も勝率が高く，平均クリア時間も対魂魄妖夢戦を除いた 3 種類の敵に対して最も短くなった．橙とそれ以外の敵を比べたとき，最も異なる点は移動範囲である．他の敵が主に画面上部で移動しているのに対して橙は画面全体を動き回る行動をする．ICI の状態sは相対位置のみを考慮していることからより多くのパターンの相対位置に対する行動を学習することができ，行動範囲の狭い他の敵にも応用がきいたことがわかる．対象の敵を学習させて対戦した元の AI よりも高い勝率を収めたこ

表 3 ICI の学習対象以外の敵との総当たり戦の結果

学習に利用した敵	秋穣子				河城にとり				橙				魂魄妖夢			
	勝率 (%)	平均クリア時間 (秒)	平均生存時間 (秒)	全自弾の命中率 (%)	勝率 (%)	平均クリア時間 (秒)	平均生存時間 (秒)	全自弾の命中率 (%)	勝率 (%)	平均クリア時間 (秒)	平均生存時間 (秒)	全自弾の命中率 (%)	勝率 (%)	平均クリア時間 (秒)	平均生存時間 (秒)	全自弾の命中率 (%)
秋穣子	76	14.2	14.7	43.6	1	5.3	4.2	74.2	29	26.5	22.6	18.9	0	–	7.1	9.4
河城にとり	86	11.8	11.6	51.8	6	7.4	7.0	50.3	40	29.3	23.1	19.1	0	–	10.2	8.1
橙	93	9.4	9.7	56.7	49	6.5	8.5	67.3	62	21.3	17.5	25.6	8	11.9	8.3	25.6
魂魄妖夢	82	13.1	12.3	43.1	1	12.5	5.8	47.1	27	23.6	18.0	19.9	1	10.9	9.0	16.8

とから，学習対象によっては他の敵にも充分に通用することができることがわかった．

7. 議論

どの AI も秋穣子，橙と比べて河城にとり，魂魄妖夢に対しての勝率が低かった．考えられる原因は自機にせまる敵弾が一方向からではなく複数方向から来ることが考えられる．河城にとりは敵から発射される弾に加えて常に画面を横切る弾の二方向から敵弾が迫る．魂魄妖夢も同様に敵から発射される弾に加えて画面最下部から上方向に発射される弾の二方向から敵弾が迫る．このことからこれらの敵に対しては少なくとも二方向から迫る敵弾を避けなくてはいけなくなり，AI が避けきれず被弾してしまったことが考えられる．実験結果より以下に DCI と ICI について議論する．

DCI がうまくいかなかった原因は影響マップの情報Iの分類基準が少なすぎたためであると考えられる．本論文の分類の仕方であると危険度が 1 と危険度 100 が同じIとして分類されてしまい危険度が高いときでも危険度の低いと仮定した行動をとってしまう恐れがある．また逆の行動をとる恐れもある．そのため自弾の命中によってQ値に正の報酬が与えられる計算をするため，危険度の大小を考慮せず自機の行動が自弾を当てるためだけの極端な行動をとってしまったといえる．改善点としては影響マップの情報Iの危険度の分類パターンを増やすことである．例として危険度の分類を0，低い，高いといった3パターンにして計3^9の組み合わせにするなどがある．ただし，$2^9 = 512$であり，$3^9 = 19683$となるように分類パターンを増やすほど状態sのパターンが増え，膨大な学習量の必要性がでてくることも考慮すべきである．

一方で ICI は DCI と比べて平均生存時間の長いことから自弾を当てるだけの極端な行動をせず敵弾を回避しているといえる．さらに影響マップのみの AI と比べて，ICI は全自弾命中率が高く，回避のみだけでなく自弾を当てるための動きをしているといえる．また勝率が 3 つの AI の中で最も高いことから最も攻守のバランスの良い AI であると結論付けられる．

しかし敗北がある理由としては影響マップ自体にあるといえる．ICI が通用していない敵には影響マップのみの AI の勝率が 0 となっている．従って改善点としては影響マップの格子の大きさについての見直し，危険度の計算式の改良など影響マップ自体の改善が求められる．

8. おわりに

本研究では，避けることに特化した影響マップと一度の行動により結果が反映される Q 学習を組み合わせることによってより強い弾幕 SG の AI を開発することを目的として，影響マップと Q 学習を直接組み合わせる DCI，間接的に組み合わせる ICI の 2 つの手法を提案した．DCI と ICI と影響マップのみの AI の比較実験を行った結果，ICI が最も勝率が高く，かつ影響マップよりクリア時間を短くすることに成功した．さらに ICI が他の敵に通用するかの実験を行った結果，広範囲に移動する敵を学習させた AI が最も良い結果を示した．

本研究では，相対位置を 27 通りとしたが何通りのときに最適な強さになるか，また学習の対象を 1 つの敵としたが複数の敵を学習させた結果どのような結果になるかなど議論の余地がある．さらに本研究では自機の行動を単純なものにしたが，特殊コマンドを行動の 1 つに追加したときこの手法がどこまで有効かなどの今後の課題はたくさん残っている．

文　献

[1] 野村直也, 橋本剛, "弾幕シューティングゲームを対象とした汎用的学習法," 情報処理学会研究報告：ゲーム情報学(GI), vol. 39, no. 4, pp. 1-7, 2018.

[2] David Silver and Aja Huang, "Mastering the Game of Go with Deep Neural Networks and Tree Search," *Nature*, vol. 529, pp. 484-489, 2016.

[3] 栗谷拓哉, 橋本剛, "熟練プレイヤーレベルを目指す弾幕シューティング AI の開発," 情報科学技術フォーラム講演論文集，12(2)，pp. 383-384, 2013.

[4] 平井弘一, Reijer Grimbergen, "弾幕の認識に人間の視覚特性を取り入れたシューティングゲーム AI の研究," ゲームプログラミングワークショップ 2016 論文集, pp. 158-161, 2016.

[5] 佐藤直之, 池田心, "Influence Map を用いた経路探索による人間らしい弾避けのシューティングゲーム AI プレイヤ," ゲームプログラミングワークショップ 2016 論文集, pp. 57-64, 2016.