First Person Shootingゲームに対するAIエージェントを用いたバランス調整テストの自動化手法の提案

電気通信大学大学院　情報理工学研究科　情報学専攻

小澤陽平、西　康晴

はじめに

- 基調講演のTariq King氏の話を借りれば、自動化と銘打っているがが、マニュアルテスティングである
- どんなパラメータにすれば面白いゲームになるのかを自動探索しているものではない
- パラメータを変えたときに、ちゃんと狙ったとおりになってるかなって確かめるためのプレイログの収集をエージェントにやらせるという内容になる
- 最初に学習済みモデルで効果を確かめ、
次に、学習を行いエージェントの挙動をより細かく観察した

背景：やられてばかりじゃ面白くない

- First Person Shootingゲーム（FPS）は、主人公の視点で操作して敵を倒すゲームである
- FPSでは、敵にすぐやられてばかりいるとイライラする
- だからといって、自分が強すぎてもスリルがない

背景：ゲームの面白さのバランス

- ゲームの面白さは絶妙なバランスで成り立っている
  - 難易度が高すぎても、低すぎても、面白くない
  - FPSの場合、敵にすぐやられてばかりいるとイライラするし、自分が強すぎてもスリルがない
    - 例１：　生存時間 = 敵に倒されずにプレイできる時間
    - 例２：　Kill/Death比 = 敵を倒した数/敵に倒された数

背景：莫大なバランス調整のコスト削減が必要

- ゲーム開発ではバランス調整が生命線である
  - テストプレイして調整する、の繰り返し
  - デバッグの度に、バランス調整が必要
  - バランス調整テストに、数十人ものテスターを何ヶ月も雇う必要があり、莫大なコストが掛かっている
    - 10人で3ヶ月行った場合、3000円/時×150時間×3×10=1350万円の費用がかかる[1]

- バランス調整テストを自動化したい
  - コンピュータに自動でバランス調整テストをさせられれば、大きなコスト削減になる

背景：FPSのバランス調整テストの自動化

- FPSはバランス調整テストの自動化が難しい
  - リアルタイムに操作を自動生成しなくてはならない
    - 相手の行動に合わせて、回避や攻撃といった操作を行う必要がある
    - 最適な操作をゲームが始まる前に生成しておくことが出来ない
  - 自動生成する操作内容を決めるための条件が複雑である
    - 自分のステータス、近くにあるアイテム、敵の位置や向き・射撃動作中かどうか、自分が過去に倒された場所かどうか、など
    - 単純なif-then-elseルールでは人間のような強さを再現することができない

先行研究：FPSのバランス調整テストの自動化

- FPSのバランス調整テストの自動化の研究はない
  - 非FPSであれば、最適な操作を遺伝的アルゴリズムで事前に生成しておく研究がある[2]
    - リアルタイムに操作を自動生成していない
  - 人間ほど強くない自動化ならば、単純なif-then-elseルールで操作を自動生成する研究がある[3]
    - マップ内の遮蔽物の位置だけで、操作を自動生成している

目的

- 本研究ではリアルタイム性を持ち、複雑な条件を備えるテスト対象であるFPSに対してバランス調整テストを自動化する手法を提案する

アプローチ：深層強化学習技術

- リアルタイムに複雑な条件から操作を自動生成する技術に、深層強化学習がある
  - ゲームをクリアする深層強化学習エージェントは、色々と提案されている
    - ブロック崩し
    - AlphaGo
- FPS向けの深層強化学習エージェントも活発に研究されている
  - 研究用の有名な開発プラットフォームにvizdoomがある[6]
    - 商用のゲームプラットフォームをベースにしている
    - 深層強化学習エージェント同士で対戦する大会があり、様々なアルゴリズムの深層強化学習エージェントが出場している

アプローチ：FPSのエージェントに必要な技術

- FPS向けの深層強化学習のアーキテクチャとして
  CNN＋LSTMが提案されている[4][5]

- Convolutional Neural Network (CNN)
  - CNNは、画像から有効な特徴量を検出でき、一般物体認識で高い効果が確認されている[9]
  - ゲーム画像から有効な特徴量を取得するためにCNNを用いる
- Long-Short Term Memory (LSTM)
  - LSTMはRecurrent Neural Network (RNN)を拡張したネットワークで、メモリーセルと呼ばれる記憶域を持ち、長期的な記憶を保持できる[10]
  - CNNによる画面の特徴量から各操作のスコアの出力にLSTMを用いる

アプローチ：エージェントのアーキテクチャ

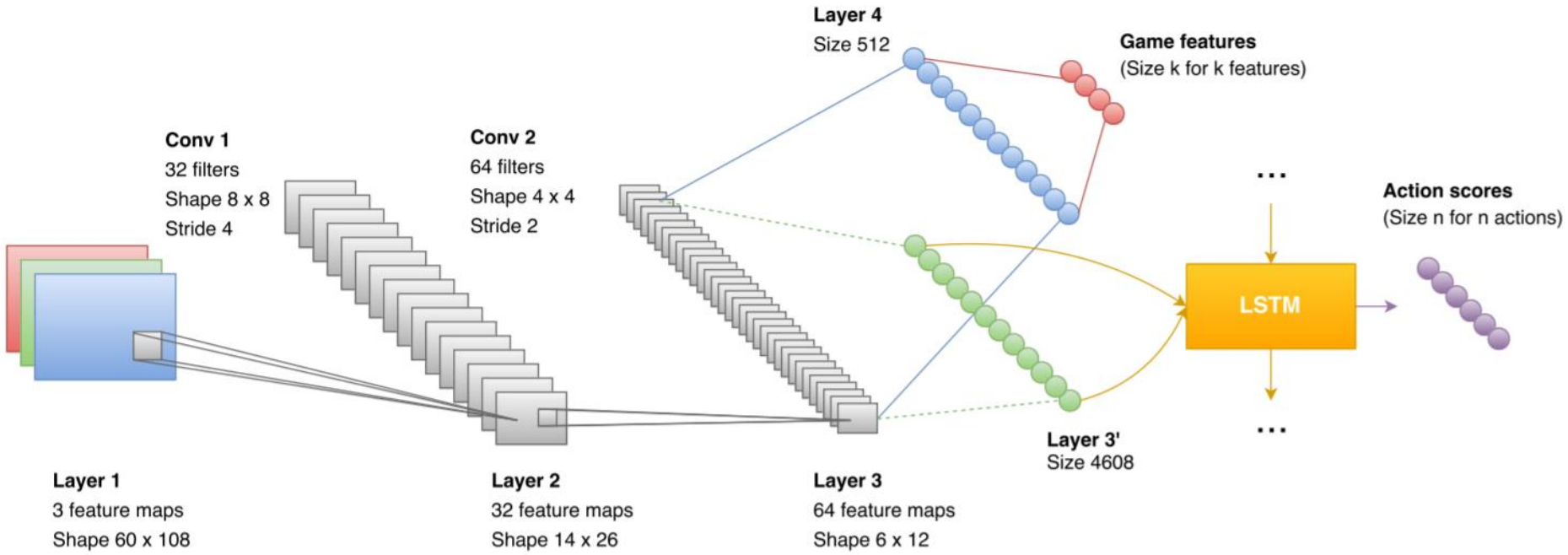


図1.提案されているCNN+LSTMアーキテクチャ[4]

- ネットワークがゲームフューチャにつながっているのが特徴
  - ゲームフューチャで敵やアイテムの有無を予測させる

アプローチ：CNN＋LSTMを備えるArnold[5]

- CNNとLSTMアーキテクチャで学習されたエージェント
- vizdoom上で実装されている
  - 大会では最強クラス
- 学習を途中でやめれば、色んな強さのエージェントを容易に作成できる
  - 同じアーキテクチャでの学習曲線を示す

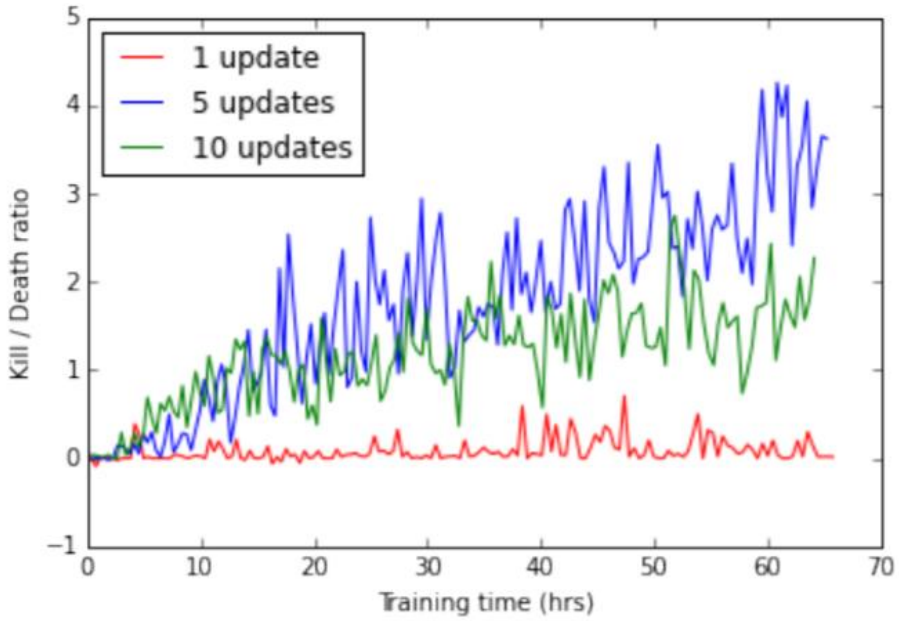


図2. CNN+LSTMアーキテクチャでの学習曲線[4]

提案手法：概要

- 本研究では、この深層強化学習エージェントを用いてFPSのプレイログを収集しバランス調整テストを行う

- 事前準備
  - ゲームドライバーの作成
    - ゲームのコントロールとゲーム情報の取得
  - エージェント作成
- バランス調整テストの設計
- テスト実行

提案手法：事前準備

- ゲームドライバーとエージェントを準備する
  - ゲームドライバー
    - ゲームの起動・終了、ゲーム情報の取得およびゲーム操作を行うAPIを提供する
      - vizdoomがこれにあたる
  - エージェント
    - ゲームをプレイできる深層強化学習エージェントを用意する
      - Arnoldの学習済みモデルおよび、同アーキテクチャで学習したエージェントを利用する

提案手法：バランス調整テストの設計

- 以下を設定する
  - 調整パラメータ、目標パラメータ及び目標値の設定
  - 調整パラメータの変更範囲およびステップ
  - 各調整パラメータでの試行回数の設定
  - プレイログから目標パラメータの算出方法

- これらの値はゲームプランナーあるいはゲームデザイナーといった難易度調整を行う開発者が行う
- 調整パラメータは多岐に渡るが、通常は幾つかのパラメータをピックアップし、調整パラメータとして変動させる

提案手法：テスト実行

- テスト設計に基づき、深層強化学習エージェントによるゲームプレイを行い、プレイログを収集する
- プレイログから、最も目標値に近い調整パラメータを特定する
- 特定には、平均や各類似度計算手法を用いる

適用例：調整パラメータの特定

- 目的

  調整パラメータと目標パラメータ及び目標値を与えた場合に、深層強化学習エージェントを用いて調整パラメータの特定が可能であることを確かめる

- 環境
  - ゲームルール：デスマッチモード
  - マップ：vizdoom competition 2017 track1[7]
  - 武器：ロケットランチャーのみ
  - エージェント:Arnold[8]の学習済みモデル

適用例：調整パラメータの特定

- 手順

① 調整パラメータとして敵の人数、目標パラメータとして生存時間、および目標値として10秒を与える

② 変更範囲を5~20, ステップを5として敵の人数n=5,10,15,20のそれぞれで死亡回数が300回になるまでプレイし、生存時間を記録する

③ 平均生存時間の差から、調整パラメータ値を特定する

適用例：調整パラメータの特定

表1．調整パラメータの特定

BOT人数[人]	5	10	15	20
目標値との差[sec]	5.40	-0.31	-1.43	-2.55
平均[sec]	15.40	9.69	8.57	7.45
標準偏差[sec]	15.92	7.16	5.37	4.57
変動係数	1.03	0.74	0.63	0.61

以上の手順により、n=10の時が、最も目標生存時間10秒に近く、調整パラメータは10と特定できた

検証：エージェントの評価

- 目的

人よりも強いエージェントが作成できるかを、ルールベースのエージェントと深層強化学習エージェントで比較する

比較にはKill/Death比率を用いる

- 環境
  - ゲームルール：デスマッチモード
  - マップ：vizdoom competition 2017 track1[7]
  - 武器：ロケットランチャーのみ
  - エージェント:Arnold[8]の学習済みモデル
  - 制限時間：60分
  - 敵の人数：7人

検証：エージェントの評価

- 手順

① ルールベースエージェントの作成を行う
  - 被験者２名が、vizdoomを15分間プレイし、エージェントのルール作成を行う
  - 被験者は画面から得られる情報のリストと、行える行動のリストを用いて、if-then-else構文でルールを記述する
  - 今回の検証では、このルールを擬似的に人が再現して行なった

② 人のプレイヤーの習熟を行う
  - 被験者2名が、人のプレイヤーとしてKill/Death比率を測定するために、操作説明と操作習熟を行った
  - できるだけ高いKill/Death比率を達成するように指示し、15分間習熟した

③ ルールベース、深層強化学習、人のプレイヤーで、Kill/Death比率を測定し、比較する

検証：エージェントの評価

- 結果

表2.エージェントの評価

プレイヤー	ルールベース		深層強化学習		人
	スコア	人との差	スコア	人との差	スコア
Kill[人]	283	-229	829	317	512
Death[人]	450	165	316	31	285
Kill/Death	0.63	-1.17	2.62	0.83	1.80

- ルールベースエージェントはKill/Death比率が0.63と低く、人並みのスキルを持っているとは言えない
- 深層強化学習エージェントはKill/Death比率が2.62と高く、人以上のスキルレベルであることがわかる

- 深層強化学習エージェントは、学習を途中で辞めることで、弱いエージェントが作成できるので、人並みをスキルのエージェントを作成できる

考察：適用例のパラメータ特定について

- 調整パラメータを特定する際に、バラツキが生じている
  - 変動係数が0.61~1.03とバラツキが生じている
- 特にbot人数が少ない方ではバラツキが大きくbot人数が多くなるに連れてバラツキが小さくなっている
- 敵キャラクターの数が少ないと接敵頻度が疎らになり、その結果生存時間がバラツキやすくなっていると考えられる

考察：深層強化学習エージェントの行動パターン

- Arnold[8]の学習済みモデルでは, 決まったルートを巡回していた
  - そのルートを巡回することが最もK/D比率が高くなると考えられる
- 人のエージェントはマップの隅々まで移動していた

- こうした敵がいないときのナビゲーションは人と異なっていた

ここまでのまとめ

- 本研究では、FPSに対する深層強化学習エージェントを利用したバランス調整テスト手法を提案した
- 深層強化学習エージェントは実際に人に近い強さのエージェントを作成できており、そのエージェントを用いて調整パラメータの特定をすることが出来た

エージェントの動きの観察

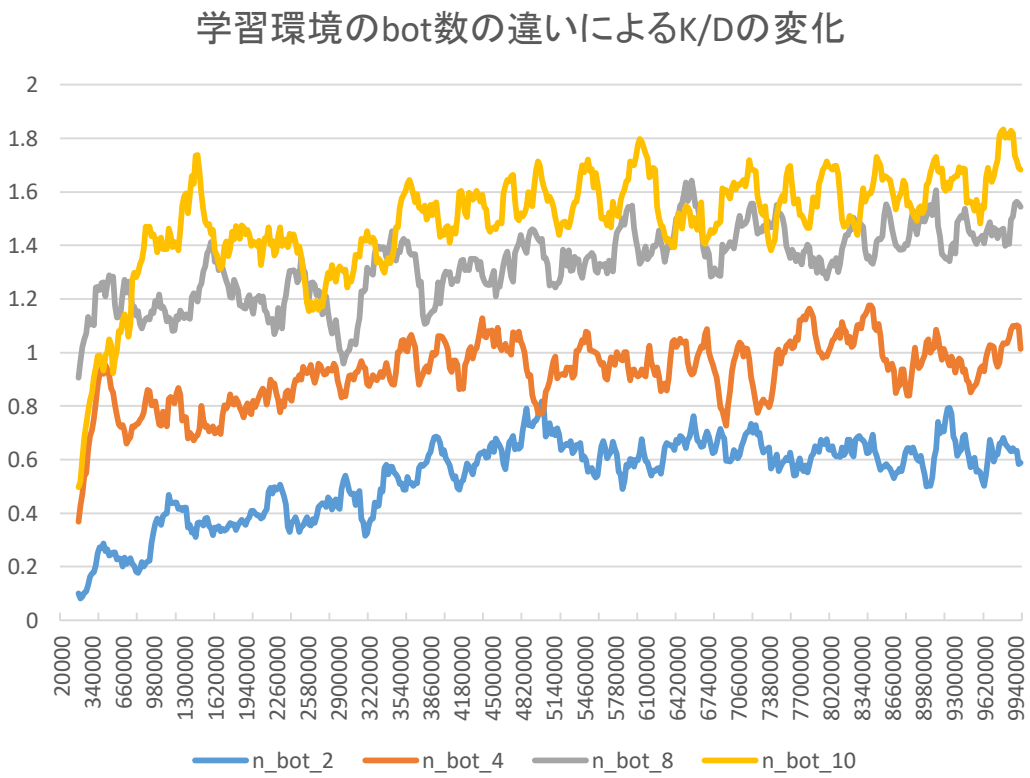
- 実際のAIエージェントの動きを観察し、バランス調整テストへの利用可能性を検討する
  - 学習を途中できりやめたAIエージェントは人並みに弱いエージェントなのか？
    - kill/death=2.0のエージェントは人に近くとも、kill/death=1.0のエージェントは人に近い動きをするのか？
    - だた、ランダムに動いてるだけではないのか？
  - 同じような動き方をしつつも、強さが異なるエージェントである必要がある
    - **高いkill/death比率のエージェントと低いkill/death比率のエージェントで動き方が大きく異なってはいけない**
  - そこで、様々なエージェント作成しkill/death比率や実際の動きを観察する
    - 作成したエージェント
      - 学習環境の違いによるエージェント
      - 学習時間の違いによるエージェント

実験1

- 目的
  - 学習環境の違いによるエージェントの強さの違いを観察し、弱いエージェントや強いエージェントが大きく異なる動きをしないか確かめる

エージェントの学習と選択

- 学習環境
  - マップ：vizdoom competition 2017 track1[7]
  - 武器：ロケットランチャーのみ
  - 1000万イテレーション学習した
  - 敵の人数：2,4,8,10
- エージェントの選択
  - 1000万イテレーションのうち、最もkill数-自殺数が少なくなるエージェントを選択した
- ボットの数が少ないほど、K/D比率が低くなる
  - これは単純にエージェントが弱いことを示すのではない



エージェントの評価

- 評価環境
  - マップ：vizdoom competition 2017 track1[7]
  - 武器：ロケットランチャーのみ
  - 制限時間：60分
  - 敵の人数：8人

結果：kill/death比率

表3. 学習環境の違いによるエージェントの差

	エージェント				
	2_bots trained	4_bots trained	8_bots trained	10_bots trained	Arnold pre trained
kill[人]	559	747	**817**	703	950
death[人]	**340**	538	520	439	378
kill – 自殺 [人]	522	701	**761**	658	892
k/d	**1.64**	1.39	1.57	1.6	2.51

- Kill/death比率としては、1.39～1.64のエージェントを作成できた
- 2_bots trainedは、kill/death比率は高いが、スコアであるkill-自殺[人]は低い

結果：既存モデルのエージェントの動き

Arnold

結果：bot人数2の環境で学習したエージェントの動き

学習環境：bots=2

- 敵がいない場合、後ろ向きに旋回し続ける
- 敵がいたら狙う

- 遠くに敵を探しに行くのではなくその場に留まるように学習された
- “待ち”タイプエージェント

結果：bot人数4, 8, 10環境で学習したエージェントの動き

学習環境：bots=4

学習環境：bots=8

学習環境：bots=10

- 時折振動があるが、概ね人に近い動きを再現できている
- 敵の人数が多い中央に入るように学習された
- “攻め”タイプエージェント

実験1：まとめ

- bot人数4, 8, 10環境で学習したエージェントの動き方は似ており、強さが違うエージェントとして扱うことが出来る
  - よりスコアが獲得できるように敵を探しに行くエージェント
  - Kill/death比率1.39～1.6
- bot人数2環境で学習したエージェントは他のエージェントと動き方が大きく異なり、kill/deathだけで単純に強いエージェントとして扱うことが出来ない
    - その場に留まり、敵が来るのを待つエージェント
- 学習環境の違いによるエージェントの差は、同一タイプの強さの違いだけではなく、異なるタイプのエージェントの可能性がある

実験２

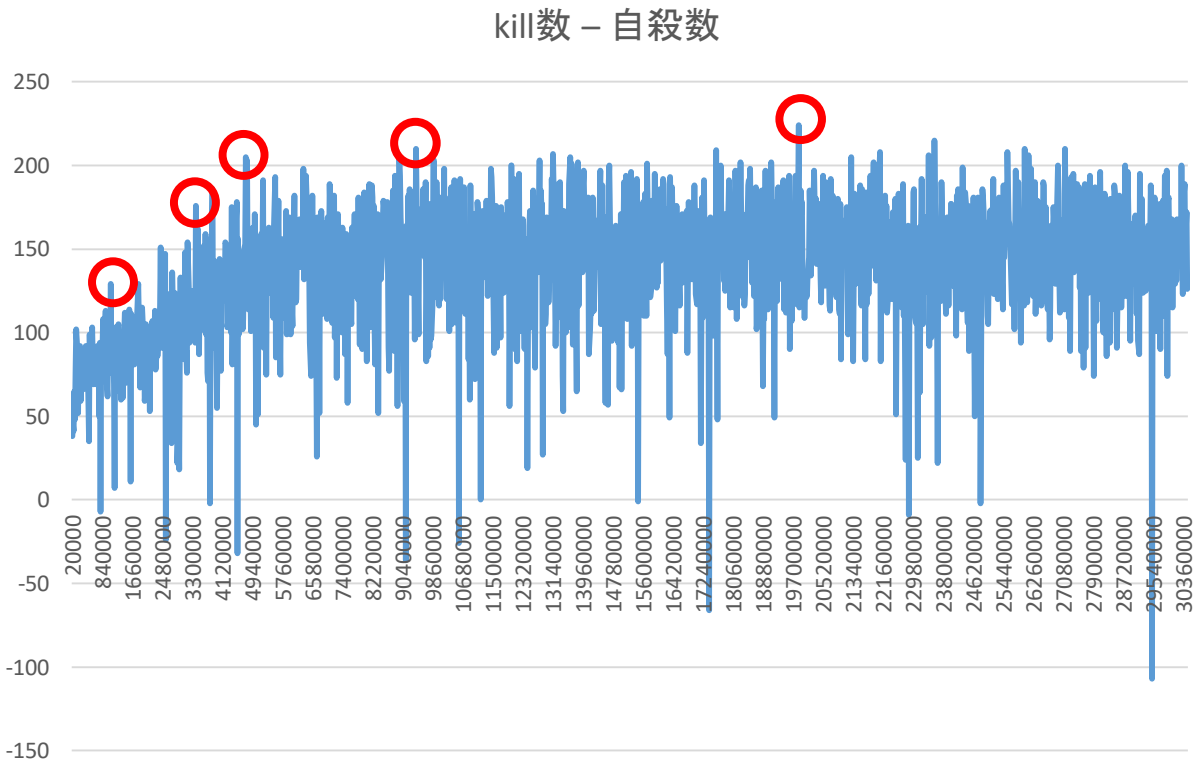
- 目的
  - 学習時間の違いによるエージェントの強さの違いを観察し、弱いエージェントや強いエージェントが大きく異なる動きをしないか

- 学習環境
  - マップ：vizdoom competition 2017 track1[7]
  - 武器：ロケットランチャーのみ
  - 敵の人数：８人
- 評価環境
  - マップ：vizdoom competition 2017 track1[7]
  - 武器：ロケットランチャーのみ
  - 制限時間：６０分
  - 敵の人数：８人

エージェントの学習と選択

- 学習途中のエージェントの挙動を観察する

- キル数-自殺数が大きくなるエージェントを抜き出し、挙動を観察した
  - 108万イテレーションのエージェント
  - 244万　　〃
  - 340万　　〃
  - 476万　　〃
  - 942万　　〃
  - 1986万　　〃



結果

表4. 学習時間の違いによるエージェントの差

	エージェント					
	108万イテレーション	244万イテレーション	340万イテレーション	476万イテレーション	942万イテレーション	1986万イテレーション
kill[人]	544	691	683	773	817	**913**
death[人]	**382**	566	533	533	520	500
suicide[人]	79	181	**47**	63	56	52
kill - 自殺 [人]	465	510	636	710	761	**861**
k/d	1.42	1.22	1.28	1.45	1.57	**1.83**

- Kill/death比率は、244万イテレーション以降は一様増加している
- 108万イテレーションは、kill/death比率は高いがスコアは低いという先程の敵が二人の環境で学習した結果と似ている

結果

表4. 学習時間の違いによるエージェントの差

	エージェント					
	108万イテレーション	244万イテレーション	340万イテレーション	476万イテレーション	942万イテレーション	1986万イテレーション
kill[人]	544	691	683	773	817	**913**
death[人]	**382**	566	533	533	520	500
suicide[人]	79	181	**47**	63	56	52
kill - 自殺 [人]	465	510	636	710	761	**861**
k/d	1.42	1.22	1.28	1.45	1.57	**1.83**

- deathと自殺は一度途中で増え、その後減少していくという学習をしている
- 244万イテレーションのkillも自滅も多いエージェントから340万イテレーションは自滅数を抑えたエージェントなっておりエージェントの成長が伺える

結果

表4. 学習時間の違いによるエージェントの差

	エージェント					
	108万イテレーション	244万イテレーション	340万イテレーション	476万イテレーション	942万イテレーション	1986万イテレーション
kill[人]	544	691	683	773	817	**913**
death[人]	**382**	566	533	533	520	500
suicide[人]	79	181	**47**	63	56	52
kill - 自殺 [人]	465	510	636	710	761	**861**
k/d	1.42	1.22	1.28	1.45	1.57	**1.83**

- kill数は最初に大きく増え、その後緩やかに上昇している

108万イテレーションの動き

108万イテレーション

- 移動量が少ない
- そのため、接敵頻度が少なくkill/death比率は高いが、絶対数は少ない
- 弾もたくさんばらまく

- まだ学習の序盤であり、人から見て意味のない行動をしているように見える

244万イテレーションの動き

244万イテレーション

- 移動量が増加
- 無意味な行動は減ったが、まだ無駄弾やループに嵌る
- Kill数は増加

- 244万イテレーションは特に自殺が非常に多い
  - 他と比べて3倍程度

340万イテレーションの動き

340万イテレーション

- 無駄弾はかなり減った
- 振動やループは残る

476万イテレーション以降の動き

- 476万イテレーション以降は見てわかる大きな違いはない
- しかし、kill/death比率は増加していっている
  - 1.45→1.57→1.83
- 徐々に動きが洗練され、その結果Kill/Death比が改善していったと考える
- 似た動きでスコアが異なるエージェントを作成できている

実験２：まとめ

- bot相手に対してkill/death比率1.45を超えたあたりからは徐々に強くなっていくエージェントが作成できた
  - 約500万イテレーション以降

まとめ

- ある程度のkill/death比率（=1.4前後）以降なら、強いエージェントと似た動きをしながらも少しずつ弱いエージェントを作成することが出来た

- 今後の課題
  - ある程度kill/death比率が出てから、異なる強さのエージェントを作成できるが、低いKill/death比率での人に近い動きの再現ができていない
  - 3次元空間を把握し、移動しているわけではないので、壁に当たり続けるといった人としては不可解な行動をする

参考文献

[1] デジタルハーツホールディングス 契約例, http://pdf.irpocket.com/C3676/xouG/pQde/qKK4.pdf

[2] 真鍋和子, “遺伝的アルゴリズムによる人工知能 を用いたゲームバランス調整”, CEDEC2017

[3] D. Karavolos, A. Liapis, G. Yannakakis, “Learning the Patterns of Balance in a Multi-Player Shooter Game”, Proceedings of FDG, 2017

[4] Lample, Guillaume and Chaplot, Devendra Singh, “Playing FPS Games with Deep Reinforcement Learning”, Proceedings of AAAI, 2017

[5] Chaplot, Devendra Singh and Lample, Guillaume, “Arnold: An Autonomous Agent to Play FPS Games”, Proceedings of AAAI, 2017

[6] M. Kempka, M. Wydmuch, G. Runc, J. Toczek, W. Jaśkowski, “ViZDoom: A Doom-based AI research platform for visual reinforcement learning”,

[7] vizdoom competition 2017, http://vizdoom.cs.put.edu.pl/competitions/vdaic-201 7-cig, 2019-01-28

[8] Arnold, https://github.com/glample/Arnold, 2019-01-28

[9] A. Krizhevsky, I. Sutskever, Geoffrey E. Hinton, “ImageNet classification with deep convolutional neural networks”, Proceedings of the 25th International Conference on Neural Information Processing Systems, vol1, pp.1097-1105

[10] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[11] H. Hasselt, A. Guez, D. Silver, “Deep Reinforcement Learning with Double Q-learning”, arXiv:1509.06461, ver3, Dec 2015

- ご清聴ありがとうございました

その他の学習パラメータ

- image size 440×225 →108×60 (RBG)
- action_combination:
- frame-skip: 4
- mini-batch-size: 32
- memory-size: 1000,000 frames
- discount factor: γ=0.99
- ε-greedy: 最初の100万イテレーションで1~0.1に線形減少
  その後0.1固定