深層強化学習の群ロボットへの応用：移動ロボット群の行動獲得の実現

○綿貫零真，松本泰輔，堀内 匡，青代敏行 （松江工業高等専門学校）

Application of Deep Reinforcement Learning to Multi-Robot System : Behavior Acquisition of Multi-Robot System

∗Ryoma Watanuki, Taisuke Matsumoto, Tadashi Horiuchi and Toshiyuki Aodai
National Institute of Technology, Matsue College

Abstract– Deep Q-network (DQN) is one of the most famous methods of deep reinforcement learning. We have already realized that single wheeled mobile robot acquired behavior to avoid walls and obstacles by using DQN. In this research, we apply DQN method to multi-robot environment and we aim to realize that the robots acquire behavior to avoid walls and other robots. This task in multi-robot environment is more difficult than the task in a single robot environment, due to the increase of the state space and the influence by the actions of other robots. Through the simulation experiment, we confirm that three robots can acquire behavior to avoid walls and other robots by deep reinforcement learning from high-dimensional visual information as input data.

Key Words: Deep Reinforcement Learning，Multi-Robot System，Behavior Acquisition，Wheeled Mobile Robot

1 はじめに

深層強化学習は，深層学習と強化学習を組み合わせた機械学習の手法であり，近年大きな注目を集めている．Deep Q-network (DQN) を用いた人間レベルのゲーム AI の開発 [1] とトッププロ棋士以上の強さを誇る AlphaGo[2] がそのきっかけである．DQN を代表とする深層強化学習の特筆すべき点は，画像そのものを状態とした行動獲得を実現したことである．つまり，深層学習が持つ表現学習（特徴量の自動獲得）能力を活用することで，高次元の画像情報から縮約した表現に変換を行い，強化学習による行動獲得を実現している．

深層強化学習は，多数の試行が容易にできるゲーム AI などのシミュレーション環境での研究が多く，実機環境への応用はあまり進んでいない．深層強化学習を実機ロボットの行動獲得に適用した研究として，Levine ら [3] は様々なマニピュレーションのタスクにおいて，深層学習とマニピュレータの軌道最適化に基づいた強化学習を用いて物体の把持方法を学習する手法を提案しているが，多大な学習コストを要している．丸山ら [4] は，セマンティックセグメンテーションと DQN を組み合わせた手法を生活支援ロボットのナビゲーション問題に適用しているが，実機環境での成功率はまだ高くない．我々 [5] [6] は，実機の車輪型移動ロボットを製作し，ロボットに搭載したカメラによる視覚情報のみに基づき，実機環境において壁や障害物を避ける行動の獲得を DQN により実現している．

上記の研究事例は，いずれも単一のロボットの行動獲得を目指したものである．しかし，将来的には，複数のロボットが存在する環境が，家庭や倉庫，工場，福祉施設などで想定でき，他のロボットとの協調行動の学習が重要となる．深層強化学習をマルチロボット環境に適用した研究として，Preferred Networks 社の分散深層強化学習に関する技術デモ [7] がある．Preferred Networks 社は，複数のロボットが存在するシミュレーション環境と実機環境において，全方位センサや天井に取り付けたカメラを用いて壁や他のロボットの位置や向きなどを計測し，ロボットが互いに避ける行動の獲得を実現した．しかし，ロボットにカメラは搭載しておらず，ロボット自身の視覚情報は用いていない．また，天井にカメラを取り付ける必要があるため，実験可能な場所が限られる．

我々は，既に 3 台の実機の車輪型移動ロボットを製作し，深層強化学習による群ロボットの行動獲得を実現する研究を進めている．その第一段階として，3 台の車輪型移動ロボットが存在するシミュレーション環境において，深層強化学習の代表的な手法である DQN を適用する．具体的には，各ロボットに搭載したカメラによる視覚情報のみに基づき，ロボット間で通信することなしに，各ロボットが個別に DQN による学習を進め，壁や他のロボットを避ける行動の獲得を実現する．また，DQN に対して，経験強化型の強化学習である Profit Sharing 法の導入と，タスクの難易度を段階的に上げながら学習を進めるカリキュラム学習の導入により，群ロボットにおける学習の効率化を実現する．

2 Deep Q-network (DQN)

2.1 DQN の概要

強化学習は，エージェントがある環境内で試行錯誤を繰り返しながら最適な行動を学習する枠組みであり，観測可能な状態 s_t に対してエージェントが行動 a_t をとることで報酬 r_t が確率的に得られる環境で学習を行う．

DQN では，以下の式 (1) で定義される行動価値関数 $Q(s,a)$ を畳み込みニューラルネットワーク (Convolutional Neural Network: CNN) で近似する．

$$Q(s,a) = \boldsymbol{E}[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t = s, a_t = a] \tag{1}$$

これは状態 s において行動 a をとった後に将来にわたって得られる割引累積報酬の期待値を表す．なお，r_t は時刻 t における報酬，γ は割引率を表す．

強化学習は行動価値関数をニューラルネットワークなどの非線形関数で近似しようとすると不安定で発散

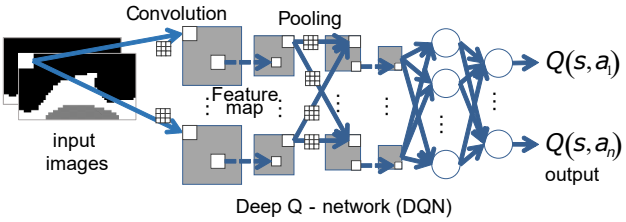


Fig. 1: Structure of DQN

しやすく学習が行いにくい．この原因は，連続する状態 s_t と s_{t+1} の間の相関が高いこと，行動価値関数の更新が方策 π に大きく影響することが挙げられる．この問題を解決するために，Experience Replay を用いて CNN を学習する．Experience Replay では，エージェントは経験 $e_t = (s_t, a_t, r_t, s_{t+1})$ を得る毎に，データセット $D_t = e_1, ..., e_t$ (Replay Memory) へ保存する．学習を行う際は，ミニバッチサイズの個数だけ経験 e_t をランダムに取り出して学習する．

本研究では，TD 誤差 δ_i に対して損失関数 L_i として，以下の Huber 損失 $L_i(\delta_i)$ を使用する．

$$\begin{aligned} \delta_i(\theta_i) &= r_t + \gamma \max_{a'} Q(s', a'; \theta_i^-) - Q(s, a; \theta_i) \\ L_i(\delta_i) &= \begin{cases} \frac{1}{2}\delta_i^2 & \text{if } |\delta_i| < 1 \\ |\delta_i| - \frac{1}{2} & \text{otherwise} \end{cases} \end{aligned} \tag{2}$$

Huber 損失は，二乗誤差関数に比べて，傾きが緩やかであるため，学習が安定する．ここで，θ_i は CNN のパラメータを示す．目標値 $y_i(\theta_i)$ に学習中のパラメータ θ_i を用いると，学習を行う毎に目標値が変化してしまい学習が不安定になる．その対策として，DQN では学習対象の学習ネットワーク (learning network) とは別に教師ネットワーク (target network) を導入し，この教師ネットワークの出力を目標値に用いる．教師ネットワークには，一定ステップ前の学習ネットワークのパラメータ θ_i^- を用いる．損失関数は確率的勾配法の一種である RMSpropGraves[8] を用いて最小化される．RMSpropGraves による更新式を以下に示す．

$$g_{\theta_i} = \frac{\partial L_i(\theta_i)}{\partial \theta_i} \tag{3}$$

$$n_{i+1} = \alpha n_i + (1-\alpha) g_{\theta_i}^2 \tag{4}$$

$$g_{i+1} = \alpha g_i + (1-\alpha) g_{\theta_i} \tag{5}$$

$$\delta_{i+1} = \mu \delta_i - \frac{\eta g_{\theta_i}}{\sqrt{n_i - g_i^2 + \epsilon}} \tag{6}$$

$$\theta_{i+1} = \theta_i + \delta_i \tag{7}$$

DQN で用いる CNN の構造を Fig. 1 に示す．Huber 損失を Fig. 2 に示す．また，DQN の処理手順を Fig. 3 に示す．

2.2 群ロボットへの適用方法

本研究は，同一の環境内に複数のロボットが存在するマルチロボット環境に DQN を適用する．群ロボットに対する適用方法を Fig. 4 に示す．

各ロボットは個別の DQN を持つ．個々のロボットは自律的に行動を決定し，自身の状態や選択した行動を他のロボットに伝えるなどの通信は行わない．各 DQN の入力は各ロボットに搭載したカメラの画像とし，出力はロボットがとり得る行動に対する行動価値関数とする．また，全てのロボットは同時に（同期して）状態の観測や行動の実行を行う．

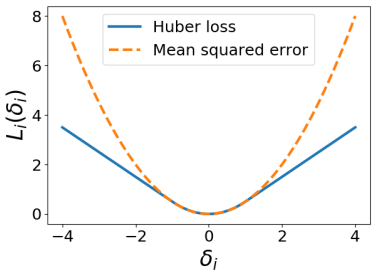


Fig. 2: Huber loss function and mean squared loss function

for episode = 1, M **do**
  Observe initial state s_1
  for t = 1, T **do**
    Determine action a_t using action selection
    Execute action a_t and observe reward r_t
    and next state s_{t+1}
    Store the transition (s_t, a_t, r_t, s_{t+1}) in replay
    memory D
    Sample randomly minibatch of transitions
    from replay memory D
    Calculate the loss function
    Update the parameters of learning network
    $Q(s,a)$ using RMSPropGraves method
    Update target network $\hat{Q} = Q$ every C steps
  end for
end for

Fig. 3: Algorithm of DQN

2.3 Profit Sharing の導入

DQN では行動価値関数の更新を次状態での行動価値関数とそのとき得た報酬のみに基づき行う．特に，行動価値関数の学習がほとんど進んでいない学習初期においては，報酬がほとんど得られないため，有意義な学習ができず，学習の速度が遅い．

そこで，DQN の学習を高速化するための手法として，Profit Sharing 法[9] を適用する．Profit Sharing 法の概要を Fig. 5 に示す．Profit Sharing 法は得られた報酬を過去数ステップに対して分配する手法である．過去 K ステップにわたって報酬を分配する場合，時刻 t に分配の対象となる報酬 r_t が得られた時点で時刻 $t-i$ における報酬 r_{t-i} は以下の式によって更新される．

$$r_{t-i} = r_{t-i} + \beta^i r_t \quad (i = 1 \sim K) \tag{8}$$

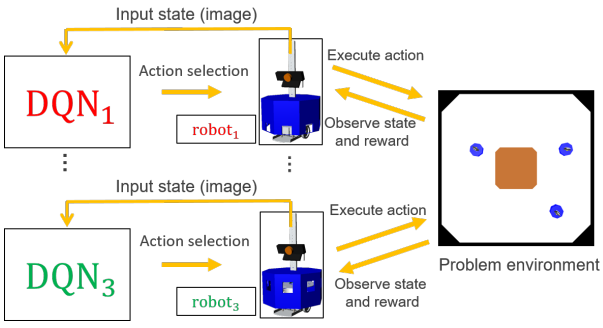


Fig. 4: Method to apply DQN to multi-robot system

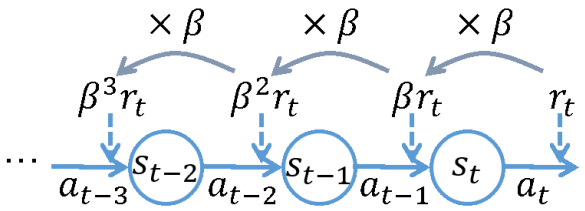
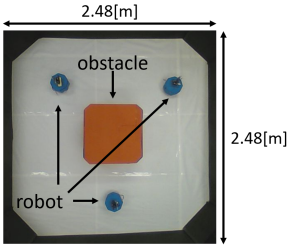
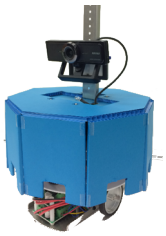


Fig. 5: Framework of Profit Sharing method



(a) Problem environment (from above view)



(b) Wheeled robot

Fig. 6: Problem environment and Wheeled robot

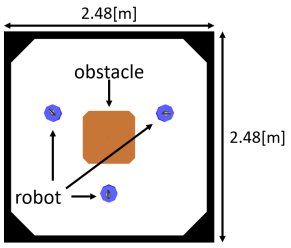
ここで，β は減衰率 $(0 < \beta < 1)$ であり，β をかけることによって報酬を割り引く．本来の報酬を得たステップだけでなく，その過去数ステップについても報酬による更新を行うことができるため，学習の立ち上がりが早くなる．

2.4 カリキュラム学習の導入

群ロボット環境において，最適な行動は他のロボットが選択する行動に依存する．全てのロボットが同時並行で学習を行うとき，常に他のロボットの行動方策が変化するため，問題の難易度が高く，学習が不安定になると考えられる．学習の効率化と安定化を図るため，問題設定（タスク）の難易度を段階的に上げてゆくことで，学習を行う手法であるカリキュラム学習の適用を検討する．初めに，単一のロボットのみが存在する環境で DQN の学習を行う．単一であること以外は群ロボットと同一の問題設定とする．その後，その 1 台が単一の環境で学習した DQN のパラメータを引き継ぐとともに，他のロボットが DQN の学習を行う．その際，一定ステップ数の間，パラメータを引き継いだロボットは学習を行わない．他のロボットの行動方策の変化が緩和され，学習の安定化が期待される．

3 問題設定

本研究では，群ロボットを対象として，実機環境で DQN を用いた学習を行い，他のロボットや壁を避け，大きく進む行動の獲得を目標とする．実機環境における問題環境として，Fig. 6(a) に示すように，縦横 2.48[m] の正方形のコースを準備し，中央に 0.62[m] 四方の障害物を設置した．また，Fig. 6(b) に示す実機の車輪型ロボットを 3 台制作した．各ロボットは同サイズであり，Table1 にその仕様を示す．実機環境での実験の準備段階として，シミュレーション環境での実験を行い，DQN の適用方法や問題設定を検討する．実機環境をもとに，Fig. 7 に示すシミュレーション環境を構築した．シミュレーション環境の構築には，Cyberbotics 社の汎用ロボットシミュレータ Webots[10] を用いた．各ロボットは車輪型ロボットとし，取り付けたカメラの画像のみを用いて行動を決定する．カメラは実機ロボットに



(a) Problem environment (from above view)

(b) Wheeled robot

Fig. 7: Problem environment and Wheeled robot

Table 1: Specifications of Wheeled robot

Diameter of body	22[cm]
Height	34[cm]
Diameter of wheel	4.8[cm]
Camera	1 direction
Distance sensors	8 directions

搭載するカメラと同じ視野角 120 度のカメラを用いる．カメラ画像は，画像中の道の部分を白色，壁の部分を黒色，自身および他のロボットのボディ部分を灰色の計 3 色に変換したものを入力として用いている．ロボットは側面の 8 方向に距離センサを搭載しており，壁や他のロボットとの距離を測定する．距離センサの値は報酬の計算にのみ用い，DQN の入力としては用いない．シミュレーション環境内に 3 台のロボットを置き，個々のロボットが独立して行動を実行する．

3.1 行動と状態と報酬の定義

ロボットは Fig. 8 に示す以下の 5 種類の行動のうち 1 つを行動 a_t として 1 ステップ毎にとる．

$$a_t = \begin{cases} a_1 & (\text{直進：高速}) \\ a_2 & (\text{直進：中速}) \\ a_3 & (\text{直進：低速}) \\ a_4 & (\text{左折}) \\ a_5 & (\text{右折}) \end{cases}$$

ここで，各行動の 1 ステップでの移動距離は以下の通りである．直進 (高速) は 4.8[cm] 前進する．同様に，直進 (中速) は 3.2[cm] 前進し，直進 (低速) は 1.6[cm] 前進する．右折は前に 2.7[cm]，右に 0.12[cm] 進み，ロボットは右に 5.2[deg] 回転する．左折は前に 2.7[cm]，左に 0.12[cm] 進み，ロボットは左に 5.2[deg] 回転する．

報酬は，直前の行動と壁や他のロボットとの距離をもとに，次のように決定する．

$$r = \begin{cases} 15 & : \text{壁などから遠く，行動が前進 (高速)} \\ 5 & : \text{壁などから遠く，行動が前進 (低速)} \\ 10 & : \text{壁などから遠く，行動が上記以外} \\ -20 & : \text{壁やロボットに衝突した場合} \\ 0 & : \text{その他} \end{cases}$$

ここで，壁などから遠い状況とは，自身の側面から壁や他のロボットまでの距離が 10[cm] 以上と全ての距離センサが示した場合とする．壁や他のロボットに衝突した状況とは，自身の側面から壁や他のロボットまでの距離が 1.5[cm] 未満であると 1 個以上の距離センサが示した場合とする．各ロボットは多くの状態で高い

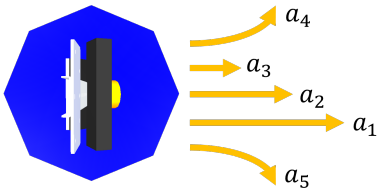


Fig. 8: Candidate actions of Wheeled robot

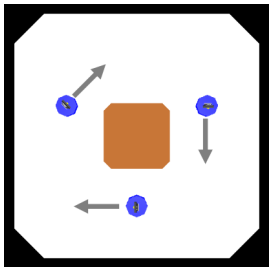


Fig. 9: Starting points and directions of three robots in evaluation (test run) phase

報酬を得るために高速を選択し，他のロボットとの接近時に低速を選択する行動の獲得が期待される．

4 シミュレーション実験

4.1 実験方法

本研究では，3 台のロボットを対象とするが，そのうち 1 台のロボットの速さを変更する．全ての行動において，1 ステップあたりに進む距離を他のロボットの半分とする．行動の種類と報酬の与え方は変更しない．全てのロボットは環境内を時計回りに回る．ロボット間の速さに差があるため，高い報酬を得るために他のロボットを追い抜く行動を獲得することが期待される．壁や他のロボットに衝突した場合には，全てのロボットについて学習を停止し，4.2 節で述べるリスタート行動を行う．このリスタート行動は，実機実験での適用を考慮して，各ロボットが自身のカメラ画像のみから，行動を取ることができるように設計した．

DQN の入力は現在と 1 ステップ前のロボットに搭載したカメラ画像を用いる．入力画像は 48×27[pixel] のグレースケール画像である．畳み込みとプーリングを行う層は 2 層あり，各畳み込み層のフィルタのサイズは 8×8，ストライドは 1，枚数はともに 32 枚である．プーリングは全て 2×2 の領域に対して MAX プーリングを用いる．全結合層のユニット数は 256 個，出力層のユニット数は行動の種類と同じ 5 個である．DQN は，Preferred Networks 社が公開している深層学習フレームワークの Chainer[11] と強化学習用ライブラリの ChainerRL[12] を用いて実装した．

学習時の行動選択手法は ϵ-greedy 法を用いる．ϵ-greedy 法では，小さな値である ϵ の確率でランダムな行動を選択し，残りの確率 $1-\epsilon$ で行動価値が最大の行動を選択する．本実験では，ϵ の値は初期値を 0.1 とし，12,500 ステップで 0 になるように線形に減少させる．

Profit Sharing 法を壁や他のロボットに衝突した時の報酬に対して適用する．衝突から過去 5 ステップに対して，減衰率 $\beta = 0.7$ によって報酬を分配する．

最初の 500 ステップは初期探索として，各ロボットは学習せず，Experience Replay で使用する状態と行動と報酬のデータを収集する．Replay Memory のサイズは

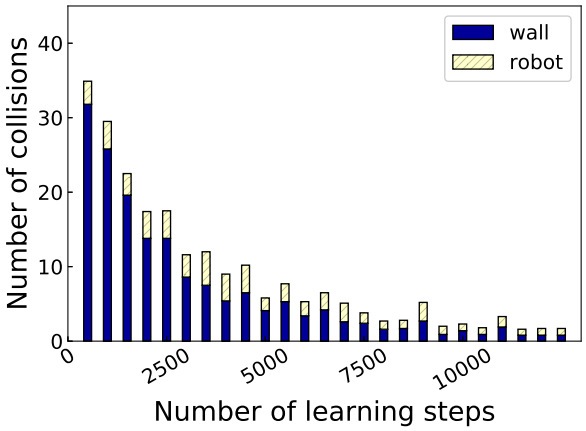


Fig. 10: Change of number of collisions at every 500 steps in learning phase)

100,000 とし，ミニバッチサイズは 32，教師ネットワークの更新間隔は 500 ステップ，割引率 γ は 0.99 とした．RMSpropGraves を最適化手法として用い，パラメータは学習率 $\eta = 0.0001$，$\alpha = 0.99$，$\mu = 0.95$，$\epsilon = 0.0001$ とした．

学習 500 ステップ毎に 10 回のテストを行う．毎回のテスト開始時に各ロボットを Fig. 9 に示す 3 箇所のスタート位置にランダムに置き直す．テスト時は greedy 法を用いて，行動を決定する．1 回のテストの終了条件は，壁や他のロボットに衝突した場合，または衝突せずに 500 ステップ経過した場合とした．

4.2 リスタート行動

学習中にロボットが壁または他のロボットに衝突した際に，ロボットをスタート位置に置き直すのは，実機環境では人間の労力と時間がかかる．そこで，実機環境での適用を考慮して，シミュレーション環境においても，ロボットが自分自身でリスタートする行動をとることを導入した．具体的には，ロボットが壁または他のロボットに衝突した際，ロボットの各側面と壁または他のロボット間の距離について，ロボットの前方右側が最も短くなるまで旋回し，22[cm] の後退を行う．後退後，決められた方向に 90[deg] 旋回する．旋回方向はロボットのカメラ画像によって決まる．画像中央の 75% 以上が茶色の場合は反時計周り，75% 以上が黒色の場合は時計周りとし，それら以外の場合は 8.7[deg] 旋回した後に再度判定を行う．全方向の距離センサが 20[cm] 以上離れていると示した場合リスタートを完了し，そうでない場合はリスタートを再度行う．全てのロボットがリスタートを完了した後に学習を再開する．

4.3 実験結果

DQN による行動獲得実験を 10 セット行った．学習時の 500 ステップ毎の衝突回数の推移 (10 セットの平均) を Fig. 10 に示す．テスト時において，壁や他のロボットに衝突せず連続して行動したステップ数の推移 (10 セットの平均) を Fig. 11 に示す．なお，Fig. 10 と Fig. 11 は，Profit Sharing 法を用いているが，カリキュラム学習は用いていない場合の実験結果である．Fig. 10 の棒グラフ中で塗りつぶした部分は壁との衝突回数

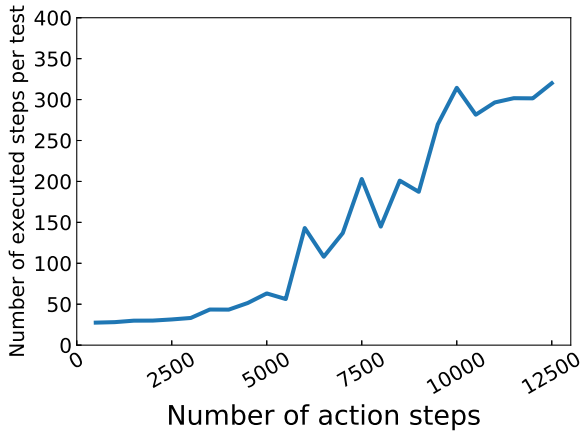


Fig. 11: Change of number of action steps without collision in evaluation (test run) phase

を表し，斜線を引いた部分は他のロボットとの衝突回数を表す．学習が進むにつれ，学習初期に壁への衝突回数が大きく減少し，中盤以降はロボットへの衝突回数も減少している．ロボットは壁を避けて道なりに進むことを学習し，続いてロボットを避けることを学習していると考えられる．学習後期の壁への衝突は，他のロボットを避ける行動によって発生するものである．また，テスト時においても衝突までのステップ数が着実に増加している．テスト時には，Fig. 9 に示す 3 箇所のスタート位置にロボットを置き直し，衝突までのステップ数が 100 ステップ前後でロボット同士の接近が発生するため，学習の後半は他のロボットを回避する行動を獲得できていると考えられる．

学習後のロボットの行動例を Fig. 12 と Fig. 13 に示す．Fig. 12 では後ろの速いロボット B が進路を変更し，前の遅いロボット C を内側から追い抜く行動を獲得している．Fig. 13 は，3 台のロボットが密集し，衝突した行動例である．本研究における報酬設定では，他のロボットが得る報酬は，自身の学習に影響を与えない．Fig. 13 の場合，速いロボット A にとっては，自身以外のロボットの衝突させてエピソードを終えることは，自身の累積報酬を高くすることに繋がる．ロボット間の通信をしない今回の問題設定では，このような他のロボット同士が接近していて衝突する可能性が高い状況では，協調行動を獲得することは本質的に困難であると考えられる．

学習により各ロボットが獲得した行動価値関数について評価する．速いロボット A の Fig. 14 の状態に対する行動価値を Fig. 15 に示す．Fig. 14 の状態において，最も行動価値が高い行動は「右折」であり，前方の遅いロボット C を内側から（右折して）追い抜く行動を獲得していると考えられる．2 番目に高い行動は「前進 (低速) 」であり，減速して前方の遅いロボット C との距離を保つための行動と考えられる カリキュラム学習の有無による学習性能の比較を Fig. 16 に示す．これも 10 セットの実験の平均である．本研究におけるカリキュラム学習では，遅いロボット 1 台がまず単独で DQN の学習を行う．その後に，他の速いロボット 2 台が DQN の学習を進め，6,000 ステップ経過後は遅いロボットも含め，3 台のロボットが学習を進める．Fig. 16

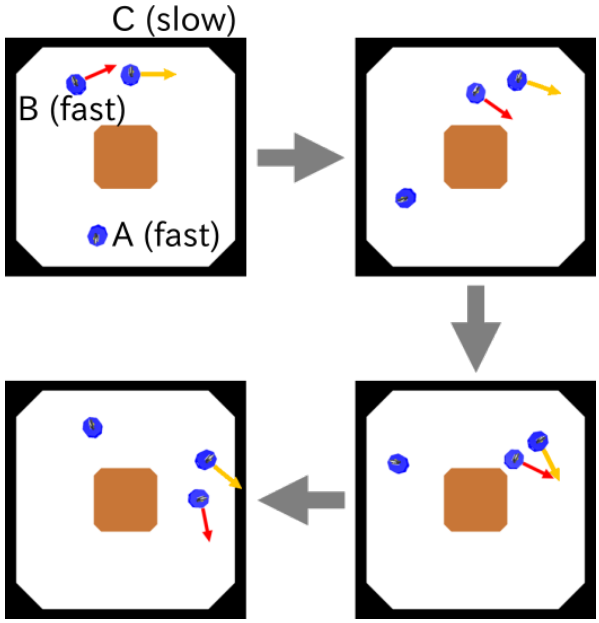


Fig. 12: An example of overtaking behavior after learning (at every 20 action steps)

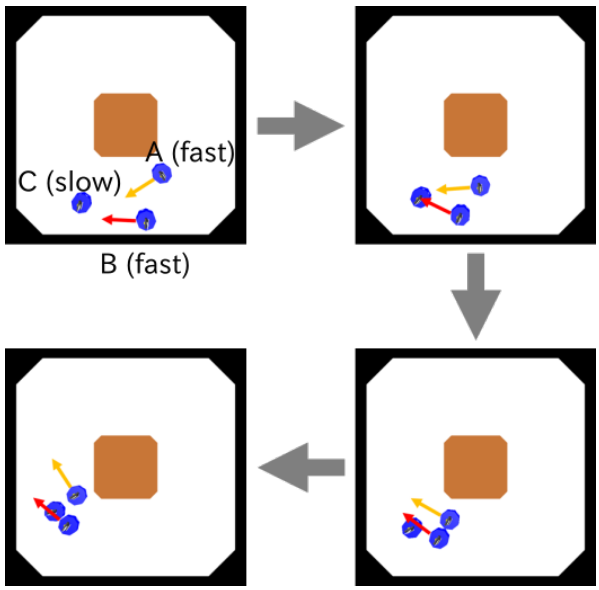


Fig. 13: An example of collision betwwen robots after learning (at every 10 action steps)

より，カリキュラム学習を適用すると，立ち上がりが早くなっていることが分かる．学習以前に遅いロボットが壁を避ける行動を獲得していることを考慮しても，中盤において安定してステップ数が増加している．これは，遅いロボットが方策を固定しているため，速いロボットが獲得すべき行動も限定したものとなり，安定した学習ができているためと考えられる．6,000 ステップを超えると，遅いロボットが学習を始めることによって，学習が不安定になる可能性もある．しかし，実際の学習結果が大きく落ちていないことから，本来の問題設定においても引き続き学習でき，カリキュラム学習の適用が学習の高速化に効果があると考えられる．

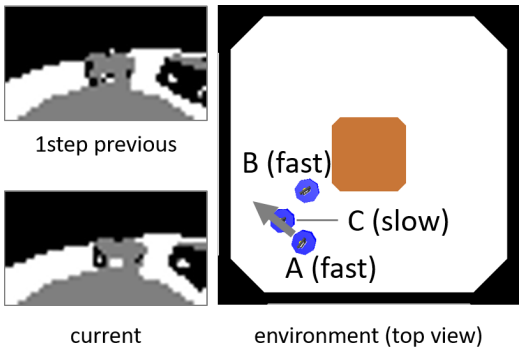


Fig. 14: An example of state of high speed robot A

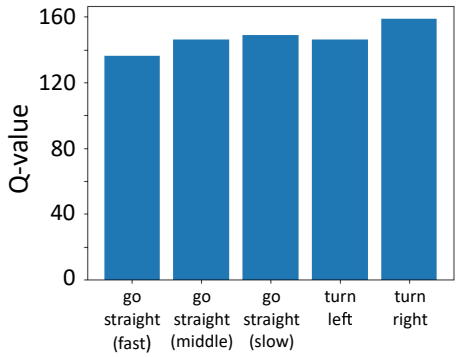


Fig. 15: Action values of high speed robot A at the state shown in Fig. 14

5 まとめ

本研究では，深層強化学習の代表的な手法である Deep Q-network (DQN) を群ロボットに適用し，行動獲得を試みた．各ロボットが，壁や他のロボットを避けて進む行動の獲得を実現した．本研究における具体的な成果は，以下の通りである．

- 3 台の車輪型移動ロボットがいずれも時計方向に回る問題環境において，各ロボットが DQN による学習を行うことにより，壁や他のロボットに衝突する回数が着実に減少することを明らかにした．
- 3 台のロボットのうち 1 台のロボットの速度を他のロボットの半分に設定し，遅いロボットを他の速いロボットが追い抜く行動が獲得できた．なお，3 台のロボットの速度を全て同じとした設定では，等間隔で距離を保ちながら 3 台のロボットが高速で周回する行動が獲得されることも確認している．
- カリキュラム学習を適用し，まず速度が遅いロボット 1 台のみを学習させ，その 1 台のロボットのパラメータを固定させて学習を行うと，ロボットが取るべき行動が一定になり，学習が安定することが分かった．

しかし，現時点では学習の終盤においても，他のロボットや壁に衝突する回数が少し残っている．本研究における問題設定では，ロボット間での通信は行わないため，他のロボットがとる行動を知ることができない．他のロボットが真横から視野内に入ってくることなどがあり，そのような場合の状態（カメラ画像）の遷移は，自分自身の行動だけではなく他のロボットの行動にも依存している．つまり，非マルコフ環境の問題設定となっていると考えられる．そこで，今後は，他のロボットとの通信を取り入れた学習を行うこと，自動車のウィンカーのように選択行動を他のロボットに視覚情報で伝えること，などを検討している．さらに，シミュレーション環境での結果をもとに，実機環境での実験を現在進めている．

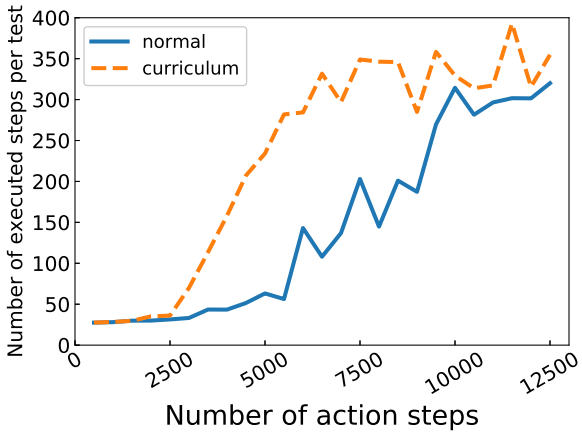


Fig. 16: Comparison of learning performance with and without curriculum learning

参考文献

1) Volodymyr Mnih, *et al.*: Human-level Control through Deep Reinforcement Learning, *Nature*, Vol.518, pp.529-533 (2015)
2) David Silver, *et al.*: Mastering the game of Go with deep neural networks and tree search, *Nature*, Vol.529, pp.484-489 (2016)
3) S. Levine, *et al.*: End-to-End Training of Deep Visuomotor Policies, *Journal of Machine Learning Research*, Vol.17, pp.1-40 (2016)
4) 丸山祐矢，古川弘憲，村瀬卓也，山内悠嗣，山下隆義，藤吉弘亘：「セマンティックセグメンテーションを用いた深層強化学習による自律移動の獲得」，第 35 回日本ロボット学会学術講演会 (2017)
5) H. Sasaki, T. Horiuchi and S. Kato: Experimental Study on Behavior Acquisition of Mobile Robot by Deep Q-network, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, Vol.21, No.5, pp.840-848 (2017)
6) 加藤大地，佐々木 光，堀内 匡，青代敏行：「深層強化学習の実機ロボットへの応用：車輪型移動ロボットの行動獲得の実現」，計測自動制御学会 第 45 回知能システムシンポジウム資料，C3-4 (2018)
7) https://research.preferred.jp/2015/06/distributed-deep-reinforcement-learning/
8) Alex Graves: Generating sequences with recurrent neural networks, arXiv:1308.0850 (2016)
9) 宮崎和光，木村 元，小林重信：「Profit Sharing に基づく強化学習の理論と応用」，人工知能学会誌，Vol.14, No.5, pp.800-807 (1999)
10) https://www.cyberbotics.com/
11) https://chainer.org/
12) https://github.com/chainer/chainerrl
13) 綿貫零真，堀内 匡，青代敏行：「深層強化学習を用いた群ロボットの行動獲得の試み」，第 28 回インテリジェント・システム・シンポジウム講演論文集，pp.125-126 (2018)