Vol.2019-ICS-193 No.10
2019/2/27

RoboCupサッカー2Dシミュレーションの守備行動での強化学習における報酬設計の検討

阿部 宇志[1,a)]　清 雄一[†1,b)]　田原 康之[†1,c)]　大須賀 昭彦[†1,d)]

概要：近年, 世界で行われているサッカーの試合において, セットプレーの重要性が高まっており, 攻撃やその攻撃に対する守備戦略においてアイデアが求められている. セットプレーにおいては, 攻撃に関して様々な工夫をもたらしてゴールを奪い取るチームが見受けられるが, それに対して最善の守備行動について確立されたチームは多いとは言えない. そこで本稿では, ロボットで行うサッカー大会として知られるRoboCup サッカーのプログラムを使用し, コーナーキックの守備における最善行動についての研究を行った. ここでは, それぞれのエージェントがコーナーキックの守備において点を取られないことに適した行動を選択するようにするため, 方策オン型 TD 学習と分類される強化学習の一つである Sarsa[3] のアルゴリズムを用いて, 提案するプログラム設計と先行研究でのプログラム設計との比較を行うこととした. また, Q 値を選手間で共有することや, ヒューリスティクスを導入した強化学習 [2] を行うことで, 学習効果向上を検討する実装も行った.

1. はじめに

近年, サッカーにおいて CK, FK, PK といったセットプレーが重要視されてきている. スポーツのデータ分析を手掛ける『Opta』[7] によれば, 2018 年に行われた W 杯では全 169 ゴールのうちセットプレーでの得点が 73 ゴールで総得点の約 43%となっており, 各チームがセットプレーにおいて工夫を凝らすことが増えてきている. しかし, 多くのチームがセットプレーにおける攻撃を工夫する反面, 守備に対する知見が多くなく, 目を向けていく必要があるといえる. ロボカップにおいても, パスを回すことや, 得点を多くとることなどの攻撃に関する研究は多くなされているが, ボールを奪うことやゴールを守ることなどの研究は攻撃に比べると多くはない. さらに, セットプレーにおける場面を限定した研究はさらに少ないため, 研究の価値がある分野となっている.

また, 今回はセットプレーでの守備行動の最善行動を得るために, 強化学習のアルゴリズムを用いた. スポーツでの事例は多いとは言えないが, 囲碁や将棋, チェスといったゲームの中で過去のデータを活用した機械による学習が注目を集め, 人間も強化学習の結果から多くの知見を得ている. スポーツにおいても, 機械での学習の研究成果が人間の知見として得られるようになれば, 戦略の幅も増えることになるはずである.

本稿では, そういったセットプレーにおける守備やロボットの学習における研究の重要性から, 強化学習の中でも学習の性能に関わる重要な要素である報酬設計に着目し, RoboCup サッカー 2D シミュレーションの守備行動での強化学習における報酬設計の検討を行った.

2. RoboCup サッカー

2.1 RoboCup サッカーとは

ロボカップサッカーは, 西暦 2050 年までにサッカーの世界チャンピオンチームに勝てる完全自律型ヒューマノイドロボットのチームを作ることを目標とし, 人工知能やロボット工学など様々な分野の技術の推進を目的としたロボットのサッカー大会である [8]. ロボカップサッカーの中でも, コンピュータの仮想フィールド上で試合が行われるシミュレーションリーグ, 小型ロボットリーグ, 中型ロボットリーグ, 4 足ロボットリーグ, 2 足歩行のロボットで行われるヒューマノイドリーグ, Mixed Reality（マイクロロボット）リーグに分かれており, 本稿ではその中でもロボカップ最古参の歴史を持つ, 2D シミュレーションリーグでのチーム開発を行った.

[1] 電気通信大学情報理工学部総合情報学科, Chofu, Tokyo, 182–8585, Japan
[†1] 現在，電気通信大学大学院情報理工学研究科情報学専攻, Chofu, Tokyo, 182-8585, Japan
[a)] abe.takashi@ohsuga.lab.uec.ac.jp
[b)] sei@is.uec.ac.jp
[c)] tahara@is.uec.ac.jp
[d)] ohsuga@uec.ac.jp

2.2 RoboCup サッカー 2D シミュレーション

RoboCup サッカーシミュレータは 1993 年に野田五十樹氏によって初めて開発され, オープンソースのプログラムとして無償で利用することができる [11]. シミュレータはいくつかのプログラムが合わさって動いており, 実際のシミュレーションは rcssserever と呼ばれるサーバプログラムによってなされている.

シミュレーション実行時の各プログラムの関係は図 1 のようになる. エージェントはエージェント自身が首を振ることで得られる知覚情報のメッセージを rcssserver から受け取り, エージェント自身が行いたい行動コマンドのメッセージを rcssserver へ送信する. この送受信の繰り返しによってフィールド上の状態が変化し, シミュレーションが進行していく. エージェント間の情報共有は rcssserver を介してのコミュニケーションでのみ成り立っており, 全てのエージェントは独立して情報の通信が行われる.

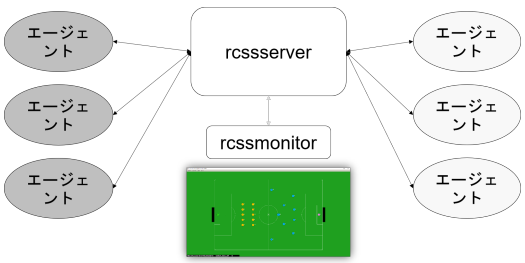

図 1　シミュレータの仕組み

また, rcssserver は離散時間シミュレータとなっており, 100 ミリ秒に 1 回, 物体の位置が更新される. サッカーのフィールドは図 2 のように x 座標と y 座標でもって表され, ボールの位置, エージェントの位置も座標で示される.

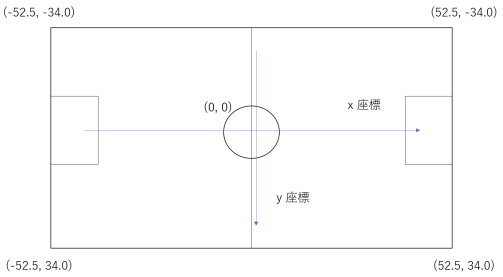

図 2　rcssserver の座標系

2.3 参考プログラム

本研究では, 『Java でつくる RoboCup サッカー選手プログラム』(大島真樹著) [9] に付属されているプログラムを改良して実装を行った. このプログラムは 1 つ 1 つの行動をプログラムごとに分け, それらを組み合わせて 1 体のエージェントをフィールドに表示させている. エージェントの行動として, 決められた守備位置に移動することやボール座標を特定し, その座標に向かうことなどの基本的なものが付属されている. 主たる守備行動のパスカットや敵のマークなどの行動は付属されていないため, 行動は自らのプログラムによって行うこととしている. また, CK の守備に対して, 攻撃側のチームはオープンソースのバイナリファイルとして公開されているものから秋山英久氏によって開発されたチームの HELIOS2016 を使用した.

2.4 関連研究

RoboCup で強化学習を使った考え方の中で最も多く使われているタスクが Stone ら [4] や荒井ら [10] により研究が行われている keepaway タスクである. このタスクにおいては, 20×20 の正方形の領域内で, keepers チームと takers チームに分かれ, keepers チームが takers チームにボールを奪われないようパスを回す. また, keepers チームである keeper は単独でボールキープを行うことはできず, パスを回し続けなければいけないというマルチエージェントタスクとなっている. この keepaway タスク内では学習アルゴリズムとして Sarsa[3] をベースとしたアルゴリズムを採用しており, 学習時間が経過するにつれてパスがつながれる時間が増加していく. 本研究でもこれに倣い, 学習アルゴリズムとして Sarsa を採用して強化学習を行った.

また, 強化学習にヒューリスティクスを導入したエージェントの守備行動に関する研究 [1] がなされている. これは Sarsa のアルゴリズムに類似した Q-Learning[6] のアルゴリズムを使用された研究となっており, エピソードが経過するごとにゴールされる回数が減少していく. ここで比較対象とされていた Q-Learning でのアルゴリズムを本稿では比較対象として使用し, 実装条件を参考とした.

3. 強化学習の手法

3.1 Sarsa

ここで本稿で使用したアルゴリズムである Sarsa 法 [3] について触れる. Sarsa は, State, Action, Reward, State(next), Action(next) の頭文字をとった方策オン型 TD 学習と分類される強化学習である. ある状態 s において行動 a をとる有効性を行動価値関数 $Q(s,a)$ を用いて評価している. また, ここで使われている $Q(s,a)$ はある状態 s において行動 a を選択することの良し悪しを比較する判断材料とされており, Q 値と呼ばれている. 時刻 t において, 状態 s_t のエージェントが状態 s_{t+1} に行動 a_t をとって遷移したとき, Sarsa では Q 値を以下の式で更新する [12].

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha[r + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)] \quad (1)$$

α は学習率で, $0 \leq \alpha \leq 1$ の中で重みを付けることで, 過去の学習への重みづけを行う. r はある状態で行動をとった際の報酬を示している.

Sarsa のアルゴリズムを以下に示す [5].
(1) 全ての Q 値を初期化する.
(2) 初期状態を観測する.
(3) 行動選択法により行動 a_t を選択する.
(4) 行動 a_t を実行し, 報酬 r を受け取る.
(5) 遷移後の状態 s_{t+1} を観測する.
(6) 行動選択法により行動 a_{t+1} を選択する.
(7) 式（1）によって Q 値を更新する.
(8) s_t に s_{t+1}, a_t に a_{t+1}, t に t+1 を代入する.
(9) (4) から (8) を繰り返し, エピソード（試行）の終了ならば繰り返しを終了する.
(10)学習終了まで (2) から (9) を繰り返し, 学習終了条件になった場合, 繰り返しを終了する.

3.2 Q-Learning

Sarsa と類似した考え方に Q-Learning[6] がある. Sarsa との違いとして, Q 値を実際に起こった次の行動の Q 値ではなく, 次の状態におけるそれぞれの行動の Q 値の中における最大値を用いて更新していくアルゴリズムとなっている. 時刻 t において, 状態 s_t のエージェントが状態 s_{t+1} に行動 a_t をとって遷移したとき, Q-Learning では Q 値を以下の式で更新する.

$$Q(s_t,a_t) = Q(s_t,a_t)+ \alpha[r+\gamma max_{a'\in A(s')}Q(s_{t+1},a_{t+1})-Q(s_t,a_t)] \quad (2)$$

Q-Learning のアルゴリズムを以下に示す [5].
(1) 全ての Q 値を初期化する.
(2) 初期状態を観測する.
(3) 行動選択法により行動 a_t を選択する.
(4) 行動 a_t を実行し, 報酬 r を受け取る.
(5) 遷移後の状態 s_{t+1} を観測する.
(6) 式（2）によって Q 値を更新する.
(7) s_t に s_{t+1}, t に t+1 を代入する.
(8) (3) から (7) を繰り返し, エピソード（試行）の終了ならば繰り返しを終了する.
(9) 学習終了まで (2) から (8) を繰り返し, 学習終了条件になった場合, 繰り返しを終了する.

3.3 行動選択法について

先述したような強化学習のアルゴリズムでは, エージェント自身が行動を選択する際に行動選択法を指定する必要がある. Sarsa や Q-Learning では多くの場合 ϵ-greedy 法を用いて強化学習を行う. ϵ-greedy 法とは, 確率 ϵ で行動をランダムにとり, それ以外の確率ではその状態における Q 値が最大である行動を選択する行動選択法である. もし, ある時間 t における状態 s_t において最大の行動価値関数である $maxQ(s_t,a)$ をもつ行動 a のみを選択する行動選択法の場合, それ以外の行動で得られる報酬を学習することができない. よって, ϵ の確率でランダムな行動を選択する手法をとるのが一般的である. また, ϵ は, $0 \leq \epsilon \leq 1$ で設定され, ϵ を固定する場合や学習が進むにつれて ϵ が減少していく場合も多く, 学習条件によって設定の仕方は様々である.

3.4 Heuristics Accelerated Q-Learning(HAQL)

Bianchi ら [2] によって Q-Learning をベースとし, さらにヒューリスティクスを適用した Heuristics Accelerated Q-Learning (HAQL) といった手法が研究されており, 前述の Luiz らによる研究では, この手法を用いてエージェントの守備行動の強化学習を行っている. この研究では, 学習の高速化を狙うため, ドメイン知識をもった開発者によってある状態において行動選択の優先度の高い行動 a' に対し, あらかじめ行動 a' をとりやすいようにする工夫がされている. 学習が進んで Q 値が大きくなっていくにつれて, ドメイン知識よりも Q 値による影響を大きくしていく行動選択法である. これらを実現するために, この手法では ϵ-greedy 法に新たな条件を課す. ϵ-greedy 法では, 確率 ϵ で行動をランダムにとり, それ以外の確率ではその状態における Q 値が最大となっている行動を選択する行動選択法であった. しかし, ここでは Q 値によって行動選択するのではなく, ある時間 t における状態を s_t, 選択されうる行動を a_t, ヒューリスティクスによる影響を ξ とした場合の $Q(s_t,a_t)+\xi H_t(s_t,a_t)$ で表される値の最大値をもつ行動を選択する. $H_t(s_t,a_t)$ はある状態 s_t において, よりよい行動だと思われる行動 a_t に関して, 以下の式で求める. それ以外の行動に対しては $H_t(s_t,a_t)=0$ とする.

$$H_t(s_t,a_t) = max_a Q(s_t,a) - Q(s_t,a_t) + \eta \quad (3)$$

ここで行動 a はある状態 s_t において考えられる各行動の中で Q 値の最大値をもつ行動を指す. Bianchi らによる研究では, $\eta = 0.01$ とされている.

このように, 行動選択法にヒューリスティクスを用いることで, 開発者のドメイン知識が正しく, 適切に条件が当てはめられていれば, 学習は高速化し, 適切な条件をあてはめることができなかった場合, 逆に学習の速度は遅くなるという性質を持っている.

4. 提案概要

4.1 プログラム構成

提案概要を述べる前に, 本研究におけるプログラム構成について触れておく. 本プログラムでは, Sarsa をベースとしてプログラムの作成を行った. また, エージェント自身が得た情報から報酬を決定しているため, Q 値の更新はエージェント自身のプログラムによって行われている. ロボカップでは, 遠くの物体の認識に対して誤差を含んだり, ボールが見えないと位置を把握できないなど, フィールド

の完全な状態をエージェントが得ることができなくなっている. そのため, チーム全体で報酬が与えられる時間と報酬の値を一定にできるように, エージェント自身が各時間における状態と行動, そして得られた報酬を一時的に保存し, エピソード終了時に Q 値を更新することとした. なお, Q 値は後述の状態表現 1296 個 × 行動集合 4 個の配列によって保持され, データとして保存されている. また, 後述のように本プログラムではボールに向かって動くなどのあらかじめ定められたマクロ行動によって行動選択を行う. 観測する状態に変化が見られたら, ϵ-greedy 法によって行動選択を行うこととし, 変化が見られない場合は時間が 1 つ前の行動を選択することとした. 以下に, 本プログラムで実装したアルゴリズムを示す.

(1) 全ての Q 値を初期化.
(2) 状態 s_t を観測し, 状態を保存.
(3) 時間が 1 つ前と状態が違う場合, ϵ-greedy 法により行動 a_t を選択し, 時間が 1 つ前と状態が同じ場合, 時間が 1 つ前の行動と同じ行動を選択する. (初期行動は ϵ-greedy 法により行動 a_t を選択.)
(4) 行動 a_t を実行し, 報酬 r を受け取り, その時間における行動と報酬を保存.
(5) (2) から (4) を繰り返し, エピソード (試行) の終了ならば繰り返しを終了する.
(6) それぞれの時間 t において保存されている状態 s_t, 行動 a_t, 報酬 r のデータで式 (1) を用いて Q 値を更新し, 保存する.
(7) 学習終了まで (2) から (5) を繰り返し, 学習終了条件になった場合繰り返しを終了する.

4.2 状態表現

前述した keepaway タスクにおいては, 各ステップにおいて各選手間との距離, 他の選手同士の距離, 味方と敵の距離, 敵とパスコースの間の角度を使用して状態を表現している. 本研究では, 以下のような距離を用いて状態を表現している.

- ゴールとボールの距離
- ゴールと自分の距離
- ボールと自分の距離
- マークしている敵と自分の距離

これらのそれぞれを 6 つの状態に分け, $6^4 = 1296$ 個の状態にそれぞれの状態は分類される. また, 第 2 章で述べたように, ロボカップにおいてはエージェント自身が首を振って得られた知覚情報によって状態を把握している. 本プログラムでは, エージェント自身によって得られる情報によって状態を決定しているため, ボールや敵の座標を把握できない場合が存在する. ゴールとボールの距離, ゴールと自分の距離, ボールと自分の距離では座標を把握できないという状態をそれぞれ分割された 6 状態のうち 1 状態とすることとした. よって, マークしている敵と自分の距離は距離を 6 分割, それ以外の状態変数では距離を 5 分割し, 対象の座標を把握できないという 1 状態を加えて 1 状態変数において 6 状態を表現する.

4.3 行動集合

ロボカップでは, エージェントの基本行動として体の向きを変える turn コマンドや体が向いた方向に動く dash コマンド等が使われている. 前述した keepaway タスクでは, こういった基本行動から学習するわけではない. ここで基本行動から学習を行うことは, 膨大な学習時間を要するため現実的ではない. そこで keepaway タスクでは, あらかじめ開発者によって作られたマクロ行動によって学習している. ここではこのマクロ行動を行動集合と呼ぶことにする. この行動集合は, 行動によってかかるステップ数の長さが変化する. 本研究では, これに倣い, 以下のように行動集合を設計した.

ボール奪取　ボールがある座標を探し, 移動する.
マーク　エージェントごとに指定された相手選手の座標を探し, 指定された座標に移動する.
パスカット　ボールと敵の間の座標を計算し, その座標に移動する.
シュートブロック　ボールとゴールの間の座標を計算し, その座標に移動する.

マークは敵選手の近くであり, かつゴールとの間に身を置くことを条件としており, パスカットとシュートブロックでは現在の自らの座標とそれぞれの物体間の直線までの最短距離を計算し, その座標に身を置くことを条件としている. この行動集合は, 観測した状態が変化するまでを一区切りとしているため, 各行動集合の長さはあらかじめ決められているわけではない. よって状態が変わらなければ長く行動集合を取ることになり, すぐに変化すれば短い行動集合となる.

4.4 報酬設計

強化学習において, 機械が学習するための材料となる報酬設計の重要度は高い. 本研究の比較対象としてあげた, Luiz らによる研究 [1] で取り扱われていたプログラムではボールを蹴った場合, または GK にパスをした場合 +15, 相手選手にボールを蹴られた場合 −10, 相手にゴールを決められた場合 −15 の報酬を与えている. 本研究では, 先行研究と違い, Java で構築された簡易的なチームプログラムを使用している. そのため, 今回 CK の対戦相手となる HELIOS2016 と対戦した際, ゴールされる回数が多くなること, ボールに触れる回数が少なくなること, 相手にボールを触られる回数が多くなることから, 報酬設計に工夫がいる条件となっている.

本稿では, エピソード終了時, 自チームがボールを蹴っ

たときに +50, 相手にゴールを決められたときに −15 の報酬を与えることとする. Luiz らによる Q-Learning のチーム [1] との違いとして, 報酬では相手がボールに触れることを考えないようにしている. これはサッカーの守備において, ボールを多く回されたとしてもゴールにボールが入らなければ良いとされることからこのように報酬設計を行った. また, ゴールを決められることが多いことから, 報酬のバランスをとるために, エピソード成功時の報酬を重くするような設計を行った.

4.5 プログラムに追加した特徴

本研究では 2 つのパターンで実装を行った. 1 パターン目は, 先行研究の報酬設計から報酬のみを変更し, Sarsa を用いてプログラムしたチーム. 2 パターン目は, 1 パターン目の報酬設計に加え, 後述の 2 点の特徴を追加したプログラムを用いたチームである. 1 パターン目で, 報酬設計における効果の比較をすると共に, 2 パターン目でその他の要素が学習に与える影響も検討していくこととする. ここでは, 2 パターン目のチームプログラムにおける特徴を述べたい.

1 点目の特徴としては, 初期配置のポジションでエリアを分け, エリアごとで Q 値を共有することで, エージェント自身の学習だけでなく, 他のエージェントの学習した内容を知ることができるようにするという違いである. サッカーではポジションごとで課されている責任は変わる. 例えば一番相手ゴールに近い FW の選手は守備をすることなく, 前線に残り続ける場合も条件によっては考えられる. また, エリアでも優先すべき行動は変わるが, 大まかに優先すべき行動は決められている. 自陣ゴール前では自らのマークに見切りをつけ, シュートのブロックに動く行動が最善の場合もあれば, タッチライン際ではマークを優先し, ボールが近くても, たくさんの選手で追わない工夫がされるなど, 様々な対応が考えられる. そのためポジションではなく, 初期位置によって共有することで学習効果向上が見込めると考え, エリアごとの Q 値の共有を行っている.

2 点目の特徴としては, ヒューリスティクスを強化学習に導入したことである. ヒューリスティクスを導入することで, 最適な条件提示ができれば強化学習の高速化を見込める. サッカーのような協調行動の場合, 学習初期でランダムに行動を選ぶ方法をとると, チームとして機能せず, 学習に影響が出てしまう. その点, サッカーの知識を含んだ条件指定をすることで学習をスムーズに行える. ねらいの 1 点目として述べた Q 値を共有するというプログラムの構造からも学習速度は向上すると考えられる.

5. 実装

5.1 実装結果

今回の研究では, Luiz らによる研究 [1] の報酬設計で Q-Learning のアルゴリズムを使用したチームをチーム Q-Learning と呼び, 報酬設計の比較対象として, 提案の報酬設計を用いて Sarsa のアルゴリズムを使用したチームをチーム提案 A, 報酬以外の要素の検討のための比較対象として, ヒューリスティクスを用いて Q 値を共有したチームをチーム提案 B と呼び, 比較することとする. 守備時間の開始はコーナーキックを蹴る時間からとしており, あらかじめエージェントごとにマークする敵の背番号を割り振り, 初期状態で敵のマークについた状態から始めることとした. チーム提案 B では, エリアごとで Q 値を更新しており, コーナーアークから近い 6 選手のサイドプレイヤーで Q 値を共有し, 他 4 人のフィールドプレイヤーをセンタープレイヤーとして Q 値を共有することにした. 初期配置は以下の図の通りで, 青チームが実装したチーム, 黄チームが HELIOS2016 である.

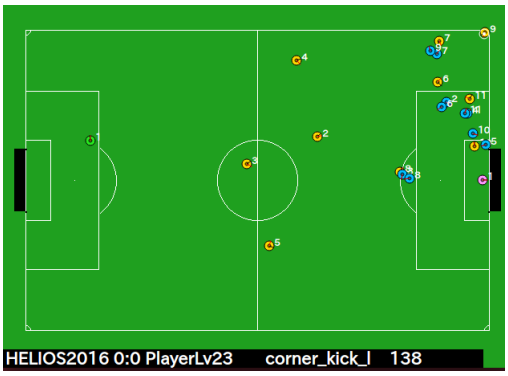

図 3 エージェントの初期配置

実装の条件として, すべてのチームにおいて, 状態表現, 行動集合は同じとしている. また, エピソードの終端をゴールキック, キックインなどのライン外にボールが出た場合, 相手のファールなどによってプレーが中断した場合, 自チームのいずれかのエージェントがボールを蹴った場合, または相手にゴールを決められた場合としており, ゴールを決められた場合以外を守備の成功と見なすこととした. 1000 エピソード繰り返し HELIOS2016 と CK で対戦させ, ゴールが決められたかどうか, ゴールされるまでの時間, 100step 以上の守備時間の回数を比較内容とすることとした. チーム Q-Learning においては Luiz らによる研究 [1] と同じ条件として学習率 $\alpha = 1.25$, チーム提案 A とチーム提案 B においては学習率 $\alpha = 0.8$ で実装を行った. また, 割引率 $\gamma = 0.9$, $\epsilon = 1/(episode + 1)$ と設定し, チーム提案 B においてはヒューリスティクスのパラメータを $\xi = 0.5$, $\eta = 0.01$ とし, 実装結果を比較している.

以下, 図 4-6 で実装結果を示す. 図 4 では, 100 エピソードごとにおける守備成功回数を示した. 前述のとおり, エピソードの終端をライン外にボールが出た場合, プレーが中断した場合, 自チームのエージェントがボールを蹴った場合, または相手にゴールを決められた場合としており, 相

手にゴールを決められなかった場合, そのエピソードにおいて守備の成功とみなす. 縦軸を 100 エピソードごとにおける守備成功回数, 横軸をエピソード数とする.

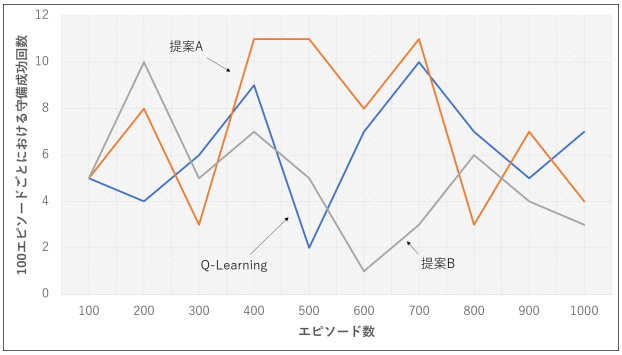

図 4 100 エピソードごとにおける守備成功回数

次に, 以下図 5-6 で守備失敗時における守備時間を示す. 当然, サッカーにおいては点を取られないことが最重要であるが, 本稿では比較検討の一つの指標として用いることとする. 図 5 では, 100 エピソードごとにおける守備失敗時の守備時間が 100step を超えた回数を示す. 縦軸を 100 エピソードごとにおける守備失敗時の守備時間が 100step を超えた回数, 横軸を経過したエピソード数とする.

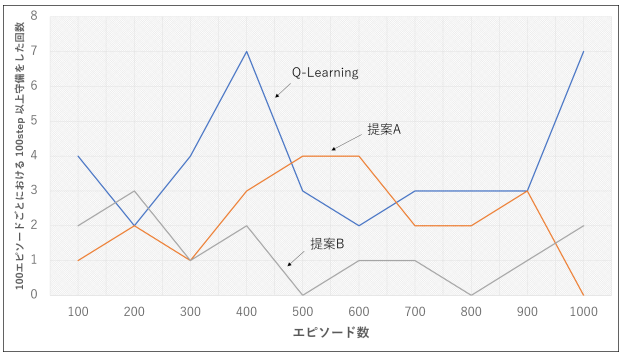

図 5 100 エピソードごとにおける 100step 以上守備をした回数

図 6 では, 20 エピソードごとにおける平均守備時間を示した. 縦軸がゴールを入れられたときの 20 エピソードごとにおける守備失敗時の平均守備時間で, 横軸を経過したエピソード数とする. ここでは, 守備成功時を除いた時間のみを使用する.

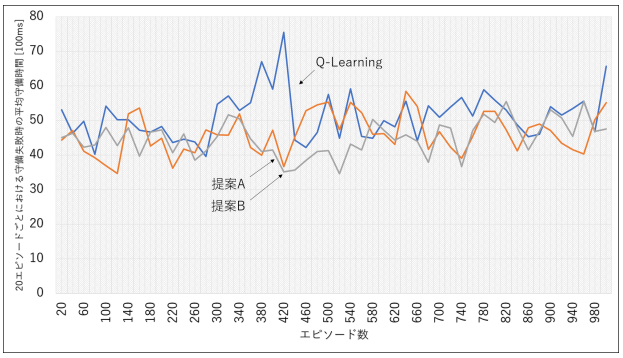

図 6 20 エピソードごとにおける守備失敗時の平均守備時間

5.2 考察

図 4 を見ると, チーム提案 A がチーム Q-Learning よりも守備の成功回数が多いことが見て取れる. チーム提案 A では, エピソード数が 300 を超えると守備の成功回数が増えている. 実際に試合を見ると, 提案 A のエージェントは, 相手が CK を蹴った瞬間にボール方向に向かい, ある一定の距離になるとパスカットを狙う行動をとっていた. これは, 報酬設計の差から見ることができる行動である. 提案 A の報酬設計では, 相手にボールを触られることを考えず, ゴールを入れさせないことを優先させている. 結果として, CK を蹴ったあとに他の選手へのパスコースを防ぎ, 相手のキックミスを誘っている. 全エピソードの守備成功回数を比較しても, チーム Q-Learning では 62 回, チーム提案 A では 71 回と, 守備の成功回数は上回っている.

しかし, チーム提案 A では, 最初にパスカットを狙う行動をとるねらいをチームとして持っているが, そこで相手がミスを犯さなかった場合, すぐに点を取られてしまうケースが多く見られた. 図 5 を見ると, 100step 以上の守備時間をもった回数では提案プログラムは, 比較対象に大きく劣る結果となってしまった. 学習後のチーム Q-Learning では, 相手にボールを触らせる回数を減らすべく各選手にマークにつき, ボールに近い選手はパスカットを狙う行動をとっていた. 図 6 を見ても, 全体的な守備時間でも比較対象に劣っており, 一度広いスペースにボールを繋がれてしまうと, 簡単にゴールへと結びつけられてしまっていたため, 報酬設計により大きな差がつくことがわかる.

また, ヒューリスティクスにおける実装では, 各比較内容においてチーム Q-Learning に劣ってしまっている. ヒューリスティクスの設計は, 正しいドメイン知識をもって行動選択の優先順位の優劣をつける必要があるが, 不適切に優劣をつけた場合, 学習に悪影響が出る. チーム提案 B では, シュートブロックが他の 2 チームと比べて非常に多く選択されていた. それにより敵のマークが手薄になり, パスがつながれてしまうケースが多く見受けられた. あらかじめ指定した条件の中に, ゴールとボールの距離が一定以下になった場合, シュートブロックを選択しやすくしていた. 今

回のヒューリスティクスでは, 簡易的な設計で, 状態分類も大雑把なものとなっていた. 細密な状態設定によって学習に好影響を与える可能性は十分に考えられる.

一般的に, サッカーにおけるコーナーキックの守備では, ボールがゴールに入る回数を少しでも減らすことが目的である. 今回の実装結果から, コーナーキックの守備においては, 相手に多くボールを触られることは結果を左右する要因ではないと考えられる.

6. おわりに

本稿では, RoboCup2D シミュレーションでのセットプレーにおける守備行動で Sarsa を使用し, Luiz らによる先行研究 [1] の報酬設計から相手にボールを蹴られる罰を省いた報酬設計で実装した. 実装の結果として, ボールを相手に多く触られることは, コーナーキックの守備において直接的な影響を及ぼさないことがわかった. さらに守備行動を改善させていくためには, チーム全体がうまく機能するための指標をシンプルに報酬として与えられるようにする必要がある.

謝辞 本研究は JSPS 科研費 JP16K00419, JP16K12411, JP17H04705, JP18H03229, JP18H03340, JP18K19835 の助成を受けたものです.

参考文献

[1] Luiz A. Celiberto Jr., Carlos H. C. Ribeiro, A. H. R. C. A. C. B.: Heuristic Reinforcement Learning Applied to RoboCup Simulation Agents, *RoboCup 2007: Robot Soccer World Cup XI*, 2007, pp. 220–227.
[2] R. A. C. Bianchi, C. H. C. R. and Costa, A. H. R.: Heuristically Accelerated Q-Learning: a new approach to speed up reinforcement learning, *Lecture Notes in Artificial IntelligenceI*, 2004, pp. 245–254.
[3] Rummery, G. A. and Niranjan, M.: On-Line Q-Learning Using Connectionist Systems, Technical report, 1994.
[4] Stone, P., Sutton, R. S., and Kuhlmann, G.: Reinforcement Learning for RoboCup-Soccer Keepaway, *Adaptive Behavior*, Vol. 13, No. 3(2005), pp. 165–188.
[5] Sutton, R. S. and Barto, A. G.: *Introduction to Reinforcement Learning*, MIT Press, Cambridge, MA, USA, 1st edition, 1998.
[6] Watkins, C. J. C. H.: *Learning from Delayed Rewards*, PhD Thesis, King's College, Cambridge, UK, May 1989.
[7] フットボールチャンネル: セットプレーから合計 73 得点。VAR 導入も影響して大幅に記録更新【ロシア W 杯】, https://www.footballchannel.jp/2018/07/16/post281822/ , (参照 2019 年 2 月 4 日).
[8] ロボカップ日本委員会: About RoboCup, http://www.robocup.or.jp/original/about.html , (参照 2019 年 2 月 4 日).
[9] 大島真樹: *Java* でつくる *RoboCup* サッカー選手プログラム, 森北出版, 2005.
[10] 荒井幸代, 田中信行: マルチエージェント連続タスクにおける報酬設計の実験的考察, 人工知能学会論文誌, Vol. 21, No. 6(2006), pp. 537–546.
[11] 秋山英久: ロボカップサッカーシミュレーション *2D* リーグ必勝ガイド, 秀和システム, 2006.
[12] 藤田義門, 中村文一, 佐藤康之: Sarsa を用いた未知平面上の未知目標状態に対する大域的フィードバック制御系設計, 自動制御連合講演会講演論文集, Vol. 57(2014), pp. 1231–1235.