方策勾配型強化学習を用いた小型二輪ロボットにおける走行制御パラメータ獲得の検証と評価

中島 輝紀[1]　安武 芳紘[2]　澤田 直[2]

概要：技術の進歩に伴いロボットは小型化し，また人工知能の発展により自律的な行動を学習することができるようになった．本研究は光センサを用いてライントレースを行う小型二輪走行ロボットを対象に，強化学習による走行制御パラメータの獲得を行う．提案する学習方法は走行状態の安定と不安定を判定し，方策勾配型強化学習によりフィードバック制御のパラメータを学習するものである．提案する手法の検証と評価には，これまでのロボットの実機を用いてきたが，今回は TOPPERS プロジェクトの仮想シミュレーション環境「箱庭」を活用して検証を行った．環境の違いによる結果の違いやシミュレータの有効性について考察する．

キーワード：ロボット，強化学習，シミュレーション環境

Verification and Evaluation of Control Parameters Acquisition for a Two-Wheel Robot Using Policy Gradient Reinforcement Learning

Abstract: With the advancement of technology, robots have become smaller and smaller, and with the development of artificial intelligence, they are able to learn autonomous behavior. In this paper, we propose a reinforcement learning method to acquire control parameters for a small two-wheel robot that performs line tracing using an optical sensor. The proposed learning method judges the stability and instability of the running state and learns the parameters of feedback control by means of policy gradient reinforcement learning. In order to verify and evaluate the proposed method, we have used the actual robot in the past. In this paper, we use the virtual simulation environment "Hakoniwa" in the TOPPERS project. This paper discusses the differences in the results due to the differences in the environment and the effectiveness of the simulator.

Keywords: Robot, Reinforcement Learning, Simulation Environment

1. はじめに

昨今のロボットは Starship Technologies 社の配送ロボット Starship や Amazon 社の DeepRacer のようにより小型化している．また，人工知能の発展によりロボットは自身でその場の環境を認識し，その時の自分の状態から次の行動を判断するようにもなってきている．ロボットが実際に次の行動を取ろうとするには，それを制御するパラメータが必要である．本研究が対象とするロボット LEGO MindStorms EV3 は搭載した光センサを用いて状態判定しながら，ライントレース走行を行う．このライントレース走行はエアコンのフィードバック制御にも出てくる PID 制御を用いている．本研究の制御するパラメータとは前進量と旋回量に関わる各 PID ゲインであり，これを走行制御パラメータと定義している．我々はこれらの走行制御パラメータを方策勾配型強化学習により導出する方法を提案している．これまで走行制御パラメータの導出は実機を用いる他なかったが，TOPPERS プロジェクトの組込みロボットを扱う単体ロボット向けシミュレータ（以下，箱庭シミュレータ）により，以上の学習法の評価をシミュレータ上で行えることとなった．

[1] 九州産業大学大学院情報科学研究科
Graduate School of Information Science, Kyushu Sangyo University
[2] 九州産業大学理工学部
Faculty of Science and Engineering, Kyushu Sangyo University

本稿では実機による機械学習の説明，および箱庭シミュレータの説明とシミュレータ上での機械学習アルゴリズムの動作検証についてまとめる．

2. 実機による機械学習

研究の対象とするロボットは，LEGO MindStorms EV3 を ET ロボコン 2019 の規約に基づき組立てた HackEV という機体である．HackEV には，マイコンを搭載したインテリジェントブロックをはじめとして，光センサ，タッチセンサ，ジャイロセンサ，超音波センサ，アームモータ，尻尾モータ，そして左右 2 個の車輪モータが搭載されている．また，ロボットの OS は TOPPERS プロジェクトが提供するリアルタイム OS である EV3RT を用いる．

今回の学習において重要なのはインテリジェントブロック，光センサ，車輪モータの 3 つである．ロボットは光センサで計測した輝度値をもとに，走行用プログラムを組込んだインテリジェントブロックでロボットの状態を判断し，車輪モータを制御つまりライントレース走行をする．

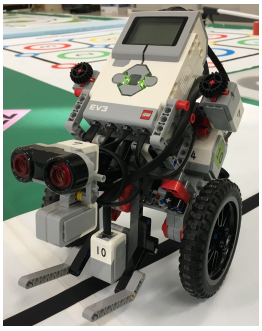

図 1 LEGO MindStroms EV3：HackEV3

2.1 ライントレース走行における PID 制御

ロボットは黒の輝度値と白の輝度値の中間の値を目標輝度値（式 1）とし，左右のモータの出力を調整することでライントレース走行を行う．ロボットの旋回量を光センサから取得した輝度値を基にフィードバック制御である PID 制御を用いて計算する．PID 制御は現在の輝度値と目標輝度値の偏差 $et(t)$（式 2）に対して，比例・積分・微分の計算をそれぞれ行い（式 3），その計算結果をそれぞれのモータに与え制御量を決定する．PID 制御を行うことで，ロボットはよりなめらかかつ安定な走行が可能となる．今回学習する走行制御パラメータは前進量，比例ゲイン（K_p），積分ゲイン（K_i），微分ゲイン（K_d）を対象としている．

$$\text{目標輝度値} = \frac{\text{黒の輝度値} + \text{白の輝度値}}{2} \tag{1}$$

$$et(t) = \text{現在の輝度値} - \text{目標値} \tag{2}$$

$$\text{操舵量} = K_p e(t) + K_i \int_0^t e(T)dT + K_d \frac{de(t)}{dt} \tag{3}$$

$$\text{モータ出力} = \text{前進量} \pm \text{操舵量}$$

2.2 状態判定

ロボットの走行状態を「安定状態」「不安定状態」「偏り状態」「脱線状態」と定め，各状態を定量的に判断するための変数を定義する．em は過去 N 回の偏差 $et(t)$ を平均した値である（式 4）．N 回は判定周期に依存する．em が正の値であれば白より，負の値であるなら黒よりを走行していることになる．ea は過去 N 回の偏差 $et(t)$ の絶対値を平均した値である（式 5）．ea の値が小さいほど，ロボットは目標値付近をなめらかにライントレース走行していると言える．

$$em = \frac{1}{N}\sum_{i=1}^{N} e(t_i) \tag{4}$$

$$ea = \frac{1}{N}\sum_{i=1}^{N} |\, e(t_i) \,| \tag{5}$$

安定状態はロボットが実際になめらかかつ安定した走行をしている状態を指し，走行の安定度を示す ea の値が安定定数 C_s を下回っていることを条件とする（式 6）．

$$ea \leqq C_s \tag{6}$$

不安定状態はロボットが一定の間隔で蛇行しながら走行している状態を指し，この際 em と走行体のぶれを累積する ea を比較することで状態を判定する（式 7）．しかし，単純な 2 値比較だけでは em が ea を上回ることはないため，em の絶対値をとると同時に不安定定数 C_{uns} を加え判定に幅を設けている．

$$|\, em \,| + C_{uns} \leqq ea \tag{7}$$

偏り状態はロボットが安定な走行をしているが，判定周期の間に目標値よりも黒よりあるいは白寄りを走行し続けた状態を指し，走行の安定度を示す ea の値が偏り定数 C_{bias} を下回っているかつ em の絶対値と ea が等しくなることを判定式とした（式 8）．偏り定数で判定に上限を設けることにより脱線状態と区別している．

$$|\, em \,| = ea \quad \textbf{and} \quad ea \leqq C_{bias} \tag{8}$$

脱線状態はロボットが白または黒の輝度を取得し続けている状態であるため，em が白か黒の輝度に近い場合に判定する（式 9）．環境の外乱によって白と黒の値に誤差が出るため，脱線定数 C_o）という定数で判定に幅を設けている．

$$\begin{aligned} &em - C_o \leqq \text{黒の輝度値} \leqq em + C_o \quad \textbf{or} \\ &em - C_o \leqq \text{白の輝度値} \leqq em + C_o \end{aligned} \tag{9}$$

2.3 方策勾配型強化学習

方策勾配型強化学習は，ランダムな方策（上昇・維持・下降）を施したパラメータで走行を行い，その際の状態の

割合によって次のパラメータを定める手法である．過去の研究においても，四足歩行ロボットにおける前進速度の向上を目的として足の各関節を制御するパラメータの学習に用いられている [1]．

図 2 は本研究で用いるロボットで，パラメータ最適化を行う手順である．手順は 2 段階ある．1 段階目は，まず走行可能なある程度最適化された前進値，K_p，K_i，K_d を初期パラメータとして設定する．設定した初期パラメータに対して，ランダムな 3 種類の方策 (初期パラメータ $\pm e$ o +0) でパラメータを変化させる．変化したパラメータを用いて走行する．報酬として，走行した際の走行状態と走行時間を取得する．以上の手順を 10 回繰り返す．

2 段階目は，まず 10 回分の報酬を走行状態と走行時間に分けて正規化する．正規化としては，作成した方策の種類ごとに報酬 (走行状態と走行時間それぞれで求め，例は speed の走行状態の報酬を基にした) の平均を求める．変化量として，維持 (初期パラメータ+0) の方策の平均が最も優れていた場合はパラメータを変更せず，別の種類の方策の平均が最も優れていた場合は，(初期パラメータ $\pm e$) の差を求める．走行状態の報酬から求めた前進量，K_p，K_i，K_d の変化量から平均と標準偏差を求める．前進量，K_p，K_i，K_d それぞれの標準化変量を求める．正規化した報酬に重みづけし，走行状態を基にした次のパラメータを求める．走行状態と走行時間から算出した次のパラメータを重みづけを行って組み合わせ，次の初期パラメータを決定する．初期パラメータを決定した次の初期パラメータに更新する．

以上の 2 段階の手順を繰り返してパラメータを最適化していく．実機によるパラメータ調整する際には実機への負担を考慮し，試行回数が少ない学習が求められる．また，本研究のような小型ロボットは計算資源が乏しいため，この強化学習の計算は機体自身で計算せずに，Excel による外部での計算処理を行っている．機体は Bluetooth 機能で走行時のパラメータを Excel 出力している．

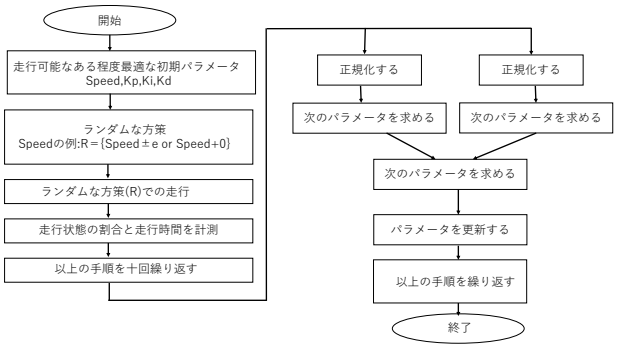

図 2 本研究のロボットでの最適化手順

2.4 検証環境

ライントレース走行ロボットには図 3 のような理想的な経路が存在する．本研究の理想的な経路では，経路上の線が交差していたり途切れたりしておらず，1 本の線で成立している．この理想的な経路は図 3 内の枠は様々な曲率を構成していることを示している．この曲率毎の経路を経路曲率としている．関連研究 [2] から経路曲率毎に区間を定め，それぞれの区間のみのパラメータを学習している．この手法により，ロボットの走行中の曲率の変化による脱線がなくなり，学習が中断する可能性が減るのに加えて，区間毎の走行状態の判定がし易くなる．最終的に全ての経路曲率のパラメータを学習すれば，どんな理想的な経路にも対応可能であると考えている．

関連研究 [2] では①〜④の区間を学習している．経路の曲率がわかると曲率半径が求まることから，学習では，①(曲率半径 ∞ cm) は図 4 の様な直線の経路を，②(曲率半径 34.0 cm) などの曲率は図 5 の様な曲線の経路を作成し，学習を行っている．

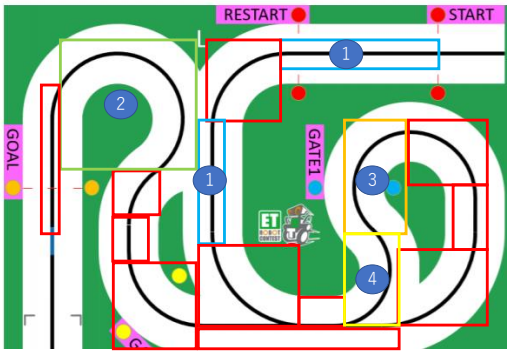

図 3 ET ロボコン 2019 競技用コース

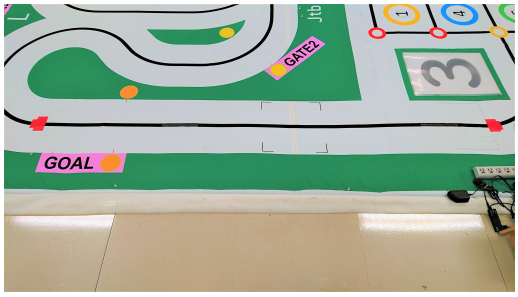

図 4 区間①の経路：曲率半径 ∞ cm

関連研究 [3] では上記の機械学習手法を別機体に適用させ，機体差に着目した検証を行っている．結果として，バッテリーやモータ，光センサの性能に差はあれど，学習によってパラメータは向上しているため，上記の機外学習手法は別機体にも適用可能であるとしている．

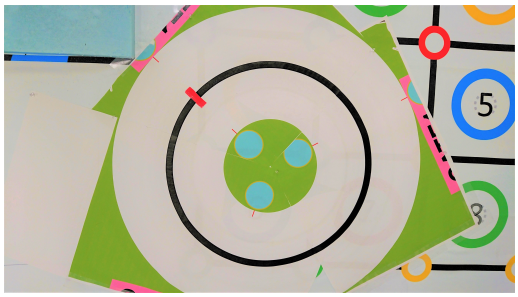

図 5 区間②の経路：曲率半径 34.0 cm など

径が実機の時と比べ大きくなっている．

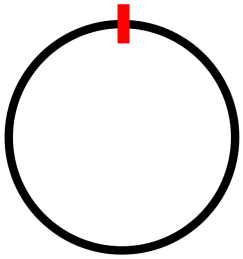

図 6 シミュレータによる機械学習：曲率半径 50cm

3. 検証

2 章で述べた学習法をシミュレータに適用した場合の学習結果をまとめ，実機の学習結果と比較を行い，シミュレータの有用性を分析する．

3.1 TOPPERS プロジェクトの箱庭シミュレータ

箱庭シミュレータは走行等のを制御処理を担うマイコンシミュレータ（Athrill）と物理計算に基づいた挙動を担う物理シミュレータ（Unity）が通信することで走行ロボットを再現している．TOPPERS プロジェクト [4] の組込みシステム構築の基盤となるソフトウェア開発の一環として開発された箱庭シミュレータ [5] を利用して，上記の機械学習手法の評価を行った．また，ET ロボコン 2020 で実際に競技で使われた大会シミュレータは箱庭シミュレータを基に作られたものである．シミュレータで走行ロボットは再現されたが，実機に存在する全ての要素の実装には至っていないことが判明している [6]．**シミュレータの特徴**

- 最高速度が 40cm/s．実機は 70cm/s
- 実機環境よりも摩擦が弱い
- 計算周期が 16ms〜17ms．実機は 4ms
- バッテリー電圧が実装されていない

このことより，シミュレータは実機の挙動と異なり，また学習結果にも影響があることが考えられる．検証準備にあたって，シミュレータ環境構築は [7] を基に行った．環境構築を行った使用 PC スペック及びプロトタイプ構成をまとめる．

使用 PC スペック

- CPU：AMD Ryzen 7 3700X 8-Core Processor
- GPU：NVIDIA GeForce RTX 2060 SUPER
- メモリ：32.0 GB

プロトタイプ構成

- プラットフォーム：Windows（WSL）
- CPU：V850
- Unity との通信方式：MMAP

上記の機械学習手法を再現するために，曲率半径 50cm の曲線の経路を用意した．計算周期のずれによりシミュレータの動作が不安定なことが分かっているため，曲率半

3.2 実機による機械学習の結果

関連研究 [2] にて，田川は実機による機械学習を行った．図 8，10，12，14 は．図 5 の区間②の経路（曲率半径 34.0cm）を学習した際の各パラメータの推移である．2.3 章で述べたように，この学習では 10 周を 1 セットとし，1 セットで得たデータを基に学習したパラメータを次のパラメータとして更新する．これを計 80 セット行っている．

走行時間の推移は図 8 のようになっている．学習に 1 セット 5000ms かかっていたのが，4000ms と約 1 秒ほど短縮している．

speed（前進量）は推移は図 10 のようになっている．初期パラメータの 45 から 60 まで向上しており，収束していないことからさらなる向上が期待できる．

PID の各ゲインは図 12，14 のようになっている．学習当初，全ゲインは下降傾向を示していたが，speed（前進量）が上昇するのに伴い，脱線を避けるために上昇している．kp と kd については上昇傾向にあると考えられる．これらに対して，ki は終始 0.1 以下であり，微量に上昇したとしても 0 あるいは 0.01 前後が最適であると考えられる．

この結果を受けて実機の学習では，speed（前進量）が収束しない限りはパラメータは上昇し続けるということが言える．

3.3 シミュレータ上の学習の結果

先行研究である実機をターゲットとした機械学習では初期パラメータとして ET ロボコン 2020 で使用したラインレースのパラメータを使用することが出来たが，シミュレータ上では元となるパラメータが無いため，初期パラメータは手動で調整したものを用いる．

シミュレータ上で先行研究と同じ機械学習のプログラムを実行すると，学習を 7 セット行い 8 セット目の学習を行っているところでコースを 10 周する前に脱線してしまい，パラメータが収束する前に学習が続行できなくなってしまった．しかし，データを取れたところまでを分析すると走行タイムが向上しており，走行パラメータが改良され

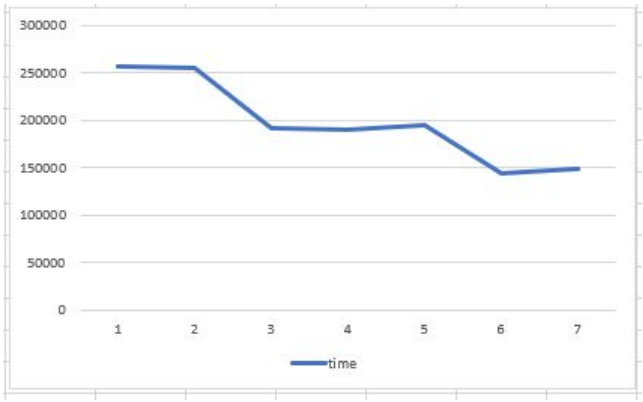

図 7 シミュレータによる機械学習：time（走行時間）の推移

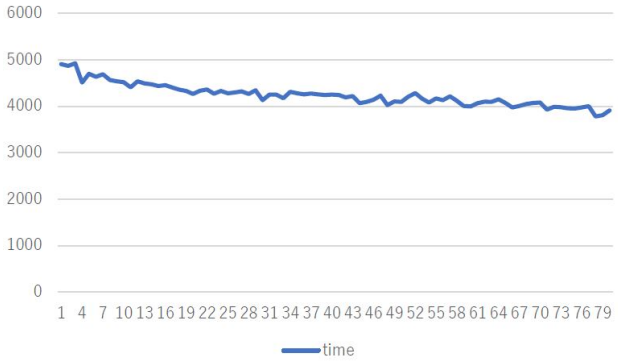

図 8 実機による機械学習：time（走行時間）の推移

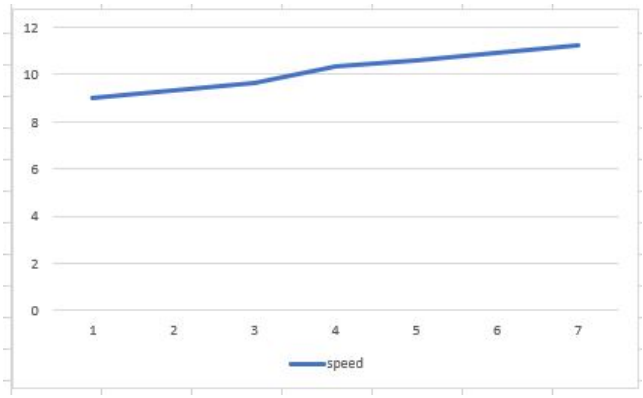

図 9 シミュレータによる機械学習：speed（前進量）の推移

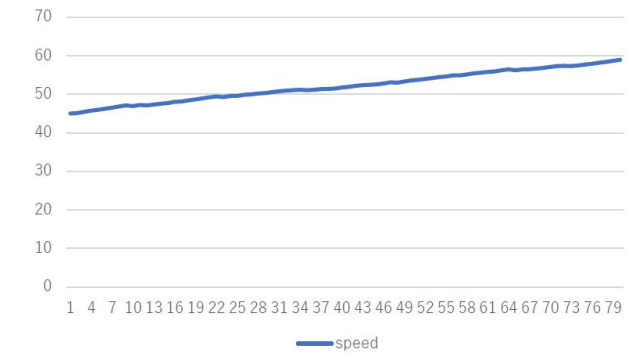

図 10 実機による機械学習：speed（前進量）の推移

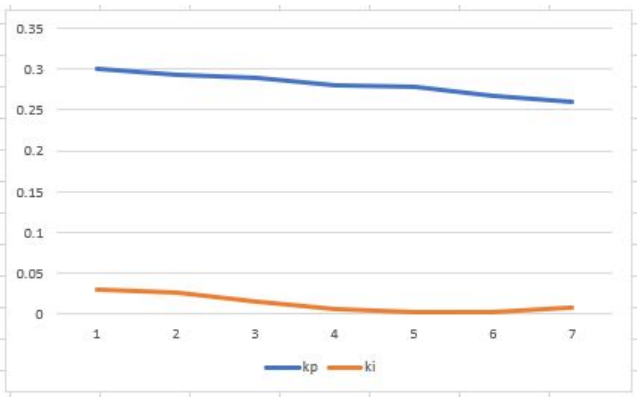

図 11 シミュレータによる機械学習：k_p（P ゲイン）,k_i（I ゲイン）の推移

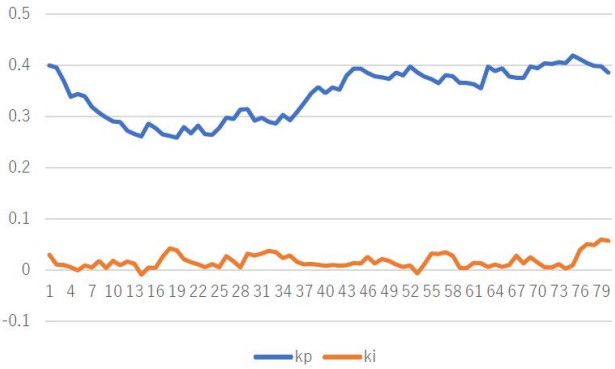

図 12 実機による機械学習：k_p（P ゲイン）,k_i（I ゲイン）の推移

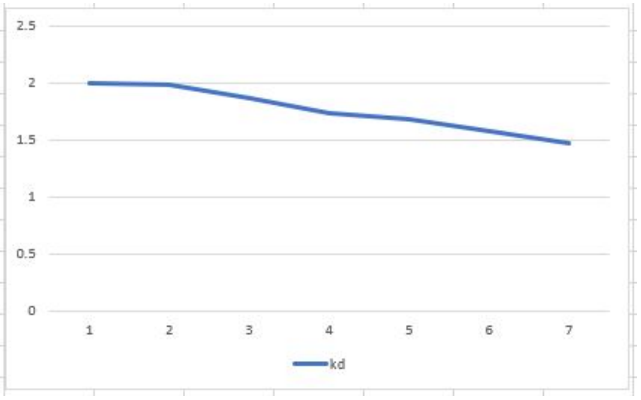

図 13 シミュレータによる機械学習：k_d（D ゲイン）の推移

て行っていることがわかる．また，speed が上がるにつれて unstable の値が上がっていっていることがわかる．この結果は田川が実機による機械学習を分析したものと類似しており，シミュレータ上でも実機と同じように学習が行えていると判断した．そのためもしこのままシミュレータ上で実機と同じように学習を行った場合，パラメータは同じように収束していっていると考えられる．結論として，シミュレータ上での機械学習は実機と同じように可能だと考える．

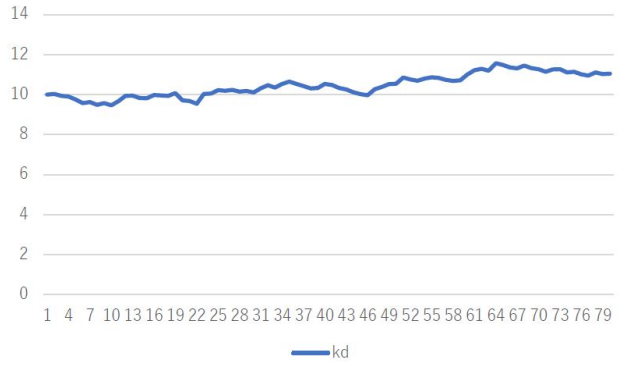

図 14 実機による機械学習：k_d（D ゲイン）の推移

3.4 シミュレータ上での動作検証のまとめ

今回の検証を行い，シミュレータ上でも強化学習によってパラメータ調整が可能であることがわかった．しかし，やはり実機とシミュレータ間では様々な相違点があり，走行にも大きく影響していたことが分かった．学習を続行できなくなった原因としては，3 章で述べたことが挙げられる．一番影響が大きかったのは車輪モータの回転量が実機とシミュレータ間で大きく異なることであったと考える．学習を行った後シミュレータ上で手動でライントレースのパラメータを変更し確認したところ，ライントレースのスピードのパラメータが 11.5 を超えたあたりで不安定になり始めることがわかった．そのため，シミュレータ上で走行パラメータの機械学習を行うためには，この車輪モータの回転量が異なるという問題を解決することが必要不可欠だと考える．実機とシミュレータ間でのモータの挙動が違うことを解決することが出来れば，シミュレータ上でも実機と同じように走行パラメータが収束するまで機械学習を行うことができると考えられる．

4. まとめ

本研究では，実機で行っていた機械学習をシミュレータ上で実行し，シミュレータ上での動作を検証した．本研究の結果としては実機と同じようには学習を行えなかったものの，シミュレータ上でも提案した機械学習を行えることがわかった．この結果を受けて，本研究でシミュレータについて検証・分析したことを基盤としてシミュレータ用に学習を作成する，もしくは走行パラメータを実機とシミュレータ間で相互に変換することが可能となれば，シミュレータ上でも走行パラメータが収束するまで学習を行うことができると考えられる．また，ロボットが経路上の赤線を光センサによる検知で一周の判定としていたものを，超音波センサによる検知にする，学習する経路を曲線だけではなく曲線と直線からなる経路にすることで，効率的な学習を行えるようになると考えている．

参考文献

[1] N. Kohl, P. Stone, “Policy gradient reinforcement learning for fast quadrupedal locomotion,” *Proceedings of the IEEE International Conference on Robotics and Automation*, pp.2619-2624, 2004.
[2] Y. Yasutake, C. Tagawa, S. Sawada, ”An Approach to Policy Gradient Reinforcement Learning with Multiple Evaluation Metrics”, Proceedings of the 34th International Technical Conference on Circuits / Systems, Computers and Communications, pp.427-430, 2019.
[3] 正岡拓海.: 個体差のある二輪走行体における方策勾配型学習の検証, 九州産業大学情報科学部情報科学科 (2020).
[4] TOPPERS プロジェクト，入手先 ⟨https://www.toppers.jp/index.html⟩（参照 2021-02-15）.
[5] 箱庭：技術資料・発表資料，入手先 ⟨https://toppers.github.io/hakoniwa/technical-links/⟩（参照 2021-02-13）.
[6] ET ロボコンミニワークショップ：シミュレータ上の走行におけるポイント　実機との差を理解しよう.
[7] 箱庭：単体ロボット向け，入手先 ⟨https://toppers.github.io/hakoniwa/prototypes/single-robot/⟩（参照 2021-02-13）.