The 32nd Annual Conference of the Japanese Society for Artificial Intelligence, 2018

市街地の自動運転における環境情報のフィルタリング

Filtering Environmental Information for Automatic Driving in Urban Area

北村清也 Seiya KITAMURA　石川翔太 Shota ISHIKAWA　荒井幸代 Sachiyo ARAI

千葉大学大学院融合理工学府都市環境システムコース
Department of Urban Environment Systems, Graduate School of Science and Engineering, Chiba University

This paper focuses on lack of explainability about the actions made by Deep reinforcement learning (DRL). DRL shows its superiority on tasks with multi-dimensional visual-input such as playing Atari games and navigation of robot. Also, DRL has been expected as an useful approach to realize a self-driving car. Though a lot of inputs would be available within urban environment, we could not specify the essential inputs to decide the appropriate action. The results derived from DRL are the tacit knowledge that is difficult to transfer to another system by means of writing it down as a symbolic way. Thus, human designer of self-driving operation may be reluctant to accept and utilize these results.

For the above reason, as a first stage to reach a symbolic representation, we propose a filtering method to specify the necessary inputs for the right actions, and show the effectiveness of it via some experiments.

1. はじめに

近年，交通事故減少や渋滞解消などへの期待から，市街地の自動運転が積極的に研究されている．市街地の自動運転では，交差点右折時の対向車や歩行者など，数多くの対象物を含む環境入力に対して適切な行動出力が求められる．この自動運転のタスクに対して，深層強化学習が期待される．深層強化学習は，環境の状態識別に有効な深層学習を行動学習の一手法として知られる強化学習に導入した手法であり，文献 [Mnih 13] にその有効性が示されている．深層強化学習によって，自動運転時に入力される周囲の環境情報から適切な行動出力が可能となる．しかし，深層強化学習は深層学習と同様，ニューラルネットワークによる関数近似器であり，入出力の関係が不明瞭なブラックボックスといえる．したがって，環境の状態に対する運転行動の妥当性が説明できない．行動妥当性の説明は，自動運転車設計に向けた信頼性を担保する上で必須であり，深層強化学習導入における課題といえる．この課題に対して，本研究では，環境の状態と行動の関係を明らかにすることを目的として，環境情報である入力の中から適切な行動出力に不可欠な属性を抽出する方法を提案する．ここで，すべての環境情報からの運転行動に必要な情報の抽出を「フィルタリング」と定義する．フィルタリングによって深層強化学習における入出力関係の説明が可能になるため，深層強化学習の自動運転導入にも貢献する．

本研究では，はじめに深層強化学習を用いて市街地の典型的なシーンに対する運転行動を獲得する．つぎに，学習後のネットワークを走行時に分析し，運転行動に必要な環境情報のフィルタリングを行う．

2. 問題設定

2.1 シミュレーション環境

運転行動の学習に用いるシミュレーション環境として TORCS(The Open Racing Car Simulator) を用いる．TORCS は自動運転タスクの研究に多く用いられているシミュレータである．本稿では，制御対象の自動車をエージェントと呼ぶ．エージェントは，周囲の環境特徴や自身の状態を表すセンサから得た状態入力をもとに行動の出力をする．対象とする運転シーンは図 1 に示すスタートラインから 400m までの直進走行区間である．

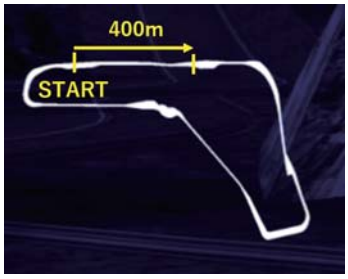


図 1: 本稿における対象シーン

2.2 深層強化学習のモデリング

深層強化学習モデル

運転行動の学習には，Deep Deterministic Policy Gradient(以下，DDPG)[Lillicrap 15] を用いる．DDPG は Actor-Critic に深層学習を導入した手法である．Algorithm1 に，DDPG による運転行動の学習手順を示す．2.1 で定義したエージェントは，Algorithm1 における $\mu(s|\theta^{\mu})$，$Q(s,a|\theta^{Q})$，μ'，Q' から構成される学習器を備える．

エージェントは状態 s_t を観測し，Actor ネットワークを介して行動 a_t を選択する．a_t を実行後，報酬 r_t，次状態 s_{t+1} を観測し，(s_t, a_t, r_t, s_{t+1}) をリプレイメモリ R に格納する．R からミニバッチをランダムサンプリングし，期待報酬 y_i を用いて Critic を更新した後，Actior を更新する．$\mu(s|\theta^{\mu})$ の更新には決定論的方策勾配法が，$Q(s,a|\theta^{Q})$ の更新にはベルマン方程式が用いられる．また，μ'，Q' は期待報酬 y_i の計算に用いられる．以上の手順を繰り返して学習を行う．

本研究において，$\mu(s|\theta^{\mu})$，$Q(s,a|\theta^{Q})$ はニューロンが 500 個，100 個，100 個の 3 層からなる中間層と入力層，出力層で構成される．入出力層のニューロン数は，状態入力，行動出力の次元によって決定される．

連絡先: 北村清也，千葉大学大学院融合理工学府都市環境システム，千葉市稲毛区弥生町 1-33

Algorithm 1 DDPG の学習アルゴリズム

Actor，Critic ネットワーク $\mu(s|\theta^\mu)$，$Q(s,a|\theta^Q)$ の重み θ^μ，θ^Q をランダム値で初期化
ターゲットネットワーク μ'，Q' の重み $\theta^{\mu'} \leftarrow \theta^\mu$，$\theta^{Q'} \leftarrow \theta^Q$ をランダム値で初期化
リプレイメモリ R を初期化
for $learn = 1$ to F **do**
  探索ノイズ $\boldsymbol{N}$ を 0 で初期化
  初期状態 $\boldsymbol{s_1}$ を観測
  for $t = 1$ to T **do**
    行動 $\boldsymbol{a_t} = \boldsymbol{\mu(s_t|\theta^\mu)} + \boldsymbol{N_t}$ を選択
    探索ノイズを更新：$\boldsymbol{N_t} \leftarrow \boldsymbol{N_t} - w_1 \cdot \boldsymbol{N_t} + w_2 \cdot random$
    $\boldsymbol{a_t}$ を実行し，報酬 r_t，次状態 $\boldsymbol{s_{t+1}}$ を観測
    $(\boldsymbol{s_t}, \boldsymbol{a_t}, r_t, \boldsymbol{s_{t+1}})$ を R に格納
    R から 128 個のミニバッチをランダムにサンプリング
    期待報酬 $y_i = r_i + \gamma Q'(\boldsymbol{s_{i+1}}, \mu'(\boldsymbol{s_{i+1}}|\theta^{\mu'})|\theta^{Q'})$ を計算
    誤差 $E = \frac{1}{128}\sum_{i=1}^{128}(Q(\boldsymbol{s_i}, \boldsymbol{a_i}|\theta^Q) - y_i)^2$ を最小化するよう Critic を更新
    決定論的方策勾配定理を用いて Actor を更新：
      $\frac{\partial J}{\partial \theta^\mu} \approx \frac{1}{128}\sum_{i=1}^{N} \frac{\partial Q(\boldsymbol{s_i}, \boldsymbol{a_i}|\theta^Q)}{\partial \boldsymbol{a_i}} \frac{\partial \mu(\boldsymbol{s_i}|\theta^\mu)}{\partial \theta^\mu}$
    ターゲットネットワーク μ'，Q' を更新：
      $\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$
      $\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$
  end for
end for

状態入力

状態入力は，表 1 に示す 5 個の属性を用いた 5 次元入力とする．これより，入力層のニューロン数は 5 であり，Algorithm1 では時刻 t において 5 次元の状態ベクトル $\boldsymbol{s_t}$ が入力として扱われる．状態入力は値域をすべて [-1,1] に正規化する．

表 1: 本実験に用いる入力次元 [Loiacono 13]

入力次元	値域	単位	詳細
trackPos	(-∞, +∞)	-	車と道路中心との距離
angle	[-π, π]	rad	道路軸に対する車の角度
speedX	(-∞, +∞)	km/h	車の縦軸における速度
speedY	(-∞, +∞)	km/h	車の横軸における速度
speedZ	(-∞, +∞)	km/h	車の鉛直方向における速度

行動出力

表 2: 本実験に用いる出力次元 [Loiacono 13]

出力次元	値域	単位	詳細
steering	[-1,1]	-	-1 が右，1 が左への最大舵角を表す舵角値

行動出力は，表 2 に示す 1 個の属性を用いた 1 次元出力とする．これより，出力層のニューロン数は 1 であり，Algorithm1 では時刻 t において 1 次元の行動ベクトル $\boldsymbol{a_t}$ が出力として扱われる．

報酬

図 2 に，エージェントへ与える報酬 r_t を示す．本実験では，報酬をペナルティに限定する．エージェントは，ペナルティをなるべく受け取らないための行動を学習する．はじめに，最も望ましくない状態としてコースアウトが考えられるため，コースアウト時に最大のペナルティ-20 を与える．つぎに，エージェントは直進走行時において，道路の中心付近を道路と平行に走行することが望ましいと考えられるため，入力次元の trackPos と angle に応じたペナルティ pos_penalty，ang_penalty を毎ステップ与える．しかし，人間の運転熟練者でも trackPos と angle が常に 0 であることはない．したがって，多少の余裕を考慮し，|trackPos| < 0.2 のとき pos_penalty = 0，|angle| < 0.01 のとき ang_penalty = 0 とする．また，他車両が存在しない直線道路では，エージェントは徐行運転をする必要がなく，低速走行はしないため，入力次元の speedX が 60.0km/h 以下のときに-5 のペナルティを加える．

```
コースアウト時
  r_t = −20

毎ステップ
  if |trackPos| < 0.2:
    pos_penalty = 0
  else: pos_penalty = −10|trackPos|^2
  if |angle| < 0.01:
    ang_penalty = 0
  else: ang_penalty = −|angle|
  r_t = pos_penalty + ang_penalty
  if speedX < 60.0:
    r_t = r_t − 5
```

図 2: エージェントに与えるペナルティ

3. 勾配を用いたフィルタリング

本章では，提案法であるフィルタリングについて説明する．2.2 より，フィルタリングは，状態ベクトル $\boldsymbol{s}$ における全要素からの行動出力に必要な属性の抽出である．本研究では，出力に対する入力の勾配（以下，勾配）を用いてフィルタリングを行う．勾配は入力値の微小変化に対する出力値の変化量である．勾配の大きい入力次元は微小変化による出力変化が大きく，影響度も大きいと考えられるため，エージェントに重要視されているといえる．勾配は入力次元の数だけ要素が存在するベクトルであり，本稿では式 (1) で表される．フィルタリングは勾配ベクトルにおける各要素の大きさを比較して行う．

$$\frac{\partial \mu(\boldsymbol{s}|\theta^\mu)}{\partial \boldsymbol{s}} = \left[\frac{\partial \mu(\boldsymbol{s}|\theta^\mu)}{\partial s_1}, \frac{\partial \mu(\boldsymbol{s}|\theta^\mu)}{\partial s_2}, \frac{\partial \mu(\boldsymbol{s}|\theta^\mu)}{\partial s_3}, \frac{\partial \mu(\boldsymbol{s}|\theta^\mu)}{\partial s_4}, \frac{\partial \mu(\boldsymbol{s}|\theta^\mu)}{\partial s_5} \right]^T \tag{1}$$

3.1 フィルタリング手順

Algorithm2 に示すフィルタリング手順によって，入力次元を n 次元から行動出力に必要な m 次元へフィルタリングできる $(n > m)$．以下，Algorithm2 について説明する．

I.L 体の安定走行を示すエージェントの獲得

深層強化学習を用いて n 次元入力により安定した直進走行を学習させる．エージェントの安定走行を判断する基準として，横加速度を用いる．ここで横加速度を，車の加速度ベクトルのうち進行方向と垂直な成分と定義する．通常，自動車の走行中は横加速度が 0.2G〜0.3G であり，これ以上大きくなると搭乗者は不快感や恐怖感を抱くといわれている [Nasukawa 08] ため，Algorithm2 中で 0.3G という基準値を設けている．エージェントが 400m を走行可能かつ $stability = good$ のとき，安定走行とする．

Algorithm 2 勾配を用いたフィルタリング

$(n, m) =$ (存在する次元, 必要な次元) $(n > m)$
$i = 0$
I.L 体の安定走行を示すエージェントの獲得
while $i < L$ **do**
  Algorithm1 を用いて n 次元入力で学習
  学習終了後，再度走行
  $(accel, ave\_accel, SD\_accel) =$
  (観測される横加速度, $accel$ の平均, $accel$ の標準偏差)
  G:重力加速度
  $over\_accel = 0$
  for $t = 1$ to $race\_finish$ **do**
    accel を観測
    if $accel > 0.3G$ **then**
      $over\_accel = over\_accel + accel/0.3G$
    end if
    if $t = race\_finish$ **then**
      走行距離 $dist$ を観測
    end if
  end for
  if $(ave\_accel + SD\_accel + over\_accel) < 0.3G$ **then**
    $stability = good$
  end if
  if $dist = 400 \wedge stability = good$ **then**
    安定走行
    $i \leftarrow i + 1$
    $agent_i$ としてエージェントを保存
  end if
end while
II. 勾配評価と評価値の更新
$X = \{x_j | j$ は n 以下の自然数 $\}$: 各入力次元の評価値
$R = \{r_j | j$ は n 以下の自然数 $\}$: 各入力次元のランク変数
$X \leftarrow 0$
for $k = 1$ to L **do**
  $agent_k$ における n 次元の勾配を分析
  分析した勾配の大きさを降順にソート，順位を R に代入
  以下の更新式により X を更新:
    $x_j \leftarrow x_j(n - r_j + 1)$
end for
III. 走行テストに用いるエージェントの選択
$M = X$ における上位 m 個の入力次元
for $l = 1$ to L **do**
  M_{all}:$agent_l$ における勾配の走行時平均上位 m 個の入力次元
  $M_{5\mathrm{m}}$:$agent_l$ における勾配の走行開始 5m 平均上位 m 個の入力次元
  if $M = M_{all} \wedge M = M_{5\mathrm{m}}$ **then**
    $agent_l$ を採用
  end if
end for
IV. M を入力とした走行テスト
if $dist = 400 \wedge stability = good$ **then**
  フィルタリング成功
else
  フィルタリング失敗
  2 行目に戻る
end if

II. 勾配評価と評価値の更新

獲得された L 体のエージェントを用いて勾配評価をする．ここで，n 個の要素をもつ X はフィルタリングに用いる評価値であり，各要素が各入力次元の評価値をそれぞれ表す．R はランク変数であり，X と同じ要素数をもつ．R の各要素 r_i は x_i の入力次元に対応しており，勾配評価で明らかとなった勾配の大きさによる各入力次元の順位が整数で格納される．勾配はエージェントを n 次元入力で再度走行させることにより求め，走行区間における平均値の大きさを比較して評価する．フィルタリング開始時に X は初期化され，更新式によって L 回更新される．

III. 走行テストに用いるエージェントの選択

L 回の更新終了後，評価値 X の値が大きい上位 m 次元をエージェントが重要視する次元として選択し，これを M とする．続いて，走行テストを行うエージェントを選択する．走行テストを行うエージェントは，L 個の中から 2 つの条件により選択する．M_{all} を $agent_l$ における勾配の走行時平均上位 m 個の入力次元，$M_{5\mathrm{m}}$ を $agent_l$ における勾配の走行開始 5m 平均上位 m 個の入力次元とすると，M は，勾配の走行時平均による順位を用いた評価値更新結果に基づいているため，$M = M_{all}$ である必要がある．また，L 回の更新結果で得られた入力次元の順位は，エージェントが重要視している入力次元の順位として信頼できると考える．しかし，$M_{5\mathrm{m}}$ を満たさずに 5 次元入力で安定走行をするエージェントの場合，走行開始時において本来重要視すべき入力次元が重要視されていないことになる．そのようなエージェントが安定走行である場合，過学習が考えられる．TORCS は必ず同じ位置から停止状態で走行を開始するため，走行開始時の過学習が起こりやすいと考えられる．本研究では，エージェントの学習と学習後の走行は同じコースを用いている．すなわち，訓練データとテストデータが同一であり，過学習による安定走行が考えられる．したがって，$M = M_{5\mathrm{m}}$ を満たす必要がある．

IV. M を入力とした走行テスト

III. で採用されたエージェントを用いて，M を入力とする走行テストを行う．採用されたすべてのエージェントが走行テストにおいて安定走行を示した場合，フィルタリングは成功となる．

4. 計算機実験

本実験では，学習に用いる入力次元 $n = 5$，フィルタリングによって獲得する必要な次元 $m = 2$，獲得エージェント数 $L = 10$ とする．

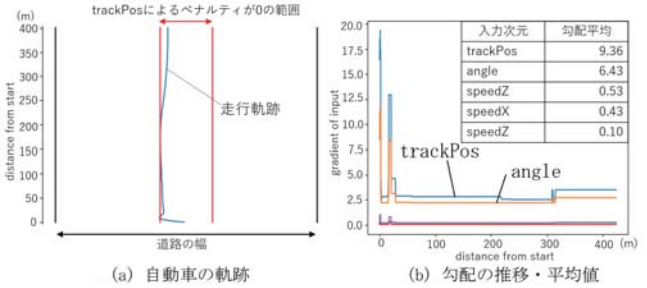
4.1 フィルタリング

I.L 体の安定走行を示すエージェントの獲得

直進走行を学習させ，安定走行を示すエージェントを 10 体獲得した．図 3(a) に示す自動車の軌跡は，獲得された 10 個のエージェントから 1 個を用いたものであり，青線が自動車の走行軌跡，両端の黒線が道路端，赤線の内側が trackPos によるペナルティが 0 になる範囲を表している．図 3(a) は，Algorithm2 で定義した安定走行ができたことを示す．

II. 勾配評価と評価値の更新

獲得したエージェントの勾配を求め，各入力次元の評価値を更新する．図 3(b) は 400m の走行時における勾配の推移を示しており，trackPos，angle の勾配が常に大きいことがわかる．図 3(b) 中の表はグラフの平均値をとったものである．これにより各入力次元に順位がつけられるため，ランク変数 R の各要素に各入力次元の順位が格納され，0 で初期化した評価



(a) 自動車の軌跡　(b) 勾配の推移・平均値

図 3: 安定走行を示すエージェント

表 3: 評価値 X の更新結果

X	入力次元	1 回目	10 回目
x_1	trackPos	5	50
x_2	angle	4	39
x_3	speedX	2	19
x_4	speedY	1	22
x_5	speedZ	3	20

値 X が更新される．表 3 に，評価値 X の更新結果を示す．本実験では，表 3 に示すとおりに各入力次元を X の要素に対応させた．表 3 における 1 回目の列には，図 3(b) による X の更新結果が示されている．10 回目の列は，獲得された 10 体のエージェントによる X の最終的な更新結果である．

III. 走行テストに用いるエージェントの選択

はじめに，評価値更新の結果と $m = 2$ より，trackPos，angle を影響度の大きい，重要視される入力次元として選択する．

続いて，2 次元入力による走行テストを行うエージェントを選択する．本実験では，10 体のエージェントの中から，8 体が採用された．

IV. M を入力とした走行テスト

採用された 8 体のエージェントを用いて，trackPos，angle の 2 次元入力による走行テストを行う．図 4 に，図 3 のエージェントを用いた trackPos，angle の 2 次元入力による自動車の走行軌跡を示す．図 4 の青線が 2 次元入力時の自動車の走行軌跡を表し，両端の黒線，赤線の内側は図 3(a) と同様である．濃桃の破線は，図 3(a) に示した 5 次元入力時の走行軌跡である．図 4 の青線は，Algorithm2 で定義した安定走行ができたことを示している．さらに 5 次元入力時と比較して，より直線的な走行をしていることがわかる．走行テストに用いた 8 個のエージェントは，すべて図 4 と同様の結果を示していた．これらより，2 次元入力で安定走行が可能であり，trackPos，angle は直進走行に必要な次元であることがわかった．したがって，入力次元を 5 次元から 2 次元へフィルタリングできた．

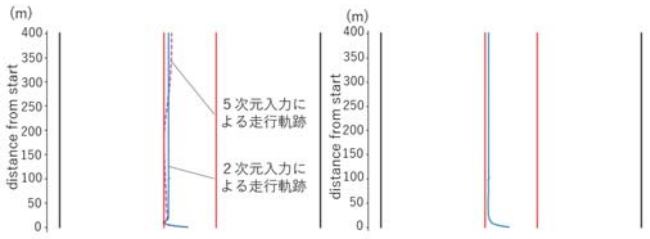


図4:2次元入力による自動車の走行軌跡　図5:2次元入力による学習結果

4.2 検証実験

4.1 より，5 次元入力から直進走行に不可欠な次元が抽出できた．ここでは，次元を削減した際の学習性能について検証する．必要な次元だけを入力した学習結果として安定走行が確認できた場合，提案法により特定された入力次元だけで安定走行が獲得可能であり，提案法の有効性が示せる．

図 5 に示すのは，trackPos，angle の 2 次元入力による直進走行の学習で獲得されたエージェントを用いた自動車の走行軌跡である．図 5 の青線で表される自動車の走行軌跡は，Algorithm2 に定義した安定走行ができたことを示している．また，図 3(a) と比較すると，より直線的な走行であることがわかる．以上により，提案法により抽出された次元だけで安定走行が獲得可能であることが示された．

5. 考察

ここでは，フィルタリングされた trackPos，angle の 2 次元入力による走行が 5 次元入力時よりも直線的であった理由について考察する．

本実験では，提案法のフィルタリングによって直進走行の学習に用いた 5 次元入力から不可欠な次元を抽出できた．直進走行を 5 次元入力，1 次元出力によって学習したときの最適行動は，tracPos，angle がともに 0 の状態で舵角 steering が 0 で出力されることである．深層強化学習ではノードの値と，ノード間の重みとの積によって値が伝播，出力されるため，重みが大きいノードほど出力に大きく影響を与えると考えられる．したがって，多次元入力における不要な次元にかかる重みがノイズのような役割をしており，本実験では，5 次元入力によるノイズを除くことでより直線的な走行，すなわち最適行動に近い挙動を示したと考えられる．

6. まとめ

本研究では，自動運転に対して深層強化学習を適用する際の課題とされる行動妥当性を説明可能にするための第一歩として，行動出力に必要な環境情報の属性を抽出するためのフィルタリングを提案した．提案法では，深層強化学習を用いて直進走行を学習したエージェントの入力の勾配を用いた．提案法によって抽出された，勾配が大きい入力次元だけを入力としても安定走行が確認できたことから，提案法によるフィルタリングの有効性を示した．

参考文献

[Mnih 13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, Martin Riedmiller: Playing Atari with Deep Reinforcement Learning, arXiv:1312.5602 [cs.LG], (2013)

[Lillicrap 15] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, Daan Wierstra: Continuous control with deep reinforcement learning, arXiv:1509.02971 [cs.LG], (2015)

[Nasukawa 08] 茄子川捷久, 宮下義孝, 汐川満: 自動車の走行性能と試験法, 東京電機大学出版局, (2008)

[Loiacono 13] Daniele Loiacono, Luigi Cardamone, Pier L. Lanzi: Simulated Car Racing Championship Competition Software Manual, arXiv:1304.1672v2 [cs.AI], (2013)