計測自動制御学会東北支部　第 250 回研究集会(2009.6.19)
資料番号 250-1

高次元連続状態空間における強化学習
－局所重み付き回帰手法を用いた価値関数近似－

Reinforcement Learning in High-dimensional Continuous State Spaces
– Value Function Approximation Using Locally Weighted Regression –

○一井宏次*　　釜谷博行**　　工藤憲昌**
○Koji Ichii*　　Hiroyuki Kamaya**　　Norimasa Kudoh**
*八戸工業高等専門学校 専攻科　**八戸工業高等専門学校
*Hachinohe National College of Technology Advanced Engineering Course
**Hachinohe National College of Technology

キーワード：強化学習(reinforcement learning), ロボット制御(robot control),
関数近似(function approximation), 移動障害物回避(moving obstacle avoidance)

連絡先：〒039-1192 八戸市田面木字上野平 16－1　八戸工業高等専門学校　電気情報工学科
釜谷博行,　Tel.: 0178-27-7283,　E-mail: kamaya-e@hachinohe-ct.ac.jp

1. はじめに

動的に変化する未知環境において，自律性や環境適応性などをもつエージェントの知的な振る舞いを可能にするには，環境との相互作用を扱うためのメカニズムを構築する必要がある[1]. そこで近年，経験をもとに自ら学習する強化学習[2]が注目されている．強化学習は探索と学習能力を持ち合わせ，関数最適化問題やロボット制御問題をはじめ，その応用が幅広く期待されている．

強化学習は価値関数を用いて，環境中での試行錯誤で得られた報酬から，環境へ適応するための行動を自律的に学習する機械学習の枠組みである．通常，価値関数は状態-行動対からなる離散値のテーブル形式で表現される．

しかし，一般に実環境での複雑な振る舞いを学習するためには入力が連続値で，かつ多入力の構成をとる場合が多い．連続値を扱うために，これまでタイルコーディング[3]や RBF(Radial Basis Function)ネットワーク[4]などの関数近似手法を用いて価値関数を表現する提案がなされてきた．だが，高次元の状態空間では，離散化した状態空間と同様に，入力次元の増加によって状態空間が拡大し，計算量やメモリが指数関数的に増加する．また，関数近似手法に必要なパラメータが多いと調整するための予備実験が大規模になることも問題といえる．

それら諸問題を改善するため，本研究では，多変量解析手法の一種である局所重み付き部分最小二乗法(LWPLS: Locally Weighted Partial Least Squares)[5]を関数近似手法とする新たな強化学習アルゴリズムを提案する．

これまでの研究では，2 次元，4 次元の低次元状態空間をもつ強化学習問題に有用であることを確認した[6][7]．本稿では，さらに多数の入力をもつ強化学習問題を取り上げる．具体的には 10 次元連続状態空間をもつ移動障害物回避問題に提案手法を適用し，シミュレーション実験から代表的な関数近似手法の 1 つであるタイルコーディングとの比較を行い，その有効性を報告する．

2. 強化学習

動物にある行動を起こしたときだけ餌などの報酬を与えるという操作を繰り返すと，その行動パターンが徐々に強化され，最終的には，実際に報酬が与えられなくても，同様な状況に置かれるとその行動を起こすようになる．このように，報酬を契機として行動パターンを学習する一連の適応現象を強化学習と呼んでいる．機械学習における強化学習はこれを工学的に応用したものである．学習と意思決定を行う学習主体はエージェントと呼ばれ，エージェントと相互作用を行うエージェント以外のすべてから構成されるものは環境と呼ばれる(Fig.1).

強化学習では，ある環境内にいるエージェントが時刻 t において，現在の状態 s_t を観測し，行動 a_t を決定する問題を扱う．エージェントはある方策にしたがって行動を選択・実行し，その結果に応じて環境から報酬 r_t を得て，次の状態 s_{t+1} へ遷移する．強化学習の目的はこのような一連の流れを通じて得られる報酬の総和を最大化することである.

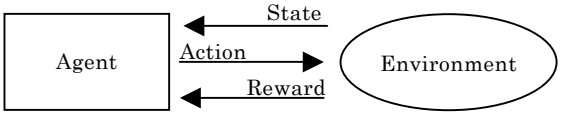


Fig.1　The agent-environment interaction

3. LWPLS

途切れなく状態が変わる連続状態空間では，各状態間の距離を定義することができる．距離的に近い状態では価値関数も近い値をもち，2 つの状態の中間あたりに存在する状態は，それら 2 つの価値関数の中間くらいの値をもつことが多い．そこで連続な状態空間をもつ強化学習問題では，価値関数の表現に関数近似を用いることができる．関数近似を用いると，学習の高速化や，今まで経験したことのない状態に遭遇しても，似たような状態での経験を生かして適切な行動選択が可能となる．代表的な関数近似手法として，タイルコーディング，RBF ネットワークなどを用いる方法が提案されている．本研究では，関数近似手法として高次元空間にも利用可能な LWPLS に着目した.

LWPLS は部分最小二乗法(PLS: Partial Least Squares)に局所重みという概念を導入したものである．PLS は主成分分析と重回帰分析の特性を結びつけた次元圧縮手法の一種としてよく知られている．特徴として，次元を圧縮して計算するため，高次元でも使用可能であること，入力との相関および出力との相関を考慮する潜在変数を用いて推定を行うため，多重共線性を回避できることなどが挙げられる．なお，多重共線性とは入力変数間に非常に強い相関があることや，一次従属な変数関係があることをいう．PLS に局所重みを導入することで，推定の際，低次元の一次式を使用しても高次の関数の近似が可能になっている.

LWPLS では入力ベクトルと出力値からなるモデル $(\mathbf{x}_i, y_i)$ を用いて，推定点 $\mathbf{x}_q$ の推定値 $\hat{y}_q$ を求める．Fig.2 に LWPLS のアルゴリズムを示す．アルゴリズムは(a)から(c)までのモデル $(\mathbf{x}_i, y_i)$ に対する射影の計算と，(d)での推定点 $\mathbf{x}_q$ に対する推定値 $\hat{y}_q$ の計算から成り立っている.

(a) $w_{ii} = \exp\left\{-0.5\left(\mathbf{x}_i - \mathbf{x}_q\right)^T \mathbf{D}\left(\mathbf{x}_i - \mathbf{x}_q\right)\right\}$

(b) $\bar{\mathbf{x}} = \sum_{i=1}^{p} w_{ii}\mathbf{x}_i \Big/ \sum_{i=1}^{p} w_{ii},\ \beta_0 = \sum_{i=1}^{p} w_{ii} y_i \Big/ \sum_{i=1}^{p} w_{ii}$

$\mathbf{X} = \left(\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, ..., \tilde{\mathbf{x}}_p\right)^T$ where $\tilde{\mathbf{x}}_i = \left(\mathbf{x} - \bar{\mathbf{x}}\right)$

$\mathbf{y} = \left(\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_p\right)^T$ where $\tilde{y}_i = \left(y_i - \beta_0\right)$

(c) Initialize: $\mathbf{Z}_0 = \mathbf{X},\ \mathbf{res}_0 = \mathbf{y}$

For $i = 1:r$

$\mathbf{u}_i = \mathbf{Z}_{i-1}^T \mathbf{W} \mathbf{res}_{i-1}$

$\mathbf{s}_i = \mathbf{Z}_{i-1}^T \mathbf{u}_i$

$\beta_i = \dfrac{\mathbf{s}_i^T \mathbf{W} \mathbf{res}_{i-1}}{\mathbf{s}_i^T \mathbf{W} \mathbf{s}_i}$

$\mathbf{p}_i = \dfrac{\mathbf{s}_i^T \mathbf{W} \mathbf{Z}_{i-1}}{\mathbf{s}_i^T \mathbf{W} \mathbf{s}_i}$

$\mathbf{res}_i = \mathbf{res}_{i-1} - \mathbf{s}_i \beta_i$

$\mathbf{Z}_i = \mathbf{Z}_{i-1} - \mathbf{s}_i \mathbf{p}_i$

(d) Initialize: $\mathbf{z}_0 = \mathbf{x}_q - \bar{\mathbf{x}},\ \hat{y}_q = \beta_0$

For $i = 1:r$

$s_i = \mathbf{z}_{i-1}^T \mathbf{u}_i$

$\hat{y}_q \leftarrow \hat{y}_q + s_i \beta_i$

$\mathbf{z}_i = \mathbf{z}_{i-1} - s_i \mathbf{p}_i^T$

Fig.2　LWPLS

まず，(a)でモデルの入力ベクトル $\mathbf{x}_i$ と推定点 $\mathbf{x}_q$ との差から局所重みを求める．w_{ii} は重み行列 $\mathbf{W}$ の対角成分であり，モデルの値をどれくらい考慮するかを決める．そして，w_{ii} の式の中にある $\mathbf{D}$ は重み係数行列であり，この値によって推定点 $\mathbf{x}_q$ からどれくらい近いモデルを考慮するのかが決まる．$\mathbf{D}$ が大きい場合には推定点 $\mathbf{x}_q$ に近いモデルのみを用い，逆に小さい場合には広い範囲のモデルを用いて推定する．(a)で求めた局所重みを用いて，(b)では加重平均を計算し，全てのモデルのデータから加重平均を引いて $\mathbf{X}$ と $\mathbf{y}$ を作る．(c)では $\mathbf{X}$ と $\mathbf{y}$ を初期値として局所線形モデルの計算を行う．$\mathbf{y}$ との相関および $\mathbf{X}$ との相関を同時に考慮して適切な潜在変数 $\mathbf{s}_i$ を求め，その後，$\mathbf{s}_i$ と $\mathbf{y}$ を結び付ける回帰係数 β_i，$\mathbf{s}_i$ と $\mathbf{X}$ を結び付けるローディングベクトル $\mathbf{p}_i$ を導出し，残差 $\mathbf{res}_i$，$\mathbf{Z}_i$ を求める．これを r 回繰り返す．ここで，r は回帰数と呼ばれる．(c)が終了したら，求めたパラメータ $\mathbf{u}_i$，β_i，$\mathbf{p}_i$ を使い，(d)で推定点 $\mathbf{x}_q$ の推定値 $\hat{y}_q$ を求める．

4. 提案アルゴリズム

強化学習では，状態，行動，報酬のやり取りと価値関数の更新を定める学習アルゴリズムと，価値関数から行動を選択する方策が必要になる．本稿ではSarsa(λ)学習を学習アルゴリズムとし，ε-greedy 方策により行動選択する．価値関数は提案手法である LWPLS により表現される．これら全体のアルゴリズムを Fig.3 に示す．

4.1 Sarsa(λ)学習

Sarsa(λ)学習は強化学習で広く使われている学習アルゴリズムの１つである．エージェントと環境との相互作用により，価値関数 $Q(s_t,a_t)$ が時間ステップごとに更新される．$Q(s_t,a_t)$ はある状態 s_t で，ある行動 a_t をとることで将来的にどれだけ報酬が期待できるかを表す．この値を Q 値と呼ぶ．

関数近似手法では，LWPLS との整合をとるためにエージェントが訪問した状態 s と，そこで実行可能な全ての行動に対する Q 値および適格度，そして Fig.2(a)で示した局所重みが 1 つのモデルとして適宜追加される．追加条件は，訪問した状態 s での局所重みの最大値がある閾値 w_{th} 以下である場合，つまり他のモデルと状態空間上である距離以上離れている場合に追加を行う．以後，追加されたモデル i の価値関数を $q_i(a)$ と表記する．ある状態の Q 値は，追加されたモデルを用いて LWPLS により推定され，更新は全てのモデルの Q 値に対して以下の式で行われる．

$$\begin{aligned}\delta_t &\leftarrow r_t+\gamma Q(s_{t+1},a_{t+1})-Q(s_t,a_t)\\ q_i(a) &\leftarrow q_i(a)+\alpha\delta_t e_i(a)\ \text{for all}\, i,a\end{aligned} \tag{1}$$

α は学習率 $(0\le\alpha\le1)$，γ は割引率 $(0\le\gamma\le1)$ である．

適格度 $e_i(a)$ は TD 誤差 δ_t を過去に訪問した状態へ伝播させるためにあり，ここでは累積トレースにより値を更新する．その際，局所重みを利用することで Sarsa(λ)学習が連続状態空間を扱えるようにした．状態 s において選択された行動を a' とすると，適格度は次式で更新される．For all i, a:

$$e_i(a)\leftarrow\begin{cases}\gamma\lambda e_i(a)+w_{ii} & \text{if } a=a';\\ \gamma\lambda e_i(a) & \text{otherwise.}\end{cases} \tag{2}$$

λ は伝播の程度を表すトレース減衰パラメータ $(0\le\lambda\le1)$ である．

Initialize $q_i(a)=0$ for all i, a
Repeat for each trial:
  Initialize $t=0$, s_t, $e_i(a)=0$ for all i, a
  Estimate Q-value from s_t using LWPLS
  Choose a_t using ε-greedy policy
  Repeat for each step:
    Update $e_i(a)$ using eq. (2)
    Take a_t, observe r_t, s_{t+1}
    Estimate Q-value from s_{t+1} using LWPLS
    Choose a_{t+1} using ε-greedy policy
    Update Q-value using eq. (1)
    If $\max_i w_{ii} < w_{th}$
      then create a new model
    $s_t \leftarrow s_{t+1}$, $a_t \leftarrow a_{t+1}$, $t \leftarrow t+1$
  until s_t is terminal state or over steps
until over trials

Fig.3 Sarsa(λ) algorithm using LWPLS

4.2 ε-greedy 方策

行動選択にはε-greedy 方策を用いる．ε-greedy 方策では，εの確率でランダムな行動を選択し，1-εの確率でその状態で最もQ値の大きな行動を選択する．

5. 移動障害物回避問題への適用

移動障害物回避問題とは，移動ロボットがスタート位置から移動障害物を避けてゴールを目指すものである．静止障害物の回避と異なり，移動ロボットの位置情報だけでは学習しにくいため，障害物までの距離情報を何らかの方法で取得することが必要となる．ここでは，測域センサを搭載した移動ロボットを想定してシミュレーション実験を行った．

5.1 問題設定

移動ロボットは自己位置と進行方向の推定が可能と仮定し，位置 x[cm]∈[0,150], y[cm]∈[0,36], x 軸と進行方向のなす角度 arg[rad]∈[-π,π]を取得できる．さらに，測域センサを仮定した距離センサから，進行方向とそれを中心にπ/8[rad]ごとに左右それぞれに 3 方向，計 7 方向の距離を取得できる(Fig.4)．距離は測域センサの誤差を考慮して 50[cm]を最大の入力とする．これらの入力により 10 次元の連続状態空間を構成する．なお，全ての入力は 0 から 1 に正規化して取り扱う．移動ロボットの直径は 8[cm]で，初期位置は (x,y)= (10,18)，arg=0[rad]とする．ゴール条件は x>90[cm]とする．行動は前進，右折，左折の 3 つとし，車輪の速度で定める．今回用いた設定は，前進が両車輪 10[cm/s]，右折が左車輪 10[cm/s]で右車輪 2[cm/s]，左折が左車輪 2[cm/s]で右車輪 10[cm/s]である．報酬はゴール到達で +30，壁や障害物への衝突で-30 を与え，その他は 0 とする．衝突時にはスタート位置から再スタートする．

ゴールに到着すると 1 回の試行が終了する．1 回の試行の最大ステップ数は 3,000 で，1 回のシミュレーションで 100 回の試行を行う．最大ステップ数に達するとゴール到達ができなくても次の試行を開始する．

移動障害物の配置は Fig.5 に示すように，2 つのパターンがあり，偶数番目の試行と奇数番目の試行で使い分ける．移動ロボットに対して手前の障害物①は x=35[cm]の位置に置かれ，奥の障害物②は x=70~100[cm]の範囲の実数値を各試行開始時にランダムに設定する．移動障害物は試行開始とともに-x 軸方向に 2[cm/s]の速度で移動する．移動ロボットが再スタートする場合はその試行時での初期位置に戻されるものとする．

シミュレータとして ENKI[8]を用い，プログラムの追加・修正を行った．1 単位ステップを 0.1[s]とし，行動はその間継続される．なお，シミュレーションの物理学計算は 0.02[s]ごとに行われ，このとき，地面との摩擦，加速特性等を考慮した．

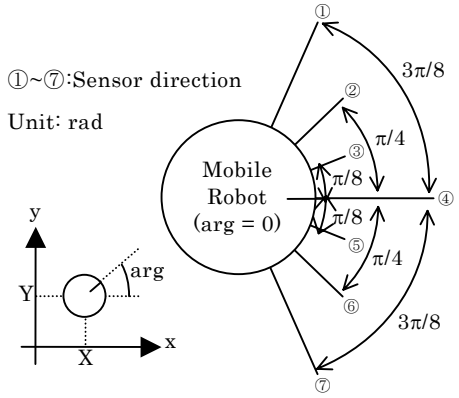


Fig.4　Sensor directions

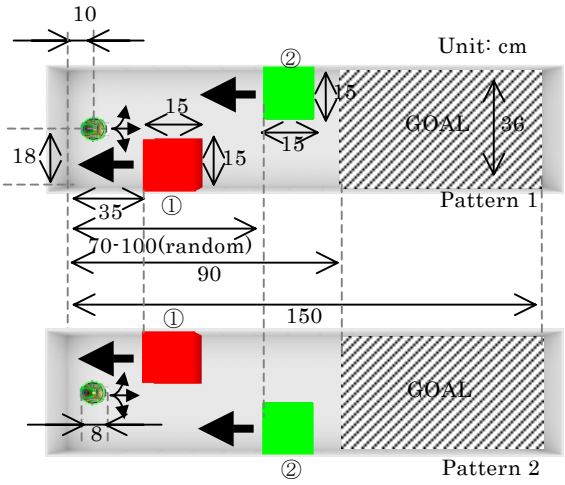


Fig.5　An environment

5.2 比較実験

提案手法（LWPLS を用いた Sarsa(λ)学習）と，従来法としてタイルコーディング(CMACs)*を用いた Sarsa(λ)学習 [2)3)] をシミュレーション実験で比較する．

タイルコーディング(tile coding)とは，格子状のタイリング(tiling)を何枚か重ねることで連続的な関数に近い近似を得る有名な手法である(Fig.6)．同じメモリ量でも，単純に状態空間を離散化するのに比べて高い解像度を得ることができ，学習性能もよいといわれている．しかし，タイル幅と形は入力次元の範囲に適するように選ばなければならない．また，タイリング数とタイルの密度を適切に決める必要がある．タイリングの密度を高くすれば，目的関数はより細かく正確に近似できるが，計算コストはより大きくなる．

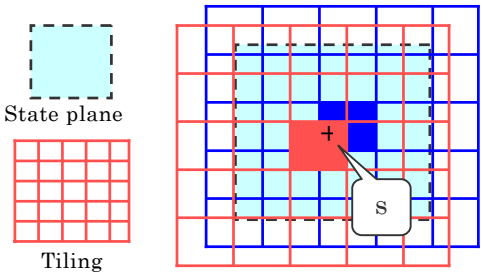


Fig.6　Tile coding (2D)

5.3 パラメータ設定

予備実験によって Sarsa(λ)学習と関数近似手法の各パラメータを決定した．

提案手法では，LWPLS のパラメータとして重み係数行列 **D** の対角成分を全て 300，モデル追加の閾値 w_{th} を 0.2，回帰数 r を 10 とした．今回は次元圧縮を行わないため，回帰数を入力次元と同じである．また，強化学習のパラメータは学習率αが 0.7，割引率γが 0.95，トレース減衰パラメータλが 0.9，ε-greedy 方策のεを 0.1 とした．εは試行回数の増加に伴って直線的に減少し，最終試行時では 0 となるよう設定した．

従来法では，タイリングを各次元で 4 つに等分割し，それぞれランダムなオフセットで 8 枚重ねる．強化学習では学習率αを 0.2 とし，その他のパラメータは提案手法と同じにした．

5.4 実験結果・考察

両手法でシミュレーション実験を 30 回行った．

まず，Fig.7 に平均学習曲線を示す．横軸が学習の試行回数，縦軸がゴールまでのステップ数で，ステップ数が少ないほど，より早くゴールに到達したことを示す．最終的には，提案手法が約 116.8 ステップ，従来法が約 140.0 ステップであった．また，提案手法がうまく収束しているのに対し，従来法では振動的で収束しきれていない．よって，提案手法がより高い学習性能を示したといえる．

次に，衝突回数の累計を Fig.8 に示す．横軸が学習の試行回数，縦軸が累計値である．最終的に提案手法が約 63.9 回，従来法が約 57.2 回となった．衝突回数では従来法の方が若干少ない．しかし，グラフの傾きをみれば，提案手法が序盤で衝突が多く，終盤では収束しているのに対し，従来法は中盤から直線的に増加し，収束していない．Fig.7 の学習曲線が振動的なのは，この点からも確認できる．つまり，動的に変化する環境に適応しきれていないと考えることができる．

Fig.9 は提案手法の平均モデル数である．横軸が学習の試行回数，縦軸がモデル数の累計である．Fig.7 とあわせて考察すれば，学習が収束するのに伴って追加されるモデルが減り，収束していることが分かる．これは対象とする問題に対して，状態空間の探索が行われる過程を示す 1 つの指標といえる．最終的に約 3,306 個のモデルを使用した．タイルコーディングでは，タイル数を L，j 次元での分割数を k_j とすると，次式が提案手法のモデル数に相当する．

$$L\prod_j k_j = 8\times 4^{10} \cong 8.39\times 10^6 \qquad (3)$$

今回のパラメータ設定では，約 839 万になる．なお，タイル数は学習開始から終

---

* CMAC(Cerebellar Model Arithmetic Computer)は小脳皮質内の情報処理機構の数学的モデルとして，J.S.Albus により提案された．複雑な入力空間マップの要素を分散記憶させる．

了まで一定である．

Intel(R) Xeon(R) E7330 2.4GHz の CPU を搭載した計算機でシミュレーション実験を行った結果，シミュレーション 1 回に要した平均時間は，提案手法が約 10 分であったのに対して，従来法では約 10 時間であった．このときの平均総ステップ数は，提案手法が約 15,724.5 ステップ，従来法が約 16,338.2 ステップである．このことから，1 ステップあたりの実計算時間は提案手法が約 0.04[s]，従来法が約 2.20[s]となる．

ここで重要なのは，1 ステップあたりの実計算時間が，行動選択のサンプリング時間である 0.1[s]以内かどうかである．提案手法は十分実現可能な実計算時間であるのに対し，従来法はそれを大幅に超過している．このことから，提案手法の方が実問題に適用する上ではるかに現実的といえる．

最後に，Fig.10 は提案手法の最終試行時の移動ロボットの位置および実行した行動を示している．序盤の x＝30[cm]あたりまでは左右の行動を組み合わせながら 1 つ目の障害物をうまく回避している．その後，中盤の x＝60[cm]あたりまでは直進している．終盤は，2 つ目の障害物を左右の行動を組み合わせてうまく回避し，ゴールに達している．結果として，移動障害物を回避しながらゴールを目指す行動パターンが学習できたことを確認できた．

6. まとめ

関数近似手法として LWPLS を Sarsa(λ)学習に取り入れ，局所重みから適格度を決定する手法を提案し，その有効性をシミュレーション実験で確認した．タイルコーディングと異なり，提案手法は直接連続状態を扱うことができ，タイル形状や重ね方等を考える必要がない．また，高次元連続状態空間を扱う制御を実機で行うにあたり，組み込み対象の計算資源の観点から，提案手法は現実的な手法の 1 つといえる．

今後の展望として，提案手法の利点を生かし，高次元連続状態を用いる複雑な問題を検討し，それに適用することや，実機への組み込みなどが挙げられる．

7. 謝辞

本研究は，科研費(19500172)の助成を受けたものである．

参考文献等

1) 浅田 稔: 強化学習の実ロボットへの応用とその課題，人工知能学会誌，Vol.12，No.6，831/836

2) R. S. Sutton and A. G. Barto 著，三上貞芳，皆川雅章訳: 強化学習，森北出版(2000)

3) R.S. Sutton: Generalization in Reinforcement Learning: Successful Examples Using Sparse Coarse Cording, Advances in Neural Information Processing System 8, 1038/1044, MIT Press (1996)

4) Jun Li and Tom Duckett: Growing RBF Networks for Learning Reactive Behaviours in Mobile Robotics, Vehicle Autonomous Systems, 285/307 (2006)

5) S. Schaal, C. G. Atkeson, S. Vijayakumar: Scalable Techniques from Nonparametric Statistics for Real Time Robot Learning, Applied Intelligence – Special issue on Scalable Robotic Applications of Neural Networks, vol. 17, no.1, 49/60 (2002)

6) 一井宏次，釜谷博行: 強化学習のための局所重み付き回帰手法を用いた価値関数近似，平成 20 年度電気関係学会東北支部連合大会講演論文集，228 (2008)

7) 一井宏次，釜谷博行，阿部健一: 局所重み付き回帰手法を用いた強化学習，平成 21 年電気学会全国大会講演論文集(第 3 分冊)，139/140 (2009)

8) http://www.e-puck.org/

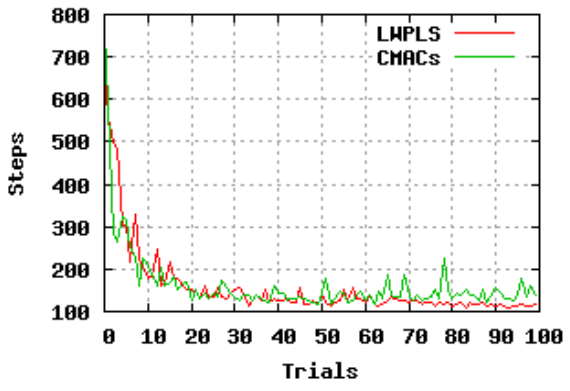


Fig.7 Average of 30 learning curves

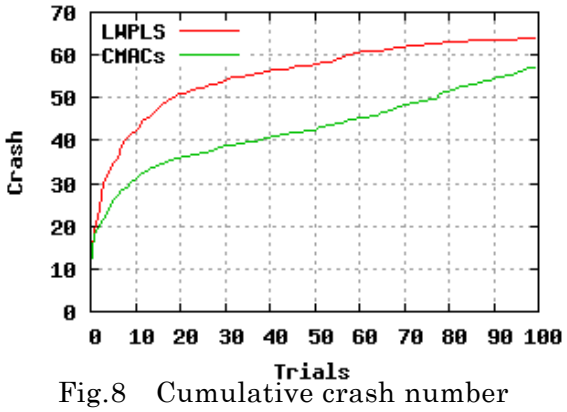


Fig.8 Cumulative crash number

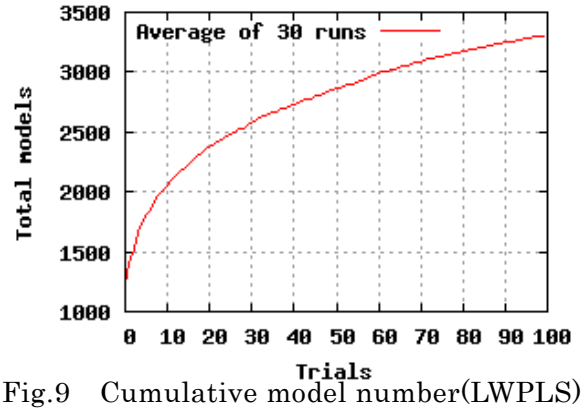
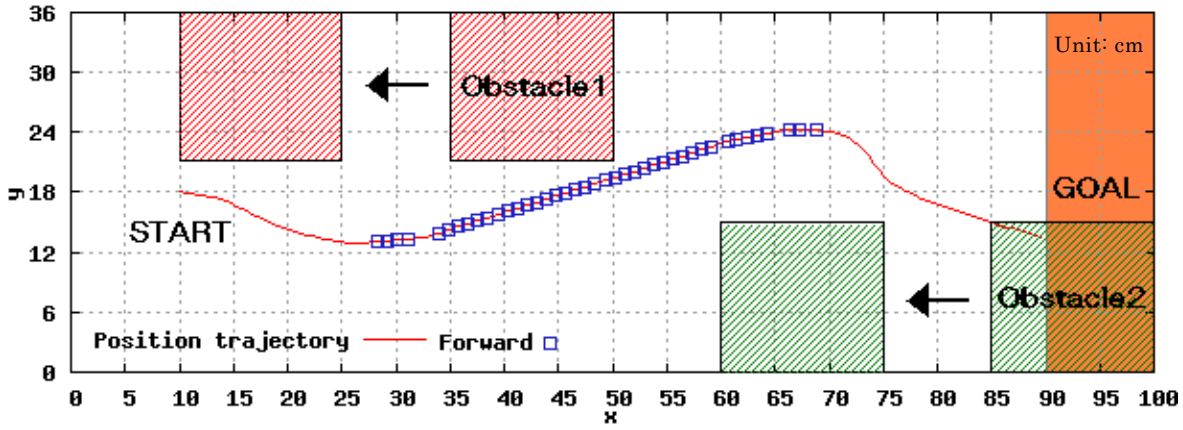
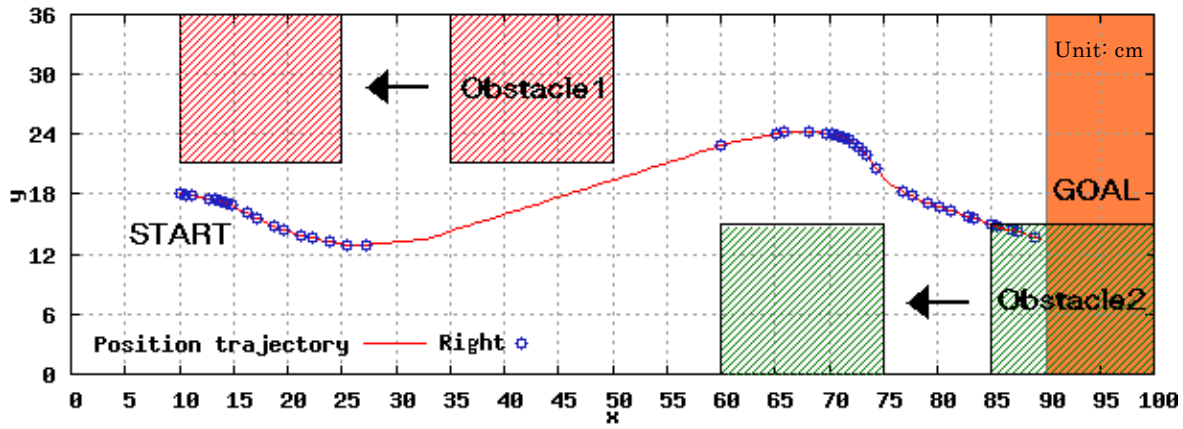


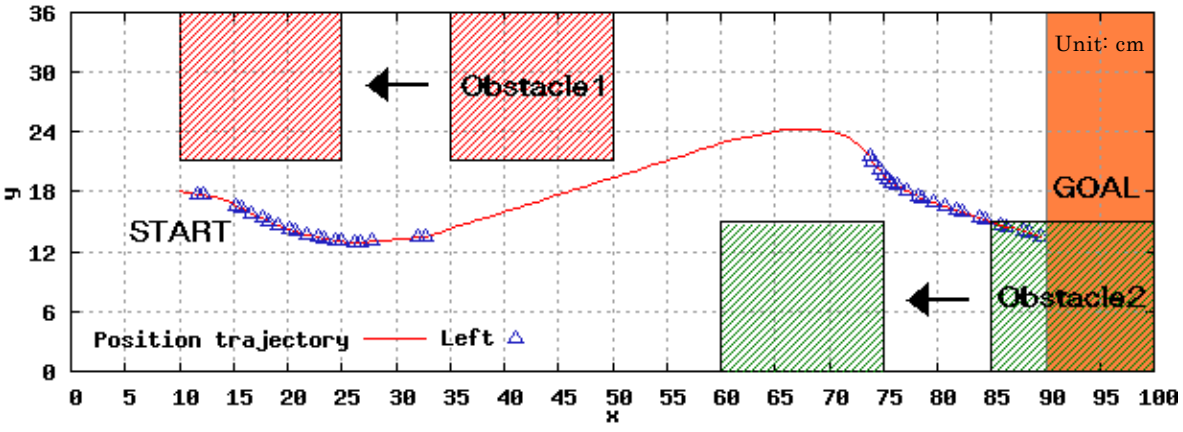
Fig.9 Cumulative model number(LWPLS)



(a) Forward action



(b) Right action



(c) Left action

Fig.10 Obtained actions (LWPLS)