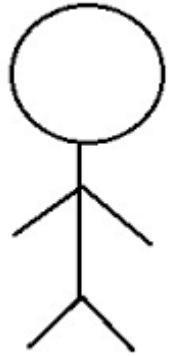


# 適応性と汎化性を考慮した 自己組織化適応制御器

平成30年 6月11日

富山県立大学 電子情報工学科4年 山本聖也

# 1. まえがき



高い適応力と汎化能力を持つ



本研究の目標  
この特性を持つ**制御器**をつくる！

(1) 制御対象の特性の変化  
に対応し得る制御器の構築



モジュール構造を持つ  
学習システムを用いる

(2) できるだけ少ない標本  
数からの汎化的な制御能力



mnSOMをもとにすることで少  
ない標本数から中間的なシ  
ステムを多数獲得できる

## 2. Self-Organizing Adaptive Controller (SOAC)

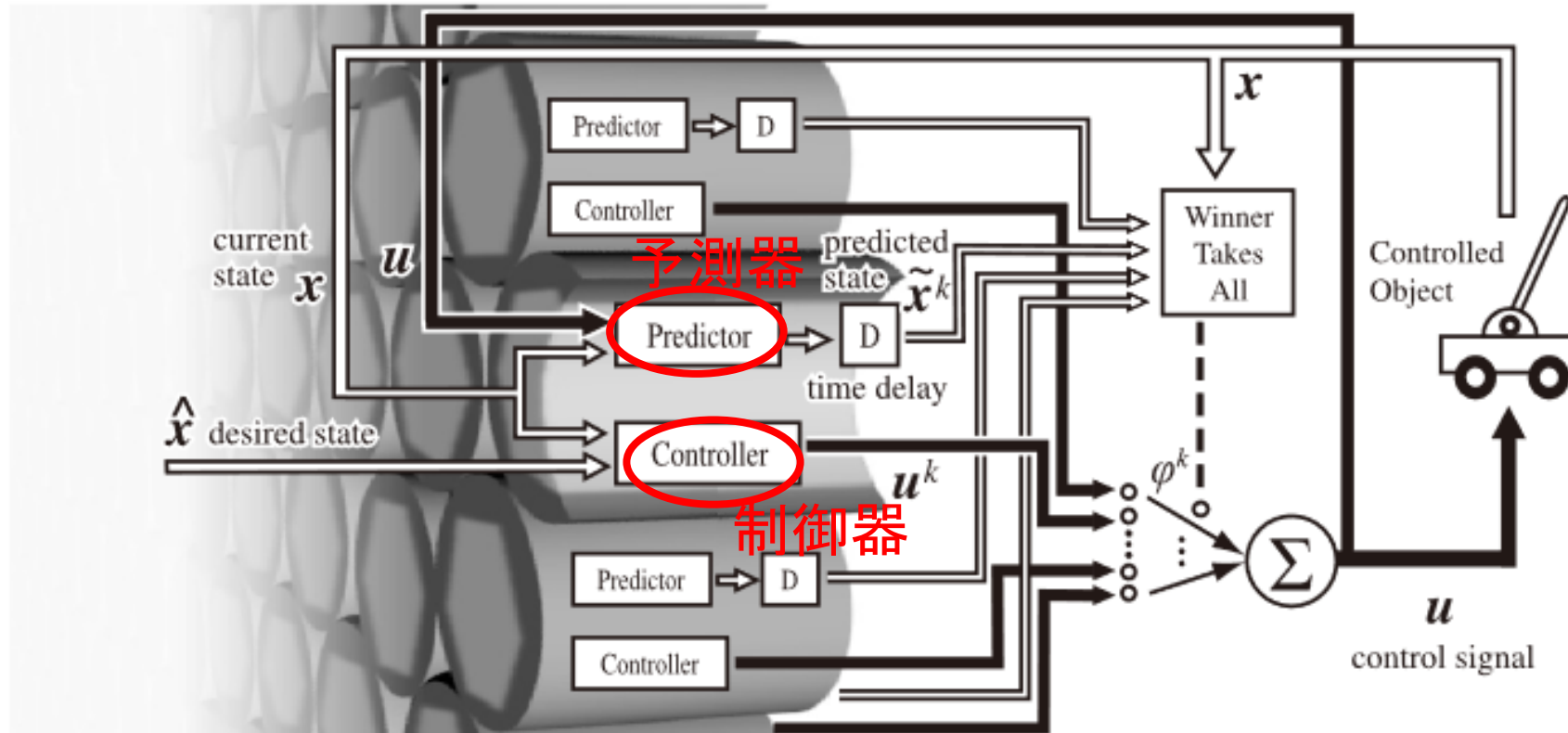


図 1 自己組織化適応制御器の基本構成図

Fig. 1 Scheme of the self-organizing adaptive controller.

- 予測器の出力

$$\tilde{x}^k(t + \Delta t) = {}^p f^k(x(t), u(t))$$

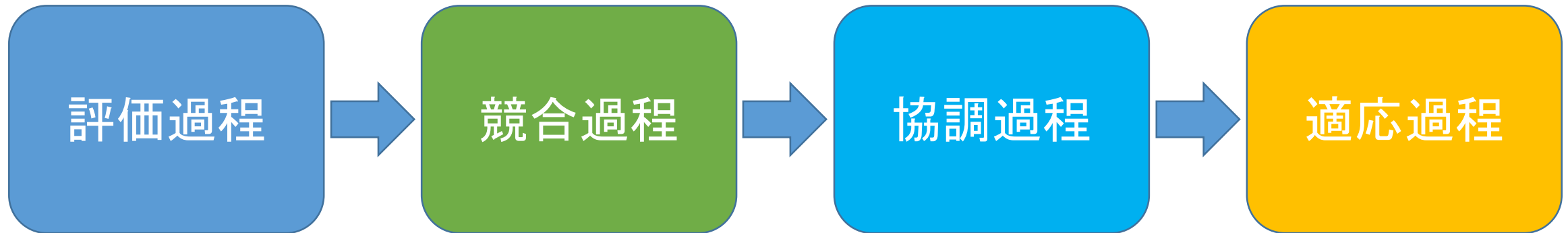
- $x(t)$ : 制御対象の現在の状態
- $\tilde{x}^k(t + \Delta t)$ :  $\Delta t$ 秒後の制御状態の予測値
- $u(t)$ : 制御信号

- 制御器の出力

$$u^k(t) = {}^c f^k(x(t), \hat{x}(t))$$

- $\hat{x}(t)$ : 目標の状態

- 学習モード(予測器)



## 評価過程

各予測器の出力と訓練データの誤差を1個すべてに対して求める

$${}^pE_i^k = \frac{1}{T} \int_0^T \|x_i(t) - \tilde{x}_i^k(t)\|^2 dt$$

$\tilde{x}_i^k(t)$ ,  ${}^pE_i^k$ : i-th 訓練データに対する k-th 予測器の出力と平均予測誤差  
T: 時系列の長さ

## 競合過程

平均予測誤差を最小とする  
モジュール

→ 最適モジュール (BMM)  
を求める

$$*_i = \arg \min_k {}^pE_i^k$$

## 協調過程

近傍関数によって学習分配率 $\psi_i^k$ を決める

$$\psi_i^k = \frac{\exp[-\|\xi^k - \xi_i^*\|^2 / 2\sigma^2]}{\sum_{i'=1}^I \exp[-\|\xi^k - \xi_{i'}^*\|^2 / 2\sigma^2]}$$

$\xi^k, \xi_i^*$  :それぞれk-thモジュールに対するBMMのマップ空間における座標

$\sigma$  : 近傍関数の半径を決めるパラメータ

## 適応過程

$\psi_i^k$ を用いて予測器の結合加重を次式によって更新

$$\Delta^p w^k = -\eta \sum_{i=1}^I \psi_i^k \frac{\partial^p E_i^k}{\partial p w^k}$$

$\eta$  : 学習率を表す正の定数

- 学習モード(制御器)  
SOACのモジュールとしてフィードバック  
誤差学習を用いる  
線形フィードバック制御器(CFC)と  
ニューラルネットワーク  
制御器(NNC)で構成されている

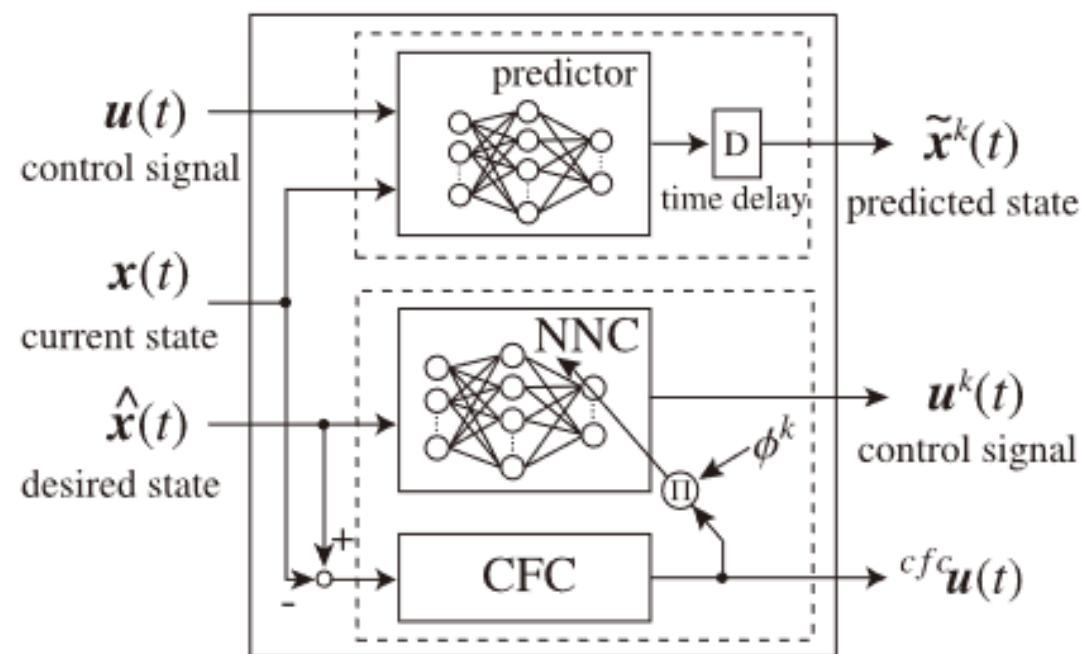


図 2 モジュールのブロック線図

Fig. 2 Block diagram of a module.

## • 2.4実行モード

制御対象の挙動 $x(t)$ と各予測器が予測した挙動 $\hat{x}(t)$ との誤差

$$p_e^k(t) = (1-\varepsilon)p_e^k(t-\Delta t) + \varepsilon \|x(t) - \tilde{x}^k(t)\|^2$$

BMM(再適合モジュール)

$$*(t) = \arg \min_k p_e^k(t)$$

各モジュールの責任信号

$$\phi^k = \frac{\exp[-\|\xi^k - \xi^*\|^2/2\sigma_\infty^2]}{\sum_{k'=1}^K \exp[-\|\xi^{k'} - \xi^*\|^2/2\sigma_\infty^2]}$$

$\sigma_\infty$ : 近傍半径の最終値

BMMが最も責任信号が大きく, マップ空間においてBMMからの距離が遠いモジュールほど責任信号が小さくなる

- 最終的に、制御対象へ与えられる制御信号は、NCCの出力と責任信号の積和にCFCの出力  ${}^{cfc}u(t)$  を加えることで得られる.

$$u(t) = \sum_{k=1}^K \phi^k u^k(t) + {}^{cfc}u(t)$$

$$u^k(t) = {}^cf^k(\hat{x}(t))$$

$${}^{cfc}u(t) = {}^{cfc}W(\hat{x}(t) - x(t)).$$

- このとき、NNCの学習則は重みベクトルを  ${}^c\omega^k$  として  ${}^{cfc}u(t)$  と  $\Phi^k$  を用いて次のように表される.

$$\Delta {}^cw^k = \eta \cdot \phi^k \frac{\partial {}^cf^k}{\partial {}^cw^k} \cdot {}^{cfc}u$$

# 3. シミュレーション

- 3.1 MPFIMとの比較

ばね・マス・ダンパ系を用いてその軌道誤差対して  
類似手法であるMPFIMとSOACの学習を行いそれをテストする

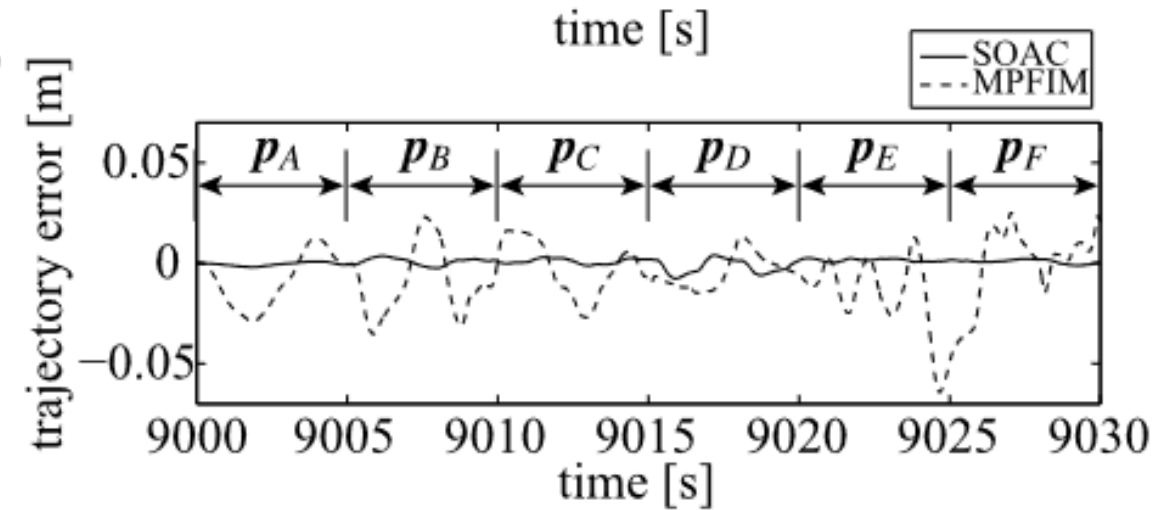
9種類の制御対象を用いてそれぞれ学習



6種類の対象を用いてテスト

- シミュレーション結果

- 上の図はそれぞれ制御を行いそれをグラフにプロットしたもののSOACの方が軌道誤差が小さいことが分かる



- 下の図はモジュール数を変更した場合の結果SOACの方が軌道誤差が小さい

Mean absolute error (MAE)

Module #		9	25	49	81
† MAE	MPFIM	0.0628	0.2152	0.3259	0.3519
	SOAC	0.0036	0.0023	0.0014	0.0014
	CFC	0.1120			

† average of 10 trials

### • 3.2倒立振子への応用

SOACがもつシステムの可視化機能についての検証を行うため、

倒立振子を用いてシミュレーションを行う。

右の図は学習モードで得られた結果

右上と左下の波形を見比べると

右上

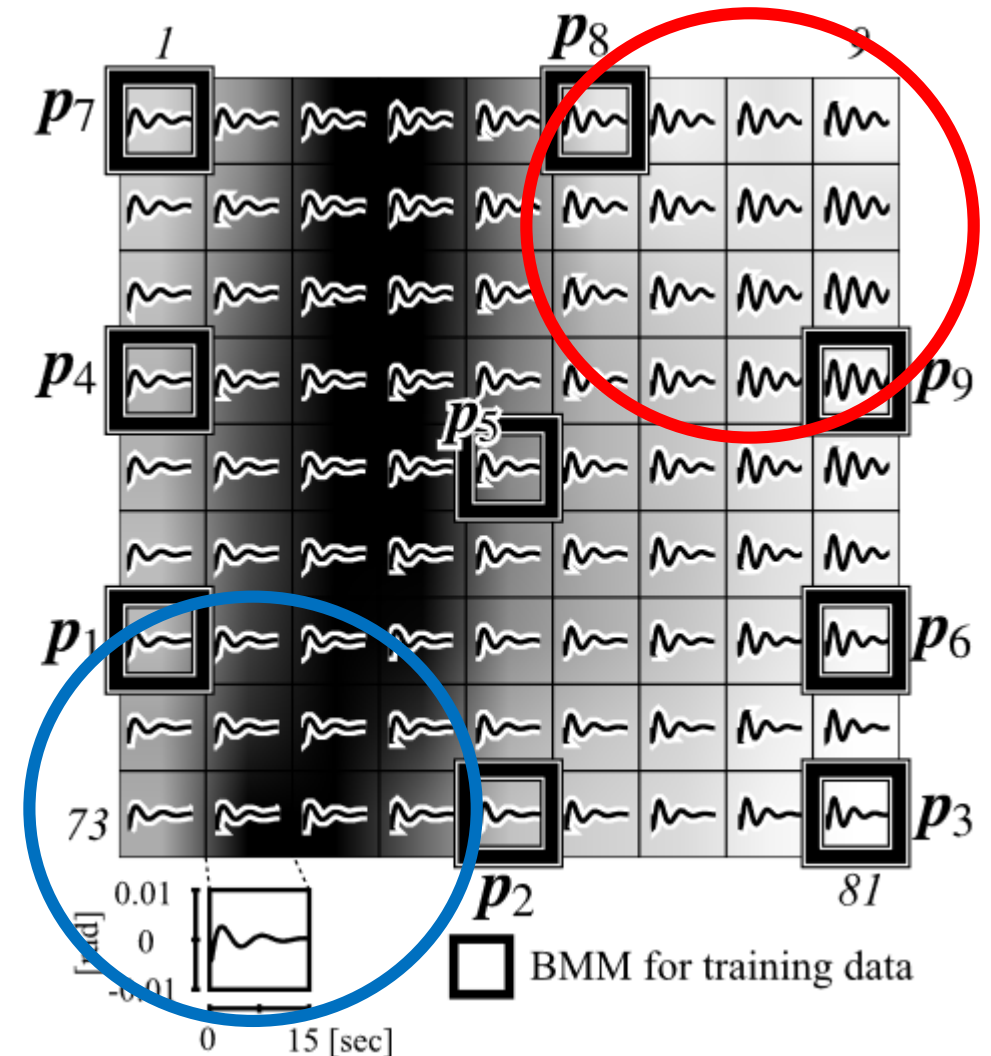
振子減衰の遅い特性をもつモジュール

左上

振子減衰の早い特性をもつモジュール

となっているためシステムの違いを

反映したマップが生成されている



- 右の図は実行モードの際の倒立振子の物理パラメータを可視化したもの

パラメータがゆっくりと変化する場合

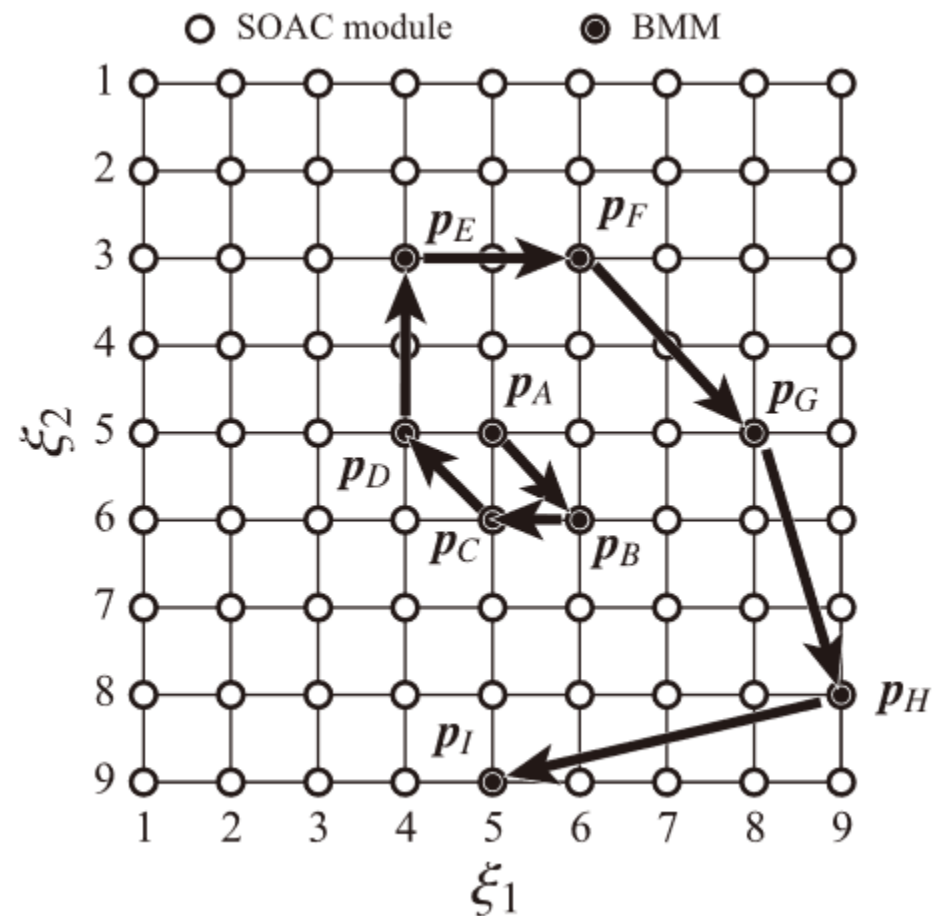
担当モジュールの変化もゆっくり,

パラメータの変化が大きい場合

担当モジュールは遠くのモジュールへと切り替わる.

その結果として安定した

制御を実現できた.



### • 3.2.2 SOACを用いたシステム解析例

システム解析の例としてパラメータ推定を行う

学習では用いていない振子 ( $p=[1.60, 1.50]$ ) ([長さ, 重さ]) を用意  
これをSOACで制御を行う際に振子の長さや重さの情報は与えずに  
位置, 速度などのセンサ情報から予測誤差に基づきBMMを決定

実際のパラメータ推定値は

$$\hat{p}^* = [1.65, 1.54]$$

となり実際のパラメータ推定値とかなり近い値となったので  
パラメータ推定の有効性を示すことができた

## 5. むすび

- シミュレーションの結果SOACは従来法よりも高い汎化性を持つことが分かった.
- 倒立振子に対して安定した制御かつパラメータの可視化に成功
- 今後の課題としてはSOACの安定性に関する考察と非線形課題への応用, そしてオンライン学習へ向けたアルゴリズムの修正などが挙げられる.