Organisation de réseaux de neurones pour l'apprentissage de fonctions robotiques complexes

Gilles HERMANN, Patrice WIRA, Jean-Luc BUESSLER

ABSTRACT

Ce papier considère le problème général de l'estimation de fonctions en utilisant une approche neuronale modulaire. Nous proposons d'utiliser des sous-réseaux de neurones indépendants pour apprendre des fonctions complexes. Ainsi, la fonction approximée est décomposée et le problème revient à estimer différentes sous-fonctions élémentaires plutôt qu'à estimer une fonction unique avec réseau de neurones unique. Cette décomposition modulaire nous permet d'introduire un certain nombre de connaissances a priori. Les sous-réseaux indépendants sont issus du schéma d'apprentissage bidirectionnel. Composée de cartes auto-organisatrices, l'approche modulaire utilisée est appliquée à un problème de contrôle robotique : une tâche de positionnement.

Keywords: Réseaux de neurones artificiels, apprentissage modulaire, carte auto-organisatrice, robotique, contrôle adaptatif, asservissement visuel.

1. INTRODUCTION

L'une des difficultés rencontrées dans l'application des réseaux de neurones artificiels (ANN) est l'estimation de fonctions complexes, telles que celles que nous pouvons rencontrer dans le monde réel. Généralement, l'absence de connaissances a priori sur lesdites fonctions implique une solution triviale qui est l'emploi d'un unique réseau de neurones. Cette méthode a le net désavantage de nécessiter des ANN de dimensions importantes. Dans ce cas, la convergence du réseau n'est plus garantie et son temps de réponse devient restrictif. La modularité peut être vue comme une solution alternative élégante.

Un résumé des différents concepts de modularité est proposé dans (Caelli et al. 1999). Les systèmes neuronaux modulaires peuvent être abordés comme une collection de sous-réseaux. Deux approches peuvent servir à l'élaboration de tels réseaux.

La première est basée sur des algorithmes qui peuvent générer des réseaux, des sous réseaux, des topologies, des connections, des poids, qui satisfont des contraintes spécifiques. La seconde approche inclut des réseaux avec des sous-réseaux fonctionnellement indépendants et qui sont conçus pour avoir des fonctions, des communications, des caractéristiques d'adaptations, spécifiques. Les réseaux de réseaux font partie de la première approche. Un tel système est alors vu comme un réseau hiérarchique. À chaque unité d'un réseau hiérarchiquement supérieur est associé un sous-réseau. Celui-ci va permettre de subdiviser le travail et d'obtenir une réponse plus fine. Cette structure est généralement bien adaptée dans les problèmes où la décomposition est évidente, et trouve plus particulièrement sa place dans la classification.

La seconde approche inclut deux types de décomposition, l'approche des mélanges d'experts (Jacobs et al. 1991), et les sous-réseaux fonctionnellement indépendants.

Les mélanges d'experts sont particulièrement populaires. Le principe est de choisir le module qui répond au mieux pour un environnement donné, parmi un ensemble de modules mis en parallèle. Un concept similaire est développé dans (Haruno et al. 2001). Cette architecture est basée sur de multiples paires de modèles directs et inverses. Ce type d'organisation n'accepte pas des connaissances a priori.

À contrario, les réseaux fonctionnellement indépendants sont compatibles avec des connaissances préalables de la fonction à apprendre. Cette approche revient à décomposer la fonction et à estimer différentes sous fonctions élémentaires. Une illustration peut être trouvée dans (Miyamoto et al. 1996). Ce concept permet de décomposer une fonction complexe en sous fonctions, et d'en construire des représentations internes.

Nous avons adopté cette approche, d'une part pour sa capacité à intégrer des connaissances a priori, et d'autre part pour sa fiabilité mise en évidence dans de récents travaux (Buessler et al. 1999). L'architecture modulaire est composée de cartes auto-organisées (SOM) dans lesquelles chaque neurone est associé à un modèle linéaire local (LLM). Un module de l'architecture sera donc un SOM-LLM, décrit dans la prochaine section.

Dans la Section 2, nous décrivons le principe de l'apprentissage bidirectionnel modulaire basé sur des SOM. La Section 3 formule le problème du contrôle robotique et valide l'approche modulaire pour cette application. La Section 4 met en évidence la validité de l'approche proposée par plusieurs simulations. Nous concluons dans la Section 5 par un résumé et en donnant les prochaines grandes lignes de nos travaux.

2. APPRENTISSAGE NEURONAL MODULAIRE

1. L'algorithme SOM – LLM

De nombreux algorithmes conviennent parfaitement à l'implémentation d'un module. Nous avons retenu, ici, deux algorithmes simples et bien connus : la carte de Kohonen étendue et l'Adaline. De manière générale, l'objectif est de réaliser l'estimation d'une fonction en utilisant une carte linéaire locale.

Les neurones, soumis à un apprentissage supervisé, sont organisés sur un treillis prédéfini $\mathcal{M}$ de dimension fixée qui lui, est soumis au principe d'auto-organisation. Chaque neurone de la carte est défini par une position $\mathbf{j}$, et est associé à trois éléments principaux :

- un vecteur poids d'entrées $\mathbf{w}_{\mathbf{j},k}^{(\text{in})}$ de l'espace $\mathcal{X} \in \mathbb{R}^n$,
- un vecteur poids de sorties $\mathbf{w}_{\mathbf{j},k}^{(\text{out})}$ de l'espace $\mathcal{Y} \in \mathbb{R}^m$,
- une matrice poids $\mathbf{A}_{\mathbf{j},k}$ issue de l'espace de sortie.

L'auto-organisation des neurones permet de discrétiser l'espace d'entrées et l'espace de sorties, respectivement avec $\mathbf{w}_{\mathbf{j},k}^{(\text{in})}$ et $\mathbf{w}_{\mathbf{j},k}^{(\text{out})}$. L'apprentissage supervisé permet de linéariser de manière locale l'espace de sorties en utilisant les poids de la matrice $\mathbf{A}_{\mathbf{j},k}$.

À l'instant *k* un vainqueur $\mathbf{i}$, résulte de la compétition des neurones. Ce neurone est celui qui représente le mieux le vecteur d'entrées $\mathbf{x}_k \in \mathcal{X}$. Le neurone vainqueur est trouvé en minimisant la distance euclidienne :

$$\mathbf{i} = \arg\min_{\mathbf{j}} \left\| \mathbf{x}_k - \mathbf{w}_{\mathbf{j},k}^{(\text{in})} \right\|, \ \mathbf{j} \in \mathcal{M} \tag{1}$$

Définissons $h_{\mathbf{i},\mathbf{j},k}^{(\text{in})}$, $h_{\mathbf{i},\mathbf{j},k}^{(\text{out})}$ et $\Lambda_{\mathbf{i},\mathbf{j},k}$ comme les fonctions de voisinage centrées autour du neurone vainqueur $\mathbf{i}$, dans leurs espaces respectifs. Les poids synaptiques de chaque neurone sont adaptés d'après les règles suivantes :

$$\mathbf{w}_{\mathbf{j},\text{k}+1}^{(\text{in})} = \mathbf{w}_{\mathbf{j},\text{k}}^{(\text{in})} + \mu_k h_{\mathbf{i},\mathbf{j},\text{k}}^{(\text{in})} (\mathbf{x}_k - \mathbf{w}_{\mathbf{j},\text{k}}^{(\text{in})}) , \tag{2}$$

$$\mathbf{w}_{\mathbf{j},k+1}^{(\text{out})} = \mathbf{w}_{\mathbf{j},k}^{(\text{out})} + \lambda_k h_{\mathbf{i},\mathbf{j},\text{k}}^{(\text{out})} (\mathbf{y}_k - \hat{\mathbf{y}}_k) , \tag{3}$$

$$\mathbf{A}_{\mathbf{j},k+1} = \mathbf{A}_{\mathbf{j},k} + \alpha_k \Lambda_{\mathbf{i},\mathbf{j},k} \frac{\boldsymbol{\varepsilon}_k . \left(\mathbf{x}_k - \mathbf{w}_{\mathbf{j},k}^{(\text{in})} \right)^T}{\left\| \mathbf{x}_k - \mathbf{w}_{\mathbf{j},k}^{(\text{in})} \right\|^2 + c^2} \tag{4}$$

où $\boldsymbol{\varepsilon}_k = \mathbf{y}_k - \hat{\mathbf{y}}_k$ est l'erreur de positionnement issue de la comparaison entre la sortie estimée $\hat{\mathbf{y}}_k$ et la sortie désirée $\mathbf{y}_k$. Les coefficients μ_k , λ_k , α_k correspondent aux taux d'adaptation. Ils varient au cours du temps et leurs valeurs sont comprises entre 0 et 1. Les fonctions de voisinage $h_{\mathbf{i},\mathbf{j},k}^{(\text{in})}$, $h_{\mathbf{i},\mathbf{j},k}^{(\text{out})}$ et $\Lambda_{\mathbf{i},\mathbf{j},k}$ ont une expression gaussienne dont le rayon varie également au cours du temps :

$$\exp\left(-\frac{\left\| \mathbf{j} - \mathbf{i} \right\|^2}{2\sigma_k^2} \right), \sigma_k = \sigma_i \left(\frac{\sigma_f}{\sigma_i} \right)^{\frac{k}{k_{\max}}} , \tag{5}$$

où σ_i et σ_f sont les valeurs, initiale et finale, du rayon, et $k_{\max}$ l'instant où σ_f est atteint.

Les poids de la matrice $\mathbf{A}_{\mathbf{j},k}$ donnent une estimation de la fonction dérivée, basée sur la règle normalisée des moindres carrés (NLMS). À l'instant *k*, la sortie du réseau est :

$$\hat{\mathbf{y}}_k = \mathbf{w}_{\mathbf{i},k}^{(\text{out})} + \mathbf{A}_{\mathbf{i},k} \left(\mathbf{x}_k - \mathbf{w}_{\mathbf{i},k}^{(\text{in})} \right) \tag{6}$$

L'expression (6) représente les deux premiers moments du développement de Taylor qui permettent d'approximer tous types de fonction.

L'algorithme du SOM-LLM est très simple, convient parfaitement à une adaptation en ligne, et converge rapidement. Cette méthode s'avère être adéquate pour le contrôle robotique. Nous utiliserons le SOM-LLM dans notre architecture modulaire.

2. L'apprentissage bidirectionnel

Dans l'apprentissage modulaire, le problème relève de l'estimation de la transformation $\mathbf{f}$ entre deux espaces, $\mathcal{X} \in \mathbb{R}^n$ et $\mathcal{Y} \in \mathbb{R}^m$:

$$\mathbf{f}\ \mathcal{X}: \rightarrow \mathcal{Y}\ ,\ \mathbf{y}_k = \mathbf{f}(\mathbf{x}_k)\ . \tag{7}$$

Chaque décomposition neuronale est spécifique à une application donnée. Comme exemple, nous présentons le cas général de la décomposition séquentielle, où chaque module est un réseau de neurones SOM-LLM.

Dans ce cas bien particulier, le problème sera décomposé en deux modules A et B mis en série, comme le montre la Fig. 1. Décomposer une fonction en deux modules implique irrémédiablement l'estimation (l'utilisation) d'une grandeur interne. De ce fait des problèmes apparaissent quand il s'agit d'utiliser cette grandeur pour l'apprentissage des modules supervisés, et que celle-ci n'est pas accessible.

Pour outrepasser ce problème, nous utilisons un apprentissage bidirectionnel. Il s'agit d'introduire un nouveau module C, et de l'associer au module B. Il prend ses entrées dans l'espace de sortie $\mathcal{Y}$ et délivre une estimation $\mathbf{z}_k^{\mathrm{C}}$ de la grandeur interne, à l'instar du module A qui prend ses entrées dans l'espace des entrées $\mathcal{X}$ et délivre lui aussi une estimation $\mathbf{z}_k^{\mathrm{A}}$ de la grandeur interne.

Le module C peut être vu comme l'inverse du module B. Notons aussi que le module C ne participe aucunement à l'estimation de la sortie, mais uniquement au processus d'apprentissage.

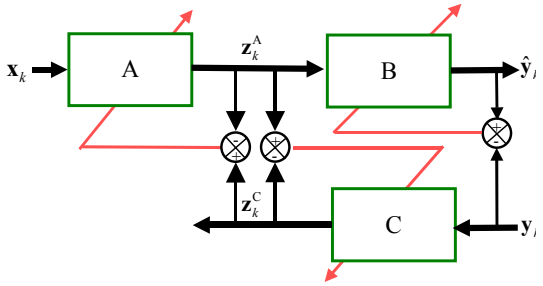

Fig. 1 : Architecture de l'apprentissage bidirectionnel

Les deux estimations de l'état interne, $\mathbf{z}_k^{\mathrm{A}}$ et $\mathbf{z}_k^{\mathrm{C}}$, servent à définir les signaux d'erreurs pour l'apprentissage des modules A et C. Le module A utilise l'erreur $\boldsymbol{\varepsilon}_k^{\mathrm{A}} = \mathbf{z}_k^{\mathrm{C}} - \mathbf{z}_k^{\mathrm{A}}$, et le module C utilise l'erreur $\boldsymbol{\varepsilon}_k^{\mathrm{C}} = \mathbf{z}_k^{\mathrm{A}} - \mathbf{z}_k^{\mathrm{C}} = -\boldsymbol{\varepsilon}_k^{\mathrm{A}}$. À l'instant *k*, le module B est supervisé par la différence entre la sortie désirée et la sortie calculée, c'est-à-dire $\boldsymbol{\varepsilon}_k^{\mathrm{B}} = \mathbf{y}_k - \hat{\mathbf{y}}_k$. C'est l'estimation $\mathbf{z}_k^{\mathrm{A}}$ qui sert d'entrée au module B, qui lui délivre l'estimation de la sortie $\hat{\mathbf{y}}_k$. La stabilité et la convergence de l'apprentissage de cette architecture modulaire est analysée dans (Buessler et al. 2002).

3. APPLICATION AU CONTRÔLE ROBOTIQUE

1. Le contexte

Notre application consiste en l'asservissement visuel d'un bras robotique à trois degrés de liberté dans son environnement 3D. La vision active est utilisée dans la boucle de retour : deux caméras montées sur une tête robotique à quatre degrés de liberté suivent l'effecteur du robot. Elles ne sont pas nécessairement centrées, et sont contrôlées séparément. Dans les expérimentations, chaque caméra tourne indépendamment autour des axes panoramique et azimutal.

Les deux caméras définissent l'espace visuel $\mathcal{V} \in \mathbb{R}^4$. À chaque itération *k*, le traitement d'image extrait la position de l'effecteur $\mathbf{v}_k \in \mathcal{V}$ et la position de la cible $\mathbf{d}_k \in \mathcal{V}$.

Les données visuelles sont fonction de l'orientation des caméras, c'est-à-dire des angles $\boldsymbol{\varphi}_k \in \Phi$ de la tête, où $\Phi \in \mathbb{R}^4$ est l'espace de ces angles.

En considérant $\Theta \in \mathbb{R}^3$ comme l'espace articulaire du robot et $\boldsymbol{\theta}_k \in \Theta$ un vecteur angles, nous pouvons définir la transformation cinématique directe $\mathbf{f}$:

$$\mathbf{f}\ :\ \Theta \rightarrow \mathcal{V},\ \mathbf{v}_k = \mathbf{f}(\boldsymbol{\theta}_k, \boldsymbol{\varphi}_k)\ . \tag{8}$$

Les coordonnées $\mathbf{v}_k$ de l'effecteur dans les images sont fonction des angles du bras $\boldsymbol{\theta}_k$ et de la position de la tête $\boldsymbol{\varphi}_k$.

Le contrôle robotique consiste à calculer la correction angulaire, en partant des mesures faites dans les images, ce qui permet de minimiser le signal d'erreur mesuré dans l'espace visuel. Ce contrôle est réalisé par l'estimation inverse $\mathbf{f}^{-1}$ de la transformation susmentionnée avec les techniques d'apprentissage. Ceci est référencé comme l'asservissement visuel adaptatif.

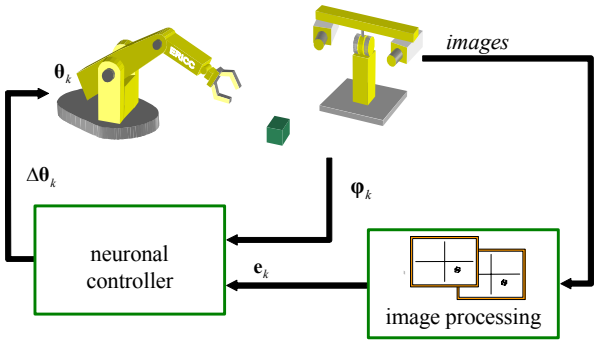

Fig. 2 : Le principe général de l'asservissement visuel

2. L'apprentissage modulaire robot—vision

La décomposition est maintenant adaptée à l'estimation de l'inverse du modèle cinématique défini par (8). De manière générale, les fonctions robotiques sont complexes, de dimension élevée, et fortement non linéaires. Afin d'en réduire la complexité, nous nous proposons d'utiliser des connaissances a priori. Ici, la décomposition séquentielle est tout à fait de mise, puisqu'elle permet de faire une estimation d'un état interne avec une représentation objective : les angles de la tête pour que les images soient centrées.

Soit $\hat{\boldsymbol{\varphi}}_k^c$ la configuration particulière des caméras qui permet de centrer un point de l'espace 3D. Le point est considéré centré quand la projection apparaît au centre des images des deux caméras. $\hat{\boldsymbol{\varphi}}_k^c$ est unique pour un point donné de l'espace 3D et peut donc être utilisé dans la décomposition de $\mathbf{f}^{-1}$. Chaque point non centré de l'espace 3D, défini dans l'espace d'entrées par $\mathbf{v}_k$ et $\boldsymbol{\varphi}_k$, respectivement les informations issues des images et les angles de la tête, peut être représenté par (réduit à) une configuration caméras centrées $\hat{\boldsymbol{\varphi}}_k^c$, un vecteur de plus petite dimension.

Estimer ce vecteur sert à construire un contrôleur robotique précis. Nous proposons la décomposition séquentielle de la Fig. 4 pour obtenir un espace interne qui correspond à la représentation avec des caméras centrées.

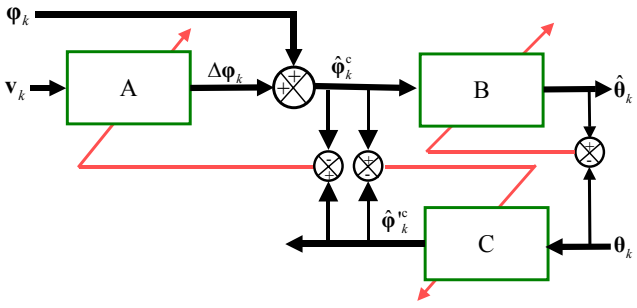

Fig. 3 : L'architecture neuronale proposée pour l'estimation du modèle cinématique inverse du système robot-vision.

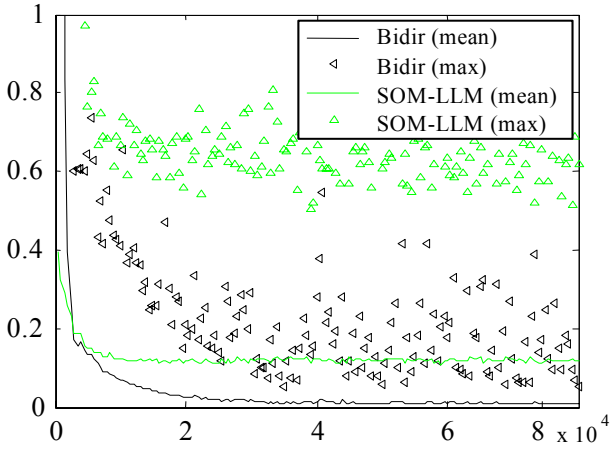

Fig. 4 : Erreurs sur les réponses des systèmes neuronaux pour l'estimation des angles du bras robotique (en rad).

Dans cette architecture, chaque module est un SOM-LLM. Le rôle du module A est de prendre un point non centré défini par le couple de vecteurs $\mathbf{v}_k$ et $\boldsymbol{\varphi}_k$, et d'en estimer la configuration caméras centrées $\hat{\boldsymbol{\varphi}}_k^c$. Le module C retourne une autre estimation des angles caméras centrées. La différence entre les deux estimations est utilisée comme signal d'erreur pour l'adaptation des poids des modules A et C.

4. RÉSULTATS

Les réseaux bidirectionnels de neurones artificiels n'ont pas de connaissances a priori sur les relations entre les données capteur et la commande issue du contrôleur. Pourtant ils sont conçus pour acquérir des connaissances suite à sa propre expérience. Les mouvements du bras corrélés aux déplacements dans les images sont utilisés pour l'apprentissage, en ligne et supervisé, du modèle cinématique inverse du robot.
L'utilisation d'un système à vision active, complique considérablement le problème par rapport à un système de vision statique. En effet, en plus des trois degrés de liberté du robot, il faut en ajouter 4 pour la tête qui supporte les deux caméras. Dans les simulations suivantes, nous allons comparer les performances d'un SOM-LLM unique à celles de l'architecture bidirectionnelle.
Le SOM-LLM est composé d'une carte de 1296 neurones. L'espace d'entrées est composé des informations visuelles et des angles des caméras, c'est-à-dire du vecteur $[\mathbf{v}_k \quad \boldsymbol{\varphi}_k]^T$. La sortie désirée correspond aux angles du bras robotique. Pour l'architecture modulaire, le module A est composé des trois cartes 1D de 12 neurones, et les modules B et C sont 3D et composés de 9*6*9 neurones. L'architecture globale utilise 1008 neurones, soit légèrement moins que le réseau SOM-LLM.
Les courbes de la Fig. 4 montrent l'amélioration des performances au cours de l'apprentissage.

Contrôleur	Erreur moyenne de positionnement dans l'espace 3D (mm)		
	X (max.)	Y (max.)	Z (max.)
SOM-LLM unique	6.536 (33.044)	14.178 (143.869)	13.643 (71.065)
Approche bidirectionnelle	1.410 (9.992)	0.469 (4.038)	0.742 (7.785)

Table 1 : La précision du positionnement caméras orientées.

Les réponses du réseau modulaire sont maintenant comparées aux positionnements de l'effecteur dans l'espace 3D par rapport à la position de la cible. La différence entre ces deux positions est appelée erreur de positionnement, et elle est représentée dans la Table 1. Nous constatons que les performances sont en faveur de l'architecture modulaire bidirectionnelle. La boucle de contrôle qui utilise l'architecture bidirectionnelle positionne l'effecteur du robot à 1.7 mm (en norme) dans l'espace 3D.
Le contrôleur basé sur le simple SOM-LLM souffre de deux inconvénients majeurs : la difficulté de déployer la carte dans l'espace d'entrées à six dimensions et la convergence de l'apprentissage. L'approche modulaire outrepasse ces difficultés à partir du moment où une décomposition cohérente et objective du problème est faite.
L'utilisation d'un système modulaire permet aux modules individuels de participer à l'apprentissage sans affecter le comportement appris par les autres modules.

5. CONCLUSION

Dans ce papier, nous proposons des sous réseaux fonctionnellement indépendants pour apprendre des fonctions complexes. Basée sur un schéma d'apprentissage bidirectionnel, cette approche modulaire neuronale est capable de prendre en compte des connaissances a priori et donc d'approximer n'importe quelle fonction complexe. Le problème est décomposé et revient à estimer des fonctions élémentaires à l'aide de modules à base de SOM-LLM. Un exemple est fourni : l'apprentissage du modèle cinématique inverse d'un système robotique. L'estimation faite permet de contrôler le robot dans une tâche de positionnement. Un contrôleur robotique basé sur le principe de l'asservissement visuel a été étudié, implémenté et validé. Respectant les contraintes de la cadence vidéo, l'approche modulaire assure le contrôle en utilisant un nombre de neurones moindre et avec une meilleure précision qu'un réseau unique.
Un avantage certain de l'approche proposée est que l'apprentissage d'un module n'affecte pas l'apprentissage des autres modules. Un module est donc autonome et en cas de décomposition objective, celui-ci peut être réutilisé. Dans notre application, les modules A et C peuvent être réutilisés pour le contrôle de la tête, pour centrer les cibles dans les images par exemple.

Nos perspectives concernent donc le développement de schémas de vision active plus sophistiqués, qui permettront de faire du suivi de cible de manière performante. Nous comptons valider ces résultats sur notre plate-forme robotique réelle.

RÉFÉRENCES

Buessler, J. L., Kara, R., Wira, P., Kihl, H., and Urban, J. P. "Multiple Self-Organizing Maps to facilitate the Learning of Visuo-Motor Correlations." *IEEE International Conference on Systems Man and Cybernetics*, Tokyo, Japan, 470-475.

Buessler, J.-L., Urban, J.-P., and Gresser, J. (2002). "Additive Composition of Supervised Self-Organized Maps." *Neural Processing Letters*, 15(1), 9-20.

Caelli, T., Guan, L., and Wen, W. (1999). "Modularity in neural computing." *Proceedings of the IEEE, Special Issue on Computational Intelligence*, 87(9), 1497-1518.

Haruno, M., Wolpert, D. M., and Kawato, M. (2001). "MOSAIC Model for Sensorimotor Learning and Control." *Neural Computation*, 13(10), 2201-2220.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. (1991). "Adaptive Mixtures of Local Experts." *Neural Computation*, 3(1), 79-87.

Miyamoto, H., Schaal, S., Gandolfo, F., Gomi, H., Koike, Y., Osu, R., Nakano, E., Wada, Y., and Kawato, M. (1996). "A Kendama Learning Robot Based on Bi-directional Theory." *Neural Networks*, 9(8), 1281-1302.