The DIAMOND Model: Deep Recurrent Neural Networks for Self-Organizing Robot Control

Simón C. Smith¹, Richard Dharmadi¹, Calum Imrie¹, Bailu Si²,³ and J. Michael Herrmann¹,²*

¹ Institute of Perception, Action and Behaviour (IPAB), School of Informatics, University of Edinburgh, Edinburgh, United Kingdom, ² State Key Laboratory of Robotics, Shenyang Institute of Automation, Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang, China, ³ School of Systems Science, Beijing Normal University, Beijing, China

The proposed architecture applies the principle of predictive coding and deep learning in a brain-inspired approach to robotic sensorimotor control. It is composed of many layers each of which is a recurrent network. The component networks can be spontaneously active due to the homeokinetic learning rule, a principle that has been studied previously for the purpose of self-organized generation of behavior. We present robotic simulations that illustrate the function of the network and show evidence that deeper networks enable more complex exploratory behavior.

Keywords: deep neural networks, autonomous learning, homeokinesis, self-organizing control, robot control

1. INTRODUCTION

Deep neural architectures (Fukushima and Miyake, 1980; Hinton et al., 2006) have reached a level comparable to human performance in certain pattern recognition tasks (Krizhevsky et al., 2012). Also in robotic applications, deep networks gain more and more importance, from state abstraction to seamless end-to-end control in complex repetitive tasks (Levine et al., 2016). Moreover, it has been speculated whether deep feed-forward networks can account for some aspects of information processing in the mammalian visual system (Serre et al., 2007), which is not to say that the brain is nothing but a collection of deep neural networks. Quite to the contrary, the brain is known to have dynamical properties that are much richer than standard deep architectures:

• Biological neural systems consist of patches of interconnected neurons which also receive re-entrant connectivity via other patches.
• Spontaneous behavior can occur at any level of depth and may spread in either direction.
• Sensory inputs are not only providing information for decision about actions, but are also analyzed for effects of previous actions.
• A hierarchical organization enables lateral transferability and flexible compositionality.
• There is little use for supervised learning.

Based on these considerations, we propose here an architecture that combines the undeniable strengths of deep neural networks with homeokinesis (Der, 2001), an approach to meet requirements of autonomous robots (see section 2). Our work connects to (Carvalho and Nolfi, 2016) where the introduction of flexibility and plasticity in a neural controller showed a good effect in a cleaning task, however, mainly based on an evolutionary approach, whereas we aim at a more principled architecture that achieves an increased flexibility by a hierarchy of identical controllers. The autonomously generate activity of higher-lever controllers provide an
intrinsic motivation (Oudeyer et al., 2007) for the lower
ones. In this way, we are able to propose a more brain-
like architecture which implicitly realizes a predictive coding
principle, compare (Adams et al., 2013) for a related approach,
at least in some parameter range, as discussed below. An early
interesting comparison is provided by (Rusu et al., 2003) which
presents a neuro-fuzzy controller for determining the behavior
of a robot in a navigation task. Their architecture had a similarly
layered structure, although the behaviors had to be predefined at
time when homeokinesis (Der, 2001) was just being developed.
More recently, differential Hebbian learning was used to explore
possible behaviors of a robot (Pinneri and Martius, 2018),
presenting a more brain-like approach at the low level, whereas
we aim a model that captures characteristics of the area-level
organization of the brain.

In the following, we will consider first the homeokinetically
controlled sensorimotor loop (Der, 2001) as the basic element
of the proposed system (section 2). In this way, we incorporate
a source of spontaneous activity. The composition of these
elements in the DIAMOND (Deep Integrated Architecture for
sensoriMotor self-Organization aNd Deliberation) architecture
(section 3) will thus be able to generate activity at all levels
and work in a fully self-supervised way, although it is also
possible to steer the system to desired behavior by very small
guiding inputs (Martius and Herrmann, 2011). The main layout
of the architecture includes a basic layer that receives information
from outside world and sends actions and is expected to
represent low-level features. There is a variable number of
deeper layers that interact only with the neighboring layers and
which represent more abstract features that are extracted from
the data through the lower layers. The architecture learns by
the homeokinetic learning rule (see below) which implies that
consistency between neighboring layers is required. We will
present a few experimental results in section 4, and discuss the
realism and performance of the architecture as well as further
work in section 5.

2. HOMEOKINETIC CONTROL

The basic element of our architecture is formed by a
homeokinetic controller, which we will describe here only
briefly, details can be found in (Der and Martius, 2012). This
unsupervised active learning control algorithm shapes the
interaction between a robot and its environment by updating
the parameters of a controller and of an internal model. The learning
goal can be characterized as a balance of predictability and
sensitivity with respect to future inputs. The resulting behavior is
random yet coherent both temporally and across multiple degrees
of freedom. The controller is a parametric function

\[ y_t = C(x_t; C) \]  

of the vector \( x_t \) of current sensory states of the robot. It generates
a vector of motor commands \( y_t \) in dependence on the current
values of the parameter matrix \( C \). The update of the parameters
is based on the sensitivity of the distance between inputs and their
predictions by means of an internal model. This model

\[ \hat{x}_{t+1} = M(x_t, y_t; M), \]  

produces a prediction of future states \( \hat{x}_{t+1} \) based on the current
input \( x_t \) or action \( y_t \) or both, and a parameter matrix \( M \).
The difference between actual and estimated state defines the prediction error

\[ \xi_{t+1} = x_{t+1} - \hat{x}_{t+1}, \]  

which gives rise to one of the two complementary objective functions that are relevant here, firstly the prediction error

\[ \mathcal{E}_{t+1} = \| x_{t+1} - \hat{x}_{t+1} \|^2, \]  

which is used to adapt the parameters \( M \) of the internal model
(2), and secondly the time loop error

\[ E_t = \| x_t - \hat{x}_t \|^2, \]  

which is based on a post-diction \( \hat{x}_t \) of previous input \( x_t \) obtained
via the inverse of Equation (2) given the new input \( x_{t+1} \), i.e., \( E_t \) is
calculated only at time step \( t + 1 \), and is related to the prediction error (4) by

\[ E_t = \| x_t - \hat{x}_t \|^2 = \eta_t \eta_t = \xi_{t+1} (J_t J_t^\top)^{-1} \xi_{t+1}. \]  

where \( J \) is the linearization of the maps from current input to next
input dependent on the current controller. As only the projection \( \eta \) of \( J^{-1} \) on \( \xi \) is relevant, the time loop error can be efficiently
estimated. The homeokinetic learning rule updates the parameter
matrix \( C \) of the controller (1) by gradient descent

\[ \Delta C_t = -\varepsilon C_t \frac{\partial E_t}{\partial C_t}, \]  

where \( C_t \) is an element of \( C \) and \( \varepsilon C \) is a learning rate.

If the representational power is of less importance than the
flexibility (Smith and Herrmann, 2019), then a simple quasi-
linear system can be considered as sufficient. Below, when we
will consider a multi-layered system, the representational power
is meant to be achieved by the interaction between the layers each
of which will consist of one instance of the current controller-
predictor unit. A pseudo-linear controller, i.e., a quasi-linear
function of the inputs with coefficients that are adaptive on the
behavioral time scale,

\[ y_t = C(x_t) = g(Cx_t + c) \]  

and a linear model

\[ \hat{x}_{t+1} = M(y_t) = M y_t + m, \]  

does thus not limit the complexity of achievable control. The
parameters of the controller and the model are now the matrices
\( C \) and \( M \) resp., which are complemented by the matching bias vectors \( c \) and \( m \). In order to incorporate limitations of actions
of the robot, the controller is quasi-linear due to the element-wise sigmoidal function $g$. Because of the simple structure of Equation (8), we can omit here the state dependency (2) and define the model $M$ only in motor space. The parameter update (7) becomes

$$\Delta C_{ij} = \epsilon_C \eta^T \frac{\partial J}{\partial C_{ij}} \eta, \quad (10)$$

and analogously for the bias term $\epsilon$. With $\mu = G'M^T(J^T)^{-1}\eta$ and $\xi = C\eta$ the learning rules for a linear controller with a linear model are

$$\Delta C_{ij} = \epsilon_C \mu_i \eta_j - 2\epsilon_C \mu_i \xi_j \eta_i \quad (11)$$
$$\Delta \xi_i = -2\epsilon_C \mu_i \xi_j \eta_i. \quad (12)$$

Simultaneously, but possibly with a different learning rate, the parameters $M$ of the linear model (9) are updated via gradient descent on the standard prediction error (Equation 4, rather than Equation 6).

$$\Delta M_{ij} = -\epsilon_M \frac{\partial E}{\partial M_{ij}} = \epsilon_M \xi_i \eta_j, \quad (13)$$
$$\Delta \xi_i = -\epsilon_M \frac{\partial E}{\partial b_i} = \epsilon_M \xi_i, \quad (14)$$

where $\epsilon_M$ is the learning rate for the adaptation of the internal model. The ratio of the two learning rates $\epsilon_C$ and $\epsilon_M$ is known to be critical for the behavior of controlled robot (Smith and Herrmann, 2019). For the architecture presented next, an optimized ratio is to be used, see also Figure 2.

3. THE DIAMOND MODEL

3.1. Deep Homeokinesis

The DIAMOND model is a generalization of the homeokinetic controller described in section 2. As shown in Figure 1, the comparison of a state variable $x(t)$ and its estimate $\hat{x}(t)$ is now repeated also for estimates of estimates etc., $x_0(t) = x(t)$, $x_1(t) = \hat{x}(t)$, $x_2(t)$, $\ldots$, where each pair of neighboring layers corresponds to a homeokinetic controller that acts onto the lower layer as its environment and receives biases from the higher layer. In the inner layers (larger $\ell$) the external information becomes less and less dominant.

In order to use homeokinetic learning in a multilayer architecture, several instances of the homeokinetic sensorimotor loop are stacked. The internal model of any lower layer serves as the “world” for the next higher layer. Likewise, estimates for input obtained at by a lower layer are the inputs for the higher layers, so each layer reproduces the elementary loop shown in Figure 1.
3.2. Simple Variant
The architecture consists of controllers for each layer \( \ell < L \) (no controller for \( \ell = L \))

\[
y_{\ell} (t) = C_{\ell+1} (x_{\ell} (t)) = g \left( C_{\ell+1} x_{\ell} (t) + c_{\ell+1} \right)
\]

(15)

and linear models that are given by

\[
\hat{x}_{\ell} (t + 1) = M_{\ell} (y_{\ell-1} (t), y_{\ell} (t)) = M_{\ell} y_{\ell-1} (t) + \hat{M}_{\ell} \hat{y}_{\ell} (t) + m_{\ell}.
\]

(16)

which simplifies for the top layer \( \ell = L \) where \( \hat{y}_{L} (t) \equiv 0 \), i.e., no higher effects are present.

In Equation (16) also the effect of virtual actions \( \hat{y}_{\ell} (t) \), \( \ell \geq 1 \) is included as follows: First, the previous prediction of a layer \( \hat{x}_{\ell} (t - 1) \) is copied into the input unit \( x_{\ell} (t) \) at the beginning of the new time step, see Figure 1. The back-propagated input \( \hat{x}_{\ell} (t - 1) \) that was used in Equations (5) and (6) is no longer needed. From \( x_{\ell} (t) \) a virtual action \( y_{\ell} (t) \) is computed that then contributes additively to the prediction (16). The controller update is here the same as for the one-layer model, and the \( \hat{M} \) matrix (not shown in the figures) is updated in the same way as the \( M \) matrix.

3.3. Main Variant
The variant with extra connections (Figure 1) has for the controller

\[
y_{\ell} (t) = C_{\ell+1} (x_{\ell} (t)) = g \left( C_{\ell+1} x_{\ell} (t) + \hat{C}_{\ell+1} \hat{x}_{\ell+1} (t - 1) + c_{\ell+1} \right)
\]

i.e., in the same way as new input \( \hat{x}_{0} (t + 1) \) that is used to calculate the prediction error is also used in the next time step as input \( x_{0} (t) \), we are also for \( \ell > 0 \) using previous predictions as new virtual input. For the deepest layer \( \ell = L \), Equation (17) is not applied, and for the penultimate layer we have simply

\[
y_{\ell} (t) = C_{\ell+1} (x_{\ell} (t)) = g \left( C_{\ell+1} x_{\ell} (t) + c_{\ell+1} \right).
\]

(18)

For the model, Equation (16) is used as above.

While the first \( C \) matrix in Equation (17) is adapted learned in the standard way (see Equations 11 and 12), the matrix \( \hat{C} \) is updated by gradient descent with respect to the prediction error for the action

\[
\mathcal{E} = (y_{\ell} (t) - \hat{y}_{\ell} (t))^2,
\]

where

\[
\hat{y}_{\ell} (t) = g \left( \hat{C}_{\ell+1} \hat{x}_{\ell+1} (t - 1) + \hat{c}_{\ell+1} \right),
\]

i.e., the input \( \hat{x}_{\ell+1} (t - 1) \) from the more inner level is used to predict the motor output \( y_{\ell} (t) \). The update equations for \( C \) are similar to Equations (13) and (14), but also contains a derivative of \( g \). Note that no loops are present in the network of Figure 1, which may not be a problem as the loops have no function (yet), and may be included later. However, it is not clear what “deliberation” could mean without these loops.

We assume that the inner (deeper) layers are updated first. The deepest layer \( \ell = L \) has no variables, just the controller and the model. According to Equation (18), no higher-level input variables are needed in order to update the variables at \( \ell = L - 1 \). In this way, virtual actions and virtual inputs are available to be used in Equations (17) and (16) to update the next layer toward the outer side, i.e., with lower \( \ell \). For the update of the matrices \( M, \hat{M}, C \) and \( \hat{C} \) the time order is not essential, if the variables are calculated as described above.

3.4. Main Variant With Deep Associations
As a further variant, which is, however, not implemented in the present simulations, a standard deep neural network can be employed to connecting the inputs \( x_{\ell} \) directly between neighboring levels. In this case a separate set of connections \( P_{\ell} \) would be learned for map from \( x_{\ell-1} \) to \( x_{\ell} \). The weights \( P \) are learned by the activations \( x_{\ell} \) that arise due to the activations of the network. In addition it is possible to add a further set of connections \( R \) that play the same role as \( P \), but for the predicted sensor values.

The network can sustain persistent activity that represents an action perception cycle. Activity in the subnetworks that are completed by recurrent connections arises by self-amplification of noise or spurious activity following the homeokinetic learning of the respective controller. It may be possible to use also the cycles more explicitly for learning, but we want to restrict ourselves here to one-step learning rule, i.e., gradients are calculated only over one The full model also includes perceptual pathways consisting of bridges between input-related units. In this way the network activity becomes shaped by standard deep feed-forward networks.

4. EXPERIMENTAL RESULTS
4.1. Active Response by the Recurrent Network
As a first test, we have considered the simple variant of the architecture (see section 3.2) when it is driven with a sinusoidal input and the “world” reproduces simply a noisy version of the motor action as next input to the robot. Typical results are shown in Figure 2 for a two combinations of the learning rates \( \varepsilon_{C} \) (11, 12) and \( \varepsilon_{M} \) (13, 14), which lead either to an abstracted reproduction of the input in the deeper layers or to a self-organization of activity that, however remains without effect in this simple variant. At lower learning rates (left column), even deeper layers respond to the original input. In this case, the internal layers are square versions of the original input. For larger learning rates (right column), the internal layers have a different response. The fifth row shows a combination of homeokinetic adaptation (the red line between 310 and 320 s) and noisy output while still following the input from the first layer. Deeper layers (lower rows), tend have a decay in the generation of motor action attributed to the squashing function.
4.2. A Wheeled Robot in the Hills

The main variant (section 3.3) is used in an exploration task, where a four-wheeled robot is expected to cover a large portion of an unknown territory (Smith and Herrmann, 2019). The hilly landscape shown left in Figure 3 can be scaled in vertical direction such that different levels of difficulty can be achieved ranging from a flat ground (level 0) to slopes that require maximal motor power (level 1) and that can cause instabilities and thus large prediction errors (4). The activity decays in a five-layer DIAMOND model for a flat arena, as the inner layers are not needed, whereas for a hilly landscape (difficulty level > 0) the inner layers did not show much attenuation. The behavior of the robot is evaluated based on a $10 \times 10$ grid overlaid to the square-shaped arena. The number of visited grid cells is averaged over five runs for each difficulty and each controller depth and represented as a coverage rate. The total coverage was in all cases below 50% such that the increase of the coverage with time was nearly linear.

Whereas a single layer can achieve a similar performance across all terrain difficulties, for increasing difficulty of the task the higher layer are more and more engaged and take advantage of the increased errors in the terrain that provide thus a potential for a more comprehensive coverage of the arena per time unit.

4.3. A Spherical Robot in a Polygonal Arena

Finally, we studied a simulated spherical robot which is controlled by three masses that a movable along internal axes, see Figure 4, left. The robot is exploring freely in a polygonal environment which was chosen to discourage circular movement along the wall. The controller picks up quickly a suitable rhythm of the internal weights that is effecting in moving the robot in any direction. Collisions with wall usually stop the robot until the emergence of a different mode of the movements of the internal weights moves the robot in a different direction. Although a more systematic study is yet to be performed, it is already obvious that adding a small number of additional layers increases the behavioral repertoire of the robot and reduces the duration of any wall collisions and re-emergence of behavior in the robot. The example is also meant to demonstrate, that the applications of the learning rule and architecture are beyond exploration of a planar arena and can be used in order to generate and to organize elementary robotic behaviors.

5. DISCUSSION

The numerical results seem to imply that a few layers are sufficient, i.e., a larger number of layers does not lead
to further improvements or may require a much longer learning time than attempted here. It should, however, be considered that the tasks and environments are all very simple, such that it is not possible to generalize this observation to more complex situations. It can nevertheless be expected that the spontaneous internal activations that were observed for suitable learning rate ratios, lead to a learning time that is approximately linearly increasing with the number of layers, and not much worse. This is suggested by earlier results with homeokinetic learning rule (Martius et al., 2007).

The present model is a representation of the idea (see e.g., Anderson et al., 2012) that it is difficult to define a clear boundary between brain and body or even between body and world. At all layers the system follow the same principles in its adaptation of the actions onto lower layers and in the learning of a model that affects higher layers. The reduction of complexity of the internal dynamics toward higher layers is counterbalanced by the autonomous activity such that the main eigenvalue at each layer will hover near unity (Saxe et al., 2014).

Although the activity is updated here in parallel in all layers, the stacked structure is clearly similar to the subsumption architecture (Brooks, 1986) as it allows for shorter or longer processing loops. It remains to be studied whether more general architectures are beneficial, especially when more complex tasks are considered.

In Figure 1, it is understood that the dynamical variables \((x, y, \text{ and } \hat{x})\) exist each in two instances, one updated by the controlling and predictive pathways, the other by the feedback within the re-estimation system. The need to disambiguate these units points to an interesting parallel to the roles of the layers of the mammalian cortex.

Finally, it should be remarked the principle of predictive coding is inherent in the architecture from the homeokinetic principle. Activity can only travel to the deeper layers if it is not already predicted by the internal model of the
current layer. In some cases this can lead to a complete decay of the activity in the deeper layers (see Figure 3), although more complex robots and more challenging environments need to be studied in order to precisely identify parallels to the predictive coding principle in natural neural systems.

**DATA AVAILABILITY STATEMENT**

All datasets generated for this study are included in the article/supplementary material.

**AUTHOR CONTRIBUTIONS**

JH, BS, and SS: conception. JH and SS: model design. RD, SS, and CI: experiments. JH, BS, and SS: writing. JH: project organization.

**REFERENCES**

Adams, R. A., Shipp, S., and Friston, K. J. (2013). Predictions not commands: active inference in the motor system. Brain Struct. Funct. 218, 611–643. doi: 10.1007/s00429-012-0475-5

Anderson, M. L., Richardson, M. J., and Chemero, A. (2012). Eroding the boundaries of cognition: implications of embodiment. Topics Cogn. Sci. 4, 717–730. doi: 10.1111/j.1756-8765.2012.01211.x

Brooks, R. A. (1986). A robust layer control system for a mobile robot. J. Robot. Automat. RA 2, 14–23. doi: 10.1109/JRA.1986.1087032

Carvalho, J. T., and Nolfi, S. (2016). Behavioural plasticity in evolving robots. Theory Biosci. 135, 201–216. doi: 10.1007/s12064-016-0233-y

Der, R. (2001). Self-organized acquisition of situated behaviors. Theory Biosci. 120, 179–187. doi: 10.1007/s12064-001-0017-9

Der, R., and Martius, G. (2012). The Playful Machine: Theoretical Foundation and Practical Realization of Self-Organizing Robots, Vol. 15. Springer Science & Business Media.

Fukushima, K., and Miyake, S. (1980). "Neocognitron: self-organizing network capable of position-invariant recognition of patterns," in Proc. 5th Int. Conf. Patt. Recogn. (Berlin), 459–461.

Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. Neural Comput. 18, 1527–1554. doi: 10.1162/neco.2006.18.7.1527

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). "Imagenet classification with deep convolutional neural networks," in Advances in Neural Information Processing Systems, eds F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Lake Tahoe, NV: Curran Associates Inc.), 1097–1105.

Levine, S., Finn, C., Darrell, T., and Abbeel, P. (2016). End-to-end training of deep visuomotor policies. J. Mach. Learn. Res. 17, 1334–1373. doi: 10.7766/jmlr.2019.14.1.040

Martius, G., and Herrmann, J. M. (2011). Variants of guided self-organization for robot control. Theory Biosci. 131, 129–137. doi: 10.1007/s12064-011-0141-0

**FUNDING**

SS was supported by the ORCA Hub EPSRC project (EP/R026173/1, 2017–2021). CI was supported by grant EP/L016834/1 from the University of Edinburgh, School of Informatics, Centre for Doctoral Training in Robotics and Autonomous Systems (http://www.edinburgh-robotics.org) from the UK Engineering and Physical Sciences Research Council (EPSRC). BS received funding from the National Key R&D Program of China (2019YFA0709503).

**ACKNOWLEDGMENTS**

We thank Georg Martius (Tübingen), Klaus Pawelzik (Bremen), Alessandro Treves (Trieste), and Hemang Kanwal (Edinburgh) for stimulating discussions. JH is grateful to CAS SIA for their kind hospitality.

**CONFLICT OF INTEREST** The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Copyright © 2020 Smith, Dharmadi, Imrie, and Herrmann. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.