A Wavelet-based Encoding for Neuroevolution

Sjoerd van Steenkiste
DKE
Maastricht University
Maastricht, NL
s.vansteenkiste@alumni.maastrichtuniversity.nl

Jan Koutník
IDSIA, USI & SUPSI
Manno-Lugano, CH
hkou@idsia.ch

Kurt Driessens
DKE
Maastricht University
Maastricht, NL
kurt.driessens@maastrichtuniversity.nl

Jürgen Schmidhuber
IDSIA, USI & SUPSI
Manno-Lugano, CH
juergen@idsia.ch

ABSTRACT

A new indirect scheme for encoding neural network connection weights as sets of wavelet-domain coefficients is proposed in this paper. It exploits spatial regularities in the weight-space to reduce the gene-space dimension by considering the low-frequency wavelet coefficients only. The wavelet-based encoding builds on top of a frequency-domain encoding, but unlike when using a Fourier-type transform, it offers gene locality while preserving continuity of the genotype-phenotype mapping. We argue that this added property allows for more efficient evolutionary search and demonstrate this on the octopus-arm control task, where superior solutions were found in fewer generations. The scalability of the wavelet-based encoding is shown by evolving networks with many parameters to control game-playing agents in the Arcade Learning Environment.

CCS Concepts

•**Computing methodologies → Artificial intelligence; Generative and developmental approaches;** *Reinforcement learning;*

Keywords

Neuroevolution; indirect encoding; wavelets; gene-locality

1. INTRODUCTION

Neuroevolution [19] provides an elegant alternative for training artificial neural networks (NNs) used in reinforcement learning (RL) tasks. Unlike in value-based methods, e.g. [26, 20], NNs can collapse the sensory processing and control into a single system that naturally deals with continuous state/action spaces. Hence, a search for optimal RL policies can be carried out using evolutionary search for NN parameters without accessing the (often hard to obtain) reward gradient [28].

NNs are encoded either directly or indirectly as strings of values (genes) and then evolved by an evolutionary algorithm. Direct encoding schemes employ a one-to-one mapping from genes to NN parameters (e.g. connectivity patterns, connection weights) so that the size of the chromosome is proportional to the NN complexity.

In indirect (or generative) encoding schemes [15, 13] the mapping from genes (genotype) to NN parameters (phenotype) is defined by an arbitrary computable function, as in early work in [22, 23], often chosen such that it (weakly) decouples the network complexity from the size of the chromosome. Indirect encoding schemes allow neuroevolution to be scaled to larger problems, as an optimal solution in a high-dimensional parameter space can be searched for in a lower-dimensional gene space [5]. Indirect encoding schemes have been successfully applied to a wide range of tasks including helicopter control [7], vision-based TORCS [16], and Atari game-playing [11].

Although proven successful in different domains, most indirect encoding schemes lack *continuity in the genotype-phenotype mapping* and as a result a small change in the genotype may cause a large change to the phenotype [17]. An encoding scheme that does not suffer from this problem encodes the network parameters as ordered sets of band-limited *Discrete Cosine Transform* coefficients (DCT-based Encoding; [17]). By using a Fourier-type mapping the DCT-based encoding is able to render the size of the chromosome completely independent of the network complexity. However, as an immediate result *spatio-temporal locality* is lost. The evolved coefficients no longer carry positional information about the network parameters and hence a local change to the genotype affects the phenotype in its entirely. Previously it has been argued that "because indirect encodings do not map directly to their phenotypes, they can bias the search in unpredictable ways." [24, 2]. Here we argue that the lack of gene locality reduces the effectiveness of traditional evolutionary operations (e.g. recombination, mutation) and therefore diminishes the efficiency of search via evolutionary computation.

In this paper, we propose a novel *wavelet-based encoding* (WBE) scheme which satisfies both spatio-temporal locality and continuity in the genotype-phenotype mapping. Similar to previous work [17] our encoding exploits spatial

GECCO '16, July 20 - 24, 2016, Denver, CO, USA

ISBN 978-1-4503-4206-3/16/07…$15.00

DOI: http://dx.doi.org/10.1145/2908812.2908905

regularity among the network weights to reduce the search space dimensionality. However, in the WBE scheme a set of localized wavelet coefficients is evolved to encode the low-frequency content in the network weights. Inspired by conventional wavelet-based image compression [4], a *lossy* reconstruction of the NN weights is obtained by applying the inverse wavelet transform to these low-frequency wavelet coefficients only.

The advantages of our method are twofold: First, any spatial regularity among the input and output weights is compressed into fewer coefficients, hence greatly reducing the search space dimensionality. Additional compression is obtained by searching for spatially correlated networks, which in the wavelet domain can be represented with even fewer coefficients. Second, the WBE makes use of a *local* mapping between the genes and network weights, which relates changes in the gene space more directly to the weight space. In our experiments we find this to be beneficial for evolutionary search.

The WBE scheme is evaluated in two reinforcement learning benchmarks: In subsection 3.1 the superior performance of the WBE over the DCT-based encoding can be observed on the octopus arm control task; in subsection 3.2 the WBE is evaluated on the stochastic Arcade Learning Environment (ALE) where it is able to learn from raw visual input and demonstrates that a reduction in search space dimension is necessary for such complex problems.

2. WAVELET BASED ENCODING

In this section we describe a novel wavelet-based encoding scheme for neuroevolution. A detailed discussion on wavelet theory and the accompanied multi-resolution analysis is beyond the scope of this paper. The reader is referred to one of [3, 25] for an excellent overview of these topics.

2.1 Wavelets and Filter Banks

Wavelets are mathematical functions for unravelling a signal at different frequency levels in a hierarchical fashion. Wavelets are defined recursively as dilations and translations of a single mother wavelet $\psi(t)$, which in the discrete case[1] take the form:

$$\psi_{m,k}(t) = 2^{-m/2}\psi(2^{-m}t - k), \quad m, k \in \mathbb{Z} \tag{1}$$

The integer translates of the wavelet function $\psi(t)$ at a particular scale m span a function space $W_m = span\{\psi_{m,k} | k \in \mathbb{Z}\}$. All W_m combined constitute a basis (localized both in time and frequency) in which any function f can be defined in terms of ψ and its coefficients γ in the space as:

$$f = \sum_{m,k} \gamma_{m,k}\psi_{m,k} \tag{2}$$

A well-known example of an orthonormal basis of wavelets is the Haar-basis [10], as displayed in Figure 1 and constituted by the wavelet function:

$$\psi(t) = \begin{cases} 1 & 0 \le t < 1/2 \\ -1 & 1/2 \le t < 1 \\ 0 & otherwise. \end{cases}$$

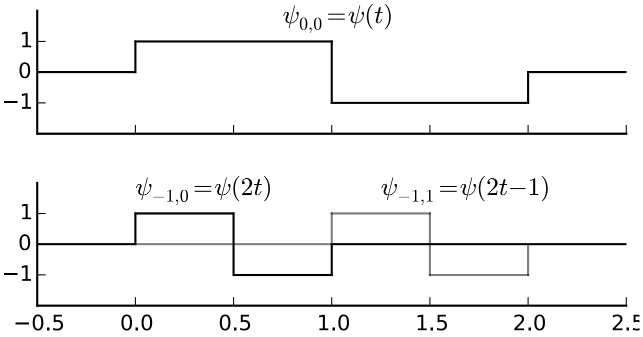


Figure 1: The Haar wavelet displayed at two different scales. We omit the scaling factor ($\sqrt{2}$) for simplicity.

The *wavelet transform* of a continuous signal $f(t)$ can be interpreted as a change of basis using a wavelet-function $\psi(t)$. It is defined as the L^2-inner product between $f(t)$ and the wavelet function $\psi_{m,k}(t)$, which in the discrete case takes the form:

$$DWT(m,k) = \langle f(t), \psi_{m,k}(t)\rangle = \int_{-\infty}^{\infty} f(t)\overline{\psi}_{m,k}(t)dt \tag{3}$$

Often no explicit expression for $\psi_{m,k}(t)$ is available, which is resolved by considering orthogonal wavelets. For these wavelets it holds that $\gamma_{m,k} = DWT(m,k)$, which can be computed without an explicit expression for $\psi(t)$. This approach is implemented by a pair of discrete time filters H_0, H_1 from orthogonal filter banks that are applied by convolution with an input sequence [25]. The filter pair is defined as a finite set of filter coefficients and uniquely determine the wavelet basis function $\psi(t)$.

Figure 2 outlines the schematics of a classic wavelet filter bank. A single pass of a signal s through the analysis filter bank computes the DWT and yields a single level wavelet decomposition. The analysis filter bank decomposes s into an ordered set of approximation coefficients a and detail coefficients b encoding the low- and high-frequency content of the input signal respectively. Since both H_0, H_1 operate at full rate, down-sampling of a factor 2 ensures that the approximation and detail signal remain critically sampled.

The DWT halves the time resolution and doubles the frequency resolution since only half of each filter output characterizes s; and a, b each carry half of the frequency band of the input signal respectively. A multi-level dyadic wavelet decomposition is obtained by repeatedly decomposing the low-frequency approximation coefficients. In doing so one essentially trades time-resolution for frequency-resolution at the lower frequencies.

The DWT as implemented by an orthogonal filter bank is lossless and the original signal can be re-obtained by filtering the approximation and detail coefficients with the synthesis filter bank (right-side in Figure 2) that computes the Inverse DWT. In the orthogonal case the synthesis filters F_0 and F_1 consist of the time-reversed impulse response of the analysis filters H_0 and H_1 respectively.

The mother wavelet $\psi(t)$ can be constructed to carry specific properties that may be beneficial to a particular application. Apart from orthogonality a desirable property is the number of vanishing moments the wavelet function

[1] The wavelet function is discretized at the intersections of the dyadic grid.

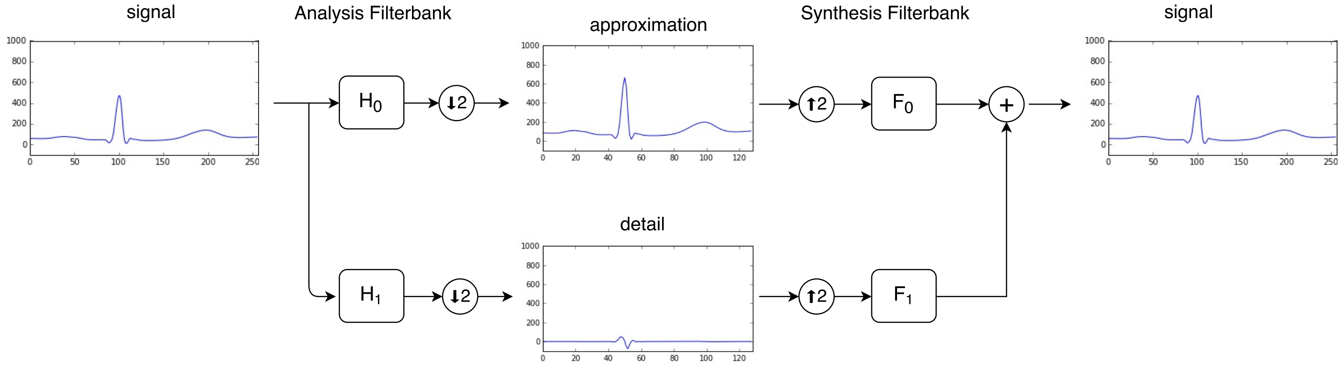


Figure 2: A classic wavelet filter bank decomposing an input signal by convolving it with the filter-pair H_0, H_1, followed by a down-sampling operation of 2. The resulting approximation and detail signal encode the low-frequency and high-frequency content of the input signal respectively. The process is reversed by up-sampling the approximation and detail coefficients and convolving them with the synthesis filter-pair F_0, F_1.

carries. Vanishing moments correspond to smoothness of the wavelet function, and to polynomial preservation of the wavelet transform. Each of these properties carry over to the filter pair, such that when $\psi(t)$ carries $p + 1$ vanishing moments any polynomial structure up to order p from the input signal is preserved in the approximation coefficients. The detail coefficients will not contain this structure.

Depending on the filter order n and the number of vanishing moments a wavelet function may have several degrees of freedom left. More specifically, a wavelet filter of order n has $2n$ degrees of freedom. Orthogonality fixes n degrees and each vanishing moment fixes an additional degree. Any remaining freedom can be used to optimize the wavelet function with respect to a particular task [9]. In our approach we optimize the wavelet function by means of evolution using a lattice structure for the design of orthogonal wavelet functions of order n with up to two vanishing moments [14].

2.2 Neural Network Encoding

The wavelet-based encoding scheme is inspired by wavelet-based image compression methods. Such methods first decompose an image using the wavelet transform, after which the wavelet coefficients are quantized and encoded to obtain a compressed representation [4]. In the quantizing and encoding step, compression is obtained by using fewer bits to encode the wavelet coefficients, yielding a lossy reconstruction of the original image when reversing this approach. The WBE similarly operates on the wavelet coefficients, however it avoids the quantization and encoding steps, and gains compression by exploiting the low energy of the detail coefficients using zeros instead.

In the WBE each network weight tensor is encoded separately as a set of positional low-frequency wavelet coefficients (i.e. the approximation coefficients). By reconstructing the network weights from just this low-frequency content a lossy reconstruction of the original weight tensor is obtained. The number of approximation coefficients to be evolved depends on the number of inverse wavelet transformations l and the dimensionality d of the weight tensor in the weight space. Hence an arbitrary weight tensor has a fixed compression ratio of $2^{ld} : 1$. A level l inverse wavelet transform of a tensor with dimensionality d is obtained by applying the synthesis filter bank from Figure 2 along each dimension l times.

As an example, consider a fully connected recurrent neural network layer with h nodes receiving a two-dimensional input of size $(m \times n)$. This layer has three weight-tensors $\mathbf{W}$ (input weight matrix), $\mathbf{R}$ (recurrent weight matrix), $\mathbf{B}$ (bias vector) of sizes $(h \times mn), (h \times h), (h \times 1)$ respectively.

The extend to which the regularity of each weight tensor can be compressed depends on the number of dimensions along which the weights are potentially correlated. In the simplest case the weights are correlated within the rows and columns, which express spatial regularity among the node inputs and across the different nodes in the layer respectively. Additional weight correlation may be present, either in the input (e.g. the input signal is a two-dimensional image) or in the layer itself (the neurons are further spatially organized), which can then be compressed by first re-organizing the weights in a tensor of higher dimension that accounts for all dimensions along which regularity occurs.

In the example the weights in $\mathbf{W}$ are organized in a three-dimensional tensor of size $(h \times m \times n)$ and hence the regularity across the two input dimensions is also compressed. The other weight tensors exhibit no additional regularity and hence keep their original shape. Depending on l there is a total of $\lceil \frac{hmn}{8^l} \rceil + \lceil \frac{h^2}{4^l} \rceil + \lceil \frac{h}{2^l} \rceil$ approximation coefficients to be evolved. The steps involved in transforming the evolved approximation coefficients to network weights for this particular example are shown in Figure 3. The low-frequency wavelet coefficients are mapped from the genome to the wavelet-space by organizing them in a tensor of suitable shape (along each dimension as $[a_l, b_l, \dots, b_1]$) forming a hypercube in the top left corner. The rest of the tensor is filled with zeros. Next the inverse wavelet transform is applied l times in order to transform the tensor to the weight space. Each tensor is now filled with weights, which after mapping (such as collapsing axes in the case of $\mathbf{W}$)[2] yields a lossy reconstruction of the weight matrices $\mathbf{W}$, $\mathbf{R}$ and $\mathbf{B}$.

The choice of wavelet basis function is important and there is often little *a priori* knowledge of what constitutes a good basis function. Here, the wavelet-basis function (i.e. the parameters of the lattice structure) is evolved alongside the wavelet coefficients, which is referred to as *dynamic* basis

[2] The weight tensor is *flattened* along the extra input dimension to yield its original shape.

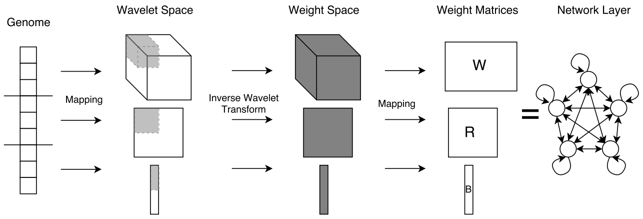


Figure 3: Schematics of the mapping from the evolved approximation coefficients in the genome to the network weights of a fully connected recurrent layer. The approximation coefficients in the wavelet space are depicted as light grey, all other entries (i.e. the detail coefficients) are zeros. A single level inverse reconstruction is shown.

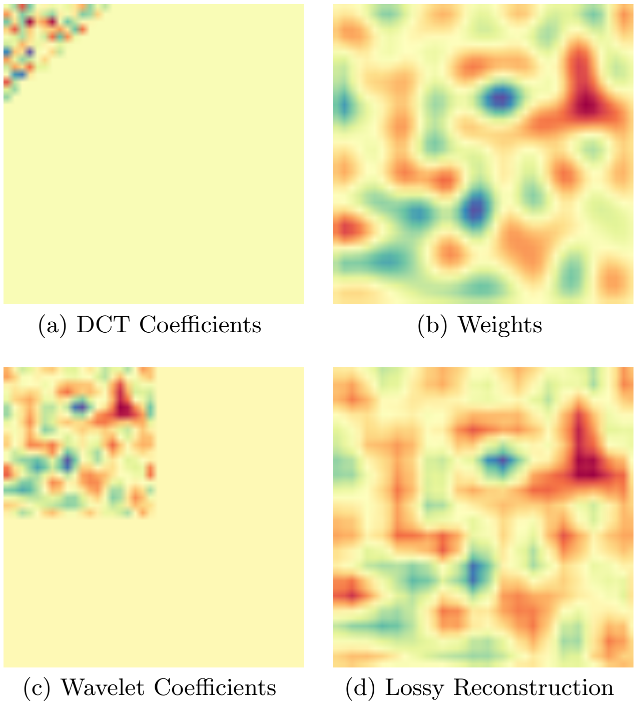
function evolution. Two alternatives are also considered, a *fixed* basis function chosen by randomly selecting values for the parameters of the lattice structure, and an *optimal* basis function that was previously evolved on the same task. When referring to the WBE by default the dynamic basis function variant is implied.

Due to the fixed compression ratio of $2^{ld} : 1$ the size of the genome is weakly coupled to the network complexity. This contrasts the DCT-based encoding, in which both are completely decoupled. As a trade-off the WBE gains spatio-temporal locality, which can be observed by comparing the low-frequency DCT- and wavelet representation of a weight matrix as displayed in Figure 4. One can clearly see that the wavelet coefficients provide positional information, whereas the DCT coefficients contain no such a correspondence. Moreover, the lossy wavelet reconstruction of the original weight matrix demonstrates how the low energy of the detail coefficients can be utilized to yield compression (in this case at a ratio of $4 : 1$) while preserving the most important aspects of the original weight matrix.

The filter order n and the level of decomposition l play a role in balancing locality and regularity. A wavelet filter of order n affects $2n$ coefficients, hence by increasing the filter order the wavelet transform becomes less local. This effect is further amplified by l. In the experiments a filter of order 2 and a single vanishing moment were chosen, leaving a single degree of freedom left for the basis function to be optimized. This choice yields quick convergence and maximal spatio-temporal locality.

3. EXPERIMENTS AND RESULTS

Two evolutionary algorithms were used to search the space of wavelet coefficients: Cooperative Synapse Neuroevolution (CoSyNE; [8]) and the Separable Natural Evolutionary Strategies (SNES; [27, 21]). CoSyNE is a cooperative coevolutionary method, which searches at the level of individual network weights. It keeps an explicit sub-population for each weight and alters the population at each generation using three evolutionary operators; Besides mutation and recombination, CoSyNE employs probabilistic permutation to co-evolve the weights within a subpopulation so that each weight forms part of a potentially different network in the next generation. CoSyNE has shown to be efficient in searching for weight values indirectly [17], here it is used in a similar fashion by searching for wavelet coefficients.


(a) DCT Coefficients

(b) Weights

(c) Wavelet Coefficients

(d) Lossy Reconstruction

Figure 4: (a) A random, band-limited set of DCT coefficients is transformed, using padding and the inverse DCT transform, into a weight matrix (b). After performing a single level wavelet decomposition on (b) the low-frequency wavelet coefficients are obtained (c). A lossy wavelet reconstruction from these coefficients is depicted in (d).

SNES maintains a (Gaussian, in this case) distribution for each gene from which it samples a population at each generation. Based on the performance of the population the mean and variance of the gene distributions are updated in the direction of the natural gradient. SNES is particularly suited for high-dimensional problems as it scales linearly with the problem dimension.

CoSyNE and SNES are similar in that they operate at the level of individual parameters, and thus allow the parameters of the wavelet-basis function to be evolved alongside the wavelet coefficients.

3.1 Octopus-arm Control Task

In this task an octopus arm having c compartments is floating in a 2D fluid environment [29]. Each compartment has constant volume and contains 3 controllable muscles. The state of a compartment is described by the x, y-coordinates of two of its corners plus their velocities. Together with the arm base rotation, the arm has $8c + 2$ state variables and $3c + 2$ control variables. The goal of the task is to control the arm in order reach a goal position with the tip of the arm, starting from three different initial positions. The arm is controlled by contracting the appropriate muscles at each step of simulated time. Two additional starting positions were used to measure generalization performance.

For this particular task c was chosen as 10, resulting in 32 control variables, and 82 observable state variables. The wavelet-based encoding is evaluated on the state control

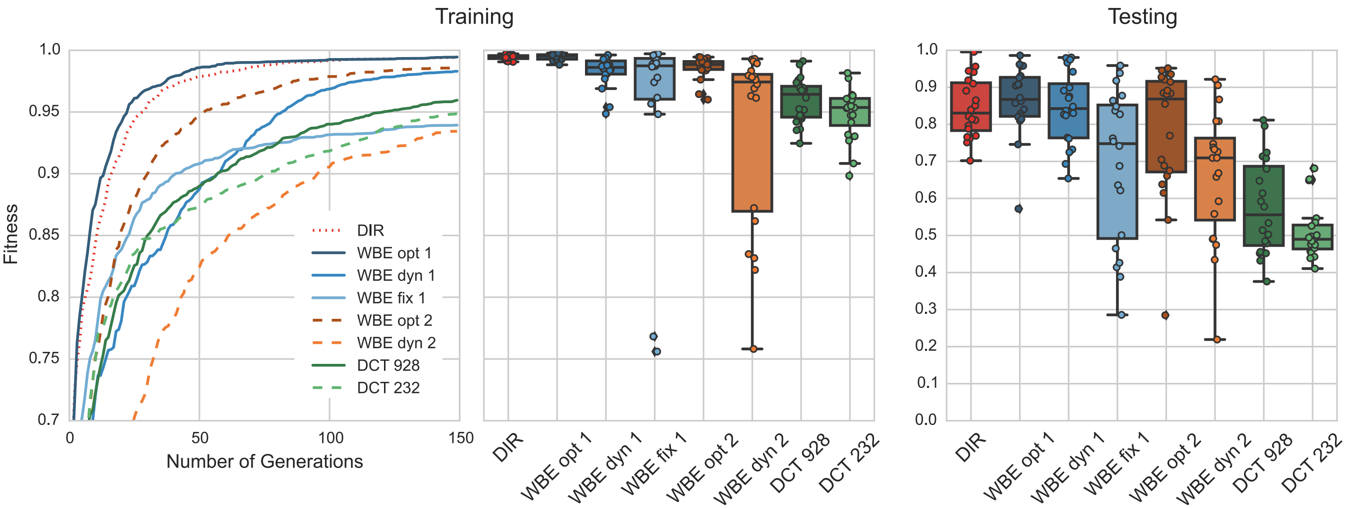


Figure 5: Fitness obtained by different encodings on the Octopus-arm Control Task. On the left the averaged champion training fitness at each generation. In the middle the champion training fitness distribution at the final generation. On the right the champion testing fitness distribution at the final generation.

task, in which the network receives the state variables as input. The controller is a fully connected recurrent NN, with a hidden state corresponding to each of the 32 control parameters, and a sigmoid activation function. The resulting network has 3 680 connection weights and hence dimensionality reduction of the weight space is not necessary. This task was primarily used to evaluate the various WBE properties, in particular the effect of spatio-temporal locality and performance compared to the DCT-based encoding.

3.1.1 Results

Figure 5 summarizes the performance of the various encodings evaluated on this task. Each encoding was evolved using SNES for 150 generations with a fixed population size of 40. The results, shown in Figure 5, are averaged over 20 experimental runs. Direct encoding (DIR) is compared to the WBE with a fixed (*fix*), dynamic (*dyn*) and optimal (*opt*) basis function for different levels, and to the DCT-based encoding using different numbers of coefficients. The same line style is used for encodings with the same amount of network parameters.

It can be seen that the direct encoding performs among the best performing encodings on this task, both in terms of training and testing fitness. The low complexity of this task allows good solutions to be found without reduction of the search space. When comparing the compressed encodings, i.e. for a single level compression (of approximately a factor 4) the wavelet-based encoding (WBE *dyn 1*) to the DCT-based encoding (DCT *928*) with the same amount of network parameters, the WBE performs significantly better, both in terms of training and testing fitness. On the other hand we observe slower convergence of the WBE, which is expected as the dynamic basis function makes this a dual optimization problem – optimal weight values and the basis function parameters (hence the transform) have to be searched for at the same time.

3.1.2 Evolving the wavelet basis function

An alternative to a dynamic basis function would be to use a random basis function at each experimental run, which is fixed throughout each generation. As seen in Figure 5 the performance of this approach (WBE *fix 1*) is significantly worse in terms of fitness. Nevertheless, the small range of the upper two quartiles in the middle plot suggests that whenever a “good” basis function is found, the fixed encoding is able to reach the same performance as when using a dynamic basis function. In fact, the WBE with a previously evolved basis function (WBE *opt 1*) has supreme performance across all the experimental runs. Hence we conclude, since a priori knowledge of what constitutes a good basis function is not available, that evolving the basis function is necessary to reach an excellent solution.

3.1.3 Further compression

Additional compression can be obtained by using a level-2 inverse wavelet transform (total compression of approximately a factor 16). WBE *dyn 2* performs significantly better in terms of testing fitness than the DCT-based encoding with the same amount of network parameters (DCT *232*). However, as opposed to the single level transform it is unable to perform significantly better (but also not worse) in terms of training fitness.

The champion training fitness distribution at the last generation in the middle plot of Figure 5 provides further insight. Here we observe that the range of the fitness in the third quartile is much larger than in the first and second quartile, due to the few cases where dual optimization of the basis function and the wavelet coefficients failed.

Similar optimization failure can be observed for WBE *dyn 1* (in the form of two outliers) yet at a much smaller scale. By using a two-level transform the choice of basis function (represented by the filter coefficients) affects exponentially more weights and amplifies the lack of performance when no good solution to both optimization problems is found. Observing that the upper two quartiles of the training fitness distribution are well above those of *DCT 232* with a similar range further confirms this.

A separate experiment was done in order to ensure that, for an increased level of compression, the WBE is able to outperform the DCT-based encoding when the additional complexity of optimizing the inverse wavelet transform is not present. In this experiment, the dynamic wavelet basis function was substituted with a previously evolved optimal basis function (WBE *opt 2*). These results are consistent with (WBE *opt 1*) where superior performance both in terms of training and testing fitness was obtained. The WBE greatly outperformed the DCT-based encoding in this task.

3.1.4 Spatio-Temporal Locality

In the light of spatio-temporal locality several interesting observations can be made. Foremost, there is a clear ordering in terms of performance from a completely local encoding (direct encoding) to an encoding localized both in time and frequency (WBE) to an encoding completely localized in frequency (DCT-based encoding). A similar ordering in terms of locality can be also be observed by analyzing the search efficiency in terms of convergence. As previously mentioned, a dynamic basis function slows down convergence due to the additional optimization problem being created. However, it can also be observed that when this is not the case (e.g. WBE *opt 1*, WBE *opt 2*) the wavelet-based encoding convergences to a superior solution much faster. This clear ordering in terms of performance and convergence serves as empirical evidence that indeed spatio-temporal locality allows for more efficient search via evolutionary computation. When substituting SNES with CoSyNE the same ranking across the encoding schemes in terms of locality, convergence and performance was preserved.

On this particular task the degree to which an encoding is local allows for a particular type of solution, which is often harder to find for non-local encodings. A typical example is shown in Figure 6, which shows snapshots of the octopus arm controlled by a network evolved for three encodings (top) and the corresponding recurrent weight matrix of the controller (bottom). In evaluating the controllers evolved using a non-local encoding (DCT-based) we observed that solutions, in which the upper and lower muscles are contracted at the same time (yielding the S-shape structure), are preferred. This is likely caused by the extensive regularity in the solution weight matrices. The direct and wavelet-based encoding both favored an alternative solution, which contracts the top muscles independently from the bottom muscles and is favorable for this particular task. The clear differences in regularity in the weight matrices found by each solution aid in understanding this very different behavior.

3.2 Arcade Learning Environment

In this task the Arcade Learning Environment (ALE version 0.5.0), which offers an API for agents to play a wide selection of Atari 2600 games [1], was used. A subset of the available games was chosen at random. The goal of an agent is to achieve a score as high as possible. An agent is represented by a NN with a fully connected recurrent layer (100 ReLU units) and a fully connected output layer (36 sigmoid units). The network computes an action every fifth frame from a visual pixel-representation of the game state. The 210 × 160 pixel input from ALE is converted to gray scale and down-sampled to size 105 × 80. This minimal preprocessing technique is similar to the one used in [18].

The agent is able to perform 18 actions: 8 directions of movement (each can be combined with the fire action), the do-nothing action and the fire action. The actions are represented by 9 buttons (each defined as 2×2 sigmoid output units) organized in a 3×3 grid. A button is "pressed" if its minimum average activation is above a pre-specified threshold (set to 0.5). If multiple buttons are pressed at the same time then the two buttons with the highest activations are registered. If one of the actions is the *fire* action then a combination action is forwarded to the game, else the button with the highest activation is used. The *do-nothing action* is activated when no other button is active.

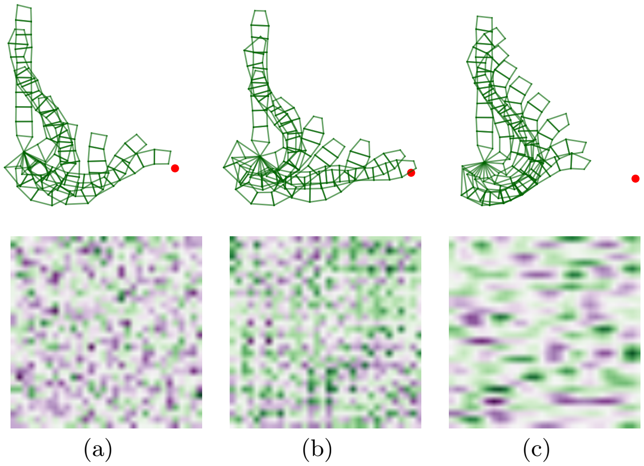


Figure 6: Snapshots of the octopus arm in action as it reaches for the goal (red dot) from the initial position with the arm pointing up (top), and the recurrent weight matrix of the corresponding controller (bottom). From left to right: direct encoding (a), WBE (b) and the DCT-based encoding (c).

The main purpose of this experiment is to show that a direct encoding scheme is no longer effective when the network becomes too large, and the use of an indirect encoding scheme is necessary. The network for this task has 853 736 weights and is well suited to demonstrate this. In this experiment we are not interested in achieving a global optimal solution to a game, hence each controller is evolved only for a small number of generations in search for a local optima.

3.2.1 Results

Figure 7 summarizes the results of the DCT-based encoding, the WBE, and the direct encoding (DIR) in this task. The WBE, and DCT-based encoding compress the total amount of coefficients by approximately a factor of 8. Five agents were evolved (100 generations, with a population size of 40) by CoSyNE for each encoding. Each agent was evaluated by playing 100 games, of which the scores are averaged, normalized and displayed in Figure 7. Stochasticity of ALE[3] yields a large variance of the scores (not displayed) and hence a significant comparison for each individual game can not be made. Nevertheless a consistent pattern, in which the indirect encoding schemes outperform the direct encoding, emerges across all the games that are evaluated.

In this paper, to the best of our knowledge, we present the

[3] With probability $p = 0.25$, the previously executed action is executed again during the next frame, ignoring the agent's actual choice.

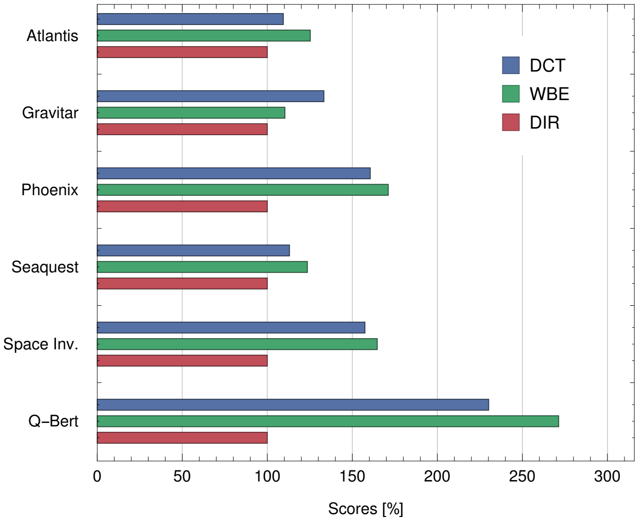


Figure 7: The performance of the DCT-based encoding and the WBE in ALE. Averages were obtained from 5 agents for each encoding, which each played the game a 100 times. Scores are displayed in terms of the performance of the direct encoding.

Table 1: Training fitness scores reported of the champion at the final generation for the wavelet based encoding (WBE), the direct encoding (DIR), and pixel-based HyperNEAT scores from [11]. Scores of the random agent are averaged over 30 experimental runs, and WBE and DIR over 5 experimental runs.

	Random	DIR	WBE	HyperNEAT
Atlantis	23 940	63 427	**82 887**	*61 260*
Gravitar	205	1 060	**1 127**	*370*
Phoenix	741	3 682	**4 643**	*1 762*
Seaquest	90	705	**763**	*716*
Space Inv.	113	734	872	***1 251***
Q-Bert	185	878	**1 143**	*695*

first scores of a neuroevolutionary agent on *stochastic* ALE using raw pixel input. A fair comparison to related work can therefore not be made, which is for the purpose of this experiment unnecessary. The closest possible comparison that can be made is to a pixel-based HyperNEAT agent on a deterministic version of ALE [11].

Table 1 reports training fitness scores and compares them to the training scores of the HyperNEAT agent as was reported in [11]. The WBE scores consistently outperform the scores of a random agent and an agent obtained by using a direct encoding. In terms of gameplay we find that when using the WBE often a local optima is found, yielding lots of points while being relatively simple in terms of strategy. Although the HyperNEAT agent is different in many ways (e.g. visual input, network architecture, randomness) it is interesting to see that in most cases WBE performs markedly better. This was surprising given the relatively simple gameplay that was obtained when evolving a controller for few generations using our encoding.

4. DISCUSSION

The behavior of genetic algorithms is difficult to understand, yet schema theory offers some insight. In particular, we are able to reason about the disruptive effects of mutation and crossover with respect to solutions as being instances of schemata [12]. It can be shown that schemata with a long defining length have a higher probability to be disrupted by crossover, where as schemata with high order have a higher probability of being disrupted by mutation. The building block hypothesis [6] suggests that genetic algorithms perform adaptation by identifying and recombining building blocks of relatively high fitness to build entire solutions. These building blocks take the form of schemata with low order and a short defining length as these are processed with minimum disruption.

The above formulation applies in the case of a local mapping, i.e. when using a direct encoding. However, when an indirect encoding is used and a non-local mapping is employed, we argue that the genetic algorithm is no longer able to make use of these building blocks. In the non-local case a single-point mutation in gene space affects the weight space entirely, and therefore its disruptive effect occurs everywhere, independently of the order of the schemata. Similarly the disruptive effect in weight space by cross-over in gene space is independent of the defining length of the schemata, as cross-over in a non-local gene space preserves none of the original values in the weight space. It follows that the previous advantage that building blocks offer in the gene space is gone, and the genetic algorithm is no longer able to use such blocks to efficiently build its solutions.

In the case of SNES it is more difficult to provide an intuitive explanation as to why a non-local encoding negatively affects the search efficiency. We speculate that discontinuity of the natural gradient updates between gene- and weight space, and the mismatch of the population distribution between gene- and weight space when sampling in gene space play a central role in this.

5. CONCLUSION AND FUTURE WORK

Indirect encoding schemes are advantageous in neuroevolution when the network complexity becomes too large. In this paper a wavelet-based compressed encoding scheme was introduced that satisfies spatio-temporal locality, and hence relates the gene space more directly to the weight space. It was argued that such a more local mapping increases the efficiency of evolutionary search when employing an indirect encoding and hence yields excellent solutions more quickly. Empirical results and an intuitive argument using the building block hypothesis with respect to traditional evolutionary operations suggest that this indeed the case.

In our experiments we found that the wavelet-based encoding using a previously evolved basis function yields superior performance. However, since such a basis function is not available a priori, a dynamic basis function was evolved alongside the wavelet coefficients. It was shown that the performance when using such a dynamic basis function is superior to an approach in which a random basis function is chosen. In comparing the dynamic WBE scheme to a DCT-based encoding scheme we found significant increased performance in favor of our method. When increasing the compression rate the choice of basis function becomes increasingly more important. It was observed that for a level

2 decomposition, in few cases, a dual solution was not found and poor performance was obtained.

A filter order of $n = 2$ was chosen in our experiments, which yielded maximum locality and reduced the complexity of the secondary optimization problem. We have empirically verified (not shown in this paper) that increasing the filter order further yields results that are in line with our previous analysis, yet were not used in practice due to the increasingly slow convergence as n increases and optimization of the basis function becomes increasingly more difficult.

Ultimately one would want to increase the filter order without slowing down the convergence and increase the compression rate without reducing stability. Hence, we argue that improvements to this particular aspect of wavelet-based encoding are necessary to provide these guarantees. Such improvements are likely to be obtained by considering a different (more restricted) parametrization of the wavelet basis function or an alternative optimization procedure of the current parametrization.

6. ACKNOWLEDGEMENTS

The authors would like to thank Klaus Greff and Rupesh Srivastava for valuable discussions and comments. This research was supported by the Swiss National Science Foundation grant "Advanced Reinforcement Learning" (#156682).

7. REFERENCES

[1] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 06 2013.

[2] H. Braun and J. Weisbrod. Evolving neural feedforward networks. In *Artificial Neural Nets and Genetic Algorithms*, pages 25–32. Springer, 1993.

[3] I. Daubechies. The wavelet transform, time-frequency localization and signal analysis. *Information Theory, IEEE Transactions on*, 36(5):961–1005, 1990.

[4] G. M. Davis and A. Nosratinia. Wavelet-based image coding: an overview. In *Applied and comp. control, signals, and circuits*, pages 369–434. Springer, 1999.

[5] J. Gauci and K. Stanley. Generating large-scale neural networks through discovering geometric regularities. In *Proc. of GECCO*, pages 997–1004. ACM, 2007.

[6] D. Goldberg. Genetic algorithms in search, optimization, and machine learning, 1989.

[7] F. Gomez, J. Koutník, and J. Schmidhuber. Compressed network complexity search. In *Proc. of PPSN*, pages 316–326. Springer, 2012.

[8] F. Gomez, J. Schmidhuber, and R. Miikkulainen. Accelerated neural evolution through cooperatively coevolved synapses. *The Journal of Machine Learning Research*, 9:937–965, 2008.

[9] U. Grasemann and R. Miikkulainen. Effective image compression using evolved wavelets. In *Proc. of GECCO*, pages 1961–1968. ACM, 2005.

[10] A. Haar. Zur theorie der orthogonalen funktionensysteme. *Mathematische Annalen*, 69(3):331–371, 1910.

[11] M. Hausknecht, J. Lehman, R. Miikkulainen, and P. Stone. A neuroevolution approach to general atari game playing. *Computational Intelligence and AI in Games, IEEE Transactions on*, 6(4):355–366, 2014.

[12] J. H. Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intel.* MIT Press, 1992.

[13] C. Jacob, A. Lindenmayer, and G. Rozenberg. Genetic L-System Programming. In *Proc. of PPSN.* Springer-Verlag, 1994.

[14] J. Karel, R. Peeters, R. Westra, K. Moermans, S. Haddad, and W. Serdijn. Optimal discrete wavelet design for cardiac signal processing. In *Proc. of EMBS*, pages 2769–2772. IEEE, 2005.

[15] H. Kitano. Designing neural networks using genetic algorithms with graph generation system. *Complex Systems*, 4:461–476, 1990.

[16] J. Koutník, G. Cuccu, J. Schmidhuber, and F. Gomez. Evolving large-scale neural networks for vision-based reinforcement learning. In *Proc. of GECCO*, pages 1061–1068. ACM, 2013.

[17] J. Koutník, F. Gomez, and J. Schmidhuber. Evolving neural networks in compressed weight space. In *Proc. of GECCO*, pages 619–626. ACM, 2010.

[18] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing Atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[19] D. J. Montana and L. Davis. Training feedforward neural networks using genetic algorithms. In *IJCAI*, volume 89, pages 762–767, 1989.

[20] M. Riedmiller. Neural fitted Q iteration–first experiences with a data efficient neural reinforcement learning method. In *Proc. of ECML*, pages 317–328. Springer, 2005.

[21] T. Schaul, T. Glasmachers, and J. Schmidhuber. High dimensions and heavy tails for natural evolution strategies. In *Proc. of GECCO*, pages 845–852. ACM, 2011.

[22] J. Schmidhuber. Discovering problem solutions with low Kolmogorov complexity and high generalization capability. Technical Report FKI-194-94, Fakultät für Informatik, Technische Universität München, 1994.

[23] J. Schmidhuber. Discovering neural nets with low Kolmogorov complexity and high generalization capability. *Neural Networks*, 10(5):857–873, 1997.

[24] K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary comp.*, 10(2):99–127, 2002.

[25] G. Strang and T. Nguyen. *Wavelets and filter banks.* SIAM, 1996.

[26] C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.

[27] D. Wierstra, T. Schaul, J. Peters, and J. Schmidhuber. Natural evolution strategies. In *Proc. of CEC*, pages 3381–3387. IEEE, 2008.

[28] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[29] Y. Yekutieli, R. Sagiv-Zohar, R. Aharonov, Y. Engel, B. Hochner, and T. Flash. Dynamic model of the octopus arm. i. biomechanics of the octopus reaching movement. *Journal of neurophysiology*, 94(2):1443–1458, 2005.