Evolving Robust, Deliberate Motion Planning With a Shallow Convolutional Neural Network

Ben Jolley and Alastair Channon

School of Computing and Mathematics
Keele University
ST5 5BG, UK
{b.p.jolley, a.d.channon}@keele.ac.uk

Abstract

Deep Convolutional Neural Networks (ConvNets) have seen great success on machine learning tasks in recent years but have shown difficulty with tasks that require long-term deliberative planning. Whereas, purpose-built hybrid network architectures have been able to solve increasingly challenging deliberate tasks in two-dimensional and three-dimensional artificial worlds. Starting from a purpose-built network and transitioning to a general architecture, like a deep ConvNet, may retain long-term deliberative planning while allowing greater flexibility in the task domain. This paper employs a standard genetic algorithm (GA) to train the weights of a ConvNet with a recurrent 3x3 filter to produce robust and deliberative motion planning. This technique resulted in an average of 98.97% completion over 10,000 runs in the most difficult deliberate task. This demonstrates that a shallow ConvNet with recurrent connections is capable of producing deliberate and robust motion planning. Further, the evolved ConvNet exhibits superior motion planning in the most challenging environments, when compared to the previous task-specific motion-planning network.

Introduction

In recent years, the success of deep convolutional networks (ConvNets) on machine learning tasks demonstrates that advanced behaviours are obtainable without careful engineering and considerable domain expertise. This success is more relevant to the ALife community than ever due to the emergence of Deep Neuroevolution (Such et al. 2017). Deep Neuroevolution is able to trains ConvNets with a simple GA and can rival the performance of state-of-the-art policy gradient algorithms such as Deep Q Networks (DQN) (Mnih et al. 2015) and Asynchronous Advantage Actor-Critic (A3C) (Mnih et al. 2016). The ability of deep networks to generalise is demonstrated clearly by the results achieved on the Arcade Learning Environment (ALE) (Bellemare et al. 2013). ALE provides various Atari-2600 games to benchmark the performance of general game players. However, ALE highlights a weakness of ConvNets when an environment requires deliberate actions. Environments requiring reactive control can perform above human level but those with sparse rewards and reliance on long term deliberative planning fall below human performance. As seen in Pitfall and Montezumas Revenge.

Reactive and deliberative behaviours in combination have shown to be achievable with hybrid networks and have been demonstrated on Atari-like 2D environments (Robinson et al. 2007, Borg et al. 2011, Borg and Channon 2017). The behaviours demonstrated on this architecture has shown to be adaptable to more complex 3D environments (Stanton and Channon 2015); even adapting to tasks outside of the intended use (Stanton and Channon 2016). However, these architectures are purpose-built and limited to specific environments whereas ConvNets are able to adapt to many scenarios. Therefore, this work aims to take the hybrid architecture and begin to generalise the network while retaining the ability for long-term deliberate planning. Core to the hybrid architecture is a static sub-network for motion planning in dynamic environments. This greatly reduces evolution's cost and exploration while limiting its functionality. Replacing this network would allow evolution to produce novel behaviours suited to the task domain.

Work in Jolley and Channon (2018) attempted to replace the static motion planning sub-network using HyperNEAT, with some success. However, this paper aims to achieve greater functionality and extend evolution's role. First, this work will incorporate avoidance in motion planning, avoided in previous work due to its complexity. Next, the static network will be replaced by a ConvNet with recurrent connections. This form of motion planning requires contextual information about the entire environment. In a neural representation, it becomes too computationally intensive for each node in the environment to have an evolvable relationship with every other node. Therefore, methods are needed to restrict the relationship between network size and evolutionary complexity. Due to ConvNets' local connections, the size of a layer can increase without the need for a larger genome and greater complexity. Then, with a novel use of recurrent links, information from local areas can iteratively spread across the environment. Finally, by utilising a simple GA, all network weights are evolved simultaneously. This is inherently a more difficult task than previously attempted.

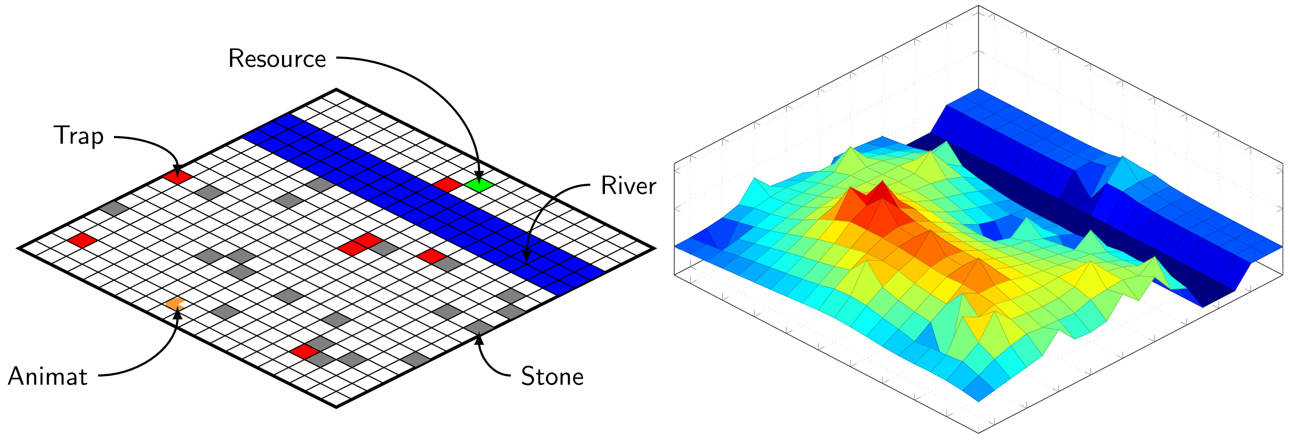


Figure 1: An RC World environment (left) with the corresponding activity landscape (right) produced via the shunting model.

Background

River Crossing Task

The River Crossing (RC) Task was devised in Robinson et al. (2007) to demonstrate high-level deliberative and reactive behaviours produced by a hybrid neural architecture. The architecture consists of a *Shunting Model* (SM) with static weights and a *Decision Network* (DN) with evolvable weights. The goal of an animat is to locate the resource on each RC world within 100 time-steps, while avoiding harmful objects. Once the resource is obtained, the animat proceeds to the next RC World. Each world increases in complexity via an expanding river obstruction between animat and resource. Animats must learn to build bridges to cross the river.

RC World RC worlds are constructed in a 20x20 bounded grid in which each cell can contain zero or one of each of four object types: stone, trap, water and resource; a cell containing none of these is deemed to contain grass. Traversing over a trap or water kills an animat; stones can be picked up and put down. Complexity is enforced by water placed across the world, creating a river obstacle. When placed on water, stones can create a bridge. Fitness is an integer from 0 to 4, determined by the number of RC worlds in which the animat reaches the resource. Animats are evaluated first of a world with river width 0 (no river), then 1, 2 and 3, stopping at first failure.

The Decision Network The DN is a standard feed-forward neural network that dictates how desirable or undesirable an object type is, and whether an animat should pick up or put down a stone. Topology consists of six inputs, four hidden nodes and five outputs. There are five binary inputs representing the presence or absence of each object type (including one for grass) and a sixth representing whether or not the animat is carrying a stone. There are four outputs corresponding to the desirability of each object type and a fifth used to determine whether the animat should pick up (positive output) or put down (negative output) a stone. The four desirability neurons use a hyperbolic tangent activation function and then values below, within and above the range [-0.3, 0.3] are converted to -1, 0 and 1 respectively. These output values are then multiplied by 15 to give *iota values*, which indicate the desirability of the object types in the environment.

The Shunting Model The SM is a topographically ordered neural network that produces a short trajectory between two positions in a dynamic environment without a learning process. First used in Meng and Yang (1998), the SM was applied to real-time robotics to solve maze-type problems by mapping the physical environment to positional neurons. Activity from desirable states propagates through the network to create an activity landscape. Peaks form at objectives and troughs at states to avoid. In the RC task the state attributes are provided via the DN and diffused via the following equation:

$$x_i^{new} = \min\left(\frac{1}{8}\sum_{j\in N_i}[x_j]^+ + I_i,\ \max_i\right) \qquad (1)$$

where x_i^{new} is the activation of neuron i; I_i is the external input determined by the iota value of the object present in cell i; N_i is the receptive field of i; $\max_i$ is the maximum iota value (15). Equation 1 is iterated fifty times to allow activity to propagate and stabilise across the 20x20 array of SM neurons, as shown in figure 1. Previous RC implementations of the SM used the *Additive Model*, but in extended work the implementation has changed for simplicity and clarity while maintaining the same behaviour (Stanton and Channon 2015).

ConvNets

ConvNets were originally proposed in LeCun et al. (1990) for handwritten digit recognition. They proved successful also in speech recognition (LeCun and Bengio 1995), object detection in natural images (Vaillant et al. 1994) and face recognition (Lawrence et al. 1997). The basis of the modern ConvNet architecture was introduced in LeCun et al. (1998) with LeNet-5. LeNet-5's success comes from deriving higher-level features from identified lower-level ones; this is achieved via local connections, shared weights, pooling and the use of layers. ConvNets can consist of convolution layers, pooling layers, non-linearity and a fully connected layer.

A filter iterates across the input vertices where convolution applies to extract spatial features. Each filter cell has an evolvable weight that is used during convolution. Depending on the task, many filters can be used to focus on key features, such as colours in an image. Outputs are then applied with a non-linear activation function. The dimension of the representation reduces by down-sampling the inputs in pooling layers. This creates an invariance to small shifts and distortions. The layer before the fully connected layer is flattened and each neuron interconnected. For example, in a classification task, the fully connected section will determine which features most correlate to a particular class.

Despite these initial successes, ConvNets' popularity would come to fruition with advances made in core computing systems. The use of GPUs allowed AlexNet (Krizhevsky et al. 2012) to train deeper and wider CovNets. In the challenging ImageNet competition, AlexNet achieved state of the art results. Various other advances were introduced in this architecture, such as dropout to reduce overfitting and ReLU to improve training times. Since then, ConvNets have been structured in many different ways. The work in Mnih et al. (2015) omitted the pooling layers to retain spatial information. Residual Networks (ResNets) have employed more than 100 layers to improve performance on visual recognition tasks (He et al. 2016). GoogLeNet's Inception module concatenates multiple filters, at varying sizes, on the same layer into a single output vector to abstract features from different scales (Szegedy et al. 2014). DenseNet connects each layer to every other layer in a feed-forward fashion which strengthens feature propagation (Iandola et al. 2014).

Experimental Setup

In these experiments, the functionality of the traditional shunting equation was replaced with a shallow ConvNet architecture with recurrent connections. Three sets of experiments were carried out for 100 runs. The first evolved the DN while the SM remained static. Next, the SM was evolved in combination with a static DN; the static DN was pre-evolved to achieve the highest fitness using the original architecture. Finally, the entire network architecture was evolved at once.

RC Task

The RC task remains similar to that of Robinson et al. (2007), with alterations introduced in Jolley and Channon (2018). Fitness is aggregated over 10 RC task attempts. Previous work has shown this to improve the robustness of solutions. To assess general performance, successful animats are subject to the Robustness Test. This simulates animats through 10^4 static RC world configurations with a river width of 3, the most difficult type of world this task offers.

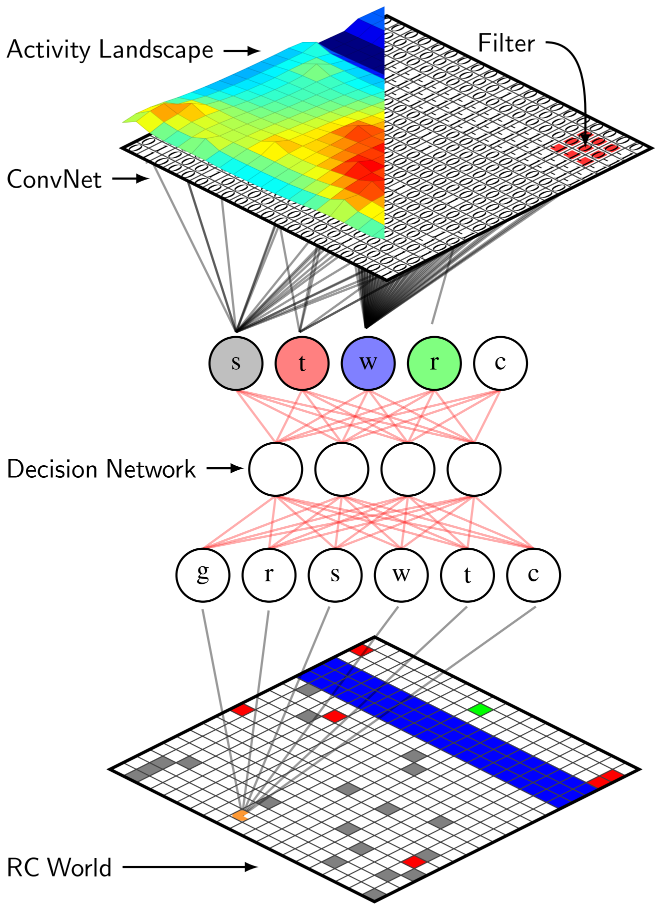


Figure 2: Neural architecture. Attributes at the agent's position (g=grass, r=resource, s=stone, w=water, t=trap, c=carrying) determine inputs to the Decision Network. Attributes of the RC World are converted to iota values via the Decision Network outputs. The iota values are feed to the ConvNet layer at the corresponding position of the attribute in the RC World. A 3x3 filter is passed across the layer to perform recurrent convolution, synchronously updating activities in the same layer. The network is activated a fixed number of times. The activity landscape is a visual representation of the output after completion. Links in red represent evolvable weights.

Network Architecture

The architecture introduced in this paper replaces the shunting model from Robinson et al. (2007) with a single layer ConvNet with recurrent connections, see figure 2. The ConvNet uses a 3x3 filter with a stride of 1 and padding of 1, to synchronously update activities in the same layer. After activation rectifier non-linearity (ReLU) applies to outputs. ReLU typically learns much faster in networks with many layers, shown in Glorot et al. (2011), which is not relevant due to our networks size. However for this experiment, the ReLU has the added benefit of replicating the shunting equation's ability to propagate only positive values. The filter takes as input cells' previous activation values added to DN outputs according to cell content (current object). This allows the propagation of values across the layer after multiple activations. The activity landscape can only function if the ConvNet output resembles the same dimensions as the input. Thus, the lack of pooling and use of padding is necessary in order for the output to remain the same spatial size.

For each world time-step, activity update in the ConvNet is iterated a fixed number of times. The RC world size instructs this number; for this work the value is 50. The ConvNet output represents the activity landscape. In the landscape, an animat follows the activation of its highest surrounding neighbour. The genotype sizes for the DN, SM and full network configurations are 44, 9 and 53, respectively.

Genetic Algorithm

This work uses an extremely simple GA, as a simple GA has been shown to achieve high quality results when evolving deep ConvNets for reinforcement learning tasks (Such et al. 2017). Our chosen GA evolves a population of 100 individuals, represented as neural network weights. In each generation, each individual's performance on the RC task is assigned a fitness value. Elitism is applied, storing 10% of the fittest individuals for the next generation. The remaining population is generated via single-point crossover and mutation, with parents selected at random from the previous generation. Each offsprings genome has a 25% chance of mutating a single parameter with an additive Gaussian noise value. Once the highest fitness has been achieved evolution ceases. If an animat does not achieve the highest fitnesss within the number of generations set, the attempt is considered a failure.

Results

100 runs for 10^4 generations are performed with each strategy on the RC task. During training, if animats are unable to complete all 10 RC tasks to the most difficult behaviour they fail the run.

- s-SM refers to the static-SM, where the DN evolves and the SM follows the shunting equation.
- e-SM refers to the evolvable-SM, where the DN is pre-evolved and the SM evolves.
- FN refers to Full Network, where both the DN and SM evolve simultaneously.

Set	Best	Worst	Mean	Stdev	Success
s-SM	1	138	43.58	30.73	100%
e-SM	43	4146	845.59	845.11	100%
FN	178	6137	673.61	1268.4	44%

Table 1: Best, worst and mean number of generations required to complete the task. Completion is defined as any animat completing the RC task at the hardest difficulty 10 times.

Table 1 demonstrates that every strategy is able to complete the RC task to the highest level of difficulty. As previously established, the s-SM is able to evolve to the highest standard of reactive and deliberative behaviours in all runs, which is also true for e-SM. Both e-SM and s-SM achieve 100% success in every run. However, all evolvable strategies on average take more generations to find a successful solution. Although FN is able to successfully train animats, these examples were a minority; FN also produced the largest deviation in the number of generations required.

Animats achieving completion at each run were evaluated in the Robustness Test with river width 3. This simulates animats through 10^4 static RC world configurations. Animat's performance on this test represents their ability to adapt to general environments. Figure 3 presents these results as well as those from the HyperNEAT architecture used in Jolley and Channon (2018). HyperNEAT and e-SM both utilise the same pre-evolved DN for direct comparison.

All strategies are statistically distinguishable from each other in their general completion ratings, established via a two-sample t-test with p values less than 0.05. s-SM still provides the most consistent results. e-SM provides comparable results with only a slight increase in mid-spread and a slightly lower average of 98.97% vs the 99.96% of s-SM. FN has the greatest deviation range but with a high completion rate of 96.72%. In the same task e-SM exhibits greater performance compared to HyperNEAT.

Another appealing aspect of the shunting equation is the efficiency of animats' movement, wherein animats move to the closest desirable object. As established in Jolley and Channon (2018), without efficient movement the actions of animats can seem undeliberate and unintelligent. So, the new architecture has to be evaluated in this area. The age of an animat after completion of an RC world is an accurate metric for judging how efficient an animat traverses the environment.

Figure 4 aggregates the average age of animats after the completion of an RC world on the Robustness Test. Results

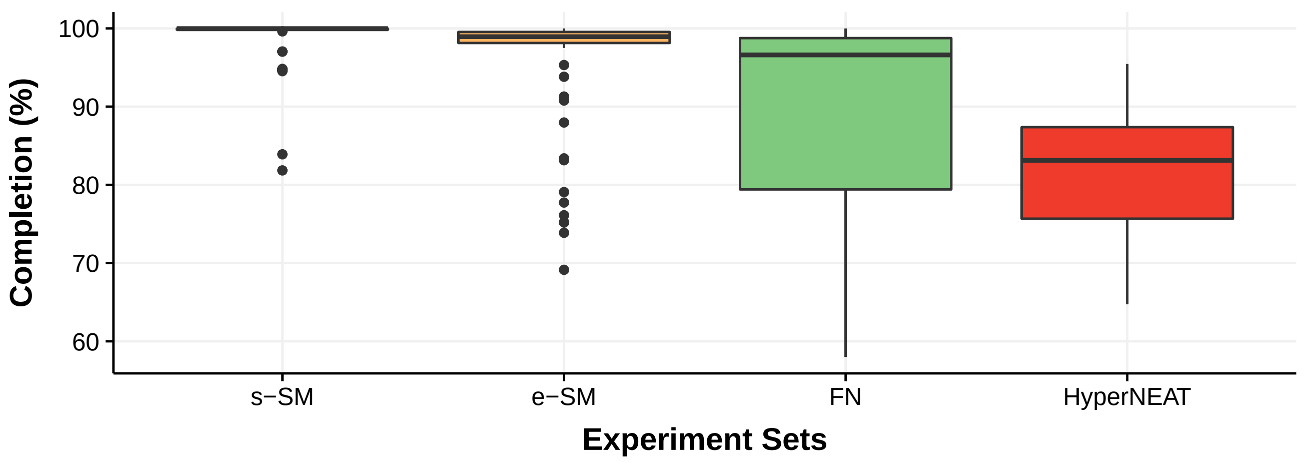


Figure 3: Completion percentage of Robustness Test across all strategies aggregated over 100 runs each. (s-SM = static shunting model. e-SM = evolvable shunting model. FN = Full Network.)

are limited to those that achieved 98% or above on the RC Robustness Test. Restricting to high completions provides an accurate comparison between results. HyperNEAT is not included as no runs achieved a completion percentage above 98%. As seen in figure 4, the s-SM stays consistent with the median of average ages at 45.2. Yet, the e-SM produces a lower median of 43.43 and the FN is able to drop further to 42.5. Although, a two-sample t-test shows that the two are statistically indistinguishable, but both have statistical significance over the s-SM. Both the e-SM and FN also have a lower spread than the s-SM but only the s-SM is able to achieve the lowest average age of 33.05.

Figure 5 provides a representation of the activity landscapes of s-SM, e-SM, and FN on the same RC world. An example has been purposely chosen in which the e-SM and FN have an age advantage over the s-SM; this is done to provide context to the improved average age. Viewing the RC world after completion of the s-SM shows two attempts at a connecting bridge. In motion, animats using s-SM locate stones close to the river and prematurely place the stone while traversing forward. This is due to the shunting equation forcing animats forward without adequate space to return to the center of the river. If stones are further from the river animats funnel to the center, as can be seen in the activity landscape when an animat is holding the stone. This is evident again when comparing stone locations from s-SM to e-SM and FN; those closest to the river in s-SM have been used to attempt an unsuccessful bridge.

Discussion

Compared to s-SM, both e-SM and FN were able to produce equally deliberate and robust motion planning in even the most difficult RC worlds, while also achieving, on average, more efficient planning. The s-SM still provides superior reliability in completion. Although, both the e-SM and FN are inherently more difficult tasks and degradation to completion average is minimal. In comparison to HyperNEAT, the average e-SM completion was higher than HyperNEAT's fittest animat. Further, e-SM in this experiment did not dismiss negative iota values like in the HyperNEAT architecture. e-SM benefits from using a ReLU to avoid the propagation of negative values. Thus, e-SM is capable of replicating the shunting equation's functionality. However, the FN informs how this architecture may adapt to future scenarios.

The FN shows that animats can learn reactive and deliberative actions simultaneously. It is reasonable to assume this architecture would adapt to similar tasks. The difficulty, in this task, comes from achieving the correct DN weights, while finding a working form of motion planning. Without a functional SM, it is difficult for the DN to converge on a correct combination of weights due to lack of feedback. Without a functional DN, the quality of the motion planning cannot be assessed. Each plays an important role in the success of the other. The larger deviation of completion results may be explained by the outliers from s-SM and e-SM in Figure 3. As expected, evolving motion planning can see instances in which the training phase presents RC worlds that do not translate to the variety of worlds in the Robustness Test. As a result, those animats have poor completion results, dropping as low as 69%.

Despite the success of FN on this task, it is uncertain how it will adapt to other architectures which use the SM. Less than half the animats trained were able to achieve the high-

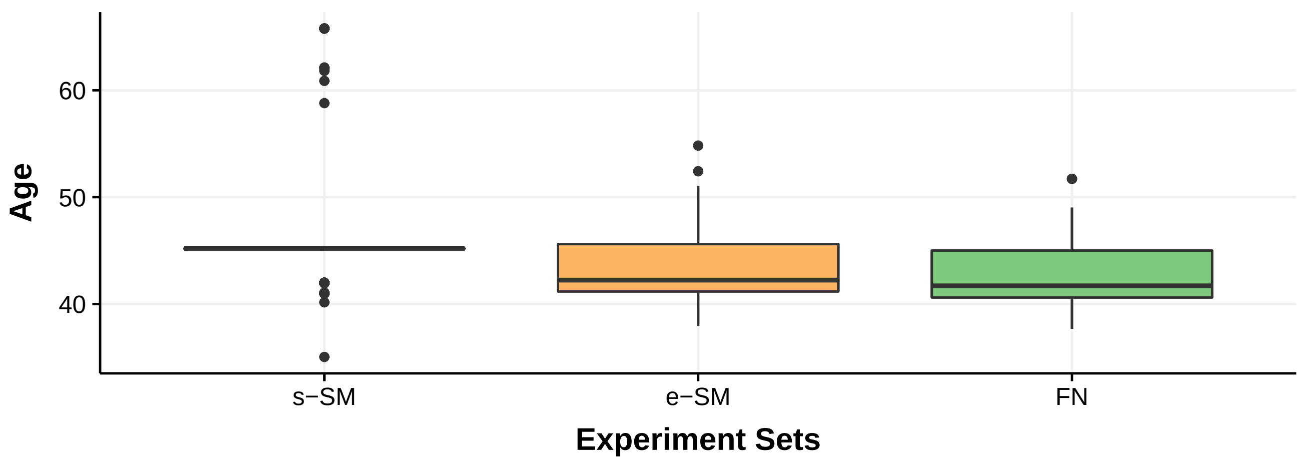


Figure 4: Mean age of animats after a successful RC world completion in the Robustness Test. All strategies are showing results from animats with 98% completion or higher in the Robustness Test. (s-SM = static shunting model. e-SM = evolvable shunting model. FN = Full Network.)

est fitness; this is a poor result compared to the perfect completions of evolving each independently. Those similar to this model, such as the RC+ task used in Borg et al. (2011), could likely see success with this network while experiencing a drop in completed runs. In contrast, a more difficult task may see issues with the completion rate. The 3D implementation of the RC task (Stanton and Channon 2015) implements the same DN and static SM with the addition of a fully connected feed-forward network, pattern generators, and outputs for joint motors. Even with a static SM, animats already see a drop to 65% of the population exhibiting the most difficult behaviour. Without an established motion planning system, the 3D implementation may find it difficult to gain traction.

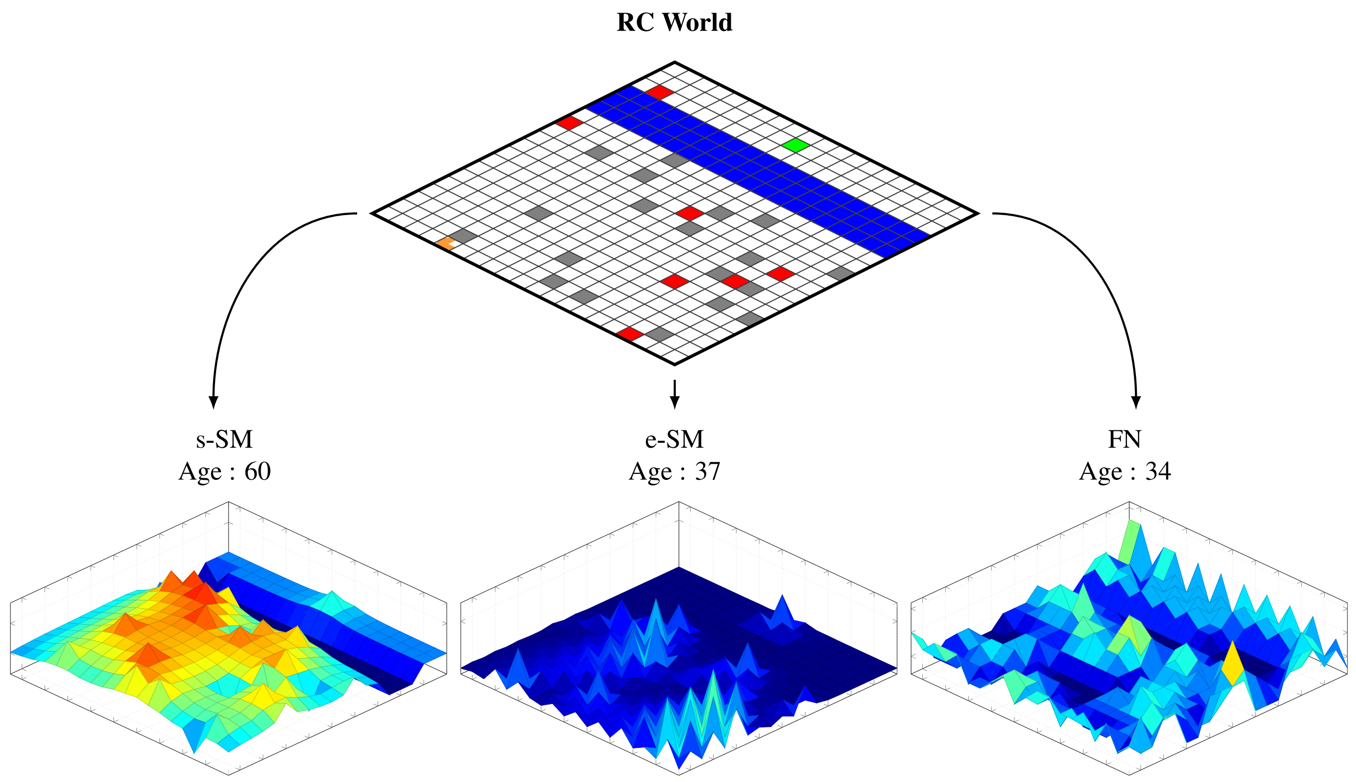
The evolvable SM demonstrated an unforeseen, novel advantage over a purpose-built system. When compared to the shunting equation, animats trained with e-SM and FN achieved instances of the highest completion rate as well as a superior average age on the Robustness Test. Analysis of the s-SM shows RC worlds with stones close to the river forced animats to create bridges in an unstructured manner. By contrast, e-SM and FN created deliberate bridge designs despite the location of the stones, see figure 5. Both evolvable strategies have activity landscapes which are difficult to interpret, thus the filter weights were examined. e-SM creates pathways to positive values via diagonal paths, whereby the corners are the most dominant of the 3x3 filter. FN operates in a similar vein to the shunting equation; the weights of the front and back of the filter are relatively equal. This forces animats forwards or backward depending on what is desirable. In motion, animats do not suffer from the long and questionable choices in the movement which appears with HyperNEAT. A highly pre-evolved e-SM could replace the SM in models that use it. This would allow superior motion planning without changing the network architecture or training.

Due to this hybrid architecture being the basis of further work, there are already working examples of greater general architectures this work can adapt. Currently, animats follow the highest surrounding iota in the activity landscape. Work in Stanton and Channon (2015) used a fully connected feed forward neural network to allow evolution to discover the relationships between iota values and movement; then, the outputs provide direct control over the animat's movement. Work in Borg and Channon (2017) generalised the DN to use RGB values as inputs, allowing greater abstraction from task-specific interactions. This also allows a fixed DN despite a varying number of object types in the world. As these considerations are addressed, the network may adapt to more complex scenarios and environments.

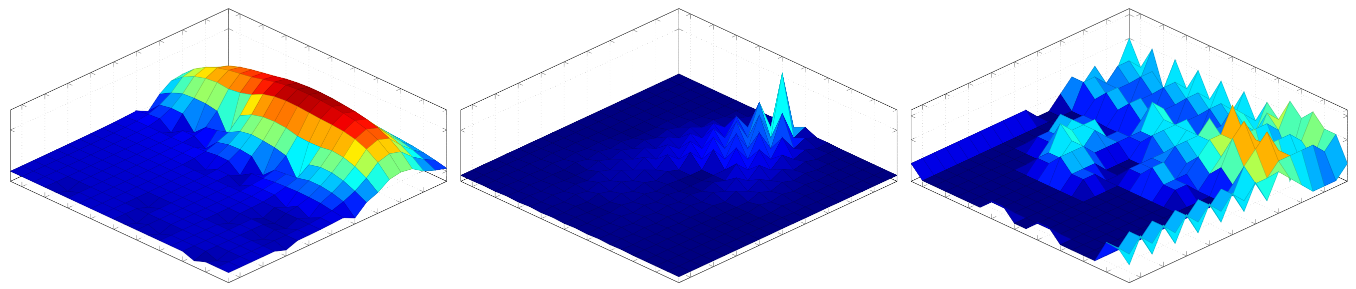
Conclusions and Future Work

This work demonstrates that a shallow ConvNet with recurrent connections is capable of producing deliberate and robust motion planning to the quality of a pre-designed solution, with greater efficiency. Further, both deliberative and reactive behaviour can be achieved by evolving the entire hybrid network simultaneously. These results were achieved with an extremely simple GA.

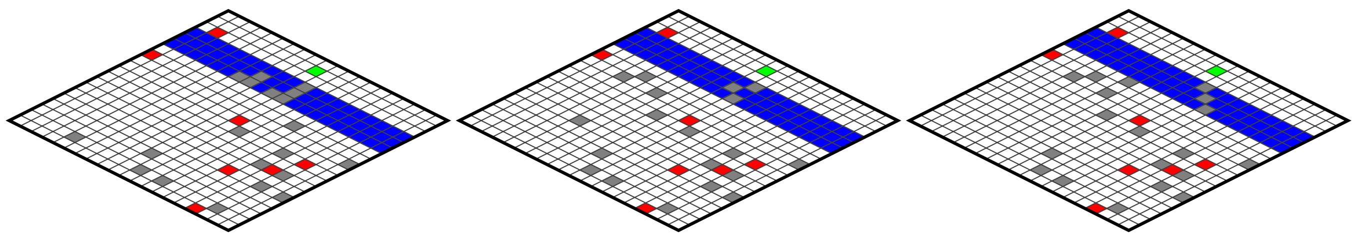
Individuals were exposed to a task of scaling complexity and required to complete the task at its most difficult behaviour multiple times. The use of a multi-evaluation fitness



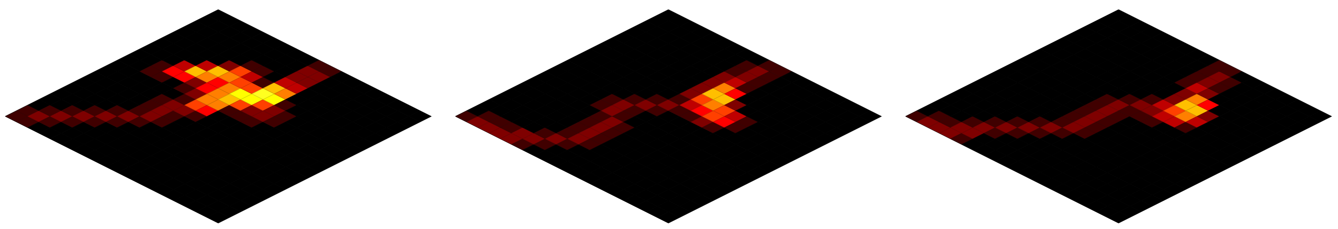
(a) Not holding Stone



(b) Holding Stone



(c) RC World after Completion



(d) Heatmap of movement

Figure 5: RC world with the corresponding activity landscapes for each strategy. Each strategy is labeled and given the age of the animat after completion. Each strategy has a completion of 98+% in the Robustness Test.

function in training, as a stopping criteria, encourages the evolution of motion planning that is adaptable to a variety of different world combinations.

This work has shown that a hybrid architecture that utilises a single layer ConvNet can achieve an average general completion that is within 3% of the static variation in a difficult deliberate task. The architecture could seemingly adapt to other tasks designed in the same manner.

The combination of the decision network and evolvable shunting model was highly successful in achieving efficient motion planning, when evolving only the shunting model and when evolving both simultaneously. The superior deliberate bridge building, achieved by each, shows potential in the future for further unforeseen, novel benefits when removing model restrictions. The limitations of the approach are the drop in successful runs when the network is evolved as a whole. Future work should continue to remove task-specific aspects in favour of greater control from the network. This would allow the architecture to adapt to unforeseen tasks.

References

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. (2013). The Arcade Learning Environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279.

Borg, J. M. and Channon, A. (2017). Evolutionary Adaptation to Social Information Use Without Learning. In *Proceedings of the 20th European Conference on the Applications of Evolutionary Computation*, pages 837–852. Springer.

Borg, J. M., Channon, A., and Day, C. (2011). Discovering and Maintaining Behaviours Inaccessible to Incremental Genetic Evolution Through Transcription Errors and Cultural Transmission. In *Advances in Artificial Life, ECAL 2011: Proceedings of the Eleventh European Conference on the Synthesis and Simulation of Living Systems*, pages 101–108. MIT Press.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep Sparse Rectifier Neural Networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 315–323. PMLR.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778. IEEE.

Iandola, F., Moskewicz, M., Karayev, S., Girshick, R., Darrell, T., and Keutzer, K. (2014). Densenet: Implementing efficient ConvNet descriptor pyramids. *CoRR, abs/1404.1869*.

Jolley, B. and Channon, A. (2018). Toward evolving robust, deliberate motion planning with HyperNEAT . In *Proceedings of the 2017 IEEE Symposium Series on Computational Intelligence*, pages 1–8. IEEE.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, pages 1097–1105. Curran Associates Inc.

Lawrence, S., Giles, C. L., Tsoi, A. C., and Back, A. D. (1997). Face Recognition: A Convolutional Neural-Network Approach. *IEEE Trans. Neural Networks*, 8(1):98–113.

LeCun, Y. and Bengio, Y. (1995). Convolutional networks for images, speech, and time series. *The Handbook of Brain Theory and Neural Networks*, pages 255–258.

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., and Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems 2*, pages 396–404. Morgan-Kaufmann.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278–2324.

Meng, M. and Yang, X. (1998). A neural network approach to real-time trajectory generation. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, volume 2, pages 1725–1730. IEEE.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. (2016). Asynchronous Methods for Deep Reinforcement Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1928–1937. PMLR.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529.

Robinson, E., Ellis, T., and Channon, A. (2007). Neuroevolution of agents capable of reactive and deliberative behaviours in novel and dynamic environments. In *Proceedings of the Ninth European Conference on Artificial Life*, pages 345–354. Springer-Verlag.

Stanton, A. and Channon, A. (2015). Incremental Neuroevolution of Reactive and Deliberative 3D Agents. In *Proceedings of the European Conference on Artificial Life 2015*, pages 341–348. MIT Press.

Stanton, A. and Channon, A. (2016). Neuroevolution of Feedback Control for Object Manipulation by 3D Agents. In *Proceedings of the Fifteenth International Conference on the Synthesis and Simulation of Living Systems (Artificial Life XV)*, pages 144–151. MIT Press.

Such, F. P., Madhavan, V., Conti, E., Lehman, J., Stanley, K. O., and Clune, J. (2017). Deep Neuroevolution: Genetic Algorithms Are a Competitive Alternative for Training Deep Neural Networks for Reinforcement Learning. *CoRR, abs/1712.06567*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2014). Going Deeper with Convolutions. *CoRR*, abs/1409.4842.

Vaillant, R., Monrocq, C., and Le Cun, Y. (1994). Original approach for the localisation of objects in images. *IEE Proceedings-Vision, Image and Signal Processing*, 141:245–250.