# Categorizing the Environment for Improved Performance at a Light Eating Task

Emanuel Schorsch

May 11th, 2012

**Abstract**

This paper investigates the possibility of integrating an evolutionary system with category awareness. The categories were preformed using RAVQ. The controllers were then evolved using NEAT for 200 generations. Within this time period both the controls and the integrated setups were able to successfully solve the given task. The simulations were all done in Pyrobot, allowing dramatically more trials to be performed. The task was to eat as many light pellets as possible before the robot ran out of energy, using mainly light sensors with limited range. While the results obtained are very ambiguous it is clear that integrating categories with evolution is no worse than performing straight evolution. Due to the design of the experiment it is possible that the simplicity of the task prevented the advantages of categorizing the world from showing through. Future work is necessary to resolve the ambiguities and further fine tune the parameters for optimal results.

## Introduction

One of the more basic problems that robots face is categorizing their environments. This is a task that humans so effortlessly excel at yet robots have significant difficulty performing this basic feature. One of the recent promising approaches towards categorizing the environment was described in *Sensory Flow Segmentation using a Resource Allocating Vector Quantizer* by Fredrik Linaker and Lars Niklasson [2]. Their paper lays out a new framework called RAVQ which can be used to abstract sensory data from an environment in order to form higher-level categories.

Their system maintains a moving average of the last $n$ inputs, which is user specified. The higher the number of inputs in the moving average the less susceptible to noise the system will be. If the environment is very static then the user would want to set $n$ to be higher. The moving average is then incorporated as a new category, ie. added as a model vector, if it is sufficiently "different" and "stable". The precise definition of "different" is determined by user choice of $\delta$. The choice of $\delta$ is the mismatch reduction criterion and it signifies that the moving average accounts significantly better for the inputs than any other existing model vector. The lower the choice of $\delta$ the more model vectors that will be added, perhaps spuriously. Finally "stability" is defined by the choice of $\epsilon$ and says that the deviation of the inputs within the model vectors should be below $\epsilon$. This prevents fleeting, meaningless categories from being created which arose due to noise. However, the environment may have meaningful categories which capture transitions between different stages, which cannot be captured unless $\epsilon$ is set higher.

Their system was very successful at identifying categories in a static environment in which the salient features were the walls, corners and a corridor. They significantly improved upon previous

systems both in terms of speed in categorizing the environment and adaptability to new aspects of the environment which hadn't necessarily existed before. However, one of the drawbacks of RAVQ is that it is mainly intended for relatively static environments and cannot adequately capture transitions between different categories.

Naturally beyond merely categorizing the environment we also want robots to be able to develop strategies for coping with whatever task we assign it. One of the earlier attempts to evolve a system which could handle a complex task was detailed in *Incremental Evolution of Complex General Behavior* by Faustino Gomez and Risto Miikkulainen [3]. Their motivation was to be able to solve more general complex tasks, such as prey capture, with neural networks. What had previously frustrated this goal is the sheer complexity of the task prevented neural networks from evolving the appropriate topology and weights. The neural network would either find a solution which only worked in the very specific environment the robot was in, or it would get stuck in a highly inferior local maximum.

Their innovation was to aid the robot's learning by breaking the task down into easier subtasks, in order to allow the robot to gradually work up to solving the most complex problem. This is necessary as in a task such as capturing prey there are many components which can't be expected to all be evolved simultaneously. It is necessary to first perhaps teach the robot to move towards the prey and then attempt to teach it the more difficult concept of remembering where the prey was previously so as to trap it even when it is not in range. This is well grounded in general developmental principles, the idea that learning occurs when an attempted task is slightly too difficult yet not too difficult. The authors attempt to find this careful balance by subdividing the task appropriately and then evolving a neural network to solve the progressively more difficult "ladder" that has been constructed.

The Neuro-Evolution system that they used is Enforced Sub-Populations (or ESP). ESP allocates a separate population for groups of neurons, and only allows a neuron to recombine with members of its own sub-population. Additionally a complete network is not evolved, rather the individual neurons are evolved and then the network is filled in. Each individual neuron has a chromosome which encodes the input and output connection weights. The top-quartile of neurons (ranked by fitness) are crossed over with a higher ranking neuron. The bottom half is filled with mutated neurons from the worst individuals in the previous generation. The advantage of the sub-population of neurons is that each sub-population specializes and does not interfere with other neuron's specialization by getting mixed together. Further, the neuron's evaluation is more consistent since it is always evaluated in the context of its "peers". Finally, ESP allows the evolution of recurrent networks. This is not possible without the contribution of the sub-population as recurrent networks are especially sensitive to context and neighbors. The neurons can then evolve with some degree of certainty as to what their neighbors, which they will be connected to, are like. However, a consequence of this is that within each sub-population diversity will decline as the trial is run.

ESP is then combined with Delta-Coding to allow the authors to achieve the successful transference of an evolved population from one task to the next harder task. Delta-Coding in essence slightly alters the chromosomes of the best individuals so as to search in the neighborhood of the best solution while still maintaining some semblance of diversity. The degree of change to the chromosome is specified by Delta, so for the transitions to relatively novel new tasks Delta would be set higher. The values of Delta were chosen according to a probability function which allowed drastic changes but only very infrequently, the exact details can be seen in their paper.

Combined with ESP the authors were able to successfully implement this system to evolve a

controller which could hunt down a prey which moved unpredictably and got a head start. This forced the robot to rely on short-term memory in order to be successful. When the controller was trained on the final task right away the performance was poor, hovering barely above chance. However, when the controller was trained on a simpler environment first, performance was markedly better when the controller arrived at the final task. The authors gradually increased the head start and speed of the prey so that the controller initially almost always had the prey in its field of vision. As the speed of the prey was increased the probability that the prey left the vision range of the controller increased forcing the robot to naturally increase its short term memory abilities.

Perhaps most interesting was the authors analysis of the results. They attempted to interpret the neural network through a lesion study, which involved disabling input connections to selected neurons. This ideally allows the authors to see whether any of the neurons play unique roles or are specialized in any way. Instead what they found was that the performance of the network was remarkably robust with performance declining slightly for each neuron removed, almost irrespective of which neuron was removed. This is remarkable in that it shows that the neural network is very interconnected and it is the interplay between different neurons in a highly complex manner that eventually generates the desired behavior.

Continuing this work Risto Miikulainen and Kenneth Stanley in *Competitive Coevolution through Evolutionary Complexication* lay out a system to better enable the evolution of complex networks. Miikulainen's previously described paper shows its influence here with the idea of beginning with simpler tasks. While the NeuroEvolution of Augmenting Topologies (NEAT) method does not use this exact technique it maintains the spirit and rationale. The general idea is to start with a small neural network and gradually allow it to complexify until it reaches a solution. This enables the creation of an adequate solution in a small and simple search space before venturing into greater complexity. This resembles the idea of Delta-coding in the previous paper which used previous fit individuals to determine the neighborhood to explore in the future.

However, one of the key differences, and the motivation for the development of NEAT is that most previous work uses a fixed length genome. This makes sense, to prevent the search space from becoming prohibitively large, but it also can prevent the network from having an adequate genome length as it needs. However, equally unsuitable would be to simply start with an arbitrarily large genome and search that enormous space. The solution the authors come up with is to explore a subset of the larger space, the subset being determined by the structure of the fit individuals from the previous generation.

This theory of elaboration, so that future searches are directed by past ones is necessary to preserve successful features of the controller. Once a robot has evolved say how to chase down prey you want to add on to that ability not forget that ability and search from scratch in an attempt to happen upon the perfect solution. The main breakthroughs in the NEAT system come from its combination of its genetic encoding, historical markings, and speciation.

Genetic encoding clearly works, as it has been demonstrated in the rather successful large scale experiment on Earth. In order to get the results nature has it is helpful to mimic many of her features. The premise of the experiment the authors plan to conduct is to start an arms race of sorts such that robots compete against each other gradually developing more complex strategies to cope with the increasing fitness of their opponents. This occurs in nature as species compete for resources each developing their own coping mechanisms. Then the final step once you have separate species with largely separate genomes is to figure out how to evolve them in an appropriate manner.

The genes in the authors system encode both the neuron's connections and the weight of the

connections. The mutation of the gene can then result in either more connections, differently weighted connections, or a new node altogether. The rate of each of these mutations is determined by the user. Mutation is only one part though of nature, sexual recombination is also needed. This is achieved through the innovation of historical markings. The authors realized that the history of a gene could be used to match up different genes in different individuals. They implemented a global innovation tracking system, which tracked new genes. This enabled similar genes to be crossed over with similar genes, preserving general function. This then allows potentially very different individuals to be recombined into a new possibly better individual.

Using this system the author's were able to significantly outperform systems which used fixed topologies. While this is not all that surprising and it would perhaps be more interesting to compare NEAT to Miikulainen's previous system combining ESP and Delta-Coding it is nonetheless clear that NEAT is very successful. The authors demonstrated the evolution of complex strategies which developed over time in response to the strategies of the opponents. This is quite relevant to my goal of developing a controller which can quickly and efficiently gather "food pellets" by incorporating sensory data with abstracted category information.

## Experiment

The experiments were all run in Pyrobot, a robot simulator. This was for convenience and speed. While ideally experiments could be run on real robots with the noise they entail it is simply not feasible to carry out the number of runs required to evolve large populations. Using the Pyrobot simulator greatly sped up each of the experiments so that multiple different trials could be run. One of the advantages of using the simulator is that multiple trials can be conducted simultaneously. This was done by not actually running a graphical simulation but instead stepping through what the robot would encounter and just presenting the data. The absence of graphics dramatically speeds up evolution times, as would be expected.

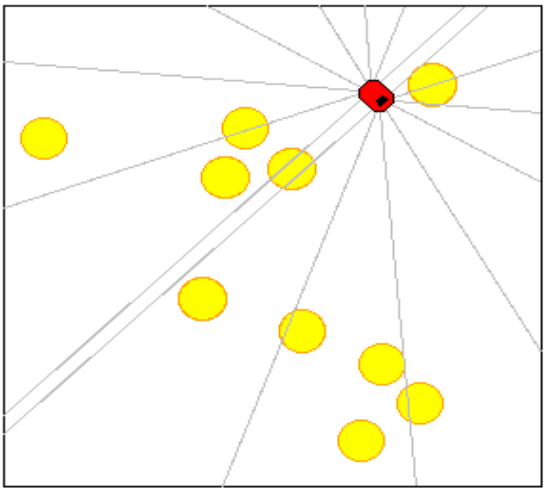


Figure 1: The Pyrobot light world

The task was to evolve a controller for a robot to eat as many light pellets as possible before

it died. The robot eats lights automatically as soon the brain detects it is on top of a light. This process occurs independently of any other action. The robot used was a red pioneer robot in a world with ten lights randomly positioned (see figure 1). The robot is initialized with 20 energy units and loses .3 energy units at each time step. If the robot "eats" a light pellet then 10 energy units are gained. The robot is positioned randomly in the world at the start of each of the three trials, with each trial also having random light placement. When the robot's energy is less than 10 a warning is initialized to $1 - \frac{energy}{10}$. The fitness function was experimented with quite a bit in earlier experiments. Ultimately an exponential fitness function was chosen, so as to differentiate between very good performances.

$$\text{Fitness } = \frac{L_1^2 + L_2^2 + L_3^2}{10^2 \cdot 3}$$

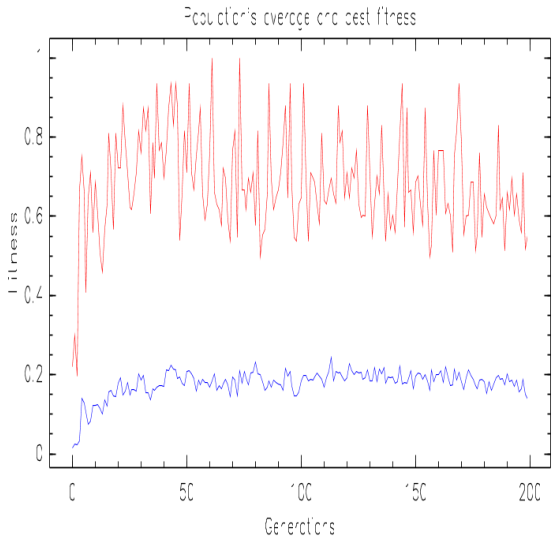
Where $L_n$ represents the light pellets eaten in the $n^{th}$ trial, and the denominator represents the maximum fitness attainable, so as to scale the fitness between 0 and 1 for NEAT.

The experiment was to compare the control and integrated robots. The difference between the control and the integrated robots were that the integrated robots had additional input nodes which were activated depending on which category the robot was in. The control robot only had sensory data. In all cases the robots had the light sensors and the warning described above as input. There were two light values, one on the front left of the robot and one on the front right. The pioneer robot has 16 sonars sensors in Pyrobot. When the sonar sensors were activated the minimum of the front four sonar sensors were input as the "front" sonar, and the minimum of the back four were input as the "back" sonar value.

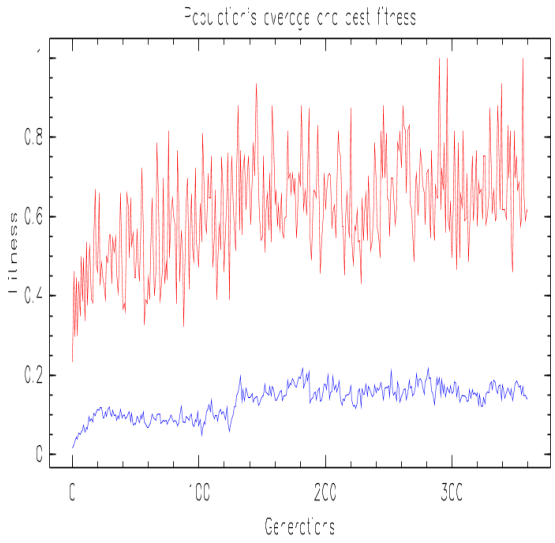
Using only light sensors and the warning value, NEAT evolved a population of feed-forward networks for 200 generations. At the end of this the best individual was run through 3 trials, while RAVQ was active. The robot fed in both the light values and the warning value to RAVQ and produced 5 categories, which were saved to a separate file. This file was then, unless specified otherwise, used by each of the robots in the main experiment. There were then two types of integrated robot, the first was the integrated-raw robot. This robot had the standard inputs, but it also had the data from the winning model vector fed in as additional input. The other integrated robot was the integrated-boolean robot. This robot instead of having the raw data fed in had an additional input neuron for each category (five) and each input neuron would be 1 if the robot was in that category and 0 otherwise. The buffer history size for the RAVQ was 2 unless otherwise specified.

The populations were initialized to a size of 100 using NEAT. The settings for NEAT were largely untouched except for the probability of adding connection and nodes which were set to .07 and .05 respectively. Although these parameters were set to be relatively low they still led to very complex networks after many generations were run.

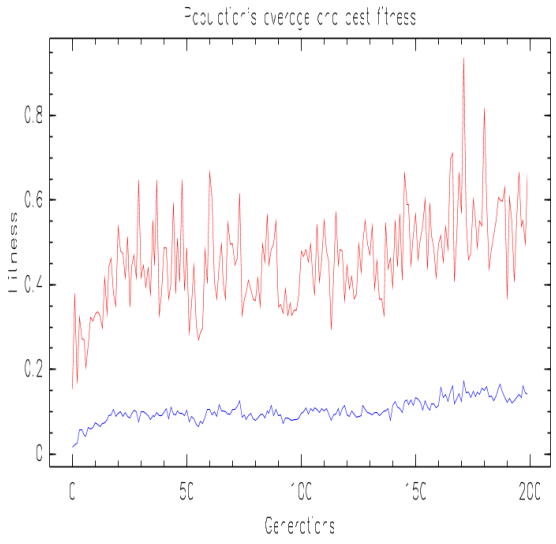
## Results and Analysis




(a) Control




(b) Raw Integrated




(c) Boolean Integrated

Figure 2: On the upper left you can see the average and best fitness of an average run of the control robot, on the upper right the integrated robot with raw model vectors as input. On the bottom is the average fitness of an integrated robot with boolean values for each model vector as input.

Once the parameters are tweaked satisfactorily, the integrated robots should be able to outperform the control as they have access to the same information, just more. In the first line of experimen-

tation each different type of controller was evolved for 200 generations. As you can see in figure 2, neither type of integrated robot outperformed the control. The control attained the highest average fitness out of the three, as well as attaining it well before the other two. This signifies that the addition of categories requires further complexity in order to cope with the additional information. This is corroborated by the fact that the control robot plateaued almost immediately whereas the integrated robot with raw model vectors as input continued to improve steadily.

Towards the end of the 200 generations, the integrated robot with raw data as input has achieved fitness on par with the control. However, in the specific trial Pyrobot had to be resumed and so accidentally ran for 360 generations. One can see that after the integrated robot has achieved parity with the control further improvement is rather hard to come by. What this suggests is that perhaps the task is structured in such a way that it is very difficult for any of the controllers to evolve a solution with fitness significantly above .2.

Regardless it is quite evident that the robots that use categories are performing inferior to the robots that do not. One possible explanation to this is that the categories that were formed were unhelpful, or merely reflected noise. On examination of the categories it was seen that they reflected reasonable situations. Categories ranged from no light, a lot of light, strong light on my right, all seemingly salient features of the environment. However, this is being judged in a top-down approach in which the human perspective is being used to analyze the potential utility of a feature that may be used in an entirely different fashion in a robot.

One possibility that I attempted to investigate was the impact of the buffer history size. What this setting is a measure of is how quickly is the environment changing. Given that lights come and go due to only a small radian shift a low buffer history of 2 was initially chosen. However, I ran two trials of identical integrated robots with just the buffer history size changed between them. One of them was set to 2 and the other was set to 5. The results were slightly, but consistently in favor of the robot with only 2 vectors in buffer history. This confirmed the intuition that the categories in the world are subject to rather rapid change.

However, on closer examination it was noticed that the categories being used were excessively spurious. Somehow when the categories were being created on the initial run through some setting had been tweaked which led to the creation of over 200 categories. This stands in stark contrast to the 5 of the main experiment. Nonetheless, the robot with shorter buffer history not only performed better but nearly outperformed the control previously discussed. One caveat to this is that these robots also had sonar sensors as well, which most likely aided their performance. This makes this side experiment rather less impressive, as what likely happened is that since the number of model vectors was so outrageously high the model vectors merely duplicated the raw sensory data. The robots were then simply evolving with two copies of all the incoming sensory information, which says nothing about the impact of categories. Unfortunately due to time constraints these side trials were not able to be re-run with the appropriate categories.

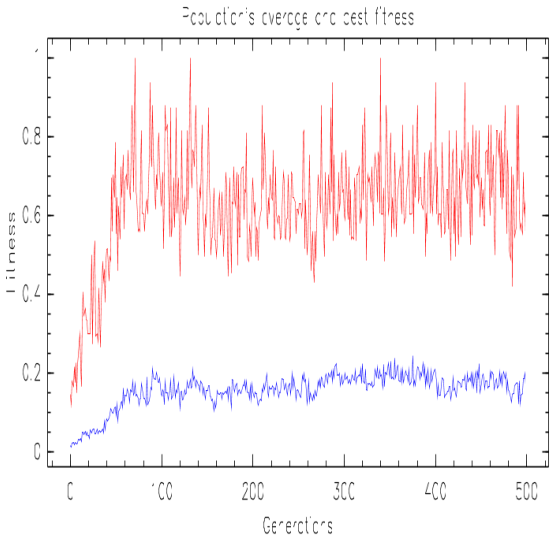



Figure 3: Control

A control was run for 500 generations (Fig 3) to test the hypothesis that the experiment was setup in such a way as to preclude evolving a fitness above .2. The control robot in addition to the standard sensors other trials were given, was also equipped with front and back sonar sensors. As can be seen in figure 4 the control never went significantly over .2. To keep the fitness function in the perspective of light pellets, there are 30 light pellets, 10 in each trial. A fitness of .2 means the robot ate approximately 5 light pellets each trial. Considering that the task gets increasingly difficult as light pellets are not re-populated this is very good performance. In addition it shows a very smooth controller qualitatively. The final evolved product has an appropriate over-arching strategy which is to circle the outside, in addition to short term tactics, which are to eat the light pellets. This is clearly demonstrated when in one trial the robot ate all the light pellets and so followed the wall on its right until it died. This shows the robot had evolved a strategy for re-orienting itself and finding the remaining light pellets even when they were out of view.

## Conclusion

It is promising to see that the additional burden of knowing the current category did not significantly impede the performance of the robot. Nonetheless, it is clear that in order to justify incorporating categories into robots, it must show some kind of advantage. The time constraints prevented the appropriate number of trials and settings from being run with the integrated robot. This being said it would be surprising if the categories or settings were so detrimental as to affect the results in a truly meaningful way. The performance of the integrated robot was somewhat invariant to changes in parameters as far I could tell.

One possible explanation for why the categories didn't seem to matter is that the task was simply not complex enough. Both the control and the integrated robots would eventually reach the "maximum" fitness of .2 and so any change in parameters to the categories wouldn't have a noticeable effect. This is a significant problem as the main advantage in categorizing the environment is

to aid decision making over sensory overflow. When the robot is drowning in magnitudes of data that is when using the preformed categories would have the biggest advantage.

# 1 Future Work

There are multiple areas for future progress with this experiment. One avenue to explore is the impact of categories. What kind of categories are the best? How is it best to input them? Perhaps even giving category information to a controller in place of sensory information, as opposed to in conjunction. These are all important as it may very well be that when the categories aren't delivered in the right way their message becomes distorted and useless.

Further the integration of RAVQ and NEAT should be tried in other more complex environments. Presumably RAVQ will be most beneficial in environments which have a lot of sensory data coming in. Regardless the task should clearly be difficult enough, and structured, so that one avoids the local maximum that occurred in this experiment. If categories could be implemented successfully it would be very promising as it would mean the data techniques for abstracting over the robots world could actually be turned around and utilized by the very same robot to aid its decision making.

# References


[1] Mikkulainen, Risto, and Kenneth O. Stanley. *Competitive Coevolution through Evolutionary Complexification.* Journal of Artificial Intelligence. 21. (2004): 63 - 100. Print.

[2] Linaker, and Niklasson. *Sensory Flow Segmentation using a Resource Allocating Vector Quantizer.* (2000)

[3] Gomez, F., Miikkulainen, R. *Incremental evolution of complex general behavior. Adaptive Behavior* (1997): Adaptive Behavior, 5, 317342.