

Joe Blaylock  
B553 Final Paper  
05/2007

I set out with manifold goals for my final project. I wanted to explore new ideas, write interesting software, and do something that I could think about as being cognitively interesting. During the earlier part of the semester, I batted around evolving Traveling Salesman approximators and writing a Q-Learning AI for Quake. I eventually settled on reproducing a paper of Randall Beer's, "Toward the Evolution of Dynamical Neural Networks for Minimally Cognitive Behavior"<sup>1</sup>. The notion that embodiment is fundamental to cognition has always appealed strongly to me, and so I'm naturally predisposed to be sympathetic to the dynamical systems model. Combined with reasonably easy access to Randy himself, my own curiosity about CTRNNs, and the nice-looking animations shown during Randy's "Frictionless Brains" talk, working with this paper was ultimately an easy choice.

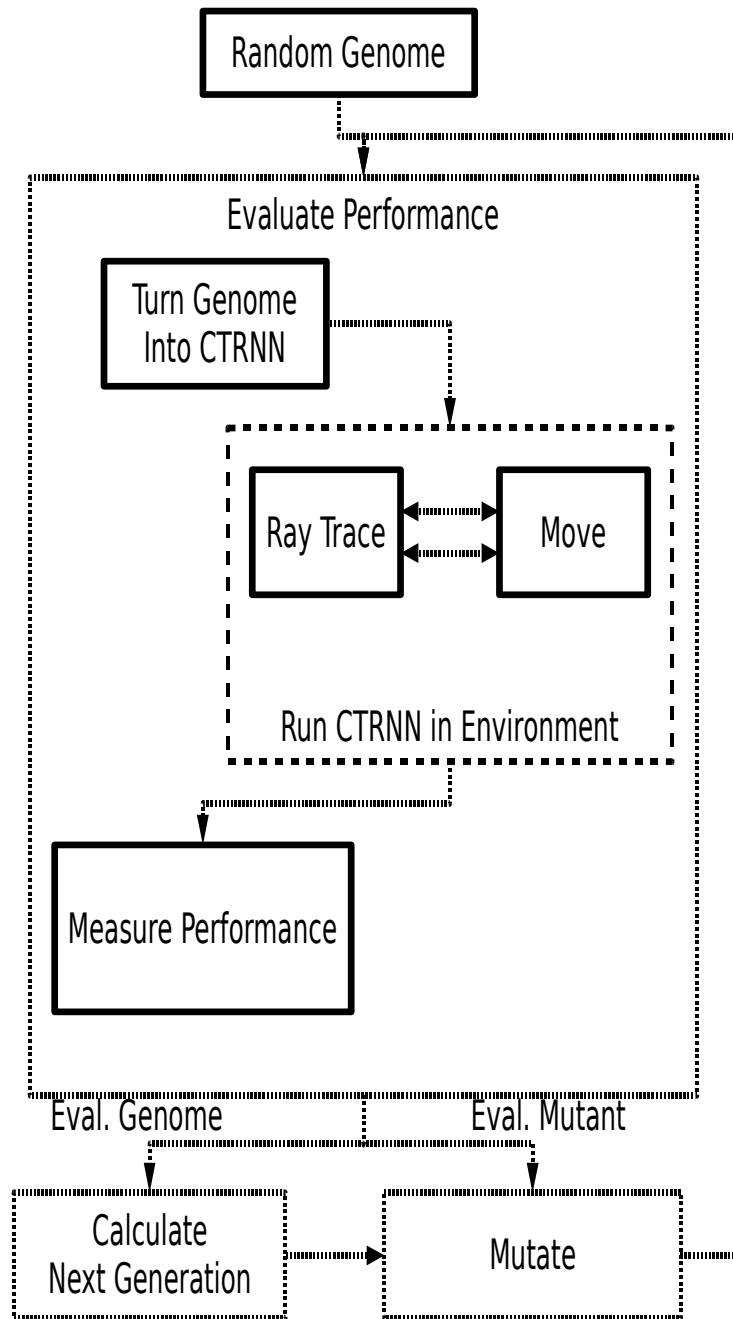
It fits in well with the orientation of B6553, too: we've already done projects using genetic algorithms to evolve neural networks, in the Bunnyverse. I set out wanting to reproduce Randy's infrastructure in Python and then use it to run the same set of experiments outlined in his paper. I counted eight of them, ranging from simple Braitenberg-vehicle behavior (chasing a ball) to object distinction and manipulator alignment. I figured that the software would be the hard part, and that running the experiments and gathering data would be fairly straightforward once the software was all working.

Fortunately, I was wrong, and it's given me the opportunity to learn a lot of useful new things. First of all, my time estimates for the programming were way off. Randy's lost the source code for those experiments over the years, but he still had his CTRNN and his Genetic Search libraries (C++) laying around, and I figured that with those at my disposal, cranking out the Python equivalents wouldn't take much time at all. Those parts were indeed fairly straightforward, once I understood what Randy's code was actually doing, and I was even able to undertake a true translation, so that the new version is properly idiomatic Python. What I never anticipated was that the environmental simulation (the specific code Randy was missing) would be so challenging.

It may be easier to see why if I take a moment to digress and discuss the structure of the code:

---

<sup>1</sup>Beer, R.D. (1996). Toward the Evolution of Dynamical Neural Networks for Minimally Cognitive Behavior. In P. Maes et al. (Eds.), *From Animals to Animats 4: Proc 4<sup>th</sup> Int. Conf. on Simulation of Adaptive Behavior* (pp. 421-429). MIT Press. (Also at <http://mypage.iu.edu/~rdbeer/Papers/SAB96.pdf>)



Genomes are instantiated into network architectures: gains, biases, weights. This network is used to drive a mobile agent with a ray-tracing eye as it tries to perform various tasks. Its performance is compared to the performance of its mutant child, and the superior of the two is kept.

The agent in the experiments is a circle, possibly with an appendage (which I never got around to implementing), which can move back and forth in a limited field. It's equipped with a CTRNN (parameterized by its genome) and a five- or seven-ray foveated eye, which works by raytracing objects moving in the environment. For a single genome evaluation, this model instantiates the agent's network, and then runs the agent multiple times (for statistical significance) with different (randomized) environmental parameters. For example, in the only experiment that I had time to complete, the agent was supposed to align its body to catch a falling ball. The ball falls in a random direction at a random speed.

Building the raytracing eye turned out to be really very challenging, particularly since I started with very naïve use of trigonometric relations (which was very computationally expensive). Eventually I used the Ray/Sphere Intersection algorithm employed by PovRay, ported to Python, with NumPy for the vector operations. Once I managed to find an eye input strategy that worked, I found I enjoyed working with the vision system which is fortunate, since it took me as long as the entire rest of the code base to get right. I had always heard that NumPy was fast, but I was very surprised when I profiled the ball-alignment experiment and found that the raytracing code (called more than every other part of the system combined) used around 5% of the total CPU time.<sup>2</sup> It's a very pleasant discovery, and I plan to use NumPy every chance I get in the future.

Even with NumPy, though, these experiments are slow. The first one in the paper object orientation by a bilaterally-symmetric feed-forward network is the simplest both conceptually and computationally, and took 26 minutes to complete. Having only finished debugging my codebase this afternoon, not much time is afforded to me for actually running experiments. However, the high degree of accuracy (99%) of the evolved agent in the ball catching task is similar to that achieved by Randy's agent in his paper, and I find this encouraging.

In the future, I'd like to extend the use of NumPy throughout the codebase, to the CTRNN and Search code, and generally optimize it for performance. The terrible time I had getting everything debugged also has me thinking that unit tests are in order. There are some warts on the API that I'd like to correct, and also some style errors. Beyond the codebase, though, I'm still fascinated by the whole embodied cognition milieu, and I'd like to continue constructing and running the experiments that Randy outlined in this, and other related papers.<sup>3</sup> Eventually, I'd like to devise my own Frictionless Brains experiments; possibly with an eye towards instantiating a CTRNN in hardware.

---

2 Mike may want to consider using a similar algorithm with NumPy for the Bunnyverse; it would relieve him of his dependence on Tkinter. I'd be happy to help out as much as I can.

3 Such as Slocum, A.C., Downey, D.C. and Beer, R.D. (2000). [Further experiments in the evolution of minimally cognitive behavior: From perceiving affordances to selective attention](#). In J. Meyer, A. Berthoz, D. Floreano, H. Roitblat and S. Wilson (Eds.), *From Animals to Animats 6: Proceedings of the Sixth International Conference on Simulation of Adaptive Behavior* (pp. 430-439). MIT Press. (Also at: <http://mypage.iu.edu/~rdbeer/Papers/SAB2000.pdf>)