2

Informing Game AI through the Study of Neurology

Brett Laming

2.1 Introduction
2.2 Critical Thinking
2.3 Neurology
2.4 Conclusion

2.1 Introduction

Human beings are fascinating machines, and the world of science is an amazing place. But as AI programmers we usually end up closeted in computer science and conditionals, rather than taking inspiration from the worlds we are trying to emulate. Math, psychology, biology, engineering, and the physical sciences all have a part to play in the inspiration and mechanisms we use in our daily jobs.

Focusing on neurology, this article aims to inspire you to think further afield, giving you a detailed understanding of the neuron while illustrating aspects that have contributed to the author's programming tool kit and critical thinking. It is not intended as a full course in the field but aims to inspire further reading. With that and brevity in mind, much will be simple illustration at the potential cost of over-simplification.

2.2 Critical Thinking

AI often involves mimicking higher cognizant behavior. Faced with hypothetical situations most people correctly start with *introspection*, answering the question, "What would I do?"

As a programmer, this then becomes, "How do I code that?" followed by, "How do I code that efficiently?" But if we remove deadlines and programming from the equation for a minute and take some time to reflect on our thoughts, then maybe better questions are, "How would I do that?" and "Why?" Think a bit more and you realize that these questions are the primary drive of the psychological and biological sciences, a point that has not gone unnoticed before [Kirby 02].

If we invest the time to answer those questions we might be more likely to arrive at physically grounded behavior. But in this high pressure world, is it really that necessary? After all, we as AI programmers have survived quite well already.

And herein lies the most dangerous assumption of all, because with advances in locomotion fidelity, emotional content, and facial expressiveness, even the slightest nuance may immeasurably affect immersion. Without trying, we will never know, and with the Internet at our ready disposal, is there really an excuse for not doing some research, even if we just take the low hanging fruit?

2.3 Neurology

We are in the business of writing AI, the predominant output of which is *behavior*. Whether this output is represented as animation, speech, or thought, it is none the less analogous to the role of the brain and wider nervous system. From robotics to cognitive science, disciplines have taken their cues from their biological counterparts.

As such, considering its involvement in everything from the senses, through higher thought, to motor output, the study of neurology is one of our best foundations in the AI world. While our understanding of this huge and complex subject is very incomplete, it has one key redeeming feature. Almost the entire nervous system, brain, or nerves, is made out of one very fundamental cell—the *neuron*—and this single building block arms us with more information than you would at first expect. As AI programmers, it is wise at first to understand as much as possible about the cognitive mechanisms and building blocks of the creatures we are trying to represent. Hopefully, this section will serve as both a primer and provide keyword hooks to help further understanding. Commonly misrepresented as a purely electrical process, this section will introduce the biophysical side that has been gaining popularity in the field of computational neuroscience. It will take you through the full signaling properties and mechanisms of the neuron in sections, interspersed with the inspiration and history of some useful AI techniques before culminating in derivation of the perceptron as the basic building block of the neural network and why it is still useful by itself. Should you wish to delve into a little more detail on the neuroscience side, *The Computational Brain* [Churchland 99] and *From Neuron to Brain* [Nicholls 92] are both excellent resources.

2.3.1 Primer: The Electrical Processing Myth

Before we begin, it is important to clarify a dangerous preconception. We often assume the nervous system communicates by electrical signals. It unfortunately does not; we have merely adopted this metaphor as a handy way of generalizing the actual process. For now, we need to temporarily forget the visualization of electrical signals running down wires from the brain because it comes with preconceptions.

In fact, our notion of this electrical signal is really just a view of the voltage (*membrane potential*) at a single point on the neuron's *cell membrane*. Obtained by a pair of electrodes seated on either side of the membrane, it can give the impression of a spatial nature when really it is just a point voltage changing over time. The notion of voltage arises because the semipermeable cell membrane separates different concentrations of ions, and being thin, exhibits capacitance that eventually gives rise to membrane potential. Drawn in graph form then, over time, it would appear as a flat line situated around –70 mV or the *resting potential*.

2.3.2 Takeaway: Don't Neglect the Temporal Aspects

A different view of the same data immediately transforms our perception. What is traditionally considered an electric signal racing around the body, when clamped to one location, merely becomes a meter registering activity. But as we shall shortly see, drawn over time this once again provides the signal with a shape. Being reminded to think about problems in both time, as well as the current state, definitely has its merits, especially in utility theory [Mark et al. 10], and if you are ever overwhelmed by AI state—for example, AI steering signals given to vehicles—try graphing sections of it over time; the patterns and problems often become very much more obvious in the time domain.

2.3.3 Primer: The Neuron at Rest

For the neuron to remain balanced at rest, a number of conditions need to be met. Two of the most important are, first, that the net charge on either side of the cell membrane must be zero and, second, the concentrations of solute particles inside (*intracellular*) and outside (*extracellular*) must balance. A common misconception then is that if this were so, there would be no difference in charge (both sides zero) and the membrane potential would be 0 mV also. So where did that –70 mV come from? The key here is the difference in *individual* ion concentrations and the clue that the membrane is semipermeable.

Two primary forces act on charged ions. *Diffusion* seeks to push ions towards areas of lower concentration, and *electrical attraction* seeks to pull them towards areas of opposite charge. So looking at Figure 2.1, we can see that overall concentrations on either side are indeed balanced, and the overall charge on either side is zero. However, there are still vastly different concentrations of all relevant ions. At rest, the cell membrane is essentially closed to sodium ions (Na^+), and proteins are too big to go through. So while diffusion might want to alter the balance, proteins and Na^+ have nowhere to go. However, the cell membrane is permeable to chlorine ions (Cl^-) and very permeable to potassium ions (K^+). As such K^+ tries to flow out of the cell by diffusion. In doing so, its exit causes a charge imbalance that sees the now more negative inside attracting the positive ions. Cl^- similarly gets pulled back to the outside because traveling inside would make the outside even more positive. When the system eventually settles and overall ions are at rest, this more negative inside, with respect to the outside, gives us our resting potential.

Before we move on you might be wondering, what if K^+ and Cl^- were left long enough that ions on either side normalized, and concentrations became neutral across the membrane? Two things spoil the plan. First, the permeability to K^+ is very much higher than to Cl^-, so K^+ is far more effective in creating an electrical pull that affects both types of ions. As such K^+ is the dominant partner, and Cl^- can be thought as just reshuffling to match. Second, a number of *active transport* mechanisms, including *membrane pumps*, work as necessary to return ions to their relevant sides.

2.3.4 Takeaway: Diffusion

Diffusion is a nice technique, and if you have used influence maps you will no doubt work out why. Most of the general tweaks for making influence maps work [Champandard 11]

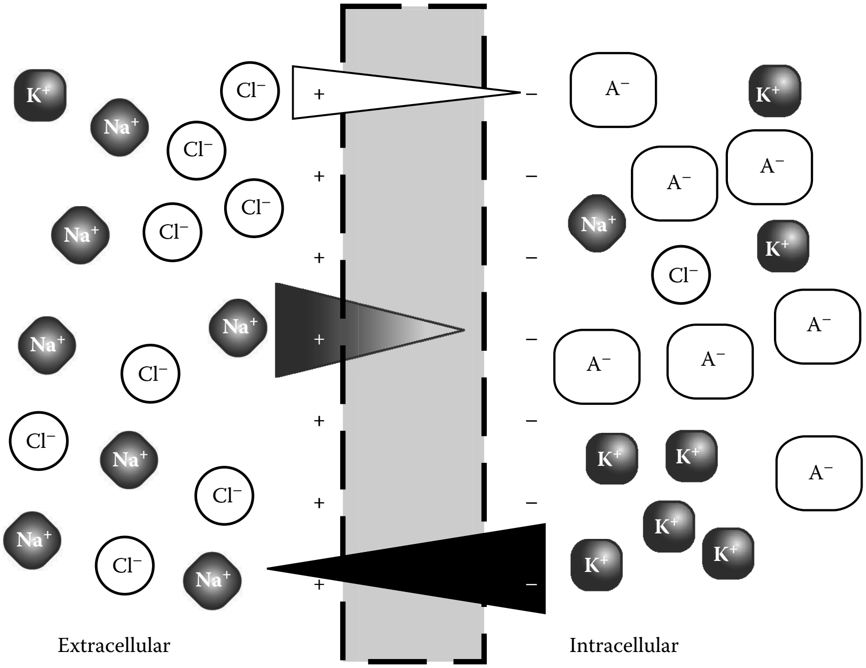

Figure 2.1

Diagrammatic representation of a neuron at rest. Four ions are represented, potassium (K^+), sodium (Na^+), chlorine (Cl^-), and an electrically charged anion (A^-) representing intracellular proteins. Notice here that total charge on either side of the cell is net zero and that the number of counted ions is balanced at 16 on each side.

can be, in actual fact, worked into the *diffusion equation*. In textbooks, the equation has a tendency to look formidable:

$$\frac{\partial c(r,t)}{\partial t} = \nabla.\left[D(c,r)\nabla c(r,t)\right]$$

But ignoring the math for a minute, it really just states that rate of change $\frac{\partial c}{\partial t}$ of concentration c at a distance r and time t depends on neighboring concentrations $\nabla c(r,t)$ and the ease of diffusion $D(c,r)$, as defined by the *diffusion coefficient D*. If the diffusion coefficient remains constant in space and time, then we obtain a simpler form, which is the heat equation:

$$\frac{\partial c(r,t)}{\partial t} = D\nabla^2 c(r,t)$$

∇^2 is known as the Laplacian operator, and it turns out that it can be computed by finite methods. If we create a grid of a variable α, for example ion concentration or influence, with constant grid cell width *r*, then the change in α after a time *dt* by diffusion is simply the sum of α for any connected neighbors minus any relatively scaled contribution before being averaged by grid cell area. If over time we then add on anything that might change this value of *α*, like a drip feed of ions or an increase of influence *k*, we obtain the following for a 2D grid of cell location *i*,*j* at time *t*:

$$\alpha\left(i,j,t+dt\right)=$$

$$\alpha\left(i,j,t\right)+\left(dt\times\left(k+\frac{D(\alpha\left(i_{-1},j,t\right)+\alpha\left(i_{+1},j,t\right)+\alpha\left(i,j_{-1},t\right)+\alpha\left(i,j_{+1},t\right)-4(\alpha\left(i,j,t\right))}{r^{2}}\right)\right)$$

Here, *k* could model anything from a decay rate to suppression by a different influence map. The beauty of this system is that it is both compensated for in time and space, which means it allows us to work with rate of change. Imagine its use in a real-time strategy domain, where production rates could be factored in, in place of production spikes!

But the best news is that this technique has already been adopted by the graphics community for many similar things due to its easy visualization in texture space. This also means it would be quite possible to get AI based simulations to work by just following similar GPU diffusion or gas examples—where the code is significantly easier to visualize than the math [Pharr 05].

2.3.5 Primer: Perturbation from Equilibrium

We have stated that the cell must always remain in equilibrium. But if it did so then the resting membrane potential would stay at ~–70 mV, and nothing would happen. As such, much processing of the neuron comes about by either electrical charge or concentration changes. Consider what happens if we increase the concentration of K^+ outside the cell. A decrease in K^+ concentration gradient now means K^+ has less inclination to leave the cell, and so less charge is required to pull it back. The relative membrane potential therefore becomes more positive, which also means less resistance to Cl^- moving in. Postperturbation, extracellular diffusion, and an increase in active transport work hard to return the neuron to equilibrium and standard resting concentrations. This results in a decay of membrane potential to resting values as illustrated in Figure 2.2. This common response-decay curve is a characteristic of most changes in local concentration.

2.3.6 Primer: The Spike or Action Potential

While a small change in membrane potential like the above is at least a signal, its effects are local and will dissipate quickly. To facilitate neuronal transmission we are going to need something of a very different magnitude.

Recall that we said that K^+ was permeable at rest and Na^+ was not. This is made possible because a number of K^+ specific *membrane channels* situated in the cell wall are in the *open state* allowing K^+ ions to be free to move if they want to. Similar Na^+ specific *membrane channels* in the cell membrane remain closed at the specified membrane potential. It turns out that there are many different *ion channels* found in neurons, sensitive to a wide range of factors, including membrane potential range.

Now consider the case where we temporarily push the membrane potential above the opening point of the Na^+ channels as seen in Figure 2.3. This opens Na^+ channels which,

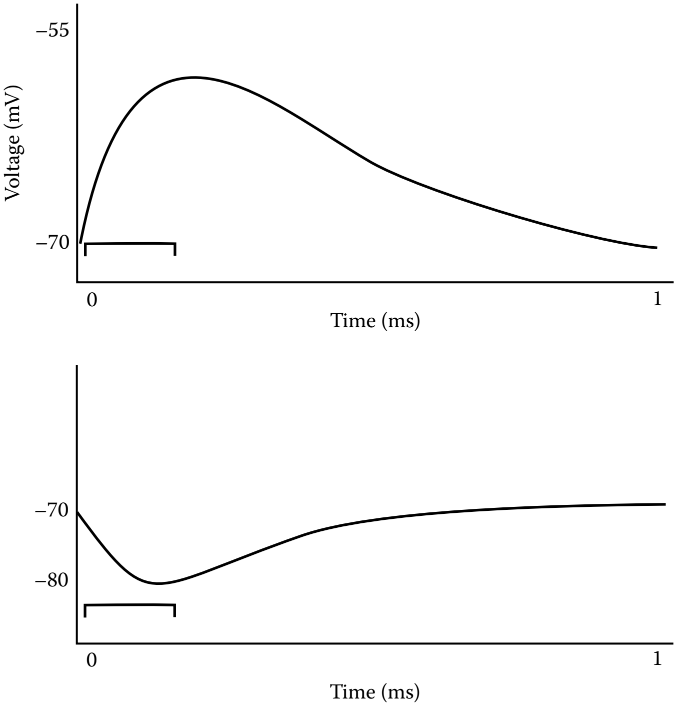

Figure 2.2

An illustrative representation of the membrane potential response curve due to a localized change in extracellular K^+ concentration (top). Notice that after an initial sharp change from the delivery, a slow decay to rest represents active processes returning concentrations to normal resting levels. (bottom) Increasing intracellular K^+ leads to a reverse effect.

combined with a massive concentration gradient into the cell and the accelerant of negative charge inside as shown in Figure 2.1, cause a massive influx of Na^+ ions. This pushes a huge swing in membrane potential towards the positive (*depolarization*). All things being equal, we might then expect the top response of Figure 2.2 albeit with much greater magnitude and duration. But as the voltage swings positive, high voltage potassium channels open and K^+ flows out of the cell also at great speed, driven by both a concentration gradient and repulsion from a more positive inside. This causes a rapid swing back the other way (*repolarization*). Because channels represent a rearrangement of lipids in the cell membrane, there is both a delay in closing and a period before which they will reopen again. In the case of the rapid efflux of K^+, this delay means the membrane potential dips below rest (*hyperpolarization*) before active processes, now working very hard in the form of a combined K^+/Na^+ pump, attempt to return the concentrations to rest. The resulting climb is called the *refractory period,* and at this stage it is performing as in the bottom case of Figure 2.2. Once channels are ready to reopen again, it will be possible for a second action potential to be generated. If the concentrations get the chance to return to rest, then this process is lossless and will result in a similar action potential if activated again. If ion concentrations haven't returned to rest, it is still possible to activate further action

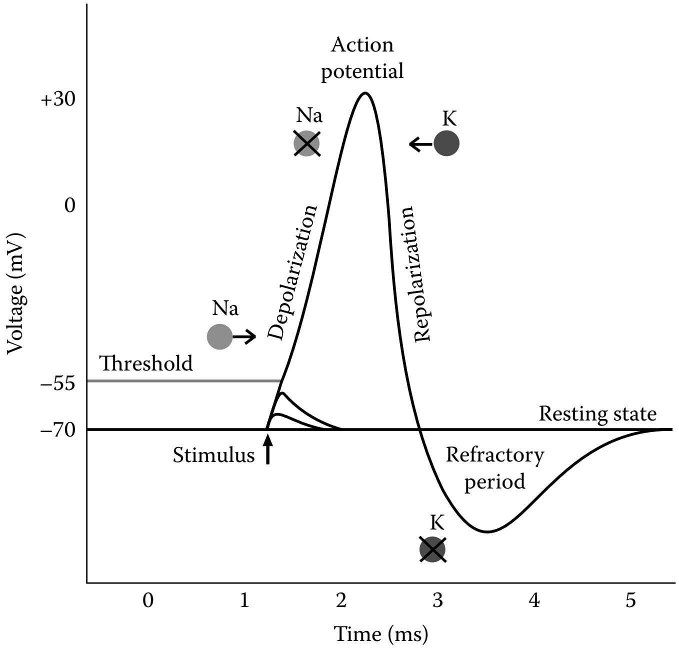

Figure 2.3

The action potential profile. Phases are marked alongside key ion changes that help produce the characteristic shape. Notice that failure to produce an action potential elicits a similar response to Figure 2.2.

potentials. However, the ion concentrations will become more and more imbalanced, with each action potential starting at a higher membrane potential, until the system will fail to respond, at which point it will have to decay to operating conditions again. This overexcitation results in a *burst* of spike activity, followed by periods where the neuron simply cannot fire.

2.3.7 Primer: Signal Transmission

So if the change in membrane potential is local to a specific location, how do signals travel? Recall that we said that the imbalance of ions causes a charge on the membrane, the same charge that gives us the membrane potential. Just like lightning, if an area of higher charge is different from its surroundings, it spreads outwards. Likewise if a change in ion concentration occurs at a particular site, then inside or outside the cell, diffusion will try to normalize the overall concentration on either side of the membrane.

If these changes are enough to push the neighboring membrane potential above the threshold for Na^+ channels to open, a similar action potential will be generated on a neighboring site on the cell membrane, causing a ripple effect as action potential generation spreads outwards. It is this "The Wave" style of propagation that gives us our notion of the electrical signals traveling around the nervous system. Because activated channels need time to recover and the refractory period helps keep the local membrane potential negative, neighboring sites cannot retrigger an action potential again in the same region

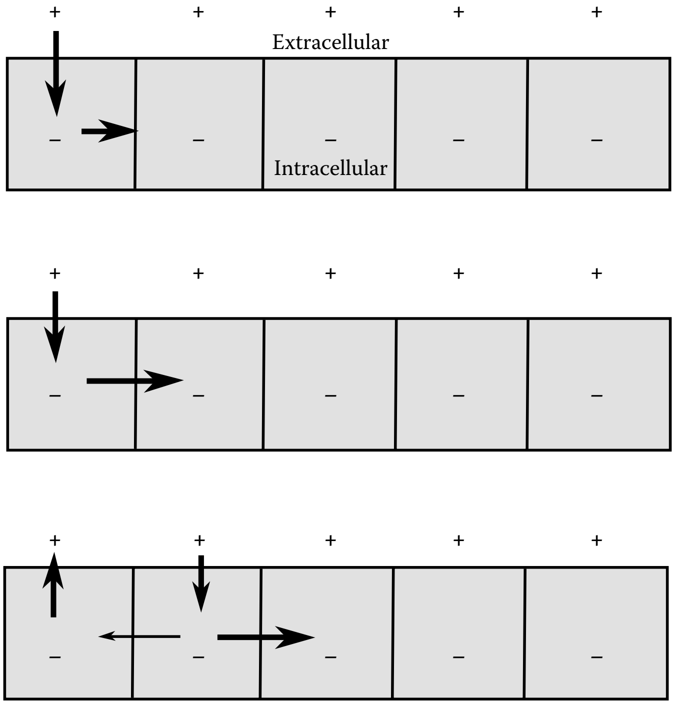

Figure 2.4

An illustration of action potential propagation through nerve fibers over time. The resulting swing in repolarization and the delay in membrane channels reopening prevent previously excited sites from firing again.

so signal transmission carries lossless in an outwards direction as illustrated in Figure 2.4. This is true of both single signal and bursting neurons.

2.3.8 Takeaway: The Schmitt Trigger and Hysteresis

At a similar time that action potential and nerve fiber transmission was being discovered, Otto Schmitt was taking inspiration from its workings. In his 1937 dissertation he introduced the thermionic trigger later to be known as the Schmitt trigger [Schmitt 38].

While it has many other uses in electronics, the Schmitt trigger has the interesting property that the switch between two states, say on or off, is based on an overlap of different activation conditions between them, in essence covering the dead zone from one to the other. Hence, in systems that might vary around a key trigger value, this system ensures that a period of change must occur before the state will switch. We call this dual threshold approach *hysteresis*. It is often a key to keeping clean control in AI systems, especially where switches in animations are concerned as best illustrated by Listing 2.1.

A key point to remember about Schmitt triggers is that they rely on a physically changing property, such as voltage or concentration, incorporated in the system. The most common mistake is to believe you can add hysteresis with a timer. It is not hysteresis and it is not protecting you; it is just changing the frequency of oscillation. Proper hysteresis might not

Listing 2.1. An example of hysteresis. If we were just to base our approach distance around a single value (5m), then every slight move outside that radius would cause us to follow. With hysteresis, once we start to follow we will continue until we are inside 2.5 m, and won't start again until we are further away than 5 m. This noise reduction means the player has some maneuvering room before we decide we need to be closer.

```
const float SEPARATION_DISTANCE = 2.5f;
const float APPROACH_DISTANCE = 5.0f;
if (approach_player = = false)
{
    if (distance > APPROACH_DISTANCE)
    {
        approach_player = true;
    }
}
else
{
    if (distance < SEPARATION_DISTANCE)
    {
        approach_player = false;
    }
}
```

be as clean as a single variable either; it might be a complicated series of conditionals that separate logic space. Yet the principle will still hold, provided there is the relevant overlap.

2.3.9 Takeaway: Compartmentalize to Solve Hard Spatial Problems

Notice how a potentially hard problem, the transmission of action potentials, can be better conceptualized by compartmentalizing the space (Figure 2.4). At this stage, each compartment represents just a local area, and the comprehension is immediately simplified. Now consider a tactical warfare simulation with a realistic AI communication system. Rather than working out a route for each agent "A" to get a message to another agent "B," it is easier to imagine all such agents on a compartmentalized grid, posting messages to neighboring agents who then carry the transmission that way. In doing so, we get some nice extras for free: the potential for spies, utterance signaling, realistic transmission delays, and bigger signaling distances covered by nonverbal gestures or field phones. In either case, by thinking about the problem as a different representation, we have instantly simplified the procedure and got some realistic wins as well!

2.3.10 Primer: Morphology

If neurons were just a wire, comprising the same action potential generating membrane channels, then life would be a lot easier. Unfortunately, most neurons have a complex shape or *morphology*, and very different ion channels along the membrane. This means that the neuron itself is responsible for a lot of implicit processing.

The morphology of a typical neuron consists of four parts: the *dendrites*, the *soma*, the *axon,* and the *axon terminals* (as illustrated in Figure 2.5).

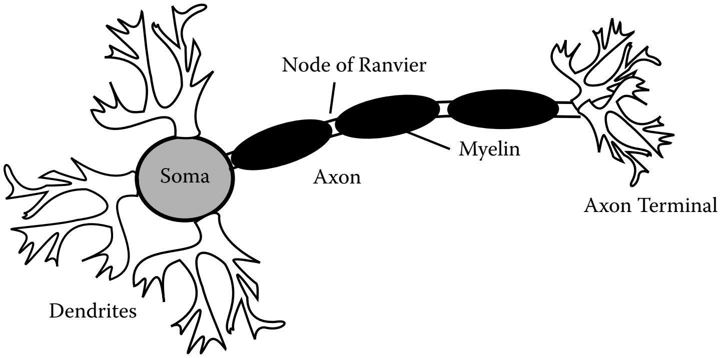

Figure 2.5

A diagrammatic representation of the key components forming the morphology of the neuron.

For the time being we can think of signals in the neuron starting as a change in membrane potential at dendritic sites that generates one of two different types of responses similar in nature to Figure 2.2. With a distinct absence of Na^+ gated channels, the voltage profile of the first response is very similar, albeit reaching higher magnitudes and therefore a longer decay. The second response is inhibitory and inverted, with its changes actively making the membrane potential more negative, but otherwise following a similar path. The near lack of action potential generation in the dendrites means transmission is generally *passive*, spreading merely by change of charge in neighboring regions. While a single signal might die out, the frequency at which dendrites are activated means that there is generally constant stimulation, be that excitation or inhibition that sums in both distance and time. If this summation of charge reaches the soma and creates a high enough potential difference, then traditional spike generation channels at the axon head carry a new signal down the *axon* to the *axon terminals*. While the distance between dendrites and soma is usually small, axons are generally much longer. To facilitate quicker conduction velocity, the axon is usually insulated by a *myelin sheath* that focus charge build-up from strong action potentials at the nodes of Ranvier. This charge is strong enough to cause excitation in a neighboring node faster than it could be carried there normally.

2.3.11 Primer: Synaptic Transmission and Plasticity

The last section mentioned that when dendrites are excited or inhibited there is a strong change in membrane potential, the general cause of which is attributed to *synaptic transmission*. As illustrated in Figure 2.6, when a conducted signal reaches the axon terminals, a change in membrane potential triggers *vesicles* (think seed pods) in the *presynaptic* cell membrane that shoot a packet of complex chemicals (*neurotransmitters*) into the medium between cells. Pointedly projected at other neurons, neurotransmitters travel a short distance through this medium to be picked up by receptors on the dendrites of neighboring *postsynaptic* neurons. This in turn triggers the opening of a number of ion channels, resulting in either an excitatory (positive) or inhibitory (negative) change in membrane potential. These changes are called EPSP (excitatory postsynaptic membrane

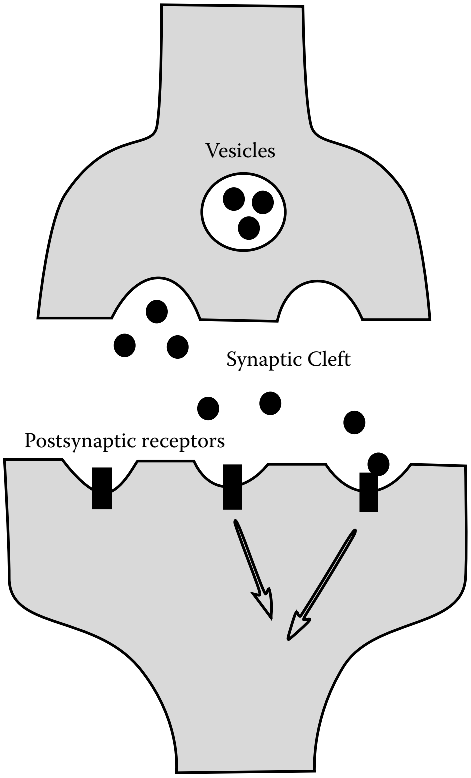

Figure 2.6

A diagrammatic representation of the synapse. In response to stimuli, vesicles in the presynaptic neuron expel their contents into the synaptic cleft. These cross the cleft to the postsynaptic neuron where receptors trigger changes in postsynaptic potential.

potential) and IPSP (inhibitory postsynaptic membrane potential), respectively, and can be thought of as high magnitude variants of Figure 2.2.

2.3.12 Primer: Plasticity and Learning

We know that the brain can learn and adapt. Dendritic excitation is generally a short term byproduct of postsynaptic potentials and action potential generation is for the most part fire-and-forget; so where is the mechanism for learning? Again the purely electrical viewpoint doesn't suffice. Only by understanding the chemical processes are we able to hypothesize on the mechanisms that might do so.

Recall that in synaptic transmission, synapses expel neurotransmitters across to other neurons. If we were to change the sensitivity of these postsynaptic neurons or the amount of neurotransmitters expelled, then obviously the resulting effect will be an increase or reduction in the postsynaptic potential. Both mechanisms could therefore be used by pre- and postsynaptic neurons to control the strength of that connection. We call this *synaptic plasticity*, and it is here that the principles of learning and adaptation are believed to lie.

2.3.13 Primer: The Hebb Rule and Rosenblatt's Perceptron

In 1949, Donald Hebb, in an effort to understand learning, made the assumption that if the axon terminals of one neuron regularly excited the dendrites of another, such that they regularly took part in firing it, then some growth or metabolic process would increase both their efficiencies in doing so [Hebb 49]. In other words, for each synapse *i* of a total number *j* contacting the neuron, the change in their strength w_i would be a proportion of how much their contributory input *x* correlated with the output *y*. Here Hebb defined output as the summation of all the input to synapses x_i times their individual strength w_i. In other words, he attempted to mimic primitive synaptic sensitivity and interaction in the dendritic tree. In line with real neurons, he hypothesized that this change would happen slowly over time (using a learning rate μ). Put into mathematics we say:

$$\omega_i = \mu x_i y$$

where

$$y = \sum_{i=0}^{j} \omega_i x_i$$

In 1957, Frank Rosenblatt took Hebb's rule of learning and applied it in algorithmic form as the perceptron algorithm, making a number of improvements at the same time [Rosenblatt 57]. First, the synaptic strength in Hebb's rule can quickly grow unstable if all inputs continue to contribute. So Rosenblatt favored learning only if the result was unexpected. He also constrained output, mimicking the firing of the axon only if a summation threshold is reached and applied a bias term to keep things moving if all other inputs were 0.

Hence the Rosenblatt perceptron can be specified as follows:

For a vector of potential inputs $v_i = \left[-1, i_1, \ldots, i_j\right]$ and a vector of weights $v_w = \left[\omega_0, \omega_{1,\ldots,}\omega_j\right]$ where ω_0 represents the bias weight, but always for a constant input ($i_0 = -1$):

$$y = v_i . v_w > 0 \textit{ then } 1 \textit{ else } 0$$

Rosenblatt's perceptron has two phases, a learning pass and a prediction pass. In the learning pass, a definition of v_i is passed in as well as a desired output, y_{ideal}, normalized to a range of 0 … 1. On each presentation of v_i and y_{ideal} a learning rule is applied in a similar form to Hebb's.

Here, each weight is affected by the difference in overall output and expected output y_{ideal} multiplied up by contributory input and learning rate.

$$\omega_{i,t+1} = \omega_i + \mu\left(y_{ideal} - y\right) i_i$$

So by presenting a series of these vectors and desired output, it is possible to train this artificial neuron to start trying to give new output in the absence of untrained conditions by just applying the output calculation without training.

2.3.14 Takeaway: The Hebb Rule and Rosenblatt's Perceptron

There are a number of key points here. First, unlike neural networks that commonly consist of many of these units, at this simple level the meaning of the parameters is understandable. In AI terms, this gives us both the glamorous, a potential prediction of player behavior, and the less so, in the tuning of parameter unknowns.

The perceptron is essentially a Boolean classifier. For n changeable variables in an input vector, it tries to train the weights to correctly classify an nD point as being on one side (1) or another (0) of an imaginary separator through that space. Hence, if $n = 1$, the separator is a value that splits i_1 . At $n = 2$, it represents a line that divides $\left[i_1, i_2\right]$. At $n = 3$, it is a plane dividing $\left[i_1, i_2, i_3\right]$ and so on.

Provided the inputs can be logically separated—the XOR function is an example that can't be linearly ($n = 2$) separated—then with enough training, weights should settle and $\left(y_{ideal} - y\right) \to 0$. If this does not eventually happen, then this tells us that either the problem is not Boolean classifiable, or we have not used the correct input. However, with some common-sense guesses on dependent variables, results are generally obtainable.

Moreover, its synaptic strength w_i is potentially understandable. Removing bias, which just seeks to modify the threshold as $w_i \to 0$, then we know that particular input we are passing in doesn't factor much on our decision. Just like at synapses, if it is less than 0 it inhibits the 1 result, and if greater than 0, it adds towards it. Now combine this with a notion of utility [Mark 09] essentially heuristic in form, and you can test what factors are critical in influencing a decision and whether you have got them right.

The key here is the training on input. For example, here is a utility equation from a recent talk [Mark 10]:

$$cover\ chance = 0.2 + reload\ need \times 1.0 + heal\ need \times 1.5 + threat\ rating \times 1.3$$

Now, these values probably took some degree of trial and error to arrive at and probably had to be normalized into sensible ranges. But consider this:

$$cover\ chance = -1 \times w_0 + reload\ need \times w_1 + heal\ need \times w_2 + threat\ rating \times w_3$$

If we present the player with a number of scenarios and ask them whether they would take cover, we can easily get values that, after enough questions, are still easily accessible in meaning and therefore can make better initial guesses at the equation that might want to drive our AI.

A final nice property is that we don't necessarily need to answer the false side of the equation if we don't want to; we could just supply random values to represent false. Imagine each time the player goes into cover we measure these values and return true, following it by random values that return false. Even if we happen to get a lucky random true cover condition, over time it will just represent noise. If we then clamp all weights to sensible ranges, potentially found by training in test circumstances, we now have a quick to compute run-time predictor of what determines when the player may think of going into cover!

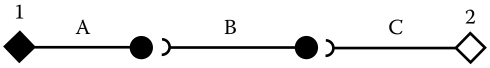

Figure 2.7

Diagrammatic view of the reflex arc. Sensory input (1) gets carried by neuron (A) to an intermediate neuron (B) and onto the muscle (2) through neuron (C).

Table 2.1 The time for various stimuli to reach the brain

Stimulus	Time to brain (ms)
Auditory	8–20 ms
Visual	20–40 ms
Touch	155 ms

2.3.15 Primer: Pathways

Our final foray into the world of the nervous system involves what happens when neurons combine at synapses and networks are formed. When we talk about these series of identifiable connections we talk about *pathways*.

The shortest and least complicated pathway, the *reflex arc* in Figure 2.7, connects a sensory neuron through a synapse to an intermediate neuron, through another synapse to a motor neuron. Other pathways, on the other hand, pass through many more neurons. Considering what we know about the signal transmission process, it seems reasonable, then, that transmission times should scale based on neuron count and travel distance.

Recall that we said one of the key dangers with neurology was thinking of signaling as electrical impulses traveling through wires. Electrical current approaches the speed of light at 3×10^8 m/s. The fastest myelinated axons can only achieve speeds of 120 m/s. To put this in perspective, Table 2.1 shows the time taken for different sensory signals to reach the brain. This is before considering any further processing or motor output. It is hardly surprising then that the senses we rely on the most are the ones located closest to our brains.

Once you start bringing in motor control, however, even for simple tasks, the times just continue to rise. Ask a friend to hold a ruler, with your thumb and finger ready to grasp it at the bottom. Let them drop the ruler unannounced and as soon as you see it move, close your fingers to grasp the ruler.

Now because $s = ut + \frac{1}{2}at^2$, where a is gravity and u is 0, reaction time can be worked out from the equation

$$t = \sqrt{\frac{2 * s}{9.8}}$$

where s is the distance traveled on the ruler where you grasped it (in meters). Do the math and the value should come out around 0.2 seconds.

The point here is, even at 30 frames per second, your reaction time to a reasonably simple coordination task is actually already 6 frames.

Compare this with receiving a burn. Your natural reaction is to pretty instantaneously pull away. Yet by our calculations, assuming touch speed, if the signal was traveling from hand to brain to hand again, this travel time would be in the range of 300 ms. So here, the answer almost certainly lies in a reflex arc transferring pain signals, through a really small pathway, to a motor neuron and therefore producing an uncontrolled involuntary reflex.

In a lot of precise or fast-acting systems, any uncompensated latency is going to have a serious detrimental effect on accuracy. Imagine the hard task of trying to grasp a moving feather in a drafty room. Here, our visual cues need to be processed into some spatial representation and transformed into motor commands to coordinate a large number of muscles that allow us to finally grasp it. Yet, remarkably, we can do this with some ease.

Trying to reduce this error in a traditional closed loop way, with the latency we have discussed, would mean we would consistently fail to react to the near-random changes in direction. Instead, the nervous system uses a different technique to cope. We call this *open-loop ballistics*.

As a simple example, saccades are the fast movement of the eyes that swing focus to different points in the visual field. Given that visual feedback to the brain takes between 20–40 ms to return and measured eye travel velocities can reach 900 deg/sec, any attempt at error correction by a closed loop is going to lead to overshoots of ~3 degrees and therefore be untenable. It turns out that in order to cope, saccades make use of a ballistic open loop. Put simply, the journey is preplanned prior to onset. Experiments show that eye muscles respond to bursts of spikes, the frequency of which roughly maps linearly to eye velocity. We also know that to move the eye, a series of spikes come in that force the eye to travel very fast to a new position, before the frequency scales down, so that it applies only enough velocity to counteract the elasticity of the muscles wanting to return.

Something then is controlling the output of these spikes. If the brain was only predicting a velocity with no feedback on error, then chances are the elastic forces trying to return muscles would mean we could not hold the position.

So it turns out that a number of things happen. First, there is a clear proof that we do not need to rely on visual stimulus to perform this operation. Just think about a point in space to the top left of your view, and you will find you can intentionally direct your gaze there by saccade. This ability to make voluntary saccades without the usual sensory input means we must maintain a mental image of the calculations involved. We call this mental image the *efference copy*. It means we can make an initial guess at velocity and start eye movement as soon as possible using the ballistic open loop. When we do receive visual information about our progress, it arrives late. However, because we know about our journey or where we should be at any time, the discrepancy between the two can be applied as a change in *gain*, strengthening or weakening a few synapses based on previous error. With a mostly linear velocity response, this change can scale the remaining journey sensibly helping tune mid-flight. With a 100 ms delay before onset and a total travel time of about 200 ms, this journey preplanning makes sense with respect to the timings.

It turns out that you get the same notion of a separate mental representation when you try to grasp something. Again, an internal representation is definitely at play. If you close your eyes and then go to pick something up, even blind and moving your head, you will still have reasonable success. You are once more running off a mental representation, open loop feedback only coming late from muscle sensors (*proprioceptors*) providing a notion of yourself in space.

2.3.16 Takeaway: Constrain to the Nervous System

We know that with the exception of reflexes, most response is comparatively slow. We should always bear in mind that some of the quickest processing at 200 ms or so is still 6 frames at 30 frames per second. This means that distributing AI planning over frames is not only viable, it may be more biologically plausible.

We know that neural latency almost always prohibits direct feedback and indeed various systems have worked around this with open loop feedback. This tells us a few things.

First, the flow of information at a single frame level is one-way, because there is no natural system for returning information. Hence, sensory input passes to the brain and then onto motor control. This leads to the notion of one-way information flow in the frame, through sense, think, and act stages [Laming 09].

By much the same reasoning, we should probably get out of the habit of requesting more information, such as a spatial query, midway through AI processing. If we need that info, it's probably better precached with other sensory info for all to use. Considering our brain is comparatively sluggish anyway, we can easily spread this vast information gathering over separate frames and threads.

We also know that in the majority of control cases, such as animation, movement by the nervous system involves initial planning, setting off, and then refining mid process. Consider a long-jumper trying to hit a launch board. They may do their best to plan for hitting the board upfront, but they are constantly reacting to change and error on the way up, which is why it is never an exact launch position. Preplanning exact paths by animations then, while it might give excellent results, may not be the most realistic. Some planning is clearly good, but continuous adaptation by techniques such as steering and velocity avoidance has its part to play [Laming 09].

Finally, we know that, although some information is stored in memory, signals to muscles (essentially our animation cues) are just the output of neurons being fired each frame. Hence, it should be possible to model these signals in AI without necessarily storing them. It is this that lies behind the reasoning for stack-based control signals [Laming 09].

There are obviously plenty more gems out there hidden inside the recommended reading. For example, in the high definition, facially close-up world of our AI character's eye gaze, pupillary response will not only sell emotion [Mark 12] but will provide a more subtle alternative to the traditional notion of head tracking. With the addition of accurate latency, this should start to give anticipation for free!

2.4 Conclusion

Ultimately this chapter is a *primer* in neurology. Its purpose is to introduce a world of neuroscience, neurons, and neural nets to those that may never have learned about them in detail before. By looking at the beauty encapsulated by the neuron, nervous system, and its signal processing potential by shape, make-up, and connections, it hopefully inspires you to think about the biophysical and psychological nature of other AI topics you may encounter.

In doing so it occasionally interrupted with a *takeaway* section, providing the history, background, and concepts behind some gems the author has used and written about before, illustrating the relevant reasoning and critical thinking that drove those decisions.

In this article we discussed potential uses for diffusion and the importance and proper use of hysteresis. We also explored the *perceptron*, the base unit of neural networks, what it actually represents, and how we might make sensible use of it without going overboard. Finally, we looked at some AI architecture design considerations which can be extrapolated from the nervous system as a whole.

You should now have a simple overview that allows you to make sense of the various references and surrounding literature and realize the benefit of exploring outside the computer science box.

References

[Champandard 11] Champandard, A. J. "The Mechanics of Influence Mapping: Representation, Algorithm & Parameters." http://aigamedev.com/open/tutorial/influence-map-mechanics/

[Churchland et al. 99] Churchland, P. S. and Sejnowski, T. J. *The Computational Brain*. Cambridge, MA: MIT Press, 1999.

[Hebb 49] Hebb, D. O., *The Organization of Behavior*. New York: John Wiley & Sons, 1949.

[Kirby 02] Kirby, N. "Solving the right problem." In *A.I. Game Programming Wisdom*, edited by Steve Rabin, pp. 21–28.Hingham, MA: Charles River Media, 2002.

[Laming 09] Laming, B., "AI Architecture and Design Patterns" Session, AI Summit, Game Developers Conference, 2009. Slides available online http://www.gdcvault.com/play/1460/(307)-from-the-ground-up.pdf.

[Mark 09] Mark, D. *Behavioral Mathematics for Game AI*. Course Technology PTR, 2009.

[Mark et al. 10] Mark, D. and Dill, K. "Improved AI Decision Modeling through Utility Theory" Session, AI Summit, Game Developers Conference, 2010.

[Mark et al. 12] Mark, D. and Schwab, B. "Less A More I: Using Psychology in Game AI." Session, AI Summit, Game Developers Conference, 2012.

[Nicholls et al. 92] Nicholls, J. G., Martin, A. R., and Wallace, B. G. *From Neuron to Brain*. Sunderland, MA: Sinauer Associates, 1992.

[Pharr et al. 05] Pharr, M. and Fernando, R. *Programming Techniques for High Performance Graphics and General Purpose Computation.* Chapter 31. Reading, MA: Addison-Wesley, 2005. Available online (http://http.developer.nvidia.com/GPUGems2/gpugems2_chapter31.html).

[Rosenblatt 57] Rosenblatt, F. "The Perceptron—A Perceiving and Recognizing Automaton." Report 85-460-1, Cornell Aeronautical Laboratory, 1957.

[Schmitt 38] Scmitt, O. H. "A thermionic trigger." *J. Sci. Instrum.* 15, 24, 1938.