Free Running Subsumptive Regular System (SRS22)

Richard Keene Jan 1, 2022

A Free Running Subsumptive Regular System (SRS) is described, and several conjectures are presented. The goal is to establish an approach to Artificial General Intelligence (AGI) that is self-directed, always active, has aspects of cognition, and can be implemented on current systems. Cognitive behavior is extrapolated. The system is theorized to exhibit behaviors of self-clocking, neural avalanche, inner vs. outer attention, reacting to the environment based on experience and more. This paper is strongly oriented toward a practical implementation. This paper is a refinement and reworking of Artificial Cognition, from the conference Intelligence in Neural and Biological Systems, 1995. INBS'95, Proceedings (Keene, 1995) (<https://dl.acm.org/doi/10.5555/850955.854064>).

SRS Subjective Law: Success is defined as creating an AI that is interesting to relate to and convinces people that it is thinking.

Table of Contents

[1 Overall Architecture 4](#_Toc149216461)

[2 Definition of Terms and Information Flow 5](#_Toc149216462)

[2.1 Terms 5](#_Toc149216463)

[2.1.1 Subsumption 6](#_Toc149216464)

[2.1.2 Cortex Regular Pattern Matcher System 7](#_Toc149216465)

[3 Loops in the System 9](#_Toc149216466)

[3.1.1 Lockup and Unlock 9](#_Toc149216467)

[3.2 Free Running Emergent Behaviors 10](#_Toc149216468)

[3.3 The Importance of Random Noise 10](#_Toc149216469)

[4 Learning 11](#_Toc149216470)

[4.1 When to Learn 11](#_Toc149216471)

[5 Concept Maps 12](#_Toc149216472)

[5.1 Attention: Internal and External 12](#_Toc149216473)

[6 Multi-SRS systems 13](#_Toc149216474)

[6.1 System Creation 13](#_Toc149216475)

[7 Global Pattern Connectivity 14](#_Toc149216476)

[7.1 Connectivity Triples ⅄ 14](#_Toc149216477)

[7.2 Meta-Maps and Anonymous M 14](#_Toc149216478)

[8 The Reaper and Connection Routing 15](#_Toc149216479)

[9 Implementation Notes and Conjectures 16](#_Toc149216480)

[9.1 Synchronization 16](#_Toc149216481)

[9.2 Neuron Pulse vs. Value 16](#_Toc149216482)

[9.3 Redundancy 16](#_Toc149216483)

[9.3.1 Computer Memory Locality 17](#_Toc149216484)

[9.3.2 Multiple Muti-SRS systems 17](#_Toc149216485)

[9.3.3 Biological implementation of G 17](#_Toc149216486)

[9.3.4 Derivation of X, Y, Z coordinates 17](#_Toc149216487)

[9.3.5 Connectivity Triples 17](#_Toc149216488)

[9.3.6 Distance from the Spinal Cord 17](#_Toc149216489)

[9.4 Avalanche and Generative Adversarial Networks and/or Autoencoders 17](#_Toc149216490)

[10 References 18](#_Toc149216491)

# Overall Architecture

The SRS22 system has two main sections.

First is the Subsumptive System where the external environment is received, modeled, and reacted to. The Subsumptive System is hard-wired as stimulus-reflex Maps, also called Concept Maps.

The second section in the architecture is a Cortex that sits above the Subsumptive System and has the task of predicting the future state of the environment and reacting in advance. This modifies the state of the Subsumptive System and is where learning takes place.

Inputs from Real World

Outputs to Real World

Cortex Layer

Cortex Layer

Cortex Layer

Cortex Layer

**G**  
Global Goodness

Figure - Overall System Layers

# Definition of Terms and Information Flow

**F** (Transform Function)

**I** (Inputs)

Change Rate

**O** (Outputs)

World and Hardware  
and/or  
Other Subsumptive Modules

**M**  
(Concept Map)

**OF** (Output Transform)

Subsumptive System, Single SRSUnit

Figure - One Subsumptive Module, also called SRSUnit.

Cortex Unit, Single SRSCortexUnit

**S**  
Input from Subsumptive System and from the Cortex.

One or More   
Neurons

**C**   
Charge

[**T**]  
Targets with **⅄**distribution

Stimulus out to Subsumptive System and Cortex Units.

**Q**  
Match Quality

Learning  
Function

Reaper

**S’**

**L**  
LinkWorld and Hardware  
and/or   
Other Subsumptive modules

**W**  
Weights

**So**Stimulus from other Cortex Neurons

**N** Neuron Model

Figure 3 – A single Cortex Unit.

## Terms

The sensory information **I** is fed into a **Subsumptive Unit**. The preprogrammed function **F** transforms the input to some useful representation or ‘concept’. The output of **F**,and the average rate of change for the sensory information , are fed into **M**, the **Concept Map**. The concept map is the input to an (optional) output transform function **OF** and operates some hardware out to the world. This is the hardwired subsumptive system. By hardwired we mean to say there is no learning, rewiring or change to **F** or **OF** over time. A Subsumptive Unit may also be the input to other subsumptive units.

The **Cortex Unit** or Regular Predictor is “regular” (as in “orderly”) in that it is an array of neurons with simple perceptron style learning. The neurons exhibit learning and change over time. The cortex neurons are continuously trying to learn what state the inputs are in for a given stat of a single element in a target Concept Map, or alternately, the state of a single other cortex neuron. Thus

Eq 1. For Neuron N : C = fatigue{C + Ʃ Inputs \* W + Ʃ So \* Q – decay}

The Neuron **N** has a associated with matching and learning a pattern. While pattern matches are immediate, the delay between seeing a pattern and using **G** to decide to learn the pattern is significant in system operation.

The goal of computing **S’** at time **T**, predict the future state of **M** or other **N** at (and thus indirectly the future state of the external environment) and react earlier (by and in advance of the actual environmental state to gain an evolutionary and survival advantage. It is hypothesized that reacting to the future environmental state is also the goal of all cognitive systems that have any ability to learn.

The Global Goodness Function **G** evaluates the environment for how well a goal of survival or other goal is currently being met. **G** is derived from, and establishes, the reason the AGI exists. It is likely that a more nuanced set of Goodness functions will be developed.

An **SRS Module** or **Concept Map** is one instance of Fig. 2. A full system will have many modules and a large variety of **F** and **M**. The interconnection of modules forms the Subsumptive system. The core design of the full system focuses on **F** and interconnectivity of Concept Maps.

The term Concept is used here because the map **M** represents some abstraction or concept of the environment or further abstraction of up-stream Concept Maps. As a concrete example a low-level concept map (or just “map”) might take raw video frames as input, and extract edges. The next downstream map might take the edges and extract edge orientation around the center of the image frame thus representing the concept of “radial angle” at the center of the attention spot in the view.

### Subsumption

One concept mentioned herein is **subsumption**. Subsumption (Thompson, 2009) (Brooks, 1985) (Wikipedia, Subsumption Architecture, 2021) is to place something in a larger context such that the lesser context is encompassed. In our use herein we mean to take simple functions that are stimulus-reflex and provide simple behavior. We then override or subsume them when some situation arises where more complex and nuanced behavior is needed. The lower-level behavior is temporarily dormant or hidden/blended while the higher-level behavior is manifest.

#### Subsumption and the Need for Forward Additive Neurons

In a directed analog network (consisting of Op-Amps and resistors) one can take inputs and combine them and generate outputs. The classic example of this is “Photovores” where a small robot seeks light sources. One can use subsumption and generate rather complex analog networks to arrive at complex behaviors with very biological looking movement. The problem with evolving such a system is that one change anywhere in the analog network upsets the entire network and most all the network’s resistors need to be recalculated. If a random new path is added (mutation) without recalculating the entire network, the chance of having a functional system is very near zero.

With neurons that are pulsed, if the new mutation’s neurons are not active the neurons in effect do not exist. One can refer to this as **neural hiding**. (Not related to “hidden layers” in neural networks.) A random mutation to the pulsed neural network has a larger probability of not being completely destructive to the system operation and survival and might even be a survival benefit. In a computer system we add values forward into the next SRS representing pulse rates. See section 8.2 for more on pulse vs. value.

Using subsumptive design and implementation strategies results in a system that can be designed and modified much more easily. One can incrementally refine the overall system. One can even snapshot the entire state of the current system and learning, and then modify the system and add or remove SRS and reload the system and continue. This is a trick that evolution cannot do and instead must produce a new generation to test mutations.

In from  
World and Hardware

Out to  
World and Hardware

Subsumptive System

Concept Map

Concept Map

New Concept Map Subsuming  
When Active

Concept Map

Concept Map

Figure 4 – A Subsumptive system

### Cortex Regular Pattern Matcher System

The cortex is observing the state of the entire system and tries to predict the future state of the subsumptive system and stimulate the subsumptive system in the future state in advance of the environment arriving at that state.

Each neuron in the cortex has many inputs that are the state at semi-random points in the entire subsumptive system and cortex. Each neuron has one output which is a semi-random targets in the subsumptive system and cortex that it is trying to predict and then stimulate. The neuron weights are evolved over time from observing the inputs for how good of a match there is, and then using the charge of the target neuron as the correct value.

If there is a strong match the output targets are stimulated according to the strength of match and the weights are adjusted.

In from  
World and Hardware

Out to  
World and Hardware

Overall System – A Subsumptive Regular System

Concept Map

Concept Map

Concept Map

Concept Map

Concept Map

Cortex

Subsumptive System

One Cortex Unit

S

S’

Figure5 – A Subsumptive Regular System

# Loops in the System

There are several feedback loops in the system.

In from  
World and Hardware

Out to  
World and Hardware

Feedback Loops

Concept Map

Concept Map

Concept Map

Concept Map

Concept Map

Cortex

Subsumptive System

N  
One Cortex Neuron

S

S’

Environment

*Change Rate*

Q  
Match Quality

Figure 5 - Feedback Loops

The first and most obvious loop is the World and System interaction shown in light gray. This type of interaction had been discussed in detail by other authors. See (Chan, 2001) for just one example.

The much more interesting feedback in the system is between the Cortex Unit(s) **N** and the Concept Map(s). With high and low **Q** the system simply acts as a subsumptive network with no learning. This is simple stimulus, transformation, and reaction.

In the case of a static (boring) environment is low so in effect **F** is turned off. The cycle of **N** matching, stimulating **M** to that pattern (possibly subdued by **Q**) and then repeating the cycle will cause a **free running** cycle through related patterns. The loop also causes a virtuous cycle of match, weakly stimulate M, and repeat. This will cause a vague pattern match to **avalanche** into a definite pattern. Somewhat related reference is (Yi Li, 2018).

### Lockup and Unlock

The **NM** loop has the potential to “lockup” in a single state or a repeating chain of states. There are several mechanisms to prevent this. First is that cortex neurons become **fatigued** and have greatly reduced stimulus strength, that decays over time. The decay interval is directly related to the of **N**. The actual match strength is a fatigue factor, that has a decay factor toward 1 when selected (used), and a decay factor toward 0 when not selected (idle). Also stated as “the neuron becomes tired”.

Another mode of breaking out of pattern lockup is when the environment changes and **F** gets precedence over **N**. Interestingly, this may not happen in a single tick of time, but can also be a gradual avalanche, and both **N** and **F** can be blended into an output response and in effect switch to some other chain of patterns; “switch to another train of thought”.

A third case is that the SRS is a subsumptive and hierarchical system of SRSUnits and an SRSUnit can be the input to other SRSUnits and can also have other SRSUnits as its input. Each SRSUnit has its own and all SRS run asynchronously so another **N** and/or **M** could break a given Cortex Neuron out of a lockup. This would happen when the upstream SRS has an abrupt change and rises. See section 5.

## Free Running Emergent Behaviors

The system free runs in the above loops and each SRS and the Cortex Neurons are asynchronous, and each has its own . The SRS implementation operates on a **tick** that is some fraction of .

Some **M** may be considered as “low level” modules with short such as in the above edge detection example.

Other much longer SRS that are “higher level concepts” with longer . High level SRS may take longer to react to external stimulus and thus the situation can emerge where the low-level SRS are “in the real world” and high-level SRS in a state more related to Cortex matching and avalanches. Thus, the high levels are on an independent or semi-independent train of thought not directly related to the current world state.

There will be a fuzzy boundary relative to time scales that will change moment by moment. Thus, the system may maintain external robotic control while at the same time be strategizing what higher level actions to predict toward and act on. Interestingly this arrives at the same behavior as the popular three level style of robotic control, but from a completely different reasoning and logic.

## The Importance of Random Noise

The system will run in **NM** cycles and respond to the environment as programmed. It is useful to have occasional completely random match hits in the matcher if both and **Q** are low. High-level concept maps should have this behavior, though it should be very infrequent. This disturbs the system out of idleness. This also give the system a propensity to think, rather than be idle. One can have this randomness occur more often in a calm and idle system. Very calm environments will allow the system to self-process and possibly make subtle connections of concepts and relationships that otherwise would be missed. This also would be processing that would cement the identities of **Ṁ**. See section 6.2.

# Learning

One of the most difficult challenges is to know when to learn.

## When to Learn

Eq 2: Learn Probability: **L = {>0│0}**

The **N** should not try to learn every single pattern that is ever in **M**. Rather it wants to learn patterns the in the future cause higher **G**. Thus, the derivative of **G** averaged over , if increasing indicates a good pattern to learn. Since an event at **T** +  (where **T** is “now”) may have a very poor **G** or decreasing **G** at **T**, we want to look at **G** nearer to **T +** , one possible time frame would be the average

Eq 3:

of **G** as in Eq 2. Since we need time dependent **G** values a fifo (first in first out buffer in this case at ¼ intervals) must be kept for this algorithm. An alternative to a fifo might be a long and short term **G** buffer (two single values) that follow **G** averaged over two time spans. This also would give the derivative easily. The actual implementation is a fifo history of **G** over several minutes and fine intervals for efficiency. This imitates hormonal release being the goodness factor.

# Concept Maps

A SRS Unit is also called a Concept Map. The map **M** is the current representation of some abstraction of data. This could be something simple and immediate like stereo sounds direction, or something complex and longer term like overall movement or travel goal. The map **M** could also be any dimension as an array including 0 (a single value), 1 a linear array e.g. stereo sounds direction, 2D or higher dimensions.

## Attention: Internal and External

The map blending equation (Eq 1)

Eq 1:

Is implemented such that on each tick of the SRS **M** gets updated. We use a SRS tick interval so as to not have to update **M** for every SRSon every cycle or tick of the overall SRS System. This lets one distribute the processing of longer SRS.

A **M** has a **decay** factor **Fd**, which is how much the value of all neuron decays to 0. A value of 1.0 is no decay, and low values toward 0.0 are fast decay.

The decay and fatigue together make the system more dynamic over time and allow for sensing novel stimulus and later ignoring unchanging stimulus.

The attention can be external with high or internal with high **Q**. If is high then the SRS is acting purely as a subsumptive directed graph of functional transformations. One can force high when designing and debugging an SRS. With high **Q** the SRS is imagining some environmental state. Forcing **Q** high and low generates an autistic system with no external attention. It is interesting when **Q** and are balanced. They system is then generating actions based on past experience blended with the current environment and thus has volition. The blended **M** represents a solution that will tend toward higher **G** in the future given the current environmental state. This balance between **G** andis key to a useful operational system.

# Multi-SRS systems

## System Creation

The overall system is designed by intuiting what mappings of the environment might be useful to attaining a higher **G**. A useful **F** would be any transformation that arrives at some abstraction of what is going on in the environment that might promote survival and a higher **G**. One can name particular **F** as a terminology for system design.

As an example, one might generate an SRS called “Sound Approaching” that takes the derivative of audio input volume in stereo, and direction, and maps to a 1D array of 360 values representing how much sound is increasing from a given heading in degrees. Other SRS then can use Sound Approaching to compute further abstractions or generate reactions such as fleeing from the approaching sound.

With a subsumptive design one can then design another SRS that is “Pheromone Sensing” (maybe our robots emit a weak radio signal to indicate reproductive readiness) and overrides Sound Approaching. This makes the robot flee sounds unless there is a signal of reproductive readiness in which case it seeks the sound instead. Thus, one can design freely and connect the SRS network to achieve hierarchical behaviors without having to invent the entire system at once.

One could also design the system with two independent multi-level SRS systems with different goals, and some limited inter communication (each system is tied into a relatively few SRS in the other system) and have an overall system that “thinks” in two modes and thus a “split brain”. The common SRS between the two systems would be low-level and the separate system SRSs would be more high-level. Not being limited by evolutionary history one could design a system with many such sub-SRS-Systems.

# Global Pattern Connectivity

The SRS design described above uses clusters of **N** as an array of copies of previously seen **M** for conceptual simplicity. It is assumed that a given **M** at timer **T** can be used to predict the future **M** at time **T +** . This may not actually be true. For example, a current **M** representing the center of image edge angle will not very well predict what the edge orientation will be at **T +** . Such a prediction requires a more global pattern match.

A better predictor is to snapshot the entire multiple SRS systems and have a huge number of **N** to detect known overall patterns. Obviously, this would be such vast amounts of data that one could not store it and a biological brain cannot simply have a few million extra copies if itself. A much more realistic approach is to have a fewer **N** be a rough approximation of the global state.

## Connectivity Triples ⅄

If a single **C** has a pattern that is 10% of **M** and **N** that are very close to the target **N** and then another 75% of fairly closely related concepts, and the remaining 15% of connections being to pointers in the entire global system, one could arrive at useful predictions. This series 10:75:15 (always adding up to 100) is a **connectivity triple** and is represented by the symbol **⅄** an upside down Y representing 3 parts. The three elements are **⅄1,** **⅄2, ⅄3.** One could posit connectivity quadruples or more elements. The triple seems to communicate the intended connectivity well.

Interestingly, the **⅄** is a representation of local vs global connectivity and directly relates to the type of predictions made. A **⅄** of 50:49:1 will give predictions that are immediate and concrete. A **⅄** of 5:5:90 will be a very global holistic prediction. Different cortex sections should have varying connectivity triples.

In biological systems local connectivity requires shorter neurons than far connectivity. So **⅄** with large **⅄3** is expensive but also are very valuable for overall pattern matching and prediction, and thus high IQ. Due to evolutionary history, the overall pattern matching and in particular **⅄** with large **⅄2** were probably added rather late. Thus the **S’** implementation in a biological brain has **S’** and the pattern matcher as an added layer on the outside of the multi-SRS survival-reflex system and is the cortex. Each cortical neuron then has a target area deeper in the brain that it is trying to predict and stimulate the inner brain sections into the future state. One then would expect cortical neurons to have vertical long connection into the inner brain as observed.

## Meta-Maps and Anonymous M

One interesting effect falls out of this description. If the **N** is trying to predict the future **M**, one could also have **N** that are trying to predict the future **C** of other **N**. In effect the matcher becomes a **meta-map** and very abstract concepts that have no direct environmental correlates could get set up in the Matchers. This refactors into the concept of an SRS that has no **F** but is simply an **M** and is an anonymous concept **Ṁ** (M with a dot over it). This is not a designed SRS and **F**. One simply has a large array of initially identical SRS **M**. Over time and learning these **Ṁ** begin to represent abstract concepts. This might correlate to the cortex processing abstract ideas like “loyalty” or “Hobbit”.

# Subsumptive Architecture Not Needed

The purpose of the subsumptive Concept maps is to compute abstractions on inputs, and de-abstract concepts toward the outputs. Biological brains do this because they must use neurons to perform computations. In a program there is no need for the subsumptive layers. Instead, abstractions can be computed directly in code, such as generating the edges in the fovea region of the camera. Then a Concept map can simply represent the state of the abstraction from the code.

This flattens the system design and simplifies how it is implemented. It also greatly increases execution speed.

Cortex

N  
One Cortex Neuron

S

S’

Q  
Match Quality

Out to  
World and Hardware

# The Reaper and Connection Routing

In the SRS System it is certain that some old learned patterns of **N** are no longer relevant and are never a match. It is also possible that some of the inputs to a **N** are not correlated with the target **N** or **M** and are incorrect connections.

An algorithm called the **Reaper** periodically checks the activity level over time of the cortex **N** and if never used in a very long time, re-randomizes the connections and target, thus reuses the **N**. Also input weights that are near zero and have been so for a long time are reconnected to some other random source. Over time the system can reroute and grow similarly to neuroplasticity.

In addition, the cortex needs to be grown gradually. If the entire cortex and all the possible **N** are instanced when the system is created, the many **N** will be random and overwhelm the subsumptive system. Thus the cortex gradually adds new **N** over time at some TBD rate. Initially the system is just the subsumptive system and **N** get created with near **⅄** ratios. Much later **⅄** with far connections are created.

In addition, the subsumptive system is being added to by a programmer and new concepts created. It is necessary to have the cortex by dynamic and always growing.

## Continuous Learning

One option that needs to be tested is where instead of using Eq. 2 and goodness to decide when to learn, is to always learn. On every tick create one or more new cortex neurons. Then the reaper algorithm decides which get kept. If a single cortex neuron gets used several times it can solidify the connection so the reaper leaves it in place.

More TBD as development continues on the SRS22 test implementation.

# Implementation Notes and Conjectures

It is imagined that implementing a single SRS and an entire SRS system would be done using CUDA on one or more GPUs.

## Synchronization

The various SRS do not need to do any locking or synchronization. In the best case a GPU thread that wants to modify a **M** can just do so and since the total elements of all **M** is very high compared to the number of threads, there will be few collisions. In the case of a collision, the reader of the **M** might get a previous value or even a corrupt value. On the next SRS tick, it will get the right value and the error is just slight noise in the overall system.

Note: The actual implementation currently is two phase per system tick. A “compute next state” and then “copy next to current state”.

One would synchronize the GPU on every simulation tick (not SRS tick) to copy in and out external world inputs and outputs.

Also, the reaping of stale **N** and **M** would have to be on a sweep basis. This is quite like sleeping and cleaning up neural connections. One disables direct muscular activity so one does not thrash about in sleep. So, one would expect the system to occasionally halt for cleanup and garbage collection (once a day?).

## Neuron Pulse vs. Value

In biological systems one evolutionary reason for neurons being pulsed is that the pulse rate can be relatively immune to oxygen, sugar, or other factors. Pulsed neurons also enable some time critical short-term computation and measurement such as sound direction. Pulse summing enables neural hiding, which is a important feature of the system.

In a computer simulation short time critical events can be accounted for by having **F** that directly compute the time differences. Subsumption can also be achieved with simple addition of values rather than actual pulses. **F** and **OF** are optional. The implementation has all maps as subclasses of SRSUnit being the generic Concept Map module. Some SRSUnit are of the same class if they have general input connections to other SRSUnit and not to hardware IO.

As a side note, neurons have a normal activation range of 0.0 to 1.0 but may go outside that range. It is still to be determined whether allowing values out side 0 to 1 is a correct approach.

## Redundancy

The redundancy to compensate for cell death and damage is not needed in computer digital system. Thus, it is expected that an SRS could achieve AGI equivalent to the human brain with an order of magnitude fewer elements and connections. (1% as many neurons and connections?) See (Glassman, 1987) for an interesting read on redundancy. Also (FABRIZIO DE VICO FALLANI, 2012) Biological Correlates

Here are some random notes on biological systems that do not fit cleanly in other sections, or that are even less than conjectures.

### Computer Memory Locality

The entire SRS22 algorithm is opposed to current computer memory design. The running algorithm with each **N** being sparsely connected to the entire rest of the system will cause cache thrashing. In a GPU such highly random memory access is a major performance killer. Much research will need to go into this problem and how to preserve memory caching. Some new style neural computers are being designed to combat this problem.

### Multiple Muti-SRS systems

The brain is obviously split into two hemispheres, the left and the right. The hemispheres operate as fairly independent thinkers but with two styles. No, the author will not start making grandiose statements as to what each half does. See (McGilchrist, 2012).

### Biological implementation of G

Is **G** the equivalent todopamine in organic brains? It has been shown that dopamine increases cause learning. An event followed by a reward is the fastest teaching technique. The technique of event, then punishment, then reward after changed behavior has been shown to be less effective in psychological studies.

### Derivation of X, Y, Z coordinates

One could use the X,Y,Z location of sections of the human brain as a guide for SRS concept coordinates. In a program SRS can have the same X, Y, Z where in a biological brain, neurons must be spatially separate. Thus, one could use the location of hearing in the brain coordinates in millimeters or Talairach Coordinates. This also might get the closer to human though reproduction and a system one can understand. (Wikipedia, Talairach coordinates, 2020)

### Connectivity Triples

Connectivity Triples are a simplified method to talk about how connected some section is, and to encode that connectivity into program meta-code. In a biological brain the effect of triples would be embodied in how aggressively cortical neurons connect further away vs. nearby and would be a continuous curve of probability to connect at a given distance.

### Distance from the Spinal Cord

One might conjecture that neural paths close to the spinal cord have short and the furtheraway sections represent more abstract concepts and longer .

### Fatigue

Synaptic Fatigue in biological neurons directly mirrors the fatigue in the algorithm. It is interesting that the need for fatigue was thought of first, and much later discovered as a known factor in biology/

## Avalanche and Generative Adversarial Networks and/or Autoencoders

There is some similarity between avalanche into some system state and GANs (Goodfellow, 2014). In particular, if the output of a GAN is fed into the input of itself the GAN will then constantly generate new versions of its scene or domain. In effect the GAN generates a scene then cascades onto some other scene ad. Infinitum.

An effect that looks very similar (but is not an avalanche) is interpolating in latent space directly. (See https://www.reddit.com/r/deeplearning/comments/urbfv8/latent\_space\_interpolation\_of\_an\_ascii\_art/)

# References

Brooks, R. (1985). A Robust Layered Control System for a Mobile Robot. *Massachusetts instuitute of Technology*, 25.

Chan, S. (2001, October 31). *Complex Adaptive Systems.* Retrieved from web.mit.edu: https://web.mit.edu/esd.83/www/notebook/Complex%20Adaptive%20Systems.pdf

FABRIZIO DE VICO FALLANI, J. T. (2012). REDUNDANCY IN FUNCTIONAL BRAIN CONNECTIVITY FROM EEG RECORDINGS. *International Journal of Bifurcation and Chaos, Vol 22, No 07*.

Glassman, R. B. (1987). An hypothesis about redundancy and reliability in the brains of higher species: analogies with genes, internal organs, and engineering systems. *Neurosci Biobehav Rev*.

Goodfellow. (2014). *Generative Adversarial Nets.* Proceedings of the International Conference on Neural Information Processing Systems (NIPS 2014).

Keene, R. (1995). *A New Model for the Cognitive Process.* IEEE, INBS '95: Proceedings of the First International Symposium on Intelligence in Neural and Biological Systems (INBS'95).

McGilchrist, I. (2012). *The Master and his Emissary – The divided brain and the making of the western world.*

Michael Pereira, D. P. (2022). A leaky evidence accumulation process for perceptual experience. *Trends in Cognitive Sciences*, 11.

Thompson, T. (2009). Improving control through subsumption in the EvoTanks domain. *Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Games, CIG*.

Wikipedia. (2020). *Talairach coordinates*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Talairach\_coordinates

Wikipedia. (2021, Nov). *Subsumption Architecture.* Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Subsumption\_architecture

Yi Li, G. W. (2018). DeepIM: Deep Iterative Matching for 6D Pose Estimation. *arXiv:1804.00175*.