Chapter 10
Noble Ape's Cognitive Simulation:
From Agar to Dreaming and Beyond

Thomas S. Barbalet
Noble Ape, USA

ABSTRACT

Inspired by observing bacterial growth in agar and by the transfer of information through simple agar simulations, the cognitive simulation of Noble Ape (originally developed in 1996) has defined itself as both a philosophical simulation tool and a processor metric. The Noble Ape cognitive simulation was originally developed based on diverse philosophical texts and in methodological objection to the neural network paradigm of artificial intelligence. This chapter explores the movement from biological observation to agar simulation through information transfer into a coherent cognitive simulation. The cognitive simulation had to be tuned to produce meaningful results. The cognitive simulation was adopted as processor metrics for tuning performance. This "brain cycles per second" metric was first used by Apple in 2003 and then Intel in 2005. Through this development, both the legacy of primitive agar information-transfer and the use of this as a cognitive simulation method raised novel computational and philosophical issues.

ARTIFICIAL LIFE, NOBLE APE AND AGAR

There is no coherent, universally accepted history of artificial life. The term artificial life was coined by Christopher G. Langton in the late 1980s (Langton, 1997). From the late 1980s to the early 1990s a number of popular and academic books covered the topic of artificial life either as a surveying of the art (Emmeche, 1991; Levy, 1992) or covering the author's particular interests in artificial life (Dawkins, 1987). Contemporary practitioners of artificial life tend to attribute one of these books as the basis for their development - Dawkins' Biomorphs (1987) or Dawkins' inspired possibilities (Ventrella, 2005; Barbalet & Stauffer, 2006; Barbalet & De Jong, 2007) or Sims' Blockies (1994) (Barbalet & Klein, 2006). Dawkins, Sims and the inspired practitioners'

DOI: 10.4018/978-1-60566-705-8.ch010

simulations were based on genetic algorithms.

Noble Ape was framed in the broadest possible surveying of these books. No particular book was the focused basis for the development. In fact, Dawkins' earlier work (1976) was considered over Dawkins' later work (1987) with regard to the social framing of Noble Ape. Without any guiding artificial life text, the foundational theme of Noble Ape was that artificial life empowered the developer to take from any area of interest and assemble a testable simulation environment to see how these theories inter-played. This was a view born in isolation.

In 1996, the open source Noble Ape Simulation was created to produce a rich biological environment and to simulate the movement and cognitive processes of the Noble Apes, a sentient ape-like creature that wandered through the simulated environment (Barbalet, 2005c). One of the primary interests of the Noble Ape Simulation was the development of simple societies and whether the environment could contribute to the structure of the societies created. Would a harsh environment with limited food and water create an authoritarian society? Would an environment teeming with food and water produce a casual and contemplative society? What would happen through transitions of famine or war? How would the societies develop through these experiences?

If there was a seminal theme through the original development of Noble Ape it was through the ideas of Logical Atomism (Russell, 1956). These ideas were not developed in the age of contemporary computing however they appeared applicable through the description of sense-data processing and the idea of atomic sense information. Logical Atomism presented the idea that sense data was provided in discrete processable quantities. Through providing sense data over time, the development of a coherent self could be generated. These ideas were further refined in Noble Ape Philosophic (Barbalet, 1997) into a coherent means of taking the external world and making an internal created representation. It is important to note that this internal representation can be without observable reference in terms of relationships between the external information (or the sense data presentation of the external information) and the internal representation. This still gave no indication of the method of processing. In terms of the cognitive simulation it defined vision (shorthand for all external sense data), the identity which was the material of the cognitive simulation, fear and desire. Fear and desire were the two operators that acted on the identity to manipulate the vision information over time.

The artificial life project closest to Noble Ape was PolyWorld (Yaeger, 1994). Like Noble Ape, PolyWorld was an example of the "intelligent agents in a simulated environment" class of artificial life simulations. Although the projects were completely independent, they shared a number of the same high-level concepts - computational genetics, physiology, metabolism, learning, vision and behavior.

The primary distinctions between Noble Ape and PolyWorld related to two components. Noble Ape contained a more detailed simulated environment, including an undulating topography, a changing weather simulation and a biological simulation containing a diversity of simulated flora and fauna. The other distinction, and the subject of this chapter, was the means of simulating the intelligent behavior of the agents. PolyWorld used a neural network model of intelligence. Noble Ape did not.

The motivation not to use a neural network intelligent agent model in Noble Ape was due to Kirsh (1991). Kirsh asserted that simple processes could provide "concept-free" intelligence which was shared through all intelligent life from simians to insects. This seemed plausible and also linked well with Russell's account of Logical Atomism. Kirsh's position was highly critical of traditional artificial intelligence methods. Whilst Kirsh did not name neural networks explicitly, the tenor of his text was clearly against "highly cerebral activities" in intelligence modeling. Rather than being

a technical paper, Kirsh wrote a philosophical critique of artificial intelligence which motivated much of the early cognitive simulation development in Noble Ape.

Without any clear technical description of the means of concept-free sense-data processing based on these foundational ideas in the initial Noble Ape development, the only other simulation mechanism available to the author was agar simulations that had been developed prior to the Noble Ape development as a means of learning to program real-time graphics applications.

Through high school, the author took biology courses that utilized agar studies. Materials like the author's hand and contact swabs had been place in agar dishes. The contaminated agar dishes had then been left to grow in warm, dark environments. These agar dishes were studied in terms of population growth and levels of bacteria based on the known population growth rates. These kinds of agar studies gave the author an early insight into the agar medium and bacterial growth.

With the author's interest in computer simulations, agar simulations had been particularly useful as they provided real-time movement over the screen as the agar's simulated bacterial plumes took hold and died back. These simulations represented an idealized biological environment where one or potentially more populations of organisms colonized the energy rich agar environment. In its simplest form, there was a numerical array representation of the energy stored in each agar cell and a population number indicating the population density at that particular agar cell. Various algorithms were used to produce population growth, energy consumption and movement into adjacent agar cells. These agar simulations were a population cell occupation step from simpler cellular automaton simulations. Although the agar simulations did share some of the same spatial spread processes.

In addition to artificial life books, the early Noble Ape development was also heavily influenced by "write your own computer game" books (Isaaman & Tyler, 1982; Howarth & Evans, 1984). These books discussed how to create fictional environments and related mythos associated with these kinds of games. From the inception of Noble Ape, there needed to be a strong narrative relating to the simulation. If the entities in the environment were merely agent dots on the screen, the user would be less likely to bond with the agents and the simulation. The agents had to be humanoid and the name "Noble Ape" gave a strong emotive resonance with the user.

The development of the cognitive simulation followed a familiar pattern to other aspects of the Noble Ape development. A philosophical vision directed by apparently unrelated source code that converged back on the philosophical vision. It is this element of cyclically re-enforced philosophical experimentation through source code that has kept the author's interest in the Noble Ape Simulation for more than a decade and quite possibly many more decades to come.

The development of the cognitive simulation through agar simulation methods required a clear vision with regards to the two primary elements, fear and desire. Once the algorithm was established, the cognitive simulation needed to be tuned and the method chosen to tune the simulation required a three dimensional, real-time visual description of the cognitive simulation. Once the simulation was tuned, it was adopted by Apple and then Intel as a means of providing a rich processor metric.

The cognitive simulation development did not end after tuning or corporate use. It continues to this day and offers a number of interesting future projects both through Noble Ape and by third-parties. Like PolyWorld and a number of other contemporary mature artificial life simulations, Noble Ape is intended to continue development for many decades to come. It is not intended to be a stand-alone complete project subject to conclusion-testing. It continues to provide insights to users and developers alike. Some of these aspects will be discussed here, however

Noble Ape should not be thought of as a static or historical work.

This chapter is offered as an example of how to take a divergent set of concepts from a number of sources, including inspiration from the natural world, and produce a coherent and unique solution. It is intended to show not only the particular use in the Noble Ape Simulation but also the potential for this method to be used in other applications.

AGAR INFORMATION TRANSFER AND COGNITION

The author's interest in agar simulation predated Noble Ape. It came from three sources - high school biology coursework (as discussed), developing computer anti-virus software and as a means of showing time-evolving graphics. Writing anti-virus software created an interesting byproduct - a fascination in the heuristic analysis of infections over networks and how these infections appeared to replicate biological viruses. The computational and biological infection graphs showed similar properties and it intrigued the author sufficiently to write simple agar simulations to confirm that the simulated biology could replicate these growth rates.

As with the similarities between Noble Ape and PolyWorld, the agar simulations shared a number of traits with cellular automaton simulations (von Neumann & Burks, 1966). Rather than single cellular automaton inhabiting each array cell, a population of bacterial agents and a corresponding energy value for that agar cell over an array of agar cells was the basis of the agar simulations.

At the same time, the author began to experiment with simple real-time graphics and in particular the color capabilities of VGA. This produced colorful agar bacterial growth simulations that showed the growth and infection patterns on a fullscreen display.

As an enhancement to the long-term visual interaction, the agar simulations had end wrap-around properties. This eliminated most problems with boundary conditions and created effectively an infinite simulation space for the simulated bacterial spread. In addition to simulated bacterial consumption of energy, energy regeneration and growth was added. The idea that the simulated agar including the bacteria and energy in the system resulted in a zero sum was critical. The bacteria once dead would be the food of the next generation. The bacteria-rich agar had to flourish through wave after wave of bacterial plume. These agar simulations were integer-based both in terms of the energy level and the bacterial population but also in terms of the grid coordinates. Each grid coordinate represented an integer value of bacteria population and energy.

Through these experiments many algorithms to produce regenerating agar growth for great visual effect were developed. All the simulations had an information transfer function similar to:

$$I_{t+1}(x,y) = (I_t(x-1,y-1) + I_t(x,y-1) + I_t(x+1,y-1) + I_t(x-1,y) + (a * I_t(x,y)) + I_t(x+1,y) + I_t(x-1,y+1) + I_t(x,y+1) + I_t(x+1,y+1)) / b \quad (1)$$

where I was the information transfer meta-agar, a was a weighting value to show the primacy of the central reference and b is a normalizing value. x and y were the cell coordinates. Whilst not referring to cellular automaton, this equation is the simulated-agar equivalent of a Moore neighborhood (1962).

This formula did not exactly replicate the random fluctuations in the agar simulation but it gave a good general equation for information transfer over time. The experimentation with agar simulations continued. The parameters that went into the growth and death of the agar simulation and the algorithms associated with energy consumption and bacterial survival became less important. The agar simulations all shared a similar pattern of

information transfer between the distinct population and energy cells. This information transfer became more interesting than the underlying agar simulation.

This required a new mathematical method to describe the information transfer rather than the traditional agar simulation metrics of population and energy fluctuation.

Over time the agar populations would move outward through the agar at a rate of spread which was algorithm dependent but always adhered to the same characteristics. The spread described a spatial wave-like property of the agar. This population information transfer was an ethereal property of the agar. This analysis begged the question - what would happen if other kinds of information were placed into this meta-agar information transfer environment?

This meta-agar had both time resistive qualities and also dissipation qualities which could be represented through quantized spatial and temporal units as follows:

$$dI_{t+1}(x,y)/ds = I_t(x-1,y) + I_t(x+1,y) + I_t(x,y-1) + I_t(x,y+1) \quad (2)$$

$$dI_{t+1}(x,y)/dt = I_t(x,y) - I_{t-1}(x,y) \quad (3)$$

where *x* and *y* were the cell coordinates. *I* was the information transfer meta-agar. *dI/ds* referred to the spatial derivative and *dI/dt* was the time derivative. Obviously through various simulations there were constants of environment applied to both *dI/ds* and *dI/dt*. This did not change the underlying spatial and temporal properties. The *dI/ds* algorithm is analogous to a Von Neumann neighborhood (1966).

It is important to understand these two equations in terms of information transfer.

dI/ds propagates information through the meta-agar and also defines how information can be collected through the meta-agar. Aside from information flowing from the center, there is also a collection of information where the information gradient is particularly favorable. There is nothing that says that the total information has to remain constant through these functions. The dI/ds can be tuned to favor optimizing information piling around like values. This is analogous to simplification and amplification.

dI/dt can also be tuned to allow for cyclical resonance which, in concert with dI/ds, can propagate particular information with greater intensity. This is analogous to wave mechanics in physics and the primary reason the physics algebraic conventions are adhered to in describing the derivatives.

Noble Ape was created relatively rapidly. It was formulated and first announced within a three week period. This was achieved as a number of the components of Noble Ape had been developed through other coding projects. These components included the terrain visualization, the real-time mouse and keyboard interface and also file parser code. Part of the underlying rationale behind developing Noble Ape was to take the divergent pieces of software the author had developed and put them together.

The cognitive simulation was no exception. It could have gone one of two ways. Prior to Noble Ape in 1994, the author travelled through Malaysia and Thailand. This provided a trilingual transfer between the English language in the larger cities to the Bahasa (Malay) language in the regional parts of Malaysia to the Thai language. The three languages were distinctly different but they all mapped onto the same underlying meaning. This begged the question - could a general language analysis algorithm be developed to parse vast quantities of a particular language and generate a syntactical-check? More importantly also a meaning-check to construct meaning from a substantial body of text. This idea became LaSiE (the LAnguage Simplification Engine). As the Philosophy of Language was also alluring to the author through reading Bertrand Russell and his student, Ludwig Wittgenstein (1953), Noble Ape's

cognitive simulation could have been an abstract engineering of LaSiE. LaSiE was in part a specialized neural network algorithm, but also as the name suggests it had a substantial phoneme reduction component to simplify the neural network. Following extensive algorithm testing and modification relating to the best and most substantial electronic texts of the time (Starr, 1994) amongst others, the results were inconclusive. In contrast, the meta-agar simulation provided more positive results. The potential for this meta-agar to yield an abstract but successful cognitive simulation seemed likely.

There were still a number of problems. The information transfer did not map well to visualization information. A considerable amount of time was spent exploring how the spatial visual information could be translated into a two dimensional cognitive simulation. The original vision method had an eye simulator creating both simulated eyes' two dimensional photographic renderings of the outside world back into the cognitive simulation. This was not feasible computationally.

The solution was to translate the outside world into a scanning obstacle line - not like a radar or sonar - but like a piece of string that stretched to the nearest obstacle as a radial scan of the vision. This idea was central to the development of the Psi visualization method (Rushkoff, 1999).

It became evident that what was missing was an additional dimension in the cognitive simulation - it needed to be three dimensional (plus time evolution, obviously). The addition of a third dimension clouded how the visualization information would be put into the cognitive simulation and also how point and line (versus linear) actuators would be placed in the cognitive simulation.

In three dimensions, the Noble Ape cognitive simulation was transformed to:

$$I_{t+1}(x,y,z) = (p * I_t(x,y,z)) + (q * dI_t/ds(x,y,z)) + (r * dI_t/dt(x,y,z)) \quad (4)$$

where

$$dI_t(x,y,z)/ds = I_t(x-1,y,z) + I_t(x+1,y,z) + I_t(x,y-1,z) + I_t(x,y+1,z) + I_t(x,y,z-1) + I_t(x,y,z+1) \quad (5)$$

$$dI_t(x,y,z)/dt = I_t(x,y,z) - I_{t-1}(x,y,z) \quad (6)$$

where x, y and z were the cell coordinates. p, q and r were weightings to produce a normalized value for I_{t+1}. dI/ds was labelled "desire" and dI/dt was labelled "fear". The origins of these names have already been discussed with regards to the Noble Ape Philosophic document. The algorithmic distinction of these two components were subdivided and named for both methodological and emotive reasons.

The methodological reasons related to the mathematical primitives that also came through the biological simulation. Noble Ape's biological simulation was based on quantum mechanics. The landscape provided the wave function for operators to be applied to determine the surface area, directional gradient and other factors similar to a wave function in quantum mechanics. The landscape height map could be operated on to provide additional biological information. The surface area of the landscape at a particular point indicated the amount of grass or moss that was present. A greater surface area (characterized by steep angles) meant trees could not grow. Similarly the east/west traversing of the simulated sun provided an accurate indication of insect movement and also where the most sun would be found over the land area. These biological operators resolved down to basic orthogonal primitives.

The same was observed with fear and desire. The algorithmic effect of fear was a very rapid time evolving reaction. Desire in contrast dissipated in all directions leaving a shadow of previous information. Fear and desire were orthogonal and they contributed different properties to the information transfer.

The final orthogonal component was the representation of the information transfer itself. This

could be represented by contour lines through three dimensional space. This captured the shape of the evolving information transfer and indicated the stability of the cognitive simulation. It was that information transfer contour that was so useful in tuning the cognitive simulation.

The emotive reasons behind naming the two components of the equation fear and desire was it allowed someone who was a mathematical novice to get a clear understanding of what these properties did in the cognitive simulation. The description of fear as an instantaneous reaction to rapid changes in stimulus gave a solid emotive connection to both what the algorithm was attempting to model in the cognitive simulation but also with the user's own experience. Similarly the discussion of desire as being something which motivated future judgements, future expectations, future understanding and also desire as the great simplifier, seemed to both describe the algorithmic reality and resonate with the user's experience and understanding.

TUNING AN INSTRUMENT IN A NON-LINEAR KEY

There is an ethereal quality in tuning a detailed non-linear simulation that combines elements of art, science and experience. Other simulators, too, find it difficult to describe what is explicitly needed to be able to tune artificial life simulations (Barbalet & Stauffer, 2006; Barbalet & Daigle, 2006). Before exploring the tuning of the cognitive simulation, it is important to explore the basics in simulation tuning and then discuss the importance of visualization to give accurate feedback.

In 1997, a high school science teacher in rural Australia contacted the Noble Ape development to ask if it would be possible to take just the ecological simulation component of Noble Ape and put it in a program that his high school students could use. The ecological simulation in Noble Ape contained the animal groups found through the author's travels in Malaysia (Cranbrook, 1987). As already noted, the biological simulation dynamics were based in principles from quantum mechanics. The transition in simulated flora and fauna populations were covered with surface area integrals and convolutions. The teacher's requirements were better served with extended Lotka-Volterra equations (Volterra, 1931) where the notion of predator-prey was expanded to include plants, herbivores, carnivores and omnivores. From this the collected species were phylum-grouped into bushes, grass, trees, seed eating birds, sea birds, birds of prey, insect eating birds, insects, mice, frogs, lizards, apes, cats and fish. The fish represented the ocean fish population surrounding the Noble Apes' island and thus could be maintained as a constant food source. This provided a rich and complicated set of non-linear interactions that evolved over time.

The feedback provided to the high school teacher, in terms of teaching his students how to tune the biological simulation, was to start with a reduced subset of the species and look at how these groups interacted over time. This in some regard was analogous to learning to juggle by starting with a couple of balls. Traditionally Lotka-Volterra mathematics had been taught with two dimensional plotting to show stability conditions in joining cycles. For so many species interactions, this was impossible. The method of finding reoccurring values for all populations at known time deltas was not applicable with plant, herbivore, carnivore and omnivore interactions. After tens of hours of play, the author was able to get five species stability with occasional bursts into six or seven species stability. It was a bridge too far for the high school teacher and students.

Lotka-Volterra equations related to scalar population numbers. This was in contrast to the Noble Ape Simulation method which used area-integration to show population predation through overlapping habitat. The property of stability of simulation over areas and volumes is a different technique, but with similarities to the scalar

analysis in terms of time-cycle changes.

When development began on Noble Ape, the visualization of the cognitive simulation was achieved with planar scans through the three dimensional brain representation. This provided little meaningful information bar a great aesthetic draw to the user.

By 2000, a real-time rotating three dimensional visualization method that showed the changes in the cognitive simulation to a high level of detail was implemented (Barbalet, 2004). This, in turn, enabled the cognitive simulation variables to be tuned with real-time graphical feedback. Whilst this was not analogous to the cyclical graphs of Lotka-Volterra tuning, it enabled stable constants to be found for both the awake and asleep aspects of the cognitive simulation)(Figure 1).

The simulation needed to model the properties of sleep in terms of dreaming and problem resolution analogous to human and mammalian sleep cycles. The Noble Apes would need to be able to have some dreams they could remember and use the sleep environment to distill long-term goals. Ideally the quantifiable transition between awake and asleep could be transitioned for additional feedback.

The elements of the cognitive simulation had already been defined through the Noble Ape Philosophic document. The implementation of constants for the sleep and awake states of the cognitive simulation had to be regulated around those orthogonal ideas. This created an interesting mix of applied method weighed against philosophical methodology.

This method of creating graphically stable simulations with applied theories was well defined through the biological simulation tuning and the author's earlier work with agar simulations. The constants that emerged would ultimately be scalar weightings to multi-dimensional equations. The philosophy of fear and desire, the identity and ultimately Logical Atomism was also a heavy consideration in the simulation tuning but the simulation needed to operate with stability first and foremost. With these apparently competing ideas, the simulation constants found in March 2001 following months of simulation tuning were particularly interesting (Barbalet, 2008).

In the awake brain simulation:

$$I_{t+1} = ((I_t * 0) + (dI_t/ds * 171) + (dI_t/dt * 146)) / 1024 \quad (7)$$

In the asleep brain simulation:

$$I_{t+1} = ((I_t * 501) + (dI_t/ds * 86) + (dI_t/dt * 73)) / 1024 \quad (8)$$

Whilst awake the cognitive simulation existed purely on instinct coming from desire (the *dI/ds* multiplier) and fear (the *dI/dt* multiplier). Desire played a slightly heavier role in both awake and asleep states, but the fear and desire elements were both roughly halved during sleep with a substantial residual identity maintaining through the sleep versus none in the awake state.

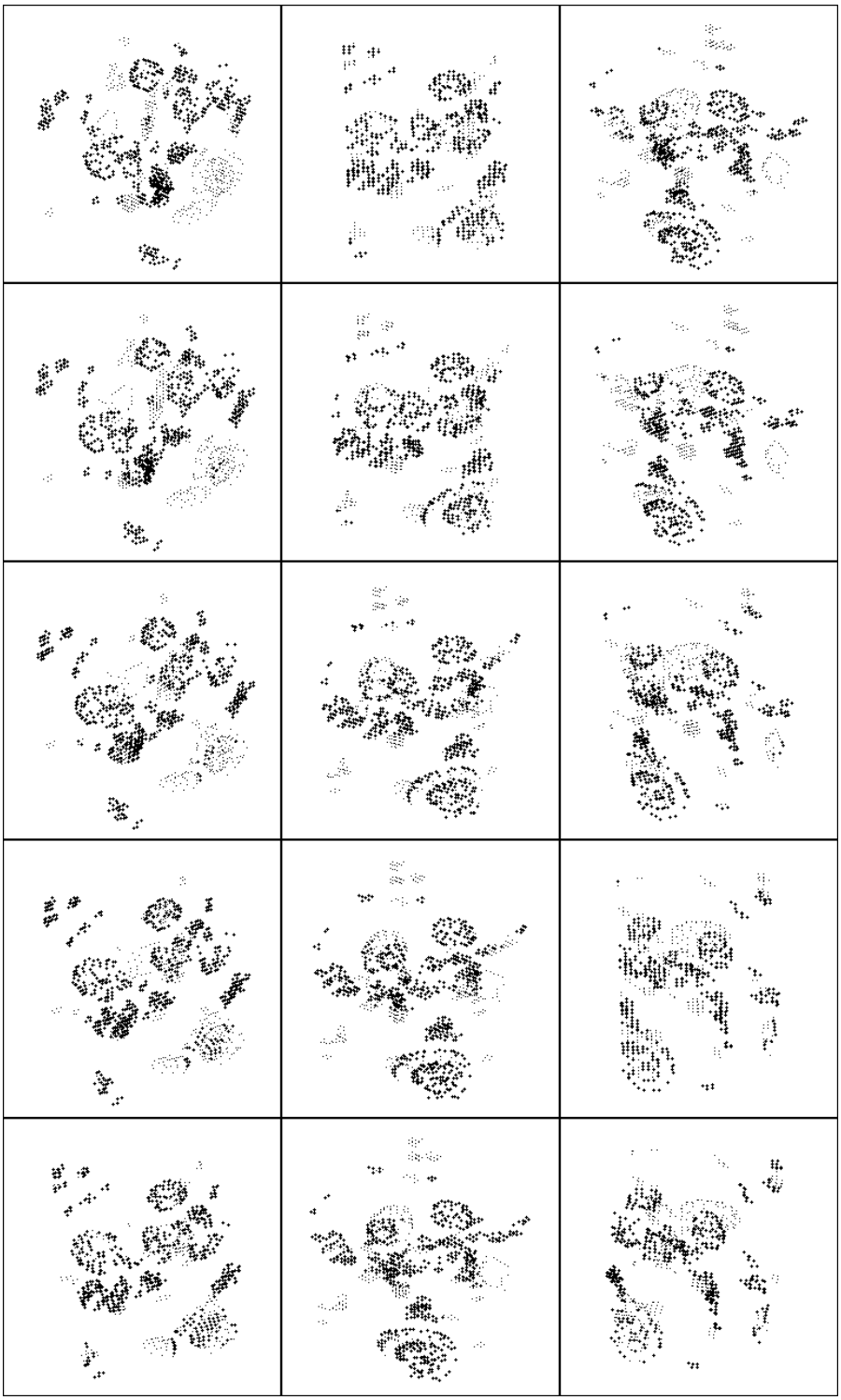
For Noble Apes that were unable to sleep, in conditions of drowning or for other similar extreme reasons, the effects in the cognitive simulation would be quite dramatic indicating the level of trauma the Noble Ape was suffering.

Two important points should be noted with the final six constants that came through the applied analysis of the three dimensional graphical output of the cognitive simulation.

These results may not be unique. In fact there is a good range around the constants (tested to +/- 5 in all cases) that yielded the same kind of stability. It is to be determined if there are unique properties associated with these number ranges. It has been a useful byproduct to allow users the freedom to choose their own constant weighting or variations of cognitive simulation constants to allow experimentation with the outcome through these ranges and their range boundaries.

The subjective nature of the non-mathematical to mathematical mapping of fear and desire and the narrative historical description of the Noble

Figure 1. An example of the Noble Ape cognitive simulation visualization through general movement paused at 11.25 degree rotations



Ape Simulation development could indicate an implicit yet predetermined outcome. This is a fair criticism. The results captured the almost circular philosophical outcomes of creating a mathematical modeling method that described two orthogonal operators and the insight into awake and asleep states of the resultant mathematical simulation.

Both these points merit further investigation.

The apparent dampening of the cognitive simulation through the large total divisor (1024) may appear a point of concern for information persistence. This analysis comes through passive observation. It should be noted that this dampening actually forces dynamic structure propagation through the cognitive simulation which explicitly beats this dampening. This can be observed in the Noble Ape Simulation but merits further descriptive investigation and analysis (Figure 2).

Through tuning the cognitive simulation, the description of sleep in a quantifiable sense may be considered with some interest. The sleeping Noble Ape has a roughly halved awake-state combination of fear and desire but also maintains a half residual identity component which doesn't exist when awake. As with real animals, the Noble Apes remain relatively paralyzed during sleep. It would be an interesting experiment to simulate the dreamt movements of Noble Apes as an additional psychological tool to understand the Noble Apes thought processes a little better.

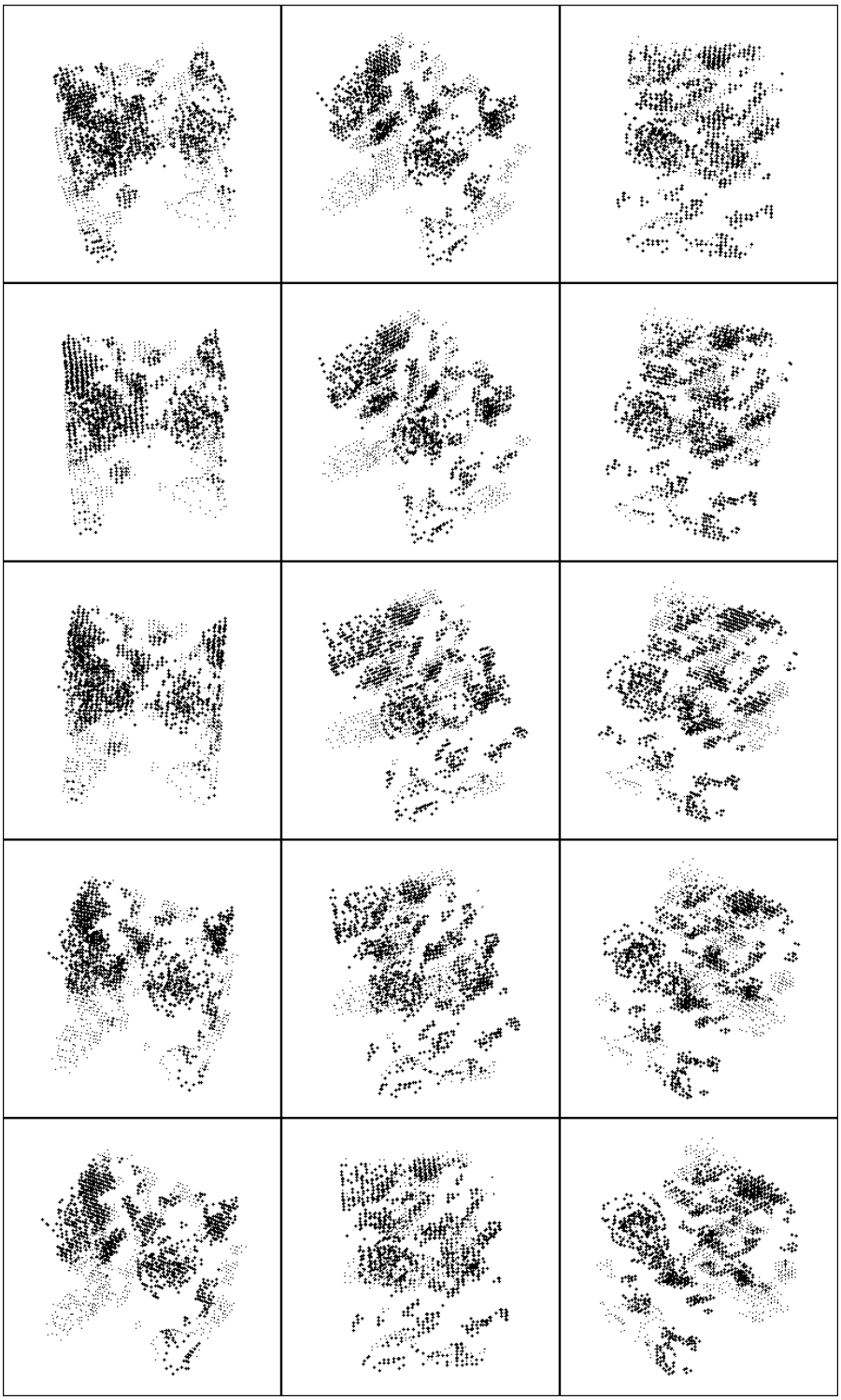
APPLIED USE OF THE COGNITIVE SIMULATION

Without the interest of two engineers at Apple, Nathan Slingerland and Sanjay Patel, Noble Ape's cognitive simulation would have continued as an open source curio. In early 2003, they asked to use Noble Ape as an example for new optimization techniques that Apple wanted to display to their third party developers. The two techniques related to thread balancing the cognitive simulation and rewriting the cognitive simulation optimized for AltiVec vector processing (Barbalet, 2005b).

The cognitive simulation had been optimized heavily to minimize the mathematical instructions that went into the tight loop focused around quadrants characterized as positive and negative, lower and upper halves (Barbalet, 2008). These minimized the use of binary ANDs on edge limits. The brain is defined as a 32 x 32 x 32 byte array (32768 bytes) with a previous time cycle cached brain (old brain).

```
#define  B_SIZE  (32768)
#define  B_WR  (B_SIZE - 1)
#define  F_X  (1)
#define  F_Y  (32)
#define  F_Z  (1024)
#define  B_Z  (B_SIZE - F_Z)
#define  B_Y  (B_SIZE - F_Y)
#define  B_X  (B_SIZE - F_X)
/*
 *  The basic brain formula is;
 *     b(t+1) = a*l + b(t)*m + (b(t)-b(t-
1))*n;
 *
 *  The waking mind works differently to the
sleeping mind. This is quantified
 *  with two distinct but similar equations.
There are two versions for the awake
 *  and asleep states, in this function it
is simplified to;
 *     b(t+1) = a*l_a + b(t)*l_b - b(t-
1)*l_c;
 *
 *     where, l_a = l, l_b = m+n, l_c = n
 */
#define  B_FN(ave, bra, obra)
((((ave)*l_a)+((bra)*l_b)-((obra)*l_c))>>10)
/* positive and negative, lower and upper
halves */
#define  B_P_LH  (br[loc+F_X]+br[loc+F_Y]+b
r[loc+F_Z])
#define  B_P_UH  (br[loc-F_Z]+br[loc-
F_Y]+br[loc-F_X])
#define  B_N_LH  (br[(loc+F_X)&B_
WR]+br[(loc+F_Y)&B_WR]+br[(loc+F_Z)&B_WR])
#define  B_N_UH  (br[(loc+B_Z)&B_
WR]+br[(loc+B_Y)&B_WR]+br[(loc+B_X)&B_WR])
typedef  unsigned char  n_byte;
typedef  unsigned short  n_byte2;
typedef  long  n_int;
void brain_cycle(n_byte * local, n_byte2 *
constants) {
  n_byte  *br = local, *obr = &local[B_
SIZE];
  n_int  l_a = constants[0], l_c = con-
stants[2];
```

Figure 2. An example of the Noble Ape cognitive simulation visualization resolving multiple dynamic structures paused at 11.25 degree rotations



```
  n_int  l_b = constants[1] + l_c, loc = 0;

  while (loc < F_Z) {
    n_int  average = (B_P_LH + B_N_UH);
    n_byte  obr_tmp = obr[loc];
    n_byte  br_tmp;
    obr[loc] = br_tmp = br[loc];
    br[loc++] = (n_byte)(B_FN(average, br_
tmp, obr_tmp));
  }
  while (loc < B_Z) {
    n_int  average = (B_P_LH + B_P_UH);
    n_byte  obr_tmp = obr[loc];
    n_byte  br_tmp;
    obr[loc] = br_tmp = br[loc];
    br[loc++] = (n_byte)(B_FN(average, br_
tmp, obr_tmp));
  }
  while (loc < B_SIZE) {
    n_int  average = (B_N_LH + B_P_UH);
    n_byte  obr_tmp = obr[loc];
    n_byte  br_tmp;
    obr[loc] = br_tmp = br[loc];
    br[loc++] = (n_byte)(B_FN(average, br_
tmp, obr_tmp));
  }
}
```

The Apple engineers took a single cycle of the brain calculation and optimized it into a 16-bit pipeline which allowed eight independent brain cycles to be calculated simultaneously (Barbalet, 2005b).

```
    average = br[loc_MINUS_X] + br[loc_
MINUS_Y]; // 16-bit precision
    average += br[loc_MINUS_Z] + br[loc_
PLUS_X]; // 16-bit precision
    average += br[loc_PLUS_Y] + br[loc_
PLUS_Z]; // 16-bit precision
    obr_tmp = obr[loc]; // 8-bit precision
    obr[loc] = br_tmp = br[loc]; // 8-bit
precision
    nbr_tmp = var_a * average; // 32-bit
precision
    nbr_tmp += var_b * br_tmp; // 32-bit
precision
    nbr_tmp -= var_c * obr_tmp; // 32-bit
precision
    nbr_tmp = nbr_tmp >> 10; // 32-bit pre-
cision
    br[loc] = nbr_tmp; // 8-bit precision
```

The Apple engineers proposed a 16-bit arithmetic pipe where the following was divided into high 16-bit and low 16-bit arithmetic.

```
    nbr_tmp = var_a * average; // 32-bit
precision
    nbr_tmp += var_b * br_tmp; // 32-bit
precision
    nbr_tmp -= var_c * obr_tmp; // 32-bit
precision
    nbr_tmp = nbr_tmp >> 10; // 32-bit pre-
cision
```

The brain cycles per second is equal to the number of individual brain cycles calculated every ten seconds divided by ten. If there were a troop of sixty apes and in ten seconds there were six full simulation cycles, the ape brain cycles per second would be;

60 * 6 / 10 = 36 ape brain cycles per second.

There were two reasons the ape brain cycles per second metric was useful. It provided a thorough arithmetic pipeline through the vector implementation which indicated the processor was heavily loaded. Similarly Noble Ape's multi-window real-time graphical feedback environment shared the processor load with the ape brain calculations. This showed the impact the GUI interaction had on a separate but interdependent section of the code.

Intel's optimization for SSE, implemented by Justin Landon, Michael Yi and Pallavi Mehrotra, also included a simplification in the desire calculation to limit the switching between referencing the current brain and the previous brain cycle's memory. This minimized the memory traversal through the calculation.

```
static inline void brain_avg( __m128i *avg_
hi, __m128i *avg_lo, int loc, char *br, char
*obr ) {
  int _v1 = ( loc + F_X ) & B_WR;
  int _v2 = ( loc + F_Y ) & B_WR;
  int _v3 = ( loc + F_Z ) & B_WR;
  int _v4 = ( loc + B_Z ) & B_WR;
  int _v5 = ( loc + B_Y ) & B_WR;
  int _v6 = ( loc + B_X ) & B_WR;
  int _v1_lo = _v1 & ~0x0F;
  int _v6_lo = _v6 & ~0x0F;
  int _v1_hi = ( _v1_lo + 0x10 ) & b_eand;
  int _v6_hi = ( _v6_lo + 0x10 ) & b_eand;
  __m128i a_hi = _mm_load_si128( (__m128i*)
```

```
(( _v1_hi < loc ? obr : br ) + _v1_hi) );
  __m128i a_lo = _mm_load_si128( (__m128i*)
(( _v1_lo < loc ? obr : br ) + _v1_lo) );
  __m128i a = _mm_or_si128( _mm_srli_si128(
a_lo, 1 ), _mm_slli_si128( a_hi, 15 ) );
  __m128i b = _mm_load_si128( (__m128i*)((
_v2 < loc ? obr : br ) + _v2) );
  __m128i c = _mm_load_si128( (__m128i*)((
_v3 < loc ? obr : br ) + _v3) );
  __m128i d = _mm_load_si128( (__m128i*)((
_v4 < loc ? obr : br ) + _v4) );
  __m128i e = _mm_load_si128( (__m128i*)((
_v5 < loc ? obr : br ) + _v5) );
  __m128i f_hi = _mm_load_si128( (__m128i*)
(( _v6_hi < loc ? obr : br ) + _v6_hi) );
  __m128i f_lo = _mm_load_si128( (__m128i*)
(( _v6_lo < loc ? obr : br ) + _v6_lo) );
  __m128i f = _mm_or_si128( _mm_srli_si128(
f_lo, 15 ), _mm_slli_si128( f_hi, 1 ) );
  __m128i zer = _mm_setzero_si128();
  __m128i tmp1 = _mm_add_epi16( _mm_unpack-
hi_epi8( a, zer ), _mm_unpackhi_epi8( b, zer
) );
  __m128i tmp2 = _mm_add_epi16( _mm_unpack-
hi_epi8( c, zer ), _mm_unpackhi_epi8( d, zer
) );
  __m128i tmp3 = _mm_add_epi16( _mm_unpack-
hi_epi8( e, zer ), _mm_unpackhi_epi8( f, zer
) );
  *avg_hi = _mm_add_epi16( _mm_add_epi16(
tmp1, tmp2 ), tmp3 );
  tmp1 = _mm_add_epi16( _mm_unpacklo_epi8(
a, zer ), _mm_unpacklo_epi8( b, zer ) );
  tmp2 = _mm_add_epi16( _mm_unpacklo_epi8(
c, zer ), _mm_unpacklo_epi8( d, zer ) );
  tmp3 = _mm_add_epi16( _mm_unpacklo_epi8(
e, zer ), _mm_unpacklo_epi8( f, zer ) );
  *avg_lo = _mm_add_epi16( _mm_add_epi16(
tmp1, tmp2 ), tmp3 );
}
static inline __m128i SSE_ B_FN ( __m128i
aver_hi, __m128i aver_lo, __m128i sse_
local_a, __m128i sse_local_b, __m128i
sse_local_c, __m128i sse_br_tmp, __m128i
sse_obr_tmp )
{
  const __m128i mask = _mm_set1_epi32(0xff);
  __m128i sse_zero = _mm_setzero_si128( );
  __m128i brain_lo, brain_hi, obrain_lo,
obrain_hi;
  __m128i result_01, result_02, result_03,
result_04, result_05, result_06;
  brain_lo = _mm_unpacklo_epi8 ( sse_br_tmp,
sse_zero );
  brain_hi = _mm_unpackhi_epi8 ( sse_br_tmp,
sse_zero );
  obrain_lo = _mm_unpacklo_epi8 ( sse_obr_
tmp, sse_zero );
  obrain_hi = _mm_unpackhi_epi8 ( sse_obr_
tmp, sse_zero );
  result_01 = _mm_unpacklo_epi16 ( sse_
local_a, sse_local_b );
  result_02 = _mm_unpacklo_epi16 ( aver_lo,
brain_lo );
  result_02 =  _mm_madd_epi16 ( result_01,
result_02 );
  result_03 = _mm_unpackhi_epi16 ( aver_lo,
brain_lo );
  result_03 =  _mm_madd_epi16 ( result_01,
result_03 );
  result_05 = _mm_mullo_epi16 ( obrain_lo,
sse_local_c );
  result_04 = _mm_mulhi_epi16 ( obrain_lo,
sse_local_c );
  result_02 = _mm_sub_epi32 ( result_02,
_mm_unpacklo_epi16 ( result_05, result_04 )
);
  result_02 = _mm_srli_epi32 ( result_02, 10
);


  result_03 = _mm_sub_epi32 ( result_03,
_mm_unpackhi_epi16 ( result_05, result_04 )
);
  result_03 = _mm_srli_epi32 ( result_03,
10 );
  result_02 = _mm_packs_epi32 ( _mm_
and_si128(result_02, mask), _mm_and_
si128(result_03, mask) );
  result_03 = _mm_unpacklo_epi16 ( aver_hi,
brain_hi );
  result_04 = _mm_madd_epi16 ( result_01,
result_03 );
  result_03 = _mm_unpackhi_epi16 ( aver_hi,
brain_hi );
  result_05 = _mm_madd_epi16 ( result_01,
result_03 );
  result_03 = _mm_mullo_epi16 ( obrain_hi,
sse_local_c );
  result_06 = _mm_mulhi_epi16 ( obrain_hi,
sse_local_c );
  result_04 = _mm_sub_epi32 ( result_04,
_mm_unpacklo_epi16 ( result_03, result_06 )
);
  result_04 = _mm_srli_epi32 ( result_04,
10 );
  result_05 = _mm_sub_epi32 ( result_05,
_mm_unpackhi_epi16 ( result_03, result_06 )
);
  result_05 = _mm_srli_epi32 ( result_05,
10 );
  result_04 = _mm_packs_epi32 ( _mm_
and_si128(result_04, mask), _mm_and_
si128(result_05, mask) );
  result_02 = _mm_packus_epi16 ( result_02,
result_04 );
  return result_02;
}
void brain_vect_cycle( n_byte *local, n_
byte2 *constants ) {
  __m128i local_a = _mm_set1_epi16( con-
stants[0] );
  __m128i local_c = _mm_set1_epi16( con-
stants[2] );
  __m128i local_b = _mm_set1_epi16( con-
stants[1] + constants[2] );
  __m128i *br = (__m128i*)local;
```

```
  __m128i *obr = (__m128i*)&local[32 * 32 *
32];
  int i = 0;
  for( i = 0; i < B_SIZE/16; i++ ) {
    int loc = i * 16;
    __m128i avg_lo, avg_hi;
    brain_avg( &avg_hi, &avg_lo, loc,
(char*)br, (char*)obr );
    __m128i br_tmp = _mm_load_si128( br + i
);
    __m128i obr_tmp = _mm_load_si128( obr +
i );
    _mm_store_si128( obr + i, br_tmp );
    __m128i ret = SSE_ B_FN( avg_hi, avg_lo,
local_a, local_b, local_c, br_tmp, obr_tmp
);
    _mm_store_si128( br + i, ret );
  }
}
```

Both Apple's AltiVec and Intel's SSE implementations hinged on 128-bit processing pipes. It is foreseeable that these concepts will date quickly as many more orders of these processing pipes become the norm. The underlying principles should continue to be useful.

FUTURE DIRECTIONS

The cognitive simulation in Noble Ape, described here, is based on very simple definitions for the interaction between dI/ds and dI/dt. The differential interactions for dI/ds can be expanded further. In fact the dimensionality of the Noble Ape brain, as described, favors no particular direction. It was developed to optimize the propagation of information throughout the brain. We can consider the possibility of changing the cognitive algorithm to something that distinguishes between the spatial dimension. The cognitive equation then could be rewritten as:

$$I_{t+1} = (m * I_t) + (n * dI_t/dx) + (o * dI_t/dy) + (p * dI_t/dz) + (q * dI_t/dt) \quad (9)$$

where *m* is a normalizing weighting, *n*, *o* and *p* are dimensional skewing desire weightings and *q* is a fear weighting. Or consider an additional weighting that was applied to the cognitive simulation to one or all of the constant parts:

$$I_{t+1}(x,y,z) = p(x,y,z) * I_t(x,y,z) + q(x,y,z) * dI_t/ds(x,y,z) + r(x,y,z) * dI_t/dt(x,y,z) \quad (10)$$

where *p*, *q* and *r* are location dependent weightings. Transition in the cognitive simulation between the Noble Apes falling asleep and waking up could be linearly interpolated:

$$I_{t+1}(x,y,z) = p(t) * I_t(x,y,z) + q(t) * dI_t/ds(x,y,z) + r(t) * dI_t/dt(x,y,z) \quad (11)$$

where *p*, *q* and *r* are time dependent cognitive simulation weightings that linearly transition between sleep and awake states based on the time. It begs the question if such a transition would yield the kind of stability found in the awake and asleep defined states through the weighted basis of these variables.

To-date the cognitive simulation has been volatile. The brain values are not retained between simulation runs and the cognitive simulation produces the only non-determinate element to the Noble Ape Simulation. Due to the three dimensional spatial representation of the brain model, compression algorithms suited to this data will be required for network transfer of the cognitive simulation to seed additional simulations. It is possible that binary tree spatial models be used to reduce the brain space into smaller compressible cubes and transmit this spatially reduced data over networks faster. With this model either fixed or variable size cubes will be optimized for reduced bit variable length. For example, a group of values in the range of 0 to 31 would be compressed into 5-bit variable space where 5 bytes would contain eight of these 5-bit values.

The discussion of compression also raises the possibility of brain cell sizes larger than bytes being used in the future. It is foreseeable that 16, 32

and 64 bit brain cell sizes be used for more subtle changes in the cognitive simulation. Similarly the size of the Noble Ape brains could increase from 32 x 32 x 32 to 128 x 128 x 128 or greater. Other animals in the Noble Ape environment could be cognitively modeled with smaller sub-set brains. Predatory birds and cats could have 4 x 4 x 4 and 16 x 16 x 16 brains respectively.

This document has discussed the Noble Ape cognitive simulation through algorithms and language. Increasingly demonstrations of the Noble Ape Simulation and its cognitive component are given to audiences that require immediate feedback. These demonstrations are exclusively graphical and often given by demonstrators without the mathematical and philosophical grounding discussed in this document (Damer, 2008). The visual demonstration of the cognitive simulation requires descriptive and graphic distinctions between fear and desire to be drawn. These visualization methods to actively identify the different components in the shared algorithms that make up the Noble Ape cognitive simulation may also prove beneficial in teaching these techniques to an audience quickly.

The Noble Ape Simulation contains a single time-cycle scripting language, ApeScript. The Noble Ape file format and ApeScript allows the expert user to both experiment with the existing cognitive simulation, manipulate the cognitive simulation's identity, fear and desire weightings and also devise their own cognitive simulation algorithms (Barbalet, 2005a).

Whilst it has not been the topic of this chapter, some attention to comparing and contrasting the response time and dynamic adaption of the Noble Ape cognitive simulation versus more traditional neural network models is an ongoing area of interest and development. As the Noble Ape Simulation allows for multiple cognitive simulation models, the potential to hybridize elements of PolyWorld and the Noble Ape Simulation is being discussed at the time of writing. This would provide a number of additional metrics to measure the Noble Ape cognitive simulation against the neural network contained in PolyWorld.

It is possible that the cognitive simulation method described here could be implemented next to a neural network to provide distinct modes of information processing for different circumstances. The Noble Ape cognitive simulation has been optimized for immediate reaction with less attention to long-term passive movement. The comparative tests for the cognitive simulation versus neural networks may relate to a number of factors. These include;

- The average life-time of a simulated agent,
- The average energy consumed per day by a simulated agent,
- The survival of a simulated agent through difficult conditions (drought/famine and disease),
- The survival of a simulated agent through predation of a single predator species, and,
- The survival of a simulated agent through predation of multiple predator species.

The Noble Ape cognitive simulation was optimized to respond to the predation and survival concerns in particular. The ability to test the cognitive simulation against and as a hybrid with neural networks such as PolyWorld will further strengthen this aspect of artificial life. This chapter was written as an introduction to the Noble Ape cognitive simulation method. There obviously is a lot more to explore and quantize through the cognitive simulation development.

NEW KINDS OF THINKING

The Noble Ape cognitive simulation represents a new model - understanding information transfer and basic intelligence in a dynamic environment. The basis of the cognitive simulation in simple

cellular information transfer is inspired by nature through watching growth patterns in agar. The insight and tuning the method provided has been explored in the chapter.

The Noble Ape development, including the cognitive simulation, is an ongoing work. The theme of the development to-date has been juxtaposing previously unrelated mathematical models as a means of exploring what is possible through conjunction. This is characterized here through a "philosophy first" approach with the view that the mathematics will follow. This allowed the exploration of a variety of novel mathematical methods for simulation. The skills described here in terms of adapting and tuning mathematical methods should provide inspiration for exploration.

The chapter has defined the mathematics and methodology of the Noble Ape cognitive simulation. However it does not eliminate the method being used in other simulations for similar effects. The cognitive simulation shows there are a number of possible uses as a replacement to traditional neural network models.

The advice to instill in new simulators looking to embark on their own simulation development is to ignore tradition. The respect of your peers should come through radical diversity rather than similar acceptance.

REFERENCES

Barbalet, T. S. (1997). Noble Ape Philosophic. *Noble Ape Website*. Retrieved June 20, 2008, from http://www.nobleape.com/man/philosophic.html

Barbalet, T. S. (2004). Noble ape simulation. *IEEE Computer Graphics and Applications*, *24*(2), 6–12. doi:10.1109/MCG.2004.1274054

Barbalet, T. S. (2005a). ApeScript Notes. *Noble Ape Website*. Retrieved June 20, 2008, from http://www.nobleape.com/man/apescript_notes.html

Barbalet, T. S. (2005b). Apple's CHUD Tools, Intel and Noble Ape. *Noble Ape Website*. Retrieved June 20, 2008, from http://www.nobleape.com/docs/on_apple.html

Barbalet, T. S. (2005c). *Original Manuals, Noble Ape 1996-1997*. San Mateo, CA: Cafe Press.

Barbalet, T. S. (2008). Noble Ape Source Code. *Noble Ape Website*. Retrieved June 20, 2008, from http://www.nobleape.com/sim/

Barbalet, T. S., & Daigle, J. P. (2006). Interview with John Daigle. *Biota Podcast*. Retrieved June 20, 2008, from http://www.biota.org/podcast/biota_jdaigle_062506.mp3

Barbalet, T. S., & De Jong, G. (2007). Dawkins, Memetics, Commerce and the Future: Part 2 of 3. *Biota Podcast*. Retrieved June 20, 2008, from http://www.biota.org/podcast/biota_080407.mp3

Barbalet, T. S., & Klein, J. (2006). Interview with Jonathan Klein. *Biota Podcast*. Retrieved June 20, 2008, from http://www.biota.org/podcast/biota_jklein_070806.mp3

Barbalet, T. S., & Stauffer, K. (2006). Interview with Ken Stauffer. *Biota Podcast*. Retrieved June 20, 2008, from http://www.biota.org/podcast/biota_kstauffer_080506.mp3

Cranbrook, E. (1987). *Mammals of South-East Asia*. Singapore: Oxford University Press.

Damer, B. (2008). Demonstration of Noble Ape Simulation at GreyThumb Silicon Valley. *YouTube*. Retrieved June 20, 2008, from http://www.youtube.com/watch?v=YBWxFKv3zBk

Dawkins, R. (1976). *The Selfish Gene*. New York, NY: Oxford University Press.

Dawkins, R. (1987). *The Blind Watchmaker*. New York, NY: Norton.

Emmeche, C. (1991). *The Garden in the Machine*. Princeton, NJ: Princeton University Press.

Howarth, L., & Evans, C. (1984). *Write Your Own Fantasy Games for Your Microcomputer*. London: Usborne.

Isaaman, D., & Tyler, J. (1982). *Computer Spacegames*. London: Usborne.

Kirsh, D. (1991). Today the earwig, tomorrow man? *Artificial Intelligence*, *47*, 161–184. doi:10.1016/0004-3702(91)90054-N

Langton, C. G. (1997). *Artificial Life: An Overview (Complex Adaptive Systems)*. Cambridge, MA: MIT Press.

Levy, S. (1992). *Artificial Life: A Report from the Frontier Where Computers Meet Biology*. New York, NY: Pantheon.

Moore, E. F. (1962). Machine models of self-reproduction. [Providence, RI: The American Mathematical Society.]. *Proceedings of Symposia in Applied Mathematics*, *14*, 17–33.

Rushkoff, D. (1999). A technology genius has Silicon Valley drooling - by doing things the natural way. *The Guardian*. Retrieved June 20, 2008, from http://www.guardian.co.uk/technology/1999/oct/07/onlinesupplement17

Russell, B. (1956). The philosophy of logical atomism. In R.C. Marsh (Ed.), *Logic and Knowledge, Essays 1901-50* (pp. 175-281). London: Allen and Unwin.

Sims, K. (1994). Evolving Virtual Creatures. In A. Glassner (Ed.), *ACM SIGGRAPH: Computer Graphics 1994 Proceedings* (pp. 15-22). New York, NY: ACM Press.

Starr, K. (1994). *The Starr Report*. New York, NY: Public Affairs.

Ventrella, J. J. (2005). GenePool: Exploring the Interaction Between Natural Selection and Sexual Selection. In A. Adamatzky (Ed.), *Artificial Life Models in Software* (pp. 81-96). London: Springer-Verlag.

Volterra, V. (1931). Variations and fluctuations of the number of individuals in animal species living together. In R. N. Chapman (Ed.), *Animal Ecology* (pp. 409–448). New York, NY: McGraw-Hill.

von Neumann, J., & Burks, A. W. (1966). *Theory of Self-Reproducing Automata*. Urbana, IL: University of Illinois Press.

Wittgenstein, L. (1953). *Philosophical Investigations*. Oxford: Basil Blackwell.

Yaeger, L. S. (1994). Computational Genetics, Physiology, Metabolism, Neural Systems, Learning, Vision, and Behavior or PolyWorld: Life in a New Context. In C. Langton (Ed.), *Proceedings of the Artificial Life III Conference* (pp. 263-298). Reading, MA: Addison-Wesley.